Proposal #1: Vehicle Inverter & DAQ System

High-Level Description

The purpose of this project is to create a custom, 3-phase modular inverter and data acquisition (DAQ) system for a project team's, Supermileage's, competition vehicle for the Shell Eco-Marathon. This will be for the vehicle's driver, who will rely on the DAQ display for real-time feedback, and the electrical systems team, who will use the telemetry data for system optimization and fault monitoring.

The system will accept a nominal 48V battery as input and step it down to 12V, 5V, and 3.3V to power different logic components and gate drivers. The inverter will drive the vehicle's motor and feature a modular PCB architecture that should handle DC-DC conversion and signal routing. The DAQ system will feature an LCD display, have various data logging capabilities, and have a CAN transceiver to interface with the inverter and other systems.

This system is worth building because its modular design can drastically refuse cost and time to repair, while implementing a closed-loop FOC control system to reduce motor current consumption under load to make the vehicle overall more efficient during competition. The DAQ system also allows us to keep monitor of system data, making it very useful for testing and identifying issues.

Schedule

9/10 - 9/28	Finalize the selection of major components and complete breadboard prototyping of the communication protocols.
9/29 - 10/14	Finish schematic and PCB layout for the inverter baseboard, phase modules, and DAQ board; ideally order the PCBs by this time.
10/15 - 10/27	Assemble the PCBs, test power planes and verify basic logic-level functionality. Work on debugging and redesigning any potential issues with the PCB to order new ones as soon as possible.
10/28 - 11/21	Develop the embedded software for the control algorithm and DAQ CAN bus integration, while conducting more motor testing.
11/22 - 12/7	Conduct higher load testing on the motor, finalize features, debug any system issues, and prepare final integration for

	the expo.
12/7 - 12/9	Finish final report and setup for design expo.

Budget

With a team of five members, we are provided with a budget of $1,250 and should be able to stay comfortably within this budget. With us all being a part of the Supermileage student project team, we have access to various tools and components we may need in our workshop in the Wilson Center to help speed up the process at times if it's too busy in the EECS 473 lab and minimize certain costs. Below is a draft list of parts with rough estimated costs.

Component/Purchase	**Quantity**	**Cost Per Unit**	**Total Cost**
PCB Fabrication	3	$40	$120
Microcontrollers (Either STM32 or ATMega328P)	2	$15	$30
MOSFETS	12	$8	$96
Isolated Gate Drivers	8	$3.50	$28
LCD Display	1	$25	$25
Current & Voltage Sensors	8	$4	$32
Connectors & Terminals	1	$20	$20
Capacitors, Inductors, Similar Components	1	$40	$40
		Total :	~$400

Potential Issues

- The inverter must be able to handle 38A of phase current while keeping the PCB temperatures reasonable. Our current rough ideas for our PCB should be able to account for this, but thermals are still something that must be well thought out during the whole process and can be easy to misjudge.
- The motor could produce quite a bit of back EMF which can risk blowing fuses. We can mitigate this by adding various chokes in line with the fuses to protect any of our other input components, such as the DC-DC capacitors.

Evaluation/Testing

We will know if our project is successful by whether or not the inverter can reliably drive our motor under realistic load conditions using closed-loop control, without exceeding our thermal thresholds. We will be testing if the DAQ board can successfully parse CAN bus messages from the inverter by verifying that it accurately logs and displays the correct temperatures, DC bus voltages, and battery current draw on the LCD output in real-time.

Additional Stretch Goals

If we are not running into many issues and are on schedule, it would be nice for us to implement reading the steering angle, brake pressure, and suspension travel through the DAQ and potentially implementing LoRa communication.

Proposal #2 : Smart Adaptive Ergonomic Desk Controller

High-Level Description

Prolonged sitting is a major concern for at-home work, yet many users with standard standing desks forget to adjust them throughout the day or find the manual process disruptive. Our target consumers are health-conscious remote workers and individuals with specific accessibility needs.

I am proposing a smart desk controller that actively monitors user presence and sitting duration, automatically prompting the user to adjust the desk to customized ergonomic heights. The physical system will consist of a custom PCB controller, environmental sensors, and a user interface, integrated into a motorized standing desk system.

Schedule

9/10 - 9/28	Breadboard the logic components and develop interfaces with sensors and motor drivers.
9/29 - 10/14	Finish schematic and PCB layout for the custom PCB to send out to be made.
10/15 - 10/27	Assemble the PCBs, integrate hardware limit switches, and write the interrupt-handling firmware.
10/28 - 11/21	Develop the full state machine for user profiling and automatic height adjustment. Begin testing for the sensors and timers.
11/22 - 12/7	Final debugging and expo preparation.
12/7 - 12/9	Finish final report and setup for design expo.

Budget

This project is relatively inexpensive, with the largest cost being an actual standing desk system to integrate with, but we can find them for pretty low costs while still functioning adequately for our purposes. The rest mostly includes the costs for the PCB and needed sensors for which there is an estimated list of below.

Component/Purchase	**Quantity**	**Cost Per Unit**	**Total Cost**

Standing Desk System	1	$150	$150
PCB Fabrication	1	$20	$20
SPI Compatible E-Paper Display	1	$25	$25
ATMega328P-AU Microcontroller	2	$3	$6
Ultrasonic Sensor Module	2	$6	$12
Photoelectric Limit Switches	4	$5	$20
		Total :	$236

Potential Issues

- There do not seem to be any obvious implementation issues for this type of project, the most difficult part will be properly accessing the purchased standing desk system's motor control and having it adjust accurately to adapt with our system.

Evaluation/Testing

Success will be tested by validating the presence-detection logic with human trials to ensure it accurately tracks sitting time and triggers height-adjustment sequences correctly to the desired heights.

Wireless Galvanic Vestibular Stimulation Headset for Motion-Sickness-Free VR

Cybersickness is an unofficial term used to describe the feeling of nausea, disorientation, eyestrain, and headaches that people experience from digital screens and virtual reality headsets. Similar to motion sickness, it stems from a mismatch between your body's sense of physical motion and the motion you perceive visually. One early study [1] found that 80% of participants showed symptoms of cybersickness after 10 min within a VR environment. If there was a way to induce a feeling of physical motion in the body that matched whatever the user is visually seeing, then the feeling of cybersickness could be reduced.

One way that has been explored by various groups is the use of galvanic vestibular stimulation (GVS) to mimic absent vestibular cues. The vestibular system controls a person's sense of balance and spatial orientation, and GVS artificially stimulates the system by passing a very small current (a few milliamperes) between electrodes on the mastoid processes and forehead. By changing the direction of current flow, researchers have been able to induce the physical sensations of roll, pitch, and yaw movement without any physical movement being present. While GVS has been shown to reduce cybersickness in VR [2], no easily accessible, commercial products have been created to address this problem with real consumers.

Details

The proposed device for this project would be a headband-style accessory meant to be worn alongside a VR headset. The headband would have two flexible electrode pads resting behind the ears and a small hardware enclosure resting on the back of the neck. The device will be wireless and battery powered with a relatively slow rate of data transmission, so BLE is a natural choice for communication. Inside the hardware enclosure would be a custom PCB, that handles connections between a microcontroller (with BLE capability), battery, and analog circuitry. The device should be fairly lightweight, not meant to interfere with a user's VR headset.

For the analog circuitry, the device will have a current limiter, an H-bridge style circuit changing the direction of current flow, and power domain isolation from the digital logic domain. For the embedded firmware, the microcontroller will receive data from a companion PC using BLE which will tell it how to control the flow of current through the different electrodes. The firmware should also include a watchdog timer that will zero out the output of the electrodes whenever connection with the companion PC is lost or there is an unexpected delay in the firmware.

For interfacing with the device, a companion PC application will be able to pair with the device over BLE and expose a small SDK that users can use. To demonstrate the functionality of the device, the PC application will include a ready made Unity plugin that reads a game's camera motion every frame, calculates the expected physical movement sensation, and sends commands to recreate that sensation to the headset. By having this be a Unity plugin, a VR

game can be tested with the device without any major requirements from the headset or software side besides use of the plugin.

Schedule

Shown below is a rough estimate of the proposed schedule throughout the upcoming semester, from the initial planning stage all the way up to the final design expo.

9/10-9/16	**Planning** After project groups are formed, the first priority is to determine the technology stack, communication protocols, and circuit topologies we will want to use. - Consider power consumption, cost, reliability, and physical limitations as a group for choice of microcontroller and communication protocol - Select between multiple different circuit topologies for handling the current driving between the electrodes Once the design is more fleshed out, write the draft proposal by 9/16.
9/16-10/1	**Final Proposal and Initial Prototyping** After the project proposal meetings, revise the original proposal to address any concerns or suggested changes. Then start ordering parts for the hardware prototype like the microcontroller, electrodes, and analog circuitry. - Bring up a proof of concept hardware prototype showcasing at least the analog circuitry driving current through dummy resistive loads - Bring up minimal firmware (LED blinking and BLE communication) - Finalize PCB schematic
10/1-10/15	**Breadboard, Prototype, and PCB Design** Get firmware and hardware prototypes working together to have end to end communication on the breadboard - Prototype should be able to complete flow of PC command -> BLE -> change in measured current on dummy resistive load. - Create a minimal Unity scene with captured camera motion data for future command sending - Decide on device enclosure requirements and begin layout of PCB - Have breadboard prototype ready for Milestone 1
10/15-11/10	**Safety Checks, Prototype Testing, SW and PCB Development** Begin testing the prototype more in depth and make any adjustments necessary as the Unity plugin and PCB finish development - Perform various safety checks and stress tests before testing on a real users - Adjust prototypes with any necessary changes that come up while testing - Finish PCB layout after prototype adjustments by 11/3 - Finish the SW Unity plugin and test the prototype flow of Unity -> BLE command -> change in current -> user perceived motion - Assemble PCB for Milestone 2 demonstration
11/10-11/26	**Device Enclosure, End to End System Integration**

	After Milestone 2, we will begin integration of the final working system - Finish up any assembly or debugging needed for the PCB - Create test program with Unity plugin for demo - Finish enclosure and test end to end system
11/26-12/9	**Final Testing, Additional Features, Poster, and Expo** Targeting project completion 12/3. Create a poster and start on the final report. Continue to fix any remaining minor bugs.
12/10-12/11	**Finish Final Report**

Budget

Part	Quantity	Cost Per	Cost
ESP-WROOM-32	3	$10	$30
PCB	5	$10	$50
Electrodes	20	$2	$40
Analog Circuitry	1	$80	$80
Enclosure Materials	1	$40	$40
Meta Quest Headset	1	$300	$300
Shipping	1	$80	$80
Total			$620

Potential Issues

Safety is the main concern with this design. Because the electrical current runs through the head, the hardware will need to be designed carefully to ensure the amperage never exceeds the safe limits that are set (about 5 mA). The design should also include firmware watchdog timers that are constantly checking for new communication from the PC. Only after testing sufficiently on resistive dummy loads should the designers move on to testing on real users.

Evaluation

For testing, volunteers should be shown a scene in VR with lots of movement that is designed to induce feelings of cybersickness. Volunteers should watch this scene twice, once with GVS active, and once with GVS not active (randomized on whether GVS is active first or second). If

the majority of users report less cybersickness when GVS is active compared to when it is not, that will be considered a success.

References

[1] https://www.sciencedirect.com/science/article/pii/S0003687098000386
[2] https://pubmed.ncbi.nlm.nih.gov/31938844/
[3] https://pmc.ncbi.nlm.nih.gov/articles/PMC3400934/

Snap-BAC, The Party Breathalyzer Camera

Alcohol tolerance varies person to person and day to day, so one way that law enforcement gauges how much a person is affected by alcohol is through a breathalyzer test. This test measures the alcohol in a person's breath to estimate their blood alcohol concentration (BAC). BAC levels are a helpful metric to keep track of how much someone has had to drink and what they are capable of doing, but it is uncommon to have breathalyzers in social settings. To help people keep track of their alcohol intake as well spark a fun party activity, we introduce "SNAP-BAC, the Party Breathalyzer Camera"

Details

The device will look and function as a vintage style point-and-shoot camera, outfitted with a small screen on the back, a shutter button on the top, and memory storage for the pictures. But the camera will also have a mouthpiece on the side that the user can breathe into for a breath sample. The camera will be able to read the BAC of the user before taking the picture, and will stamp the final picture with their BAC. After the picture is taken, it will also connect to the internet and upload it to a website where anyone at the social event can look at a live feed of their friends and their current BAC levels. It will be framed as a party game with a live BAC leaderboard, but the readings will be genuinely useful for users to track the BAC levels of themselves and their peers in order to make good decisions for during and after the event.

The main device control will be run by a ESP32 microcontroller with an attached camera header, and the device will also have an alcohol sensor, a camera display, some basic control buttons, and a rechargeable battery. All of these internals will be held within a lightweight vintage-style camera enclosure.

Schedule

Early Stage (9/10 - 10/1):

- Part selection and granular prototypes. Pick a microcontroller, camera, alcohol sensor, and screen, then breadboard prototype each feature to get them working individually

Mid Stage (10/1 - 10/22):

- Integrate all the separate prototypes into one larger prototype for testing. Also start working on the software backend, PCB design, and device enclosure.

Late Stage (10/22 - 11/15):

- Assemble PCB, connect all the pieces together and begin final testing

Final Stage (11/15 - 12/10):

- Enclosure, final testing, last minute polishing, user testing, and prep for design expo

Budget

Part	Quantity	Cost Per	Cost
ESP32 with Camera Header	3	$10	$30
Camera Module	3	$8	$24
MQ-3 Alcohol Sensor	3	$8	$24
OLED Display	3	$5	$15
PCB	5	$10	$50
Misc	1	$50	$50
Shipping	1	$80	$80
Total			$273

Evaluation and Potential Implementation Issues

Two concerns are the accuracy of the BAC readings and the hygiene. If a high accuracy BAC reading is not possible, we should have warnings that the BAC readings are highly inaccurate and should not be used as real measurements. We should also include replaceable mouthpieces to help with cleaning of the camera mouthpieces.

For the alcohol sensing in the early stages, we can test by bringing IPA and it will be able to read the alcohol levels in the air near it. After that, we could purchase a standard commercial breathalyzer and compare the readings in a real environment.

For the camera and IoT capabilities, this will be easily testable in a lab setting as well as a real world environment. We should test that the device is easy to use while under the influence, reliable, and fun for the users to encourage usage and attention to BAC levels.

Proposal 1: Modernized Game Boy Docking System

High-level description

In 2025 Nintendo released the Nintendo Switch 2, replacing its popular predecessor the Nintendo Switch and skyrocketing to sitting within the top 25 most sold game consoles in history, with over 23 million units sold worldwide[1]. However, even with newly-released powerful game consoles such as the Nintendo Switch 2, this massive video game console market still holds a high demand for retro game console accessories and modifications that aim to provide users with a modernized experience for their childhood game devices. Among the most popular retro gaming consoles is the Nintendo Game Boy, supported by an active and enthusiastic following including hundreds of thousands of members in various dedicated communities.

Popular Game Boy modifications include upgraded displays, improved speakers, and even replacing original hardware to achieve faster clock speeds. However, many of these existing Gameboy mods require invasive and often advanced modifications to the device itself, sometimes requiring new components, soldering, or even trimming off parts of the original device housing[2]. Providing a non-invasive, plug-and-play accessory that would allow users to achieve all of these upgrades without needing any technical background or having to modify their device, taps into a large portion of the retro gaming market.

To avoid needing to modify the existing Game Boy hardware, a docking system will be used to interface with the Game Boy through the device's EXT port. The EXT port is a serial port that intends for multiple Game Boys to communicate through a SPI-like protocol. By intercepting game state and sending user inputs to the Game Boy via the EXT port, aspects such as graphics rendering, audio output, external controllers, and displays can be offloaded from the original gameboy hardware, to newer hardware that exists within the external docking system.

The improved docking system will be powered off of a USB-C input and house a custom PCB equipped with an RP2350 MCU to support HDMI output at 1280x720 resolution at 60Hz through the MCUs HSTX peripheral, and an external PSRAM chip for buffering video frames. By utilizing the RP2350's two RISC-V cores, one core will be allocated for interfacing with the Game Boy, custom peripherals, and buffering video frames, while the other core will be responsible for display output and driving the HSTX peripheral.

Along with the docking device, custom battery-powered, USB-C rechargeable controllers will provide a modern upgrade to the standard Game Boy input. Using a custom PCB designed to support multiple button inputs and IMU measurements, these controllers will allow for fast and accurate user inputs over bluetooth.

---

[1] https://en.wikipedia.org/wiki/List_of_best-selling_game_consoles
[2] https://gameboy.github.io/wiki/mods

The final docking system will allow for users to enhance their Game Boy into a modern gaming console, without having to modify their original device. Improved graphics, audio, controller input, display output, and more will be provided through this plug-and-play docking system.

Budget

The following table provides a rough estimate for the total cost of the project. Although it is hard to predict exactly what components will be needed for the project at this time, the following estimate falls safely under the total budget. This accounts for any spares that will be manufactured or extra components that are not in the table currently.

Component	Quantity	Individual Cost	Total Cost
Nintendo Game Boy (Used)	1	$75.00	$75.00
RP2350B MCU	2	$0.90	$1.80
APS6404L (8 MB external SPRAM chip)	1	$2.25	$2.25
Infineon CYW43439 (WIFI 4 and Bluetooth 4.2)	2	$5.40	$10.80
Nintendo Game Boy Link Cable	1	$16.95	$16.95
LSM6DSV (IMU)	1	$4.70	$4.70
Rechargeable 3.7V 1000 mAh Lipo Battery	1	$11.00	$11.00
MCP73871 (USB-C Charger IC)	1	$1.71	$1.71
PCB (Dock)	1	$20.00	$20.00
PCB (Controller)	1	$15.00	$15.00
Miscellaneous Components	1	$200.00	$200.00
Shipping	1	$50.00	$50.00
Total			$409.21

Schedule

Below is a rough schedule with target deadlines for various milestones. Dates are subject to change, but should roughly follow this timeline.

Details	Target Deadline
Continue researching specific components that will be needed for dock PCB and controller PCBs. Also look into specific functionalities that the system should have. Begin working on a finalized project proposal.	9/21

Order Game Boy and link cable. Demonstrate successfully intercepting game state, and communicating with the Game Boy through the EXT port using RP2350 dev board. Begin working on PCB designs for docking system PCB and controller PCB.	10/05
Order PCBs for the docking system and controller. Finalize firmware for communicating with Game Boy using the RP2350. Begin writing firmware for displaying video on an external display using the RP2350 dev board.	10/19
Manufacture docking system PCB and controller PCB. Validate hardware functionality and demonstrate battery successfully powering the controller PCB and recharging via USB-C. Respin PCB if any issues are found. Demonstrate communicating with the Game Boy using the docking PCB.	11/02
Demonstrate displaying video to external display using the docking system PCB. Demonstrate communicating between the dock and controller using bluetooth. Demonstrate capturing user input through the controller.	11/16
Finalize any remaining integration tasks. 3D print cases for the docking system and controllers. Conduct stress testing of the entire system. Begin working on expo deliverables.	11/30
Attend expo with a well-tested system and all deliverables.	12/10

Potential Issues

The potential issues that may surface are the following:

1. Default Game Boy game cartridges do not report all game state or respond to all EXT port input.
2. Interfacing with an external display using the RP2350.
3. Designing PCBs to handle power input from USB-C for the dock and from a battery for the controller.
4. Improving graphics rendering using game state received from the Game Boy.

The largest concern on this list is issue 1. In order to ensure that this does not affect the final outcome of the project, this will be tested by the 10/05 deadline using an RP2350 dev board. In the case that the original Game Boy game cartridge does not provide the necessary interface, an existing Game Boy ROM can be downloaded and modified to publish all necessary game state and respond to user input through the EXT port.

The other notable concern is issue 2. To prevent any delays, lower resolution graphics output will be tested before the 10/19 deadline using the RP2350 dev board, and higher resolution output will be tested using the external PSRAM chip on the docking system PCB after manufacturing is completed.

Proposal 2: Phased Array Directional Speaker System

High-level description

Limiting noise pollution in busy environments such as airports, museums, and stores can be impossible with countless speakers all trying to convey different messages to multiple different audiences. Along with this, such systems currently require multiple different speakers, increasing cost and system complexity. These problems all point to the need for an affordable and easy-to-use speaker capable of directing audio output to many different locations.

To meet these requirements a phased array composed of ultrasonic transducers will be used to produce high-frequency, directional pressure waves capable of travelling long distances and demodulating into audible sound upon arrival to a target audience. A custom PCB will be equipped with a 7x7 transducer array driven across seven independent channels, allowing for precise horizontal beam steering. The PCB will be equipped with an STM32G4 MCU capable of providing high resolution timers with a 170 MHz clock for high-quality audio output. The PCB will also contain a custom H-bridge capable of supporting high-frequency switching speeds and the entire system will be powered off of USB-C input.

Other features that the speaker will support include bluetooth audio control and configurable audio output direction, ensuring that the aforementioned problems are addressed.

Budget

The following table provides a rough estimate for the total cost of the project. Although it is hard to predict exactly what components will be needed for the project at this time, the following estimate falls safely under the total budget.

Component	Quantity	Individual Cost	Total Cost
Ultrasonic Transmitters (20 Pack)	3	$15.00	$45.00
STM32G431RB MCU	1	$7.50	$7.50
Infineon CYW43439 (WIFI 4 and Bluetooth 4.2)	2	$5.40	$10.80
PCB	1	$20.00	$20.00
Miscellaneous Components	1	$200.00	$200.00

Shipping	1	$50.00	$50.00
Total			$333.30

Schedule

Below is a rough schedule with target deadlines for various milestones. Dates are subject to change, but should roughly follow this timeline.

Details	Target Deadline
Continue researching specific components that will be needed for the PCB. Also look into specific functionalities that the system should have. Begin working on a finalized project proposal.	9/21
Order ultrasonic transmitters, finalize PCB design, and order PCB. Continue researching beamforming and begin working on beamforming implementation using PDM.	10/05
Manufacture PCB and conduct hardware validation. Demonstrate basic PDM signal audio output and simple beamforming. Begin working on firmware implementation for playing audio over bluetooth.	10/19
Demonstrate basic audio output via bluetooth communication from an external device. Begin working on polished beamforming implementation.	11/02
Demonstrate beamforming implementation. Develop and demonstrate ability to control audio output direction using an external bluetooth device.	11/16
Finalize any remaining integration tasks. 3D print shells for the speaker system. Conduct stress testing of the entire system. Begin working on expo deliverables.	11/30
Attend expo with a well-tested system and all deliverables.	12/10

Potential Issues

The largest potential issue that may arise is that the purchased transducers may not be capable of providing the beamforming that is needed to achieve highly directional sound. To try and detect this issue early on in the project, a simple beamforming demo is set to happen by the 10/19 demo in our schedule. If it is determined that beamforming is not realistic with the purchased transducers, either different transducers will be researched and purchased, or alternate functionalities will be added to the speaker system.

Active Cell Balancing for UMich Solar Car Team's Battery

In multi-cell battery stacks, charging and discharging potential is limited by the most extreme cells in the entire battery. For example, a battery stack cannot continue charging once the first cell reaches maximum capacity, as that cell could potentially build up extreme heat, gas, or have an explosion. This means that the total amount of energy that other cells are below their maximum is unutilized potential energy storage. Additionally, when discharging the battery and using its energy, you cannot safely continue once the first cell has reached its minimum voltage, as any further discharging of that cell will degrade and damage it, potentially to the point of extreme failure by forming copper shunts, which permanently create internal shorts and otherwise increasing internal resistance.

Minimizing cell imbalance maximizes total energy capacity of a multi-cell battery stack. This can be achieved through either passive or active cell balancing. The UMich Solar Car Team (UMSCT) currently uses passive cell balancing, which uses resistive loads to burn off energy from the highest voltage cell, which keeps the cells fairly balanced but ultimately spends energy from the battery and sometimes generates significant heat. Active cell balancing, however, maintains cell balance by redirecting charge from the most extreme cells during charging and discharging. By actively redirecting charge from extreme cells during charging and discharging rather than burning it off, cell balance does not come at the cost of total energy.

For the UMSCT in particular, this would additionally save time during their Race Crew's preparation for events because their battery would not lose SoC (State of Charge, meaning the percentage of electrical energy currently stored in a battery compared to its total capacity) during the cell balancing process prior to a race and would therefore not need to be "topped off" with additional charging. As expressed by the team's current Engineering Director, this would have another added benefit because "the team can get more sleep in preparation for the race."

The UMSCT prefers to use custom designs and devices wherever possible when creating their vehicles, as it affords them the ability to make unlimited modifications in the name of efficiency and slight improvements. This is why they would not use active balancing modules available from companies like Analog Devices which do not have the ability to be modified or reprogrammed or altered in the ways the team likes to (and often needs to) with many of their electronic systems. This project would give the team the customizability they are looking for in any new modules they would add to their new vehicles.

Details

This active cell balancing project would function by having each cell be connected to a bi-directional flyback converter, which is a typical buck-boost converter in which the inductor is split to form a transformer, electrically isolating each side while still allowing energy flow both in and out of each cell. Transistors would control the operation of each transformer, and the transistors would be controlled by a microcontroller, likely through a multiplexing system, as the number of cells in the UMSCT's typical battery would require more transistors than there are pins on a microcontroller that would make sense in this application.

The microcontroller would receive readings from the BMS (Battery Management System) board, a device which the UMSCT has already created, which measures and calculates many parameters related to the battery and its behavior including cell voltages, charge, temperatures, and current flow. This would inform the redirection of charge through the flyback converters from extreme cells to maintain close cell balance.

While not necessarily required in the scope of this project, it may become advantageous to the project and/or the UMSCT to revise the current SoC algorithm used by the team, which could make the active balancing system more effective in practice, as it could inform the rate at which cells should be charged/discharged depending on the current state of the battery for optimal performance.

Schedule

9/13-9/26	**Research and Planning** Once the group has been formed, critically we must meet with the UMSCT's Engineering Director and representatives from their High Voltage and Microsystems subteams to discuss customer needs and requirements and obtain documentation and devices pertaining to the battery, BMS, and other closely-dependent vehicle systems. Decide the best option for microcontroller, connector header(s), and flyback converter components based on UMSCT's needs and sponsor availabilities. Draft proposal due 9/18
9/27-10/10	**Design** Select remaining auxiliary components (from UMSCT component library where possible) and order selected microcontroller and flyback converter components. Loosely design schematic prototype that can be breadboarded when components arrive. Final proposal 10/1
10/11-10/24	**Prototype and PCB** Construct a functional prototype and test it on lab-safe rechargeable batteries as a proof of concept. Should be communicating with BMS over CAN and perform correct actions on the rechargeable batteries for simulated BMS data. Finalize schematic and create and order a first PCB version.

10/25-11/7	**Software** Build out remaining functionality in software on chosen devkit and BMS. All UMSCT needs should be met in a functional capacity at this point, without regard for robustness yet. Checking for accuracy and any unintended behavior that could damage cells on the real thing. Work on safe failure modes.
11/8-11/21	**Debugging** Assembly of first PCB design. Flash software and debug and issues that need to be fixed in new PCB revision. Make any schematic adjustments and components additions necessary. Look into stretch goals if time allows. Design Demo rig for the Expo.
11/22-12/9	**Final Work** Construct and test demo rig with Expo-safe rechargeable batteries for demonstration and mock-BMS device to simulate CAN messages. Reassemble new PCB if necessary. Create documentation for UMSCT. Create the poster. Fit to the last UMSCT battery to ensure needs are met.
12/10-12/11	**Finish final report**

Budget

Because many selections will rely on UMSCT needs and requirements, it is difficult to estimate what the components used will cost, though budget will likely not be a large issue, as the team will most likely be willing to subsidize development costs where shown to be reasonably necessary for the project. The costs below are overestimated to ensure the project is reasonable from a budget standpoint.

What	**Cost Each**	**Quantity**	**Cost Total**
Microcontroller	~$10	10	$100
Flyback converter	~$10	25	$250
Other components			$150
PCBs	~$15	10	$150
Shipping			$100

Issues

The concerns I foresee for this project are:

1. Strict Wilson Center oversight following battery incident with SPARK student team
2. Space constraints to fit inside previous UMSCT battery not designed to fit module
3. Creating an Expo-safe mock battery that can be provably similar to UMSCT battery

The first issue will harshly limit testing and hands-on research for the project, and is the main reason a mock battery made specifically for the Expo is almost certain to be required by the Wilson Center. The second issue is caused by the fact that the battery created for the next UMSCT vehicle is not anticipated to be completed this semester, so the completed project will need to be proven on the previous UMSCT battery, which was not mechanically sized with this device in mind, creating potential space concerns, but will not be a condemnation of this design, as the next battery can be designed for it. The final issue is caused by the first, and is a problem because it may be a large undertaking and may not fully "prove" the project in the way putting it on the real battery would. However, all three of these issues can be solved by putting effort into researching the existing batteries and upcoming design to create a mock battery that will prove the concept to the standards of the UMSCT without creating undue hazard for the expo.

Additional stretch goal

As mentioned above, it may be helpful to revise the SoC algorithm utilized by current UMSCT software to improve project efficacy if time permits.

Adaptive Windshield Wipers

I've experienced issues with not being able to get my windshield wipers to go the exact right speed I wanted, and had issues with accidentally toggling other vehicle systems when adjusting them like windshield sprayer. An adaptive windshield wiper project would solve both problems by making speed control automatic and handsfree. This would allow for continuous speed adjustment rather than a few discrete preset speeds and the driver would not need to operate any levers to control them.

The system would work by detecting the coverage of water on the windshield, and adjusting the wiper speed so that the windshield is cleared every time the windshield surface becomes for instance 25% covered by rainwater. The water detection system could either be optic, with a camera and computer vision solution, or electronic sensor on the windshield to detect presence of water. The main difference this project could create from current OEM solutions is that it would be cheaper and installable in vehicles not built for it by OEMs.

9/13-9/26	**Research and Planning** Find what's out there, decide how we can differentiate, pick a detection solution and microcontroller base depending on the detection solution.
9/27-10/10	**Design** Select specific components, create schematic, order devkit for chosen micro.
10/11-10/24	**Prototype and PCB** Construct a functional prototype as a proof of concept. Finalize schematic and create and order a first PCB version.
10/25-11/7	**Software** Build out remaining functionality in software on chosen devkit. Checking for accuracy and any unintended behavior.
11/8-11/21	**Debugging** Assembly of first PCB design. Flash software and debug and issues that need to be fixed in new PCB revision. Make any schematic adjustments and components additions necessary. Look into stretch goals if time allows. Design Demo rig for the Expo.
11/22-12/9	**Final Work** Construct and test demo rigs. Reassemble new PCB if necessary. Create the poster.
12/10-12/11	**Finish final report**

What	Cost Each	Quantity	Cost Total
Microcontroller	~$10	10	$100
Wipers	~$50	5	$250
Glass	~$20	5	$100
Other components			$150
PCBs	~$15	10	$150
Shipping			$100

Issues

1. Wipers may be more expensive than I think, and might not be usable without OEM tech
2. Glass is dangerous
3. Detection may be unreliable

Infrastructure-Independent Item Tracking and Recovery System

Everyone has misplaced an item at some point in their lives. Survey data consistently shows this isn't some minor inconvenience: one industry survey found Americans spend about 2.5 days a year looking for lost items, collectively costing U.S. households roughly $2.7 billion annually in replacement costs[1]. Small items, like keys and wallets, are consistently among the top offenders in these surveys.

The project is a possession tracking system built around a handheld finder unit and one or more keychain-sized tags attached to commonly misplaced items. The finder is a compact handheld device that the user carries with them and uses to locate a missing item. When an item needs to be found, the finder establishes a ranging link with its corresponding tag and combines UWB distance measurements with IMU dead reckoning to estimate the tag's position relative to the user. The device can then guide the user toward the item without requiring external infrastructure such as GPS, WiFi, etc.

Existing products generally depend on a phone application and, in the case of AirTag, Apple's crowdsourced "Find My" network of nearby Apple devices to relay the tag's location[2]. This creates limitations in remote or sparsely populated areas where few devices may exist to support this sort of network. Phone-based systems also make the user's phone tied to the finding process. I instead want to build a completely self-contained finding system that can operate anywhere the two devices can establish a connection.

A summary:

- The finder and item tag communicate directly, without relying on cellular networks, Wi-Fi, cloud services, or nearby phones.
- Rather than simply reporting a tag's last known location, the finder can actively estimate the item's position as the user moves toward it.
- An onboard IMU tracks the user's movement, allowing successive UWB range measurements to be interpreted in the context of the finder's changing position.
- The complete finding process occurs on the finder, making the system suitable for environments with no network connectivity.

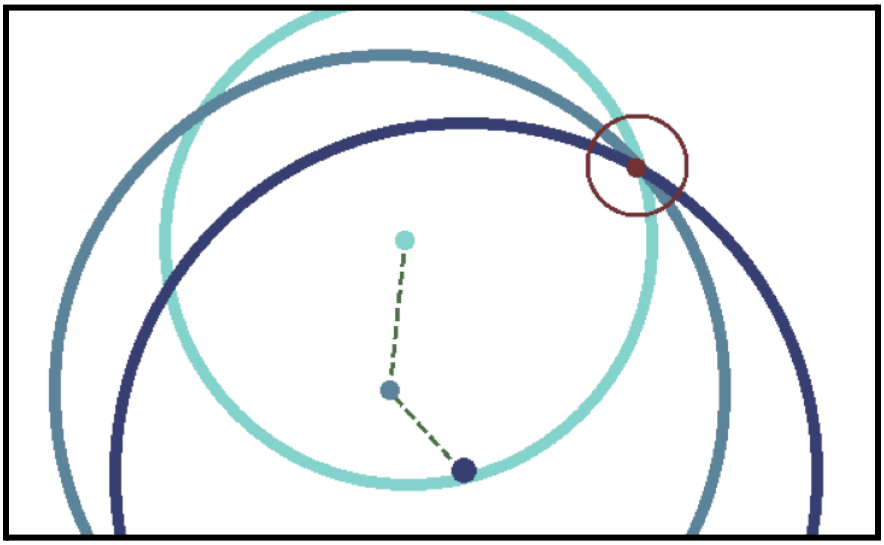
[1] https://www.prnewswire.com/news-releases/lost-and-found-the-average-american-spends-25-days-each-year-looking-for-lost-items-collectively-costing-us-households-27-billion-annually-in-replacement-costs-300449305.html

[2] https://www.apple.com/airtag/

Details

The central challenge is determining the location of a tag using only a handheld finder. GPS is not an appropriate solution because it performs poorly indoors and does not provide the spatial resolution necessary to locate an object such as a wallet or set of keys within a room. Conventional RSSI-based ranging is also unsuitable because received signal strength is strongly affected by multipath, obstructions, antenna orientation, and the surrounding environment. Instead, the proposed system is proposed to use ultra-wideband (UWB) ranging to directly measure the distance between the finder and peripheral tag. A single range measurement constrains the tag to a circle around the finder rather than uniquely determining its direction. However, the finder is mobile. By tracking its own movement with an onboard IMU, the system can combine measurements taken from multiple positions to determine the most likely location of the tag.

The system will use these measurements in a sensor-fusion algorithm. The IMU provides a prediction of the finder's motion through inertial or pedestrian dead reckoning, while UWB provides periodic absolute range measurements that constrain the position of the tag and correct accumulated drift. We will need to implement advanced filtering algorithms to properly estimate the position of the tag, like Kalman or particle filters.



In this example, the finder takes 3 distance measurements at different positions, and can triangulate the position of the object into a probable range, which will tighten with additional measurements closer to the object itself.

Schedule

Week(s)	Milestone

1-2	Finalize finder approach. Purchase eval boards, antenna modules, and IMU,display breakout.
3-4	Verify the locating using the eval boards and standalone antennas. Measure error from each source, and workshop fusion/filter algorithms. Characterize UWB update frequency, range, and LOS/NLOS behavior. Design and order finder/tag PCB.
5-6	Write system software. Bring-up manufactured PCBs. Re-spin if necessary. Port code over and begin writing application software.
7-8	In-depth testing and measurements for the final localization system and errors. Design and build enclosures. Optionally, visit a wide variety of indoor and outdoor environments to test performance.
9-10	Finish up poster, final report, finishing touches. Document work and prepare the demo.

Budget

These are very rough estimates, but we should be well within budgets.

Item	**Qty**	**Price Per**	**Totals**
Eval boards	1	$80	$80
Finder PCB	5	$4	$20
Tag PCB	10	$2	$20
UWB	5	$30	$150
Display	2	$20	$40
Other Parts +Manufacturing	2	$30	$60
Shipping	1	$80	$80
Total			**$450**

Issues

The largest issues right now are:

1. Error characterization and filtering
2. Behavior in cluttered environments
3. IMU drift

Issue 3 seems more than doable by utilizing more precise UWB range measurements to correct IMU distance measurements. Say, for example, we walk 3m according to the IMU but we get 4m closer to the object according to the UWB. Then, with reasonable confidence we could conclude our IMU is drifting slightly and apply a correction to our position/distrust the IMU more in our fusion algo.

Issues 1 and 2 will need further characterization. Especially in Non-Line of Sight (NLOS), UWB systems experience ranging bias because the signals they transmit have to take a reflected or attenuated route, which increases the arrival time of the signal and therefore positively biases the range. However, a study[3] proposes a variant of a Kalman filter specifically designed for NLOS spikes, reducing the typical spike from 0.5-1m to tenths of a meter. At the very least, UWB is far more resistant to NLOS and ranging bias than RSSI.

Evaluation and Testing

- Ranging accuracy: measure static ranged vs. ground truth distance in various environments and NLOS.
- Localization accuracy: measure error in tag distance as the finder moves in a predetermined path.
- Convergence: using the finder unit, measure how long it takes to locate a tag hidden in a realistic room compared with searching without the system.
- Fusion: compare UWB-only, inertial-only, and fused UWB + IMU localization errors.
- Operating range: determine the maximum reliable indoor ranging distance in various environments.

[3] https://arxiv.org/html/2205.05939v1#S5

Automated Tea Infusing System

Brewing specialized loose-leaf tea can be an onerous task involving precise water temperatures and infusion time. The act of serving from a traditional gaiwan (brewing cup) also carries a high risk of burns to the fingers for inexperienced tea drinkers.

This project is a compact tea infuser that is meant to entirely automate the process of brewing tea. The system stores loose tea leaves and has a water reservoir that will maintain the temperature of the water using a heating element and a water temperature sensor. The water temperature can be controlled by the user interface to support infusing a variety of teas. When a brewing session is requested, tea leaves are dispensed into the brewing cup and the water is heated to the required temperature. The water is then dispensed and the tea is steeped for a configurable amount of time.

Once the steep is complete, a servo motor attached to the side of the infusing cup tilts the spout downwards, releasing the tea into the serving mug. The mug is then removed from the machine and the tea can be drunk. A special property of these types of tea leaves is that the tea can be infused many times for an evolving taste palette. When the user is ready for a new batch, the tea leaves remain in the cup and only new water is introduced, and the infusion and serving cycle begins again. An interesting stretch goal is for the machine to discard the used tea leaves and wash itself at the end of the session, and notify the user when water is running low.

Details

One primary challenge is the dispensing of the water and regulating its temperature. PID would be the likely answer for temperature control, but dispensing of the water would require a pump or valve that is food-grade under high temperatures. In addition, the different methods of pouring water change the taste complexes of the tea; it may be worthwhile investigating a solution for different water pouring motions.

Schedule

Week(s)	Milestone
1-2	Finalize system. Order eval boards, temperature control, water heater, pump, and servo.
3-4	Prototype controlling water temperature, servo timing and motion, and tea leaf dispensing. Order PCBs.
5-6	Write system software. Bring-up manufactured PCBs. Re-spin if necessary. Port code over and begin writing application software.
7-8	Assemble and integrate the system. Design mechanical enclosures and

	user interface.
9-10	Finish up poster, final report, finishing touches. Document work and prepare the demo.

Budget

These are very rough estimates, but we should be well within budgets.

Item	Qty	Price Per	Totals
Eval boards and prototypes	1	$130	$130
Misc Parts +Manufacturing	1	$150	$150
Shipping	1	$80	$80
Total			**$260**

Issues

The only real issue that would need more investigation is food safety. All materials that come in contact with any part of the brewing process would need to be food-safe, and the water reservoir and brewing cup must withstand boiling temperatures. A simple solution to this would be to use stainless steel, which we can manufacture within the Wilson Center.

Evaluation and Testing

- Brewing accuracy. Does the system dispense the correct amount of leaves and water? Does it brew for the correct amount of time?
- Customer satisfaction. Survey customers and measure their satisfaction with ease of use, cleaning, and restocking of leaves/water.

Proposal 1: Custom Wireless Radio

High-level description

The Michigan Mars Rover Team (MRover) requires wireless communication to remotely operate their rover in competition environments. Per competition requirements, the rover must be remotely operable at distances up to 2 km with line of sight and 1 km without line of sight.[1] As such, the radio must maintain critical telemetry and control data in the worst case and support multiple concurrent camera streams whenever possible.

As a solution, we propose a custom wireless radio that provides wireless communications in multiple frequency bands for long range and high throughput. It will use a WiFi 6 2.4/5 GHz radio to transmit higher bandwidth data flows like camera streams, while a 900 MHz link will provide reliable, lower throughput communications for long distance controls and telemetry robust to environmental obstacles. The current system in use on the rover is limited to a 2.4 GHz link, which has neither the throughput of 5 GHz nor the added range of 900 MHz.[2]

The custom radio will interface with external devices through 2 main connections. A gigabit ethernet link will interface with the rover's onboard Jetson and a laptop at the base station providing a transparent connection. SMA connectors will integrate external antennas to support long range RF communications.

To support throughput upwards of 100 Mbps, significant software development effort must be put into optimizing for performance. The radio should also sort incoming IP packets and transmit them over the optimal radio link (900 MHz for controls and telemetry, 2.4/5 GHz for high throughput noncritical data). Additionally, the software will manage IP routing and provide a web interface for configuring the radio.

Implementation issues

Schedule

9/14-9/27	**Planning:** Research components that best meet our requirements and define clear project scope. Ensure components are interoperable and will not bottleneck performance unintentionally. Understand relevant regulations to ensure our design is compliant.
9/28-10/4	**Component Finalization:** Finalize component selection and begin by ordering dev kits for initial testing. Also begin designing custom PCB to integrate selected components. Dev kits will allow us to begin software integration while concurrently designing our custom PCB.

[1] https://urc.marssociety.org/home/requirements-guidelines
[2] https://www.trelink.com/wifi-frequency-ranges-2-4ghz-5ghz-900mhz/

10/5-10/25	**PCB and Software Design:** Complete PCB design and order PCB by 10/16. Ordering PCBs early will hopefully allow a second revision to be made before the PCB deadline if necessary. Develop initial software on dev kit to reduce risk after PCB assembly and accelerate overall integration timeline.
10/26-11/8	**PCB Integration:** Assemble any remaining components on the PCB and make any necessary software changes from the dev kit. If any issues arise with the custom PCB, update the design and reorder by 11/3 (PCB deadline). Design enclosure for PCB suitable for outdoor use.
11/9-11/22	**Software Optimization:** Once preliminary software shows complete functionality, begin optimizing for throughput. Test radio in a variety of conditions (best case, long range, NLOS, etc.) to ensure it meets range and bandwidth requirements.
11/23-12/10	**Final Testing:** Finalize any remaining features and iron out bugs before design expo to give plug and play reliability. Complete final report.

Budget

This project relies on several key components, including a powerful MCU and wireless transceivers. Preliminary part selections give the breakdown below. Significant margin has been included in miscellaneous parts to allow purchase of dev boards during the development process. For a team of 5 people, this comes in under budget. This parts list provides components to assemble 2 radios (one for the rover and one for the base station).

Component	Unit Cost	Quantity	Total Cost
NXP i.MX RT1170 (MCU)	$25	2	$50
NXP 88W9098 (WiFi Module)	$30	2	$60
RFDesign RFD900ux-SMT (900 MHz Module)	$85	2	$170
Microchip KSZ9131RNXI (Ethernet PHY)	$5	2	$10
Custom PCB with assembly	$100	2	$200
2.4/5 GHz Antennas	$100	2	$200
900 MHz Antennas	$10	4	$40
Miscellaneous components	$200	2	$400
Total			$1130

Technical challenges

To achieve the desired 100+ Mbps throughput, the software running on the radio must be optimized significantly to reduce overhead such as memory copying. Furthermore, this high throughput is relatively large compared to the amount of memory on the MCU, meaning buffers in the networking stack may quickly fill up if packets are not dropped efficiently. These issues seem achievable with careful software design.

On the hardware side, to achieve better range, we may want to add external amplification of the RF signals. FCC regulations allow up to 1 W of output power, while the preliminary WiFi module does not reach this limit.[3] Thus there is headroom to improve performance with amplification. That being said, as a fallback we can avoid adding this complexity to our PCB and rely solely on the range of the 900 MHz signal, so we can consider extra amplification a stretch goal. Additionally, the radio would ideally be powered by 24 V POE to eliminate the need for a separate power supply. However, this will require more complex power regulation on the PCB. As a fallback, we can include a separate power supply in addition to POE. Our PCB will also require several high speed interconnects between components, as well as traces carrying RF signals, which require more consideration than simpler traces. We may be able to avoid some of this complexity by buying integrated RF modules with antenna connectors rather than using just an IC. Our PCB will also include several components with high pin counts and small pin pitches, which will make hand assembly and modification difficult. We plan to mitigate this issue by outsourcing assembly, although this may still hinder our ability to debug.

As our project involves significant RF hardware, we will need to consider FCC regulations as well as competition requirements. For example, the University Rover Challenge imposes additional restrictions on usage of the 900 MHz band: our radio must be able to operate solely in one of the 3 900 MHz sub bands listed in the rules (902-910 MHz, 911-919 MHz, and 920-928 MHz).[4] The components we choose should support extensive configuration, enabling us to meet these requirements.

Testing Plan

To facilitate testing, we plan to develop a web dashboard for monitoring radio telemetry. Through this dashboard, we will be able to see live throughput and latency, and adjust settings including the aforementioned 900 MHz sub band. This dashboard will also include tools for benchmarking maximum throughput by generating data to send across the wireless link. We will also verify our system works in the real world by running integrated tests on the actual rover.

---

[3] https://www.ecfr.gov/current/title-47/chapter-I/subchapter-A/part-15/subpart-C/subject-group-ECFR2f2e5828339709e/section-15.247

[4] https://urc.marssociety.org/home/requirements-guidelines

Proposal 2: Everything Remote

High-level description

Many modern homes contain numerous smart devices. Beyond the traditional TV and AV receivers, increasingly many devices use wireless remotes to control smart features. As such, many people have a myriad of distinct remotes only useful for a single device in their home. This results in additional clutter and cognitive load when trying to choose and find the correct remote for a given task. To address these issues, we propose making a custom “everything remote” capable of easily controlling TVs, smart lights, ceiling fans, and more.

The everything remote will support several methods of remote control. It will incorporate an IR blaster for controlling traditional TVs and other similar devices, bluetooth for more modern streaming devices, sub-gigahertz RF for things like ceiling fans, WiFi for devices that expose web APIs, and finally Matter for modern smart lights and similar devices. With this array of interfaces, the everything remote should be able to control a wide range of devices found in a modern home.

To make the everything remote easy to use, it will support several setup modes. For simple IR devices, it will be able to record and playback pulses from existing remotes. For well documented, exposed protocols (like Roku’s External Control Protocol), extensive support will be baked into the device. Additionally, a library of reverse-engineered protocols will be included, pulling from existing open source work analyzing these interfaces. This will hopefully allow the remote to be widely applicable. Additionally, the remote will be designed to be hackable so that users can implement plugins to support more niche devices.

The physical interface of the device will include several widely-applicable buttons (directional buttons for navigation, select, etc.), as well as a small screen for displaying information to the user. It will have configuration software that allows the user to add more supported devices, manage button mappings, and configure other aspects of the device.

Implementation Issues

Schedule

9/14-9/27	**Planning:** Research devices that we can plausibly support. Select components needed to interface with these devices, and research existing software implementing protocols for these devices.
9/28-10/4	**Component Finalization:** Finalize component selection and order for prototyping. Begin designing custom PCB to integrate selected components.
10/5-10/25	**PCB Design and Prototyping:** Complete PCB design and order PCB by 10/16. Ordering PCBs early will hopefully allow a second revision to be

	made before the PCB deadline if necessary. Prototype with wireless devices on breadboard for software development.
10/26-11/8	**PCB Integration:** Assemble PCBs and integrate software. If any issues arise with the custom PCB, update the design and reorder by 11/3 (PCB deadline). Design remote housing containing PCB and battery.
11/9-11/22	**Device Integration:** Develop necessary software integration for a variety of smart devices. Port existing open source implementations and develop recording capabilities for simple messages.
11/23-12/10	**Final Testing:** Select a variety of real world devices to test with and use at the design expo. Complete final report.

Technical challenges

The main challenge of this project will be developing support for a variety of distinct devices. Many devices use proprietary, undocumented protocols that will require significant development effort to support. Although some open source reverse engineering work has already been done for many popular devices, this will need to be ported to our custom remote. Furthermore, many devices may include security features such as rolling codes that may make it impossible to control certain devices. Nonetheless, the everything remote should be able to control a significant number of devices that are either well documented or use simple IR pulses for control. However, we may have to concede on making a remote that works for truly everything.

Proposal 1: Smart Bike Lock

Bicycle theft is a large and mostly-invisible problem. The FBI's reported figures put stolen bikes at roughly 175,000 a year in the US, but survey work suggests unreported thefts push the real number closer to 2 million a year, on the order of one bike stolen every 30 seconds [1]. The outcome for a victim is bleak, fewer than 5% of stolen bikes are ever recovered [1], and a lock that's silently defeated gives the owner no chance to react and no way to find the bike afterward.

I propose building a Smart Bike Lock: a locking cable and bike-mounted electronic housing that senses an attack as it happens, sounds an alarm, and, if the bike is actually taken, reports its location over cellular using GPS. There's no physical key anywhere in the system, it unlocks either automatically over Bluetooth when the owner's phone is nearby, or by a typed PIN on a keypad on the mount. The system is split into two physical pieces connected by a single short wired harness, a sensor board embedded in the lock cable itself, and the electronics housing mounted on the bike frame.

An ideal customer is anyone who parks a bike somewhere they can't watch it: urban commuters locking up outside an office or transit stop, college students leaving bikes at dorms or bike racks between classes, and apartment or condo residents without a garage or secure storage. These are exactly the situations where theft rates are highest, and where a $30 U-lock or cable, silently cut in under a minute, gives the owner zero warning and zero recourse. A Smart Bike Lock is worth paying more for because it sells three things existing locks can't: real-time deterrence (the alarm), an actual chance to intervene before the bike is gone (the phone alert), and a shot at recovery if it isn't (GPS tracking).

Details

On the lock cable there is a small sensor board that sits permanently anchored at the cable's fixed end, inside the housing mount. The free (unlocking) end carries only a passive magnet, and a continuity wire runs the cable's full length. Together these give the cable five independent sensing capabilities:

- Continuity wire (runs the full length of the cable): trips instantly if the cable is cut.
- SW-420 vibration sensor: flags bumping, hammering, or striking of the cable.
- MPU-6050 accelerometer/gyroscope: catches slower prying, lifting, or twisting that vibration sensor alone might miss.
- Reed switch (in the housing) + magnet (at the cable's free end): senses locked vs. unlocked state.
- Strain gauge + HX711 amplifier (at the fixed end): senses tension/flex building up before a cut or snap completes.

On the bike-mounted housing:

- Motorized latch: physically grips the cable's free end and releases it electronically on a valid unlock, with no manual or key-based release anywhere in the mechanism.
- Speaker/siren: draws attention and scares off a would-be thief.
- Bluetooth via an ESP32: pairs with the owner's phone and auto-unlocks when it's in range.
- Cellular link via a SIM7000G-class module (combined LTE-M/NB-IoT + GPS): relays sensor events and the bike's GPS location to the app whenever the owner is out of Bluetooth range.

- Keypad: PIN entry when Bluetooth isn't available.
- Tamper microswitches, held closed by the assembled housing: sense if the mount itself is being taken apart.
- LiPo battery, sized for roughly 2–3 weeks of standby life between charges, with GPS/cellular duty-cycled (periodic pings rather than continuous tracking) to hit that target.
- Battery health monitoring (voltage-divider into the ESP32's ADC).
- USB-C charging.

Companion app: lock/unlock on demand, automatic unlock over Bluetooth when in range, live location tracking, battery health display, and a unified alert feed for every sensor event above.

Schedule

Dates	Milestone	Details
9/10 – 9/23	Planning	Confirm sensor/module choices (SW-420, MPU-6050, strain gauge + HX711, reed switch/magnet, continuity wire, keypad, SIM7000G-class GPS+cellular) and sketch the two-board layout (cable-end sensor board + bike-mount housing board), and draft proposal.
9/24 – 10/6	Design and part selection	Order all sensor modules, ESP32 devkit, keypad, GPS+cellular module, motorized latch, and mounting hardware. Breadboard each sensor individually into the ESP32. Final proposal due 10/1.
10/7 – 10/25	Breadboard prototype and PCB design	Get all sensors, keypad, and GPS/cellular reporting simultaneously over BLE/Wi-Fi/cellular to a bare-bones phone app. Lay out both custom PCBs (cable-end sensor board and main housing board). Begin developing 3D housing models. Order PCBs by 10/26.
10/26 – 11/10	Software development and testing on prototype	Implement per-sensor thresholding, the alarm state machine, BLE+PIN unlock logic, GPS/cellular alert reporting, and the low-battery voltage-divider check. Wire up the tamper microswitches and the motorized latch driver. Have 3D printed housing ready.
11/11 – 11/25	Software development and testing on PCB-based system	Assemble both boards, integrate into the housing and cable, and validate all detection layers. Measure false-positive rate, GPS fix time, and battery life on the core battery-only design.
11/26 – 12/9	Final testing, additional features, poster and expo	Weatherproofing and durability tests pass, finish posters, and prepare for the final report. Fix all remaining bugs.
12/10 – 12/11	Finish final report	Finish and submit the final report.

Budget

What	Quantity	Cost per	Cost Total
ESP32 dev module (BLE + WiFi)	5	$5.00	$25.00
SW-420 vibration sensor	5	$1.50	$7.50
MPU-6050 accel/gyro	5	$2.50	$12.50
Strain gauge + HX711 amplifier	5	$4.00	$20.00
Reed switch + magnet pair	5	$1.00	$5.00
Continuity sense wire (embedded in cable)	4	$8.00	$32.00
Motorized latch (electronic release)	5	$6.00	$30.00
Keypad (4x3 membrane)	5	$3.00	$15.00
GPS + Cellular module (SIM7000G-class)	5	$35.00	$175.00
Piezo speaker/siren	5	$2.00	$10.00
Tamper microswitches	10	$1.00	$10.00
LiPo battery + protection circuit	5	$8.00	$40.00
USB-C charging circuit	5	$3.00	$15.00
Custom PCB fab, 2 boards/unit (incl. 1–2 respins)	10	$20.00	$200.00
Cable, bracket, and weatherproof housing hardware	4	$25.00	$100.00
Misc. wiring, connectors, fasteners	1	$50.00	$50.00
Shipping	1	$75.00	$75.00
SIM Cellular Data Plan	1	$20	$20
Total			$842.00

Issues

- Two separate boards: all five cable-side sensors live on a small board at the cable's fixed end, so only one short harness crosses from cable to housing.
- Tuning six sensors without false alarms: vibration and movement stay soft/logged-only, while tension, cut, tamper, and an unauthorized unlock attempt all trigger hard alarms.
- Power budget: GPS and cellular are by far the biggest draws. Plan is to duty-cycle both (periodic location pings, wake-on-motion/tamper rather than continuous tracking) and size a larger LiPo to hit the standby target above. The stretch-goal solar panel would extend that further if implemented.

- Mechanical integration: the housing needs an acoustic port for the speaker, a sealed keypad membrane, and cellular/GPS antennas routed clear of the metal bike frame (plus a sun-exposed face for the solar panel, if that stretch goal is pursued).
- Weatherproofing: the housing split-line, the keypad membrane, and the cable entry port are the main routes water can get in. Plan is a gasketed housing seam, a silicone keypad overlay, and a potted/sealed cable gland, targeting roughly IP54-equivalent protection, validated with a hose-spray test.

Additional stretch goal

The main stretch goal is solar trickle-charging: a small panel on the housing's sun-exposed face to offset the draw from GPS/cellular between charges and extend standby life beyond the core battery-only target. The main board's charge circuit leaves a header open for it, so it's an add-on rather than a redesign if time allows. Additional smaller software-only stretch additions would be: selectable silent/audible alert modes in the app (a bump in a crowded rack logs quietly, while only tension, cut, and tamper events default to a loud siren), and geofencing, which automatically arms the alarm the moment the paired phone drifts out of Bluetooth range, rather than waiting for the owner to lock up manually.

Evaluation and testing

- Per-sensor validation: confirm each of the five cable-side sensing layers, plus the housing's tamper microswitches, fires correctly in isolation: vibration trips on bumping but not ordinary handling, tension trips before the cable is fully cut through, and the reed switch flags every unlock.
- False-positive / false-negative rate: run a test matrix of normal-handling scenarios (bike jostled in a rack) against simulated attacks (cutting, prying, forced unlock) and measure both error rates.
- Alert latency: measure end-to-end time from a sensor trip to the phone notification arriving, over both the BLE and cellular paths.
- Unlock reliability: confirm BLE auto-unlock only triggers within the intended range (not from a phone in a pocket walking past), and that repeated wrong PIN entries trigger a lockout rather than unlimited guesses.
- GPS accuracy and fix time: measure cold/warm fix time and positional accuracy against a known reference point.
- Battery life: measure real standby current draw, project time-to-empty against the 2–3 week target, and confirm charging behavior over USB-C (plus solar, if that stretch goal is implemented).
- Weatherproofing: pass/fail a hose-spray test on the assembled housing.

---

[1] https://thebestbikelock.com/bike-theft-statistics-us/

Proposal 2: Golf Swing Trainer

Good, repeatable ball-striking comes down to a handful of measurable mechanics: a consistent backswing-to-downswing tempo ratio, a repeatable swing plane, club path and clubface angle. Those last two are what actually determine shot shape. An out-to-in path combined with a clubface left open relative to that path is the classic slice-producing combination, while an in-to-out path with a closed face produces a hook [2][3]. Instructors generally agree that consistency in these mechanics is what separates repeatable “good shots” from inconsistent, off-target shots.

I propose a small 9-axis IMU (accelerometer + gyroscope + magnetometer) sensor that clips to a glove or grip and tracks all of this together. It times the backswing and downswing phases to compute the tempo ratio, tracks the club's swing plane through the swing, and uses the magnetometer as a heading reference to estimate club path and clubface angle at impact. It gives the golfer immediate audio/LED feedback on each swing, a tone or color change if any metric drifts off target, plus a specific cue when a swing shows the slice or hook combination, and a companion app that illustrates the swing and logs all of these over time so the golfer can see whether their mechanics are actually improving. It's a lightweight, wireless training aid aimed at real swing improvement.

An ideal customer is any golfer trying to fix a specific fault without paying for a lesson every time: mid-to-high handicap players who slice or hook consistently, golfers who practice at a range or simulator but only get feedback from where the ball went, and instructors who want objective data to back up what they're telling a student. These are exactly the golfers current tools serve poorly: a $300 to $500 commercial launch monitor gives good numbers but only after you stop and check the app, and a lesson gives you a fix but not a way to check it's sticking on your own time. A Golf Swing Trainer is worth paying for because it sells three things those alternatives can't together: real-time feedback on every single swing (the audio/LED cue), an actual diagnosis of which fault is causing the miss (the slice/hook cue) instead of a guess, and proof over time that a fix is working rather than a one-off reading (the companion app's trend view).

Details

- IMU (accelerometer + gyroscope + magnetometer): clips to a glove or grip and captures the full swing motion: timing, swing plane, club path, and clubface angle at impact.
- BLE MCU: streams swing data in real time to a small base unit or phone app.
- Audio/LED feedback unit: gives an immediate tone or color-coded cue when tempo, swing plane, path, or face angle drifts off the golfer's target on a given swing, with a distinct cue for a slice-prone (open face, out-to-in path) or hook-prone (closed face, in-to-out path) combination.
- Companion app: displays and logs tempo ratio, swing plane, club path, and clubface angle, shows trend/progress across sessions, and shot-shape tendency (slice, hook, straight), and suggests a targeted drill (like a grip check for an open face or an inside-out path drill for an out-to-in path) when a metric stays consistently off.

Schedule

Dates	Milestone	Details
9/10 – 9/23	Planning	Confirm the IMU (9-axis accel+gyro+mag), BLE MCU, and mounting approach (glove clip vs. grip sleeve). Draft the tempo-phase segmentation, swing-plane, and magnetometer-based path/face-angle detection algorithms. Draft proposal.
9/24 – 10/6	Design and part selection	Order the IMU, BLE MCU, LiPo battery, and 3D-printed mount hardware. Breadboard the IMU reading into the MCU. Final proposal due 10/1.
10/7 – 10/25	Breadboard prototype and PCB design	Get tempo, swing-plane, and path/face-angle data streaming over BLE to a bare-bones phone app. Lay out the small custom PCB. Begin developing the 3D-printed clip/mount. Order PCB by 10/26.
10/26 – 11/10	Software development and testing on prototype	Implement the tempo-phase segmentation, swing-plane, and path/face-angle algorithms, the audio/LED feedback logic, and the magnetometer hard-iron/soft-iron calibration routine on the breadboard prototype. Have the 3D-printed clip/mount ready.
11/11 – 11/25	Software development and testing on PCB-based system	Assemble the PCB, integrate into the clip/mount, and validate all four measurements (tempo, plane, path, face angle) against phone slow-motion video and alignment-stick ground truth. Add session-over-session trend tracking and shot-shape classification to the app.
11/26 – 12/9	Final testing, additional features, poster and expo	Validate shot-shape classification (slice, hook, straight) against actual ball flight across multiple users, finish posters, and prepare for the final report. Fix all remaining bugs.
12/10 – 12/11	Finish final report	Finish and submit the final report.

Budget

What	Quantity	Cost per	Cost Total
9-axis IMU (accel + gyro + mag)	3	$6.00	$18.00
BLE MCU	3	$6.00	$18.00
LiPo battery + charger	3	$5.00	$15.00

What	Quantity	Cost per	Cost Total
3D-printed clip/mount	3	$5.00	$15.00
Small custom PCB	3	$15.00	$45.00
Total			**$111.00**

Issues

- Tempo segmentation: reliably finding "top of backswing" and "impact" in noisy IMU data across different swings and users is a tuning problem rather than a feasibility unknown, since commercial swing analyzers (Blast Motion, Zepp, Garmin TruSwing) already do this successfully [3].
- Swing plane and path/face-angle estimation: integrating gyroscope data over a full swing drifts, so these estimates need a complementary filter fusing accelerometer, gyroscope, and magnetometer data rather than raw gyro integration.
- Magnetometer calibration: the magnetometer's absolute heading reference is sensitive to nearby metal (the club shaft), so a hard-iron/soft-iron calibration routine is needed before path/face-angle readings are trustworthy.
- Consistent sensor orientation: keeping the clip or sleeve oriented the same way swing to swing is the main practical detail to get right.

Additional stretch goal

A stretch goal would be to measure club impact location (heel/toe/center) using a second, small vibration/accelerometer sensor on the clubface or shaft near the head, so the app also flags mishits contributing to inconsistent shots, alongside tempo, plane, path, and face angle. It's additive (one more cheap sensor and a companion-app metric) rather than a redesign.

Evaluation and testing

Compare the device's measured tempo ratio, swing plane, club path, and clubface angle against a phone slow-motion video and alignment-stick reference lines (as ground truth) across a set of test swings, and check that all four measurements stay consistent swing-to-swing for a given golfer. Separately, confirm that the app's shot-shape classification (slice, hook, straight) matches the swing's actual ball flight, and that its session-over-session trend view reflects a golfer's real change in these mechanics across multiple range sessions, since the device is only useful as a training tool if its numbers are trustworthy, repeatable, and track real improvement.

---

[2] https://skillest.com/blog/golf-slice/
[3] https://blastmotion.com/blog/what-is-golf-swing-tempo-ratio/

Project Pre-Proposals

HW #0

Name: Nicholas Dasilva
Date: September 7, 2026

---

1. Proposal 1: Interconnected Fitness Wearables for Workout Optimization

1.1. High-Level Description

Problem Statement.

In the gym, I want to focus on making each set as effective as possible for muscle growth. During my movements I don't want to focus on counting each rep and timing my cadence of the lift. I want to focus on intensity and form and be able to see the numbers and improve over time. On top of this, tracking my workouts takes a lot of time, and it doesn't even really track what I want. I want to be able to see my range of motion, my cadence, how many reps I did and what percentage of them were good reps. I also don't want to have to setup a camera and record myself and rely on the AI to tell me what I did wrong after processing it for 10 minutes. I want instant feedback with no awkward cameras and no manual logging into some crappy app.

Proposed device.

The device I am proposing is a set of interconnected fitness wearables that are attached with lightweight straps to different parts of the body that are crucial for monitoring the movement of specific exercises. The device will be low profile allowing users to wear them without them being a distraction or getting uncomfortable. On top of the wearables there will be a mobile application that keeps track of all the important metrics when it comes to hypertrophy and building muscle such as reps, cadence, and output power. The only things the user will need to enter would be the selected workout, and the weight being used, the tech will do the rest.

At a high level, the device will operate as follows:

1. Each wearable will have its own microcontroller and IMU sensor to track movements related to each workout.
2. The microcontrollers will communicate with each other to synchronize the data and ensure accurate tracking of the workout.
3. The data will be sent to a mobile application via Bluetooth, where it will be processed and analyzed to provide feedback on the user's performance. Alternatively data can be processed on a centralized controller and then sent to the mobile application.
4. Users will be able to view their workout data, including reps, cadence, and output power, in real-time on the mobile application as well as see previously tracked workouts to ensure progressive overload.

Why it is worth building. Most high end fitness tech is either too expensive and curated only for athletes or doesn't look at the right metrics for building muscle. The device I want to build will be an affordable solution for anyone looking to optimize their workouts and improve their physique and performance in the gym.

1.2. System Concept and Major Components

The system will likely include the following subsystems:

- **Sensing/input:** A set of compact sensor boards that will track all the necessary movement data associated with selected workouts. A microcontroller with wireless capability will send that data to a device for processing.
- **Processing:** Two methods of processing can be done: a central processor that receives all data and does the post processing on it, or all the data is sent to the mobile application and the processing is done there.
- **Communication:** The wireless communication between wearables would could be Bluetooth, ZigBee, or any other low power wireless protocol. However the final send from microcontroller to mobile application will most likely have to be Bluetooth.
- **Power:** Each wearable should have compact independent battery sources that can be charged or replaced depending on the design.
- **Mechanical design:** For different parts of the body the design will be different, but the general concept would be a light weight elastic strap with a compact plastic enclosers will the sensor and communication board inside.

A custom PCB will integrate the processor and the required support circuitry. Early prototype testing will use development boards and breakout modules where useful, followed by a PCB-based final system.

1.3. Implementation Issues and Feasibility

Technical feasibility. IMUs and other motion sensors are widely used for this type of application, however the biggest challenge will be combating drift issues and the noise involved with using IMUs. This can be mitigated using different filtering and sensor fusion techniques, but it will just take some time and effort.

Reliability and usability. If the wearables are accurate and can produce consistent data, the reliability factor will weigh on the complexity of the algorithms used to process the data. Given that users are all different shapes and sizes, factoring in calibration and how to best measure the metrics that we want given the tools we have will be the biggest challenged in terms of reliability.

Mechanical and production concerns. Another large concern would be the form factor of the wearables. They need to be light weight and not get in the way of the user while they are working out. Given that, a lot of consideration is needed regarding the choice of the electronics to make sure they are compact enough to fit into the desired form factor while still meeting the functionality requirements.

1.4. Preliminary Schedule

Phase	Goals and Deliverables
Weeks 1–2: Planning and research	Assign team roles to split the workload (e.g., App Dev, PCB Design, Firmware, Mechanical). Research and finalize core components like the nRF52840 for Bluetooth and the BNO085 for motion sensing. Begin developing the mobile app front-end and test basic wireless communication on dev boards.
Weeks 3–4: Prototype and PCB design	Develop a breadboard prototype demonstrating basic IMU data reading and Bluetooth transmission to the app. Finalize the custom PCB schematic and layout. Order the PCB by the end of Week 4 to ensure we account for the roughly 10-day delay from PCB order to arrival.
Weeks 5–6: Firmware and mechanical work	While waiting for the PCBs to arrive, focus heavily on software. Develop the firmware algorithms for cadence tracking and rep counting, and refine the mobile app UI. Concurrently, 3D print prototype enclosures and test strap mounting options for user comfort.
Weeks 7–8: PCB bring-up and integration	PCBs arrive. Perform immediate hardware bring-up, verifying power and data lines. Solder surface mount components and migrate dev-board firmware to the custom boards. Begin end-to-end integration testing with the mobile app.
Weeks 9–10: Testing and refinement	Conduct rigorous real-world testing on different users to calibrate IMU algorithms for human variance. Smooth out any Bluetooth latency issues and finalize the app's real-time visualization features.
Final weeks: Expo and documentation	Finalize the working device; produce the poster and final report; perform a full demonstration rehearsal; resolve remaining minor defects.

1.5. Preliminary Budget

The project budget is expected to fit within the available budget of $250 per student. The estimate below is intentionally approximate; final part choices will follow prototype testing and design review.

Item	Estimated Cost
Microcontroller development boards and early prototype modules	$75
Sensors / input hardware	$50
Custom PCB fabrication and assembly supplies	$75
Power system: batteries, charger, cables	$25
Mechanical components: enclosure, 3D printing, fasteners, mounting	$50
Connectors, passives, spare components, and contingency	$75
Estimated total	**$350**

Budget rationale. This budget is based on a reasonably sized over estimate on each class of part to give enough overhead to ensure that we can cover any failures, testing, and iterations that may be needed. This is subject to change based on further research and prioritization of the most important features to implement.

1.6. Evaluation and Testing

Proposed success criteria are:

1. **Core functionality:** The device is capable of tracking the reps of each movement, timings of the eccentric and concentric phases, and the range of motion of the user.
2. **Accuracy/reliability:** The device should be able to match real world repetitions reliably for every set used. Each rep should be easily assessed by human eyes and the device should be able to match that. The range of motion as well as the cadence metrics should be within 5% of real world values and be consistent across multiple trials to accurately track progress over time.
3. **Usability:** Tests shall be conducted on different users about the ease of use and the satisfaction of the equipment. Scoring a 90% satisfaction or higher will be required. The test will be assessed based on the users ability to use the applications guided instructions on the use of all the equipment and its procedures.
4. **Power/physical requirement:** Each independent wearable should be able to last up to 6 hours of continuous use to cover any length of training session. Also each device should be easily chargeable and use common interfacing as dictated by user surveys.
5. **Real-world demonstration:** A user should be able to go up to a standard piece of gym equipment and enter in what the movement is and be able to execute as normal and we should see live the workout metric.

Testing will include: real world testing, controlled trials, repeated environmental tests, user feedback, comparison against a baseline, logging of measurements, etc. The final Expo demonstration will show the clearest end-to-end use case.

2. Proposal 2: Automated Toilet paper Folder

2.1. High-Level Description

Elevator pitch. The goal here is to have accurate and consistent wrapping/folding of toilet paper to dispense to the user and make the bathroom faster and much less stressful. It will also reduce the day to day use of toilet paper and save you money in the long run as having a consistent amount of toilet paper dispensing will reduce the excess use of toilet paper that is draining your wallet. Another feature would be being able to see toilet paper status within your home or business. Going to the bathroom then seeing that there is no toilet paper is always a disaster. Having detection on the device itself that notifies the user through some sort of audio or visual cue beforehand that there is no toilet paper would be extremely helpful. On top of that, having an app with a feature on the status of all toilet paper in their home or business would be extremely useful as well to keep toilet paper supplies full.

What it does and who would use it. The customer base here would be very wide, business would love this product as it can optimize the cleaning personnel's time and resources to be spent doing other things than checking stalls if they need to reload the toilet paper. It will also benefit normal homeowners and businesses alike when it comes to toilet paper usage. Wasting toilet paper is an easy way to drain your wallet, so being able to optimize that system and ensure consistency is overall a plus.

Basic device concept. The device could have multiple form factors, one of which being the toilet paper holder itself, another could be an addition that is mounted to the wall space below the toilet paper roll. However, in either configuration, an ESP32-S3 would be a great processor choice for this project as it has built in wifi and low energy Bluetooth alongside a deep sleep mode allowing the system to have an extremely long lasting battery life. The device would have a series of low power motors that would be able to unroll and fold the toilet paper with high precision alongside weight/ToF sensors to be able to detect the status on the amount of toilet paper. The status of the toilet paper will then be sent to a database with an associated ID and location to ensure differentiation of the rolls. The actual use of the device would go as follows:
User has a choice between 1, 2, or 3 for level of thickness and presses the desired button,
A set amount of toilet paper is unrolled, folded, and dispensed
The user then takes the toilet paper and the process can be repeated.

2.2. Feasibility and Implementation

Unrolling and folding technology exists, like fabric folding machines. The technical challenge lies in being able to do it in a small form factor and being able to do it consistently without allowing the end user to screw up the process or interrupt the flow. Another challenge is the toilet paper status part. Being able to detect amount of toilet paper left is doable, but will take some logic and algorithms that produce repeatable results. This can be done by using encoders and predicted amount of toilet paper, ToF sensors to check the distance between the housing and the toilet paper tube, weight based sensors with some tolerances, or even a combination of all three.

Potential implementation concerns include:

- **Primary challenge:** Mechanically feeding and folding soft, fragile toilet paper without tearing or jamming. Mitigation: Use high-friction silicone rollers, stepper motors for precise position control, and a simple accordion-fold drop mechanism rather than a complex multi-axis folder.
- **Safety, privacy, or mechanical concern:** Operating an electronic, motorized device in a highly humid bathroom environment. Mitigation: Electronics will be housed in a sealed, splash-proof 3D-printed enclosure. No privacy concerns exist since monitoring relies purely on ToF/weight sensors rather than cameras.
- **Fallback plan:** A smart dispenser that unrolls and cuts a specific length of toilet paper automatically, abandoning the complex folding mechanism while retaining the Wi-Fi roll-status tracking and database integration.

2.3. Rough Schedule

1. Research components (ESP32-S3, ToF sensors, small NEMA stepper motors) and validate the paper-feeding mechanism with a basic cardboard and roller prototype.
2. Build a breadboard version of the core functionality, using the ESP32 to drive the motors and read sensor data accurately.
3. Design and order a custom PCB; develop the Wi-Fi/database reporting firmware during the roughly 10-day delay while the PCB is fabricated.
4. Bring up and debug the PCB; integrate electronics and motors into a finalized, bathroom-safe 3D-printed enclosure.
5. Test in a representative bathroom scenario; refine dispensing speed and prepare the Expo demonstration.

2.4. Rough Budget

Category	**Estimated Cost**
Prototype hardware and development boards	$40
Sensors, outputs, and specialty components	$45
PCB fabrication and assembly supplies	$50
Enclosure, power system, and miscellaneous parts	$35
Estimated total	**$170**

2.5. Evaluation

- **Core task reliability:** The system successfully dispenses, cuts, and folds the requested length of toilet paper without tearing or jamming in 9 out of 10 controlled trials.
- **Sensor accuracy:** The ToF or weight sensor correctly identifies when a toilet paper roll falls below 10% capacity and successfully triggers an alert to the connected database.

- **Response time:** The device unrolls and dispenses the requested paper within 5 seconds of the user pressing the thickness selection button.

End of pre-proposals

uCloud: a Local Alternative to Popular Cloud Storage Leveraging Common Consumer Storage Mediums

Storage is the backbone of all modern technology. Your digital life spans years. Across those years we've all accumulated hundreds of important photos, memories, videos, and other assorted files measuring – most often – in the hundreds of GigaBytes.

While technology is progressing towards denser storage mediums and common consumer goods are likewise expanding their storage, it doesn't always meet the needs of a family for instance. And while cloud storage options are numerous and readily available, they all are transitioned towards a subscription model where your data, photos, and history are leased to you on a monthly basis. Data breaches, failures in payment, or other unfortunate realities can all contribute to the loss of memories and data important to you and your loved ones.

We leverage common consumer storage mediums such as microSD cards and USB Drives to provide an expandable at-home storage solution. Rather than shipping your photos and videos off to large companies, your photos are shipped straight back home. If you ever need more space, you can simply add another microSD, USB Drive, or daisy chain another storage module into the system.

Technical Details

uCloud is designed to be a self-hosted personal cloud storage application that leverages readily available consumer storage for a network-accessible storage option. The system consists of the hardware storage layer, the storage management stack, and a remotely accessible interface.

The hardware layer focuses on a series of ports and slots for varied common media and an interface to add and remove files from them. The hardware itself is simple with minimal interfacing to be accessible to a wide consumer base. Each connected device is detected by the hardware and concatted into a single storage pool from the user perspective. The general details for the hardware itself is largely hidden from the user.

The storage management stack is responsible for taking these devices and expanding the pool incrementally abstracting the need to reorganize files manually. This stack is also responsible for providing the abstraction used by the software networking.

The remote interface connects with a local home network as a standard device which allows files to be modified and stored from anywhere with a networked connection. Connecting to the web, HTTP/HTTPS and other networking protocols can be used to remotely transfer data. Though an endpoint could be exposed, the primary use case would be local file uploads to maintain security.

Timeline and Success

Success for a first stage uCloud for this class will look like the following and as such will be prioritized in implementation: A functional expandable storage implementing microSD and USB; safety for wired power loss to avoid data corruption; a safe procedure for removing data media with hot swap as the default mode; local network connectivity for storing files wirelessly. Stretch goals may be pursued once success as defined above is achieved.

9/10 - 10/1	**Planning** This first deadline is a large period primarily to research solutions and potential system architectures. Primarily this period will be trying to generate solutions for the primary objectives in the summary and then expanding to connecting the interfaces. The design should be finalized by the 10/1 final proposal deadline.
10/2 - 10/15	**Part Selection + Prototyping** With many of the initial design choices selected, the initial prototyping will focus on using available hardware and ordered hardware. The main goal of this prototyping period is to demonstrate storage medium detection and to establish a solid interface. Basic functionality and major components should be selected by the first milestone report.
10/16 - 10/29	**Storage Prototype and PCB Design** This period is focusing on a storage abstraction for multiple devices and demonstrating a pool data storage. Beyond that, we should develop a safe storage insertion and removal protocol with a stretch goal of hot swapping capabilities. Any additional manpower or time will be dedicated to wireless connectivity and file transferring. The PCB should be designed and submitted no later than 10/29.
10/30 - 11/12	**Local Network Connectivity and Wireless File Transfer**

	While waiting for the PCB, continue development of prototype hardware with an emphasis on local network file transferring. This time period is also for continuing progress on: storage pooling; safe storage insertion and removal; power loss safety software; and hot-swapping. The goal is to have a completed prototype with most of the software integrated into prototypes. Milestone 2 will be determined by wireless file transfer capabilities.
11/13 - 11/30	**PCB Integration and Software Testing** Assemble and bring up the custom PCBs with shipped hardware. Continue any unfinished software and integration into the final PCB design. At the end of this period all software components should be integrated and functionality should be presented as stated in the summary.
12/1 - 12/8	**Debugging and Design Expo Preparation** For the final week up until and including the expo, any last minute debugging should be finalized with stretch goals pursued on the side. The goal of this period is to have integration finalized and to focus on a successful presentation, report, and expo

Budget

Part	Qty	Cost Per	Estimated Cost
microSD sockets	9	$1.00	$9.00
USB connectors	9	$1.50	$13.50
Ethernet connectors	7	$4.00	$28.00
Ethernet Wire	3	$5.00	$15.00
Raspberry PI 3	1	$50.00	$50.00
Touchscreen	1	$35.00	$35.00
Battery	1	$10.00	$10.00
Battery Circuit	1	$15.00	$15.00
Power Cable	1	$15.00	$15.00
Passives / Headers	1	$30.00	$30.00
Other storage mediums	4	$15.00	$60.00
Total			$280.50

Issues

The most pressing implementation issues are the varied storage detection and the local endpoint file transfer.

The varied storage detection is widely implemented on most modern operating systems which points to an accessible solution at least at the application level. With a Raspberry PI for each member of the team, working at application level for testing and potentially a phase one product is very feasible.

For connecting to a local endpoint for file transfers, this is much more ambiguous from a hardware point of view. Again, with a processor designed with an operating system that can be abstracted to a software issue in the worst case.

Stretch Goals

While general wireless storage is the measure of success, stretch goals for this product include the following: daisy chaining storage expanders together, making the storage accessible from anywhere, different storage modes (file redundancy, unique archiving, etc.), mobile app integration, website integration, automatic photo backup, storage balancing, recycle bin, sync systems in different locations.

AirMonitor: An Interactive Mid-Air Touchscreen Display

Modern computer interfaces are largely restricted to physical screens – especially in consumer products. AirDisplay aims to extend the familiar touchscreen interface into free space by creating a visible aerial image that can be directly interacted with without requiring a physical surface at the image plane.

AirDisplay will function as an additional monitor rather than a standalone computer. Applications, graphics, and general-purpose computation will remain on the connected host computer. The display will receive video through HDMI and locally perform only the necessary computation for sensing and interpreting user interaction. To the host computer the device should appear only as a secondary monitor with an associated USB touchscreen.

The goal of this product is to provide an interface that can be used by ordinary applications without them being specifically designed for AirDisplay.

Technical Details

AirDisplay consists of three main subsystems: the aerial display, touch sensing, and interaction processing.

The display will take video feed over HDMI and present itself as a standard monitor to the host computer. An aerial-imaging optical element will redirect the display so that a two-dimensional image appears at a defined position in free space.

Interaction will be detected using a combination of computer vision and time-of-flight distance sensors. A camera will track the user's hand and estimate the X/Y position relative to the aerial image. A row of distance sensors positioned along one side of the interactive plane will determine when the fingertip has entered the intended touch plane.

An internal processor will perform fingertip detection, sensor polling, calibrations, and touch determination. A touch only occurs when a valid fingertip position is detected and the touch plane is broken. The processor will then report to the host through an HID message which is the only intended output of the system to the host.

The only necessary support for a host system is intended to be operating system level support for a touch screen.

Timeline and Success

Success for the first-stage AirDisplay will consist of producing a stable aerial image from an HDMI source, tracking a user's fingertip, reliably determining when the finger enters the image plane, and reporting the resulting interaction as standard touchscreen input. A successful Design Expo demonstration will allow a user to operate a normal desktop application through the aerial interface.

9/10 - 10/1	**Planning** Research aerial-imaging optics, displays, microcontrollers, cameras, and distance sensors. Prototype basic optical arrangements and determine approximate image size and geometry. Define the camera and sensor placement, USB HID architecture, and major component requirements. Finalize the overall design by the proposal deadline
10/2 - 10/15	**Subsystem Prototyping** Prototype the display and sensing systems. Produce a recognizable aerial image from an HDMI source (or predetermined image), begin camera-based fingertip tracking, and characterize the selected distance sensors. Demonstrate basic USB HID communication with a host computer.
10/16 - 10/29	**Integrated Prototype and PCB Design** Combine camera tracking and distance sensing to generate basic touch events. Develop the coordinate transformation between the camera and display. Evaluate sensor spacing and field of view. Design the custom sensor/control PCB and submit it by the PCB ordering deadline.
10/30 - 11/12	**Software Development and Testing** While waiting for PCB fabrication, continue development on prototype hardware. Improve fingertip tracking, calibration, and touch detection. Test HID touchscreen behavior.
11/13 - 11/30	**PCB Integration and Debugging** Assemble and bring up the PCB, integrate the embedded processing, camera, sensor array, display, and optical system, and construct the mechanical frame. Perform accuracy, latency,

	false-touch, and alignment testing. Resolve hardware and software issues. Debug, debug, debug.
12/1 - 12/8	**Design Expo Preparation.** Complete final calibration and reliability testing and finish the enclosure or display frame. The goal of this period is to have integration finalized and to focus on a successful presentation, report, and expo.

Budget

Part	Qty	Cost Per	Estimated Cost
Optical Element	1	$250.00	$250.00
Display	1	$40.00	$40.00
Processor	1	$25.00	$25.00
Camera	1	$40.00	$40.00
ToF sensor	8	$4.00	$32.00
PCB	1	$15.00	$15.00
HDMI Cable	1	$5.00	$5.00
HDMI Connector	1	$3.00	$3.00
Passives	1	$30.00	$30.00
Total			$440.00

Issues

The primary implementation risks are optical alignment and touch-detection reliability. The display, aerial-imaging plate, camera, and distance-sensor array must be aligned so that the sensed touch region coincides with the apparent image. Adjustable mounting and software calibration will be used to correct alignment errors.

The ToF sensors will also need early testing for field-of-view overlap, ambient-light sensitivity, and interference between adjacent sensors. Computer-vision latency and tracking accuracy will similarly be characterized before PCB design. Otherwise, the system consists primarily of low-voltage electronics and should not require unusual facilities or introduce significant safety concerns

Stretch Goals

Stretch goals include multi-touch interaction, hover detection before a touch occurs, gesture recognition, automatic calibration, higher-resolution or larger aerial imagery, audio or haptic feedback for virtual button presses.

FlowDock: A Phone Dock That Gets Louder the Longer Your Phone Is Away

Proposal #1 (primary)

High-level description

Interruptions during focused work cost more than they feel like they do at the moment. Research from UC Irvine, based on tracking real office workers over full workdays, found it takes an average of 23 minutes and 15 seconds to fully regain focus after being pulled away from a task[1]. The same line of research also found that a large share of interruptions, close to half, are self-generated rather than caused by someone else walking up to your desk[1]. This suggests that reducing self-initiated phone checking could prevent a meaningful source of lost focus during study and work sessions.

FlowDock is a small stand you set your phone on, face-down, before starting a study or work session. You press a button on the dock to start a session (basically a Pomodoro timer). While a session is running, a weight sensor built into the cradle tells the dock whether the phone is actually resting on it. If you pick the phone up and it stays off the dock, a small speaker starts playing a sound. It is quiet at first, and progressively more insistent the longer the phone stays away. The moment you set the phone back down, it stops. There is no counting of "interruptions" here, just a running clock: the longer you're away from the dock, the more annoying it gets.

The dock uses a weight sensor rather than trying to detect anything about the phone itself, which means it works with any phone, with zero setup beyond placing it there. There are no case modifications, no stickers, and no companion app to install.

The final device would use a custom PCB containing the microcontroller, the sensor interface circuitry for the weight sensor, the speaker driver, the button and LED connections, and the power circuitry. The PCB would mount inside the dock enclosure so the finished system functions as a standalone device rather than relying on a development board.

Who this is for: students or anyone doing timed focus blocks who wants something that makes stepping away from a task feel immediately annoying, instead of a phone habit they only notice after the fact. What the finished device looks like: a small desk stand, roughly the footprint of a phone, with a shallow cradle where the phone rests face-down, a start/stop button with an LED, a small speaker underneath, and a wall adapter for power.

Implementation issues

Schedule

1. Weeks 1–2: Get a devkit set up with a load cell or FSR (force-sensitive resistor) under a flat surface, and figure out a reliable weight threshold for "phone present" vs. "phone

absent." Test with a few different phone models and cases so the threshold isn't tuned to just one device, since case thickness and phone weight both vary a fair amount.

2. Weeks 3–4: Wire up a breadboard version of the full dock, including sensor, button, LED, and speaker, and get the core logic working end to end: start a session, detect phone removal, ramp up the sound the longer it's gone, and stop it the moment the phone returns.
3. Week 5: Finalize the schematic and order PCBs. Since there's roughly a 10-day gap between ordering and getting boards back, this is timed so the wait overlaps with enclosure design.
4. Weeks 5–6 (while the PCB is in transit): keep tuning the weight threshold and escalation timing on the breadboard prototype using more test phones, and design the actual dock enclosure and cradle shape.
5. Weeks 7–8: get everything working on the actual PCB, inside the real enclosure, and confirm nothing changed once it's out of the breadboard setup.
6. Weeks 9–10: run the accuracy testing described below, fix whatever breaks, add stretch goals if there's time, and prepare for the Design Expo.

Budget

Assuming a team of about five and a $250/student budget gives roughly $1,250 total to work with. Here's a first pass at how I'd expect to spend it, with margin built in for a second PCB spin:

What	Quantity	Cost per	Cost total
MCU (e.g. ATmega328P or similar)	5	$4.00	$20.00
Load cell + HX711 amplifier	5	$8.00	$40.00
Small speaker + driver	5	$4.00	$20.00
Button + LED	5	$2.00	$10.00
PCB manufacture (2 rounds × 5 boards)	10	$12.00	$120.00

Enclosure / cradle material	5	$15.00	$75.00
Misc. parts (wiring, passives, power supply)	1	$60.00	$60.00
Shipping	1	$40.00	$40.00
Total			**$385.00**

This comes in well under the ~$1,250 available, which leaves a comfortable margin for a second PCB spin or unexpected part substitutions.

Issues

The largest concerns at this time are:

1. Getting the weight threshold to reliably tell "phone present" from "nothing there" across different phone models and case thicknesses
2. Avoiding false triggers from small bumps, vibrations, or the dock being nudged
3. Picking an escalation curve for the sound that feels motivating rather than either useless or actually stressful
4. Enclosure design that holds the phone securely face-down without scratching it

Issue 1 seems addressable by testing with a handful of different phones and cases up front and picking a threshold with margin built in, rather than tuning it to just one device. Issue 2 is a smaller version of the same problem; requiring the sensor reading to remain below the removal threshold for a short persistence window before registering a "removed" event should filter out brief bumps and vibrations. Issue 3 is mostly a tuning problem once the hardware works, and something I plan to test on real people rather than guess at. Issue 4 is a minor mechanical/enclosure question rather than a functional risk so a simple rubberized cradle should be enough.

I don't see any major safety issues, and since the dock plugs into the wall rather than running on a battery, there isn't a meaningful energy budget problem either. I also don't need access to any special facilities.

Evaluation and/or testing

I plan to check three specific things to know whether this actually works:

- Presence-detection accuracy: place and remove a few different phones (different weights/case thicknesses) on the dock at least 30 times each, and confirm the sensor correctly reports present/absent 99% of the time, with no false triggers from light bumps or the desk shifting.
- Escalation timing: start a session, remove the phone for a known duration, and confirm the sound escalates on the intended schedule, and stops immediately once the phone is placed back which will be checked across several repeated trials for consistency.
- Real-world effectiveness: have a few volunteers complete matched 25-minute focus sessions with and without FlowDock, and record the number of phone pickups and the total time the phone spends off the dock/work area in each condition. I'll consider FlowDock successful if sessions using it show at least a 25% reduction in average off-dock time compared to the baseline sessions without it.

Additional stretch goals

If time allows, possible stretch goals include automatic calibration for different phone weights (so the threshold adjusts itself instead of needing to be tuned by hand), configurable focus/session settings stored in non-volatile memory so they persist across power cycles, and a small display on the dock showing the current session's total "away time," so there's a visible record of how much time was actually lost without needing to check a phone app.

[1] Gloria Mark, University of California, Irvine - "The Cost of Interrupted Work" and follow-up research on attention span, https://ics.uci.edu/~gmark/chi08-mark.pdf

---

CueTouch: A Haptic Replacement for Backstage Cue Lights
Proposal #2 ("wacky")

High-level description

Theaters have used cue lights for decades to tell an actor backstage when to make an entrance without anyone having to speak. A cue light is a small box that glows red for "standby" and green for "go," controlled by the stage manager. The problem is that this only works if the actor happens to be looking at the light at the right moment, which isn't always true because they might be mid-costume-change, turned around, or standing somewhere without a clear line of sight to the box. Wireless versions of cue lights already exist, but they're still lights. The actor still has to be constantly looking at something, and it can get chaotic backstage.

CueTouch replaces the light with a small clip-on device the actor wears on their belt or costume. The stage manager's console has an individual channel for each actor, so a specific person can be cued without accidentally cueing everyone else at once, the same way a real cue light system addresses individual people. When the stage manager sets an actor's channel to standby, that actor's clip gives a slow pulse. When it's set to go, the clip gives a distinct, sharper buzz. Both patterns are felt, not seen or heard, so it works no matter where the actor is standing or what they're in the middle of doing.

A final implementation could use a custom, compact PCB in each wearable unit, containing the MCU, wireless circuitry, vibration-motor driver, and power circuitry, with a separate PCB for the stage-manager console.

Who this is for: theaters, especially ones with actors making entrances from spots without a clear sightline to a light box, or productions that already use cue lights and want something that doesn't depend on the actor watching a specific spot. What it looks like: a small stage manager console with a channel selector and standby/go buttons, and a small clip-on receiver per actor with a vibration motor, a wireless receiver, a small MCU, and a battery.

Implementation issues

Schedule

1. Prototype wireless communication between a devkit transmitter and receiver, and confirm individual channel addressing actually works. One channel's signal shouldn't trigger a different channel's receiver.
2. Develop the two distinct haptic patterns and get the console's channel-select and standby/go controls working.
3. Design and order the wearable and console PCBs; develop the clip-on and console enclosures while waiting for the PCB.
4. Assemble and integrate the PCB-based units into their enclosures.
5. Test latency, channel isolation, interference, and haptic pattern distinguishability, and prepare for the Design Expo.

Budget-wise, for a console plus a handful of actor units (I'd aim for around 4–6 channels for a working demo), the RF modules, MCUs, vibration motors, and batteries are all inexpensive parts, and I estimate it to be around $400 including two different enclosure types and two separate PCB designs.

The real implementation issues I see are: making sure the wireless link is reliable in a backstage environment that may already be congested with other wireless gear (lighting, sound, other RF cue systems), which I'll address by testing range and interference in a realistic backstage-like space rather than just on a bench; latency, since a cue that arrives even half a second late defeats the purpose, so the wireless protocol needs to prioritize speed over things like power savings; and battery life, since the actor units need to run for the length of a full show without a recharge.

Evaluation and/or testing

I'll test three things: latency from button press on the console to vibration onset on the correct actor unit, with a target end-to-end latency under 100 ms, measured across many trials; channel isolation, confirming that cueing one channel never triggers a different actor's unit, across all channel pairs; and pattern distinguishability, having a few people wear a unit and correctly identify standby vs. go by feel alone, blind, across repeated trials, targeting at least 95% correct identification.

Wearable Teleoperated Robotic Arm

The introduction of robotics into various workspaces across Europe has led to a 4.3% decline in fatalities and a 3.2% decline in injuries[1]. The main consideration here is that these robots are not replacing the humans who are doing the work, rather they are changing what the workplace looks like. The technologies allow the robots to be the ones who handle the hazardous tasks instead of placing the workers who perform repetitive and physically intensive tasks at risk[1].

Along with improving dangerous workspaces, the introduction of robotics into healthcare, specifically for people with limited mobility has been a huge potential to improving their daily living[2]. In fact, by enabling many patients to be able to increase their independence and functionality, patients with spinal cord injuries, severe arthritis, or even the loss of a limb could now easily pick up a dropped object or reach out to close the door with ease. For many, this could reduce the need for a caretaker or the work that a caretaker would need to assist with be completely different.

What if we had a robotic arm that could perform tasks based on small gestures for both problems? Workers could control a robot in a dangerous space using simple gestures that signal what needs to get done in each scenario. Rather than removing the human aspect of their job, they can now focus on making correct decisions for a given space rather than worrying about getting injured at their next day of work. Not only would we be improving the workspaces with this technology but also bringing more mobility to many patients. Though this technology may already exist to some level; it is important to consider that many of them can cost anywhere from $20,000 to $35,000. Ideally, this product that we develop would be able to work just as well with a far smaller budget than some of the expensive gadgets in the current market.

Details

As someone is trying to accomplish a daily task, they can simply move their arm in a functional way rather than controlling the robot using some sort of button or joystick. Their arm would be wearing some sort of lightweight glove or band. This wearable would be using an IMU to sense the current state or orientation of the arm. It would also have sensors for each finger to determine the position of each finger. This data would then wirelessly stream to a microcontroller. We would want this to be done as quickly as possible. Then with the data that was just received, the controller would then compute the angles at which the joints are at and then drive the corresponding servos to the desired position. An example would be that a fully open palm could translate to grabbing onto the object in front for the robotic arm.

In a hazardous- environment, the arm would be placed near the task itself, while the operator is able to use gestures from a safe distance and the arm is able to perform the task nearby. While in the assistive role, the arm would mount to a wheelchair tray or on a side table, assisting the operator in day-to-day activities like operating a light switch, self- feeding or even retrieving objects at a reasonable distance.

Schedule

The dates selected are my best estimate that I think can be accomplished based on my 373 group's performance and is highly likely going to be modified upon group discussion.

9/9-9/25	***Planning*** On **September 9th** my current/ planned group and I plan on meeting to discuss the ideas that we have come up for this assignment. We will then most likely come up with an idea that we collectively like to find our fifth group member on the **10th of September**. Upon finalizing groups, we will be to weigh the advantages and disadvantages of using and IMU/ flex-glove sensor system and a camera- based system to determine the gesture that is being displayed by the user. We will also need to start researching servos and motors that will be used to create the movements of the robotic arm. We would need to consider the various mechanical attributes including torque as well as cost. We would also have to brainstorm methods in which data will be sent and how to implement that wireless link. If we have time we would also like the identify the microcontroller family that we would be using and start drafting our proposal for approval.
9/25-10/15	***Design and Ordering Parts*** Continue writing up our proposal making any desired changes till **October 1st.** We want to have the servos, IMU, sensors, and wireless radios picked out and ready for ordering. We also want to get our microcontroller and maybe starting with creating a mini protype with anything that seems like it could take a lot more time than intended (reading and sending the IMU data might be something new to some of us, including myself, taking more time). We want to have everything ordered by the 5th of October.
10/15-10/25	***Breadboard prototype and PCB design*** Get the end-to-end pipeline working, meaning that we want the glove to sense a gesture and send it wirelessly, causing desired movement with the servos or motors. Begin the mechanical design for the arm's links. Finalize PCB layout (we may need one for the arm and one for the glove). Plan to order PCBs by **October 25th**.
10/25-11/10	***Software development and testing*** While we wait for the PCBs to come in, we can get the software written out and tested. We might need to focus our time on getting the gestures to map to the correct angle movements on the robotic arm. This is also good buffer time for perfecting/ reprinting our robotics arm design if needed. If needed, we can order another PCB by **November 3rd.**
11/11-11/25	***Software development and testing on PCB-based system*** Combine both the PCB for the arm as well as for the glove. Start testing the design and using this time to debug anything that is not working. See if we can contact anyone to actually use our design for both intended use cases.

11/26-12/10	***Final testing, additional features, poster and expo*** Targeting project completion **December 5th.** At this point we'll get our poster done and beginning on the final report. Continue to fix any remaining bugs and make the design as robust as possible. Prepare for the expo on **December 10th.**
12/10-12/11	***Finish final report***

Budget

While a realistic budget is difficult at this time, the numbers are estimates based on very initial planning.

Item	Quantity	Cost Per	Cost Total
Microcontrollers	2	$10.00	$20.00
Servos	7	$15.00	$105.00
IMU	1	$20.00	$20.00
Sensors	5	$8.00	$40.00
Wireless modules	2	$10.00	$20.00
Custom PCBs	2	$35.00	$70.00
Battery + regulator	2	$15.00	$30.00
Shipping	1	$40.00	$40.00
Miscellaneous	1	$100	$100
Total			$445.00

Issues

The largest concerns currently are

1. Getting gesture to motion mapping with the correct kinematics
2. Interfacing with the IMU and the sensors
3. Accurately designing 2 PCB boards
4. Choosing the right parts for the project
5. Designing the mechanical aspects of the arm with CAD

Additional stretch goal

It would be nice to add sounds to help the user determine when the task has been accomplished or when it will begin to start the task. Sounds can also be very beneficial to many patients who could benefit from the assistive portion of this project.

[1] https://arxiv.org/pdf/2104.03892

[2] https://ifr.org/post/do-robots-save-lives-and-prevent-workplace-injuries

Catan- Bot

Settlers of Catan is a very popular 3-4 player game, so who could possibly want to play with 2-3 players? What if we had an extra player for this best- selling board game! When people spend so much money on the Catan extensions, why don't they spend money on the catan-bot, an extension of their board game group. The Catan- bot would be an embedded player that can manage its own resources on its turn, make trades and roll some dice. Now a full game is always available even if you can't get your roommate to join a Sunday evening game.

Details

There would be a small display that would display the bot's hand size without revealing the actual resources that it currently has, just like in an actual game. There would be a servo that would roll the dice and a few buttons where the players could keep the embedded player up to speed. For example, if a robber has been placed on one of their resources or if someone is willing to trade a card for two of theirs. There would need to be a rule-based heuristic model that would be running on a microcontroller to help determine which settlements to prioritize, which trades to make, and what resources to accept given the next best decision: building a city, building a road or getting a Development card.

Schedule

This is a modified/ simpler timeline that would indirectly follow the requirements and deadlines of the class along with the following plan.

9/24-10/13	***Build Prototype*** Build the servo that would be able to roll the dice and the button/display to update the embedded player with real time updates regarding the status of the game. We would also need to CAD any casing around the display and other parts for the embedded player to easily move around and be user friendly.
10/14-11/10	***Software Logic*** We need to be able to wire up the tracking of the game as well as the rule-based heuristic. There are a lot of rules to the game and a lot of decision making involved, therefore, this would be a very complicated portion of this idea.
11/11-11/25	***Polish and Test*** Debug any of the UI/ display and test it with real human players. This will likely result in a lot more bugs being pointed out, needing time to implement those revisions.

Budget

Item	Quantity	Cost Per	Cost Total
Microcontroller	1	$10.00	$10.00
OLED Display	1	$10.00	$10.00
Servo	1	$5.00	$5.00
Custom PCB	1	$35.00	$35.00
Shipping	1	$40.00	$40.00
Miscellaneous	1	$100	$100
Total			$200.00

Issues

The largest concerns currently are

1. The players would have to constantly be pressing buttons for the bot to be able to have “vision” of the board. The bot cannot view the board itself. This could make building the longest/ most practical road very difficult.
2. Decision making- making the heuristic competent enough to play with real players might be a very difficult task. However, this is extremely important due to the decision-making nature of the game.

Additional Stretch Goal

We could use voice recognition for the players to communicate with the Catan-bot. This would make it more user friendly rather than players having to press buttons on the display. However, this would be very difficult to accurately implement. It would misunderstand what was said and make a wrong decision, ruining the game

LumaGrid: A Low-Cost, Modular Interactive Gaming Floor

High-level description

Activate Games is an indoor gaming facility where players compete in various games that combine physical movement with video game-like gameplay. One of these games in particular is their grid room, where players move across an illuminated floor while avoiding certain tiles, memorizing patterns, and solving puzzles. Games can be cooperative or competitive, creating an engaging physical gaming experience.

As you might imagine, these commercial grade systems are quite pricey, which can be a barrier to entry for other small businesses such as smaller arcades, entertainment venues, and other organizations interested in installing this system. This project proposes a low-cost modular interactive gaming floor composed of individually illuminated and pressure-sensitive floor tiles. Each tile would likely be approximately 1' x 1', fitted with LEDs and electronics, and have a sensor detect whether a player is standing or stepping on a tile. Tile states would then be communicated to a central embedded controller responsible for maintaining game state and controlling the floor's visual response.

An initial prototype would likely be a 2x2 array, producing a floor large enough for testing purposes. If this prototype is successful, we could expand the floor to 8x8 or 12x12 since it's modular. This would also allow us to implement different game modes such as floor-is-lava, memory sequences, and puzzles-based challenges.

Implementation issues

There are a few implementation issues that need to be addressed. In regards to player safety, the flooring itself needs to be able to support people walking and jumping on it. The tile structure also needs to distribute loads so that individual sensors are not overloaded if players jump and don't land on the center of the tile. Flooring will likely be made out of polycarbonate, which is used in bulletproof glass. Testing would likely involve weights being dropped on the tiles before allowing users to play on the system.

Another issue involves sensor reliability. The system itself will need to reliably distinguish between no player on a tile, standing/jumping on a tile, briefly standing on a tile, and resist the forces players exert on the tile. The selected sensing method must therefore tolerate repeated stress while reliably detecting both brief and sustained player contact.

What is your (rough) plan for getting this working by the Design Expo?

Here is a rough draft of the roadmap that we'd likely follow to get this project working by the design expo deadline.

Planning stage (Wk 1 and 2, 9/14 - 9/25)

- Spec out tile dimensions, structural concept, sensing approaches, LED requirements, and communication protocol
- Order materials needed for initial prototyping

Prototype stage (Wk 3 and 4, 9/28 - 10/9)

- Build and test one complete tile, which includes durability, sensor reliability, LED illumination
- Expand to 2x2 prototype, retesting the above

PCB Design and Software Development (Wk 5 and 6, 10/12 - 10/23)

- Design tile-controller PCB
- Order PCBs and required components
- Write code for basic game logic, sensing algorithm, and tile communication protocol, as we wait for 10-day PCB fabrication period

Full System Integration (Wk 7 and 8, 10/28 - 11/6)

- Assemble additional tiles, PCB, electronics, and wiring
- Complete full assembly and test game reliability and functionality

Final Testing, Stretch Goals, Prepare for Expo (Wk 9+, 11/9 - 12/4)

- Correct mechanical, electrical, and software issues
- Complete safety and performance testing
- Implement any stretch goals we might have
- Prepare poster for expo and final report

Rough budget

Below is a rough draft of what materials we would need and the total costs associated with the project.

Item	Quantity	Unit Pricing	Total Price	Link
Polycarb tiles (2 pack)	30	13.99	419.7	https://www.ama
Frame materials, likely 3D printed	10	13.99	139.9	https://www.ama
Limit switches (30 pack)	8	3.99	31.92	https://www.ama
LED strips	5	13.99	69.95	amazon.com/BT
MCU, maybe ESP32 (6 pack)	2	29.96	59.92	https://www.ama
PCB, and components	1	80	80	
Electrical (wiring, Wago/busbar, fuses, battle wrap, deutsch/amphenol connectors, etc)	1	200	200	
Hardware (screws, tools, etc)	1	200	200	
Total			1201.39	

Link: EECS 473 Project - Estimated BoM

Other obvious implementation issues?

In addition to durability and sensor reliability, other implementation issues include power distribution, communication reliability, and modularity. Since there are a significant number of LEDs, these can draw a lot of current, especially as the floor scales, meaning wiring, connectors, fuses, and power-distribution hardware must be sized appropriately. Communication reliability must also be considered so that adding more tiles does not introduce excessive latency or wiring complexity. Finally, the system needs to be modular so that individual tiles, sensors, or electronics can be serviced without disassembling the entire floor.

Evaluation and/or testing

Evaluation will focus on safety, sensing reliability, and responsiveness. Each tile must withstand repeated stress without mechanical damage or deformation or electrical failure. Tiles and structure must also not create a hazardous environment for the players, which can be tested using controlled loads.

Sensor performance will be evaluated by repeatedly stepping on and around each tile and comparing the detected tile state with the known actual state. A target of >95% correct step/no-step detection during normal gameplay will be used as an initial design goal.

The system response time between a player stepping on a tile and the game registering the event should be sufficiently low such that the player does not perceive noticeable delay. A target of <100 ms end-to-end latency will be used initially.

Stretch goals

If time permits after completion of the core system, stretch goals could include additional game modes, integrated sound effects, scoreboard system, improved LED animations, and expansion to a larger number of tiles. Because the proposed design is modular, the floor should be easily expandable, perhaps 8x15 or 10x20 installations if budget and time permits.

RideGuard: Collision Warning and Airbag Safety System

High-level description

Modern vehicles include many safety systems such as airbags and collision warnings. However, the same can't be said about motorcycles, bicycles, and ebikes. This makes riders more vulnerable to collisions and injuries from accidents. While there are some expensive solutions that do exist, the high price tag can make them inaccessible to many riders.

This project proposes RideGuard, a low-cost embedded collision-warning and airbag safety system for motorcycles, bicycles, and ebikes. A front and rear mounted sensing unit would monitor vehicles and warn the user if danger is approaching. Warnings could likely be delivered in a similar manner to how modern vehicles do. Other options include LEDs, audible alerts, or handle-bar vibration. Additionally, an airbag system can also be included to reduce injury during a more severe accident.

The ultimate goal of RideGuard would therefore be to create an inexpensive safety system that could be installed on existing motorcycles, bicycles, and ebikes.

Implementation issues

Some perceivable issues include needing to distinguish between normal and dangerous situations to not trigger any false alarms, deciding when to activate the airbag system without injuring the rider, and making a low-cost solution that could work on the vast majority of bikes. Initial testing could probably be conducted on a bike or ebike using either a dummy or a human equipped with protective gear.

What is your (rough) plan for getting this working by the Design Expo?

Here is a rough draft of the roadmap that we'd likely follow to get this project working by the design expo deadline.

Planning stage (Wk 1 and 2, 9/14 - 9/25)

- Spec out sensing and crash detection technologies and algorithms, airbag systems, and bike to use
- Brainstorm how to make system compatible with modern bikes
- Order parts

Prototype stage (Wk 3 and 4, 9/28 - 10/9)

- Build a rear collision-detection prototype
- Collect sensor data and see how to improve upon prototype

PCB Design and Software Development (Wk 5 and 6, 10/12 - 10/23)

- Make PCB to include sensing, alerts, airbag, and communication systems
- Develop collision-warning and crash-detection algorithms
- Order PCB and any additional components

Full System Integration (Wk 7 and 8, 10/28 - 11/6)

- Assemble the bicycle-mounted system
- Retest the system, and make parameter adjustments as needed

Final Testing, Stretch Goals, Prepare for Expo (Wk 9+, 11/9 - 12/4)

- Correct mechanical, electrical, and software issues if not already solved
- Perform repeated collision-warning and simulated-crash tests as needed
- Turn the system into a complete bicycle-mounted demonstration unit for the expo

Rough budget

Below is a rough draft of what materials we would need and the total costs associated with the project.

Item	Quantity	Unit Pricing	Total Price
Bike	1	200	200
Sensors	3	100	300
Airbag system	2	100	200
LEDs	1	20	20
MCU	1	50	50
PCB, and components	1	80	80
Electrical (wiring, fuses, battle wrap, deutsch connectors, etc)	1	200	200
Hardware (screws, tools, etc)	1	200	200
Total			1250

Link: EECS 473 Project - Estimated BoM

Evaluation and/or testing

The system could be evaluated by repeatedly approaching the bicycle from ahead and behind at different speeds and distances. Success would require consistently detecting threatening approaches while minimizing false warnings. A successful system would likely need to reach >99% correct decision making accuracy. The airbag deployment system would also likely need to reach the same >99% threshold, in addition to deploying in time without injuring the rider.

Smart Chair for Posture Correction
Project Idea 1

High Level Description

In an increasingly digital world, as people shift to remote work and more sedentary lifestyles, poor posture is a prevalent problem facing populations. Bad posture for prolonged periods of time can often result in long-term neck and back pain, compromised spine health, and other musculoskeletal problems [1]. Students, office and desk workers, and elderly are most at risk for poor posture and the related health risks as a result of staying seated for a majority of their day with bad posture [2,3]. A smart chair aimed at correcting the posture of users through continuous monitoring and alerts could help mitigate the risks of poor posture related health issues.

Through the use of force sensitive resistors (such as textile pressure sensors) [4] embedded within the chair, ultrasonic sensors on the backrest, and software to compare the results to a baseline state described as the "good posture" state using classification algorithms, the smart chair would detect if the user has drifted to a state of poor posture. Based on the results, the chair would notify the user to correct their posture through vibrations from haptic vibration actuators. An arduino or ESP32 microcontroller would be good low-cost options for the project.

Not only is the smart chair ergonomic, it is easy to use, as on the outside it is similar to a regular office chair. Additionally, the chair will be battery powered to allow it to be used in a variety of locations. To operate the chair, the user will be able to press a button and first calibrate the chair to what is considered their "good posture" position. From there, as they sit in the chair, the chair would monitor their posture through the processes described above and send alerts as posture deviates from a set threshold.

Schedule

9/10-9/18	***Planning Stage*** After the group forming session, we will begin further developing the project as a team, forming a more comprehensive system and map out the different components involved in the Smart Chair and begin planning what microprocessor to use. Additionally, we will research the best positioning of the FSRs, ultrasonic sensors, and other sensors that might be useful. We will also research a power management system for the chair and submit a draft proposal.
9/18-10/1	***Develop Final Hardware Design + Order Components*** We will finalize our system design and select the parts needed for the project. After the proposal meeting, we will make final changes to the system design, order the selected parts and submit the final proposal.

10/1-10/15	***Prototyping stage (Milestone 1)*** Once the electrical components arrive, we will begin prototyping using breadboards and designing and testing the appropriate interfaces for the smart chair, in preparation for milestone 1. Our goal for the initial prototype will be to register changes in posture/position based on changes in force using the force sensitive resistors and send alerts in the form of vibration using the haptic vibration actuators by comparing the current posture/position to a temporary hardcoded baseline. We will move to a devboard once the breadboard prototype has been tested and start thinking about the PCB design.
10/15-11/04	***PCB Design + Ordering*** We will work on designing the PCB, keeping in mind requirements and considerations for the power management system and aim to order the PCB by 10/25/26. While waiting for the PCB to arrive, we will finish up the software for setting a baseline "good posture" measurement and handling the comparisons of realtime posture changes to the baseline. We will also decide if there is time to proceed with any stretch goals.
11/04-11/20	***Assemble + Debug PCB (Milestone 2)*** Once the PCB arrives, we will work on assembling the PCB and testing and debugging the PCB to ensure proper functionality. Once the PCB has been tested, we will integrate it with the other Smart Chair components (like the chair itself) and work on accomplishing any stretch goals that were decided upon previously.
11/20-12/05	***Final Testing + Begin Final Report/Expo Poster*** In preparation for the expo we will perform final testing and make any last minute bug fixes that arise. We will also prepare the other project deliverables such as the final report and expo poster.
12/05-12/11	***Expo + Submit Final Report***

Budget

Below is a rough budget with an estimated price breakdowns for all the components of the Smart Chair. The "quantity" column consists of bulk values (especially for FSRs and ultrasonic sensors) as the exact number has not been finalized yet. The PCB is estimated to arrive in a batch of five PCB boards. The current total estimated cost is approximately $246.38.

Part	Quantity	Unit Price	Total Price
Chair	1	$39.88	$39.88
FSRs	20	$2.99	$59.80
Ultrasonic Sensors	5	$3.95	$19.75

Haptic Vibration Actuators	10	$2.87	$28.70
Power Management + Battery	1	$48.25	$48.25
Microprocessor	1	$25.00	$25.00
PCB + Shipping	5	$25.00	$25.00

Implementation Issues

Currently, there are a couple implementation challenges that could arise. Consider the following:

1. Programming an effective classification algorithm to compare changes in posture to a baseline "good posture" reading.
2. Finding the best placement for the FSRs and the ultrasonic sensors and deciding whether to place them under the existing fabric of the chair, on top of the existing chair that we cover with an additional layer of fabric, or another location.

For issue 1, there have been a few studies done in using traditional statistical models and convolutional neural networks in classifying posture from FSR data [4] that would be worth looking into. For addressing issue 2, in the early stages of the project (before milestone 1), it would be worth taking some time to compare readings from various possible placements of the FSRs and ultrasonic sensors to see where we are able to obtain the most consistent and accurate measurements.

Stretch Goals

If there is time, a good stretch goal would be to have an attached display connected to the chair through bluetooth to display a week-long summary of the user's posture trends.

Evaluation and Testing

This project will be successful if the smart chair, once turned on and calibrated, can register changes in the posture of the user (with a 15% or smaller percent error) and send alerts in the form of vibrations. This is a key requirement the smart chair must satisfy.

[1] https://spinehealth.org/spine-posture-workplace-ergonomics/
[2] https://ieeexplore.ieee.org/document/8875249
[3] https://pmc.ncbi.nlm.nih.gov/articles/PMC12633982/
[4] https://pmc.ncbi.nlm.nih.gov/articles/PMC11086066/

Low-Cost Automated Patient Booster

Project Idea 2 (Wacky Idea)

High Level Description

Patient boosting is the process of sliding an immobile patient towards the head of the bed when they slide downwards in order to prevent pressure related injuries. To safely boost patients, healthcare workers must first place a draw sheet flat under the patient. Then, they pull the sheet towards the head of the bed, sliding the patient upwards. Not only does this process increase strain on the healthcare professionals' backs, it also risks injuring the patient as the friction from the sheet can tear the patient's skin [1]. Automating this process would both reduce the physical strain felt by healthcare workers, and reduce the likelihood of injuries to the patient.

While commercial automated patient boosters exist, they are very costly (some are approximately $9,538) [2] and require healthcare professionals to monitor the process. The automated patient booster provides a low-cost alternative with sensors to detect when a patient needs to be boosted and shifting them accordingly, reducing the need for supervision.

Through sensors like ultrasonic sensors and IMUs, and pressure mats, the automated patient booster can read a patient's position relative to the bed and if there is high localized pressure to assess if they need to be boosted [3,4]. If the patient needs to be boosted, the automated booster will initiate the motors attached to a slider sheet under the patient to move (in a conveyor belt like movement) until the patient reaches the initial position. The automated booster will also notify the healthcare professionals assigned to the patient of the change.

Schedule

9/10-9/18	***Planning Stage + Draft Proposal***
9/18-10/1	***Final Hardware Design + Order Components + Gain Access To Bed/Dummy***
10/1-10/15	***Prototyping stage (Milestone 1) + Begin Testing on Bed With Dummy***
10/15-11/04	***PCB Design + Ordering***
11/04-11/20	***Assemble + Debug PCB (Milestone 2) + Full System Integration***
11/20-12/05	***Final Testing + Begin Final Report/Expo Poster***
12/05-12/11	***Expo + Submit Final Report***

Budget

This project will require a hospital bed, sheets, and a patient dummy. We will look into renting a hospital bed and dummy or reaching out to campus resources for borrowing the bed and dummy.

The prices below are purely estimates.

Part	Total Price
Sensors	$70.75
Motor	$100
Microcontroller	$25
PCB + Shipping	$25

Implementation Issues

Consider the implementation challenges:

1. Testing will be difficult as boosting conditions with a patient dummy cannot fully simulate what it will be like with a real patient.
2. Getting timely access to a hospital bed and patient dummy on campus may be challenging (worst case plan would be to order them).
3. Working out the motor and slider sheet functionality will be tricky as one major requirement is to maximize patient safety and comfort while effectively shifting them.

It will be important to start testing with the hospital bed and dummy early on, so one of the first tasks will be to secure a testing environment with a hospital bed and patient dummy.

[1] https://medlineplus.gov/ency/patientinstructions/000429.htm
[2] hercules_repositioner
[3] https://pmc.ncbi.nlm.nih.gov/articles/PMC11479332/
[4] https://pmc.ncbi.nlm.nih.gov/articles/PMC6820230/

FreshGuard: A Sensor-Based Food Freshness Monitoring System

High-Level Description

Food waste is a significant problem in the United States. The U.S. Department of Agriculture estimates that approximately 30–40%[1] of the U.S. food supply is wasted. This also wastes the land, water, labor, energy, transportation, and storage resources used to produce that food. Spoilage is one contributor to food loss, and improved storage is one method identified by the USDA to reduce food waste.

FreshGuard is designed to reduce avoidable household food waste by giving users more information about the actual condition of food stored in a refrigerator. Instead of relying only on expiration dates, appearance, or smell, FreshGuard would monitor physical and chemical changes associated with food deterioration and alert users when food should be consumed soon.

For the prototype, FreshGuard would monitor three representative food categories: meat, bread, and fruit such as bananas. Each would be placed in a small sensing compartment. A preliminary design could use an ammonia-sensitive gas sensor for meat, a carbon-dioxide sensor for bread, and a volatile-organic-compound sensor for bananas. Temperature and humidity would also be monitored to provide environmental context.

Sensor measurements would be collected by a microcontroller-based system and compared with experimentally determined fresh and deteriorated reference values to generate an estimated freshness score and a status such as Fresh, Use Soon, or Spoilage Detected. The final prototype would integrate the sensing compartments, custom PCB, and display into a single refrigerator-based system that evaluates food based on its current condition rather than only how long it has been stored. The primary users would be households that regularly store perishable foods and want a better way to monitor when those foods should be consumed.

Possible stretch goals include adding a camera to detect visible changes such as discoloration or mold and adding a load cell to estimate how much food remains, allowing messages such as “Approximately 60 g of banana remaining - use soon.”

Implementation Issues

FreshGuard would consist of four primary subsystems: the food-sensing chambers, the custom microcontroller-based PCB, the freshness-estimation software, and the user interface.

The sensor chambers would provide the main inputs to the embedded system. Preliminary candidates include an ammonia-sensitive sensor for meat, an NDIR CO_2 sensor for bread, a VOC sensor for fruit, and temperature/humidity sensing in each chamber. These choices are preliminary because one important part of the project would be determining whether these sensors produce sufficiently repeatable changes as the selected foods deteriorate.

The processor would reside on a custom-designed PCB, as required for EECS 473 projects. The PCB would provide the required communication interfaces, power regulation, connectors, and circuitry to communicate

[1] U.S. Department of Agriculture, “Food Waste FAQs,” https://www.usda.gov/about-food/food-safety/food-loss-and-waste/food-waste-faqs, accessed Sept. 7, 2026.

with the sensors and user interface. Depending on the final sensor selection, communication could include I²C, UART, ADC inputs, or other common embedded interfaces.

Cost and production

The project should fit comfortably within the course budget of $250 per student, with the main costs coming from sensors, prototyping hardware, PCB fabrication, and containers.

	Items	Estimated Cost
1.	Gas/CO_2/VOC sensor modules	$120–180
2.	Temperature/humidity sensors	$20–40
3.	Processor development boards for prototyping	$20–40
4.	Display/user-interface hardware	$15–30
5.	PCB fabrication and PCB components	$100–180
6.	Containers, fans, connectors, wiring and miscellaneous hardware	$40–70
7.	Spare/replacement components	$40–70
8.	Stretch-goal camera/load-cell hardware	$30–60
	Estimated Total	$385–670

For a five-person team, the total budget would be about $1,250, leaving room for replacement parts or PCB revisions. Final production would mainly involve PCB assembly and modifying the sensing containers to safely expose sensors to the chamber air while protecting the electronics from food and condensation.

Reliability

Reliability is probably the largest technical challenge. Gas-sensor outputs may vary with humidity, temperature, sensor warm-up time, and interference from compounds other than the one being targeted. It would therefore be risky to assume that one fixed manufacturer threshold directly represents food spoilage.

To address this, the prototype would first collect experimental measurements from food samples as they deteriorate. Fresh samples would establish an initial baseline, and measurements would then be recorded periodically as the samples age. Several samples of each food type would be used where possible so that the system does not depend on one unusual piece of food. Temperature and humidity measurements could also be used to compensate for environmental differences.

If one candidate sensor produces weak or inconsistent results during early testing, a different sensor could be selected before the PCB is finalized. A further contingency would be to support two food categories reliably rather than keeping three categories that do not produce meaningful measurements.

Mechanical issues and facilities

The mechanical requirements should be relatively modest compared with projects requiring moving mechanical systems. The prototype could use commercially available food containers modified with small

openings for sensors, wiring, and airflow. Sensors would be mounted so that they measure the air inside the chamber without contacting the food directly.

No highly specialized facilities should be required beyond the EECS 473 lab, soldering, and electronics resources. If custom brackets or enclosure pieces become necessary, simple 3D-printed parts could be used, but the project would not depend on complex mechanical fabrication.

Safety

Because testing requires food to deteriorate, spoiled food presents a basic biological safety concern. Test samples would remain inside closed containers during data collection whenever practical, and the samples would not be eaten after testing. Containers would be cleaned or disposed of appropriately.

Electronics would also need to be isolated from food liquids and refrigerator condensation. Sensors and PCB electronics would not directly contact food, and exposed circuitry would be placed outside wet areas or inside a protective enclosure.

Energy

Energy consumption is not expected to be a major limitation for the initial prototype because FreshGuard is intended to operate inside a refrigerator and could use an external low-voltage power supply. Therefore, battery life does not need to be a fundamental design constraint. Power consumption would still be considered when selecting sensors and designing the PCB, particularly because some gas sensors may require heaters or continuous operation.

Rough Schedule:

Period	Work
Sept. 10 – Sept. 25: Planning and sensor selection	Compare candidate sensing approaches for meat, bread, and fruit. Select likely sensors, processor family, display, and chamber design. Order prototype sensor modules and begin defining how freshness will be evaluated.
Sept. 26 – Oct. 12: Sensor prototyping and initial experiments	Interface sensors with development boards and verify that measurements can be collected reliably. Construct temporary food chambers and begin gathering measurements from fresh and aging food samples. Determine whether each proposed sensor produces useful changes.
Oct. 13 – Oct. 26: Algorithm and PCB design	Develop the first freshness-score algorithm from experimental measurements. Finalize required hardware interfaces and design the custom processor-based PCB. Test sensor interfaces on the development-board prototype before ordering the PCB.
Late Oct. – Early Nov.: PCB fabrication / continued software work	Order the PCB. During the approximately ten-day manufacturing period, continue food experiments, refine the freshness thresholds and weighting algorithm, write sensor drivers, develop the display software, and test the complete system using development boards.
November: PCB assembly and system integration	Assemble and bring up the PCB. Connect the sensing chambers and user interface. Verify communication with each sensor, debug hardware and software issues, and run the complete freshness-estimation system using the custom PCB.
Late Nov. – Design Expo: Validation and stretch goals	Test the final system using additional food samples, measure classification performance, improve the enclosure, and prepare the Expo

	demonstration. If the core system works reliably, add camera-based visual monitoring and/or load-cell quantity sensing as stretch goals.

This schedule also deliberately puts the risky part (determining whether the sensors detect useful food changes early) before committing those interfaces to the PCB.

Evaluation and Testing Regarding Real-World Issues

First, a calibration dataset would be collected. Several samples of meat, bread, and bananas would be placed in the sensing chambers and monitored over time. Gas concentration, temperature, humidity, and time would be recorded as each sample transitions from fresh toward obvious deterioration. These measurements would be used to determine appropriate baseline ranges, normalization values, and classification thresholds. The exact weights and reference values would be based on the collected measurements rather than being arbitrarily selected beforehand.

Second, FreshGuard would be tested using new food samples that were not used to develop the thresholds. This would determine whether the system generalizes beyond the original calibration samples.

A successful minimum system would satisfy the following goals:

1. The custom PCB can reliably communicate with all required sensors and continuously collect their measurements.
2. The selected gas sensors show repeatable and measurable changes as the corresponding foods deteriorate.
3. FreshGuard can automatically convert sensor measurements into an estimated freshness score and one of at least three conditions: Fresh, Use Soon, and Spoilage Detected.
4. When tested on new samples, the system correctly distinguishes the defined food-condition categories at a target accuracy of approximately 80% or better.
5. The final system operates as one integrated device rather than requiring a computer to manually process individual sensor measurements.

At the Design Expo, several food samples could be prepared beforehand at different stages of deterioration. A user could select or place a sample into the appropriate chamber, after which FreshGuard would display the measured conditions and estimated freshness status. This creates a demonstration that is observable rather than simply showing previously collected graphs.

FireTrace: Wearable Return-Path Navigation for Firefighters

High-Level Description

Firefighters entering an unfamiliar building may have difficulty retracing their path during low-visibility conditions caused by smoke or darkness. FireTrace is a wearable navigation system designed to record a firefighter's movement through a building and then provide directional guidance back toward the original entry point.

The prototype would use motion sensors such as an IMU to detect changes in direction and estimate movement while the firefighter travels through the building. The system would store the sequence of major turns and movements taken during entry. When the firefighter activates a return mode, the device would reverse that path and provide directional cues through vibration motors or small visual indicators mounted on the glasses.

The finished prototype would consist of wearable glasses or a helmet-mounted unit containing the sensing hardware, a microcontroller-based PCB, and haptic feedback components. The goal is not to create a complete indoor mapping system, but rather a compact path-retracing device that could help guide a user back along the approximate route they originally traveled.

At the Design Expo, the system could be demonstrated using a small maze or obstacle course. A user would first walk through the course while FireTrace records the path, then activate return mode and attempt to navigate back to the starting point using only the device's guidance.

Implementation Issues

The main components would include an IMU, microcontroller, haptic motors or LEDs, battery, and custom PCB. The largest technical challenge is dead-reckoning drift, since small IMU errors can accumulate over time and reduce position accuracy. To reduce this problem, the prototype could focus on recording major turns and estimating relative movement rather than attempting precise continuous indoor positioning.

A rough implementation plan would be:

1. Test IMU-based turn and movement detection.
2. Develop an algorithm to record and reverse a path.
3. Build a wearable prototype using a development board.
4. Design and order the custom PCB.
5. Integrate haptic feedback and test the system in a controlled maze.
6. Refine the algorithm and prepare the Design Expo demonstration.

The project should remain within budget because the main hardware consists of an IMU, microcontroller, haptic motors or LEDs, battery, wearable enclosure, and custom PCB. A preliminary prototype is expected to cost approximately $150–250**,** leaving substantial room within the course budget for replacement parts or PCB revisions.

Safety would be handled by limiting testing to controlled indoor environments rather than real emergency conditions. No actual smoke or hazardous environments would be required for the prototype.

Evaluation and Testing

The system would be evaluated by having users walk through several predefined routes containing turns and straight segments. FireTrace would record each route and then guide the user back to the starting point.

A successful prototype would:

- correctly detect major turns,
- store and reverse the recorded path,
- provide understandable directional feedback,
- guide users back to the starting point in at least 80% of controlled test routes**.**

For the Expo, the clearest demonstration would be a small maze where someone follows the device's return instructions after completing an unfamiliar route.

Why It Fits EECS 473

FireTrace has a real-world emergency-response application and requires sensing, embedded processing, wearable hardware, haptic feedback, algorithm development, and a custom microprocessor-based PCB. It also tackles an existing navigation problem in a different way by focusing on retracing a user's own path rather than building a full map of the environment.

Portable Battery-Powered PoE Power and Network Diagnostic Hub

When working with Ethernet-based embedded devices, including sensors, IP cameras, controllers, and other network equipment, one of the first objectives would be to simply determine whether or not the device is working correctly. Testing a device may require supplying it with PoE connection, in order to check voltage and current consumption, as well as to confirm that an Ethernet link has been established and network data is being transferred. Normally, completing these tasks require an array of equipment, such as a PoE switch or injector, an AC power source, a multimeter, etc.

However, this becomes difficult when a device needs to be tested somewhere that does not have an easily accessible AC wall outlet or an inverter/extension cord that can provide power. This especially applies to remote, unestablished, or “off-the-grid” locations where the goal is simply to bring up a device, confirm that it operates correctly, or diagnose a problem.

An effective way of addressing this issue is to take a more portable means of power. Particularly, via a portable, battery-powered PoE power and network diagnostic hub. Instead of requiring several separate tools, this power supply device would power an Ethernet device and immediately evaluate its operation from a single portable unit.

Details

The device would contain an internal rechargeable battery system capable of independently powering both the diagnostic hub and the connected Ethernet equipment. USB-C Power Delivery charging would allow the user to recharge the battery with the universal charging format. An adjustable DC power-conversion stage would generate the required output voltage from the battery, with the primary low-voltage power stage targeting approximately a 5-10A operating range depending on the final design and connected load. This adjustable power would also make the system more flexible than a battery-powered PoE injector with some fixed output.

A controller would manage the power system and provide the diagnostic functions of the device. Voltage and current sensing would allow it to continuously measure the power consumption of connected equipment, while an Ethernet interface would monitor whether a network link has been successfully established and whether the connected device can communicate over the network. For equipment such as IP cameras, the network connection could also allow a laptop to access and configure the connected camera while it is being powered by the hub. This would allow a user to power a camera, confirm that it is drawing an appropriate amount of current, verify its Ethernet connection, and access the device over the network without first requiring its normal PoE infrastructure.

The system would also provide configurable input and output connections. These connections could be used to interface with external sensors, switches, triggers, relays, or other hardware associated with the device being tested. This expands the portable power supply to beyond simply supplying power and allows it to interact with the external interfaces commonly found on embedded network equipment.

For the convenience of the user, a display and controls would provide immediate access to information such as battery level, output voltage, output current, total power consumption, Ethernet link state, and fault conditions. The combination of battery power, adjustable voltage, USBPD charging, Ethernet connectivity, configurable I/O, and electrical monitoring would make the device more than a portable PoE injector.

In terms of the market, there are not many PoE-related products that combine all the functions previously mentioned. While there are advanced portable devices, such as the TREND Networks PoE Pro, in the market that are high functioning, they are fundamentally designed to test an existing PoE source (with a battery that powers the tester itself) as opposed to acting as a rechargeable source for the device under the test.

This portable-battery powered PoE device would serve as a tool for powering, communicating with, controlling, and diagnosing Ethernet-based embedded devices wherever power and networking equipment are inconvenient or unavailable.

Schedule

September 10 - September 23, 2026	This stage is primarily meant for planning and research. Following group formation, the expected functionality from the power supply device would be determined, as well as any base level and stretch goals. Then, research would have to be done for the battery system, USBPD charging, adjustable DC-DC conversion, PoE power delivery, Ethernet communication, voltage/current sensing, and configurable I/O.
September 24 - October 11, 2026	This stage is primarily for initial designing and part selection. The processor would be selected, as well as the other major components, including for the battery, charging, power-conversion, sensing, Ethernet, and display. The respective subsystems would also have to be prototyped accordingly.

October 12, 2026 - November 2, 2026	This stage would be meant for prototype integration and PCB design. The working prototype would have to be developed. Major functions would have to be demonstrated independently, but the integration for the battery, conversion, output, monitoring, measurement, and display would begin shortly after. The PCB schematic and layout would also have to be developed and verified, which will be followed by the order.
November 3, 2026 - November 15, 2026	This stage is meant for testing the prototype and developing the firmware. While waiting for the PCB to arrive, the user interface should be developed, along with voltage/current monitoring, battery monitoring, and configurable input/output behavior. The power stage would have to be tested with loads and then the actual PoE devices would follow. Designing and fabrication of the enclosure would also begin.
November 16, 2026 - November 29, 2026	This stage is intended to be for PCB-based development and testing. The ordered PCB would have to be verified before any assembly. The previously developed firmware would be transferred to the PCB and the power supply device would be tested with actual PoE equipment.
November 30, 2026 - December 10, 2026	This stage is for final testing, having any additional features, and presenting the product. The system should be completed by around the first 3 days of December, with the remainder of time being used for testing and presentation preparation. The portable enclosure would have to be completed and there would be testing done under various battery levels, PoE loads, network conditions, etc. The voltage/current accuracy and battery runtime would also have to be measured. For the Expo, the poster would have to be completed and a demonstration would have to be prepared.

Budget

Component	Quantity	Estimated Unit Cost	Estimated Total Cost
Microcontroller	1	$25 - $40	$25 - $40
Battery	1	$25 - $40	$25 - $40
USBPD Charging	1	$10 - $20	$10 - $20
DC-DC Converter	1	$20 - $35	$20 - $35
Ethernet/Connectors/ PoE Circuitry	1 set	$20 - $30	$20 - $30
Voltage/Current Sensing	1 set	$10 - $20	$10 - $20
OLED, Buttons, Connectors	1 set	$10 - $20	$10 - $20
PCB	2 sets	$10 - $20	$20 - $40
Enclosure/Packaging	1	$15 - $30	$15 - $30

The total cost comes out to be around $155 - $275. The upper bound exceeds the limit per person, but considering that the price is for the worst-case situation, it is fair to assume that the budget will still be in the appropriate range.

Issues

1) High-power DC-DC conversion
2) Battery charging and safety
3) PoE delivery and protection
4) Ethernet signal integrity
5) Heat management

The first issue would be the largest challenge due to the required voltage conversion and high current, but existing converter designs make it seem relatively reasonable. In terms of the second issue, it should also be manageable using dedicated battery-management circuitry. More research from the internet can be referenced for this. The third issue would require proper current limiting and protection for connected devices. The fourth issue would depend on the PCB layout to isolate Ethernet signals from any switching noise. The fifth issue would be addressed through simply having a good choice of components for temperature monitoring and enclosure design.

Additional Stretch Goal

If time allows, multiple independently monitored PoE outputs could be added, allowing for the power supply device to power and measure several Ethernet devices simultaneously.

Custom Intelligent SIP Intercom

Intercoms are communication devices that are used in a variety of systems and networks where communication is needed between any set starting location and another endpoint. Session Initiation Protocol, or SIP, intercoms are different as they allow for the devices to be internet network-based and act as IP phones rather than just separate units controlled by analog wiring. These devices are on the market, distributed by companies such as TEC Solutions, but they are generally fixed-purpose devices with limited hardware and software customization.

This custom SIP intercom would be an Ethernet-connected intercom designed to provide the core functionality of a commercial SIP intercom while allowing additional features to be added depending on the application. The finished device would contain a microphone, speaker, physical call button, Ethernet connection, status indicators, and programmable external I/O.

Details

When the call button is pressed, a controller would initiate a SIP call to a predetermined destination, such as another custom intercom device, computer, agent booth, or control center. Audio would be transmitted over the Ethernet network using some transportation protocol, like RTP, which delivers audio and video data over network transport layers and allows real-time and two way voice communication. A microphone, audio codec, amplifier, and speaker would form the physical audio interface.

The custom design would also allow external devices to interact with the intercom. Inputs would connect to buttons, alarms, or sensors, while outputs would control things like relays or indicator lights. This would make the intercom more configurable than a basic intercom and would allow its functionality to be adapted depending on the situation.

Schedule

September 10 - September 23, 2026	This stage is primarily meant for planning and research. Make sure that functionality is determined and that processor, Ethernet interface, audio hardware, and SIP/RTP implementation options are researched.
September 24 - October 11, 2026	This stage is primarily for initial designing and part selection. Major components should be selected and all of the basic Ethernet and audio functionality should be established.
October 12, 2026 - November 2, 2026	This stage would be meant for prototype integration and PCB design. Basic SIP/RTP

	communication and audio should be implemented. The custom should also be designed and ordered by the deadline.
November 3, 2026 - November 15, 2026	This stage is meant for testing the prototype and developing the firmware. All call, audio, I/O, and fault handling development should be continued while waiting for the PCB.
November 16, 2026 - November 29, 2026	This stage is intended to be for PCB-based development and testing. The PCB should be brought up and Ethernet, audio, controls, and external I/O should be integrated.
November 30, 2026 - December 10, 2026	This stage is for final testing and presenting. Have the enclosure and packaging completed and have repeated tests for calls and audio quality. Any remaining issues should be fixed, and the demonstration for the Design Expo should be prepared and ready.

Budget

Component	**Quantity**	**Estimated Unit Cost**	**Estimated Total Cost**
Microcontroller/Processor	2	$15 - $30	$30 - $60
Ethernet/Connectors	2 sets	$10 - $20	$20 - $40
Audio Codec	2	$5 - $15	$10 - $30
Microphone/Speaker/Amplifier	2 sets	$10 - $20	$20 - $40
Buttons/LEDs/External I/O	2 sets	$5 - $10	$10 - $20
PCB	1 set	$15 - $30	$15 - $30
Enclosure/Packaging	2	$10 - $20	$20 - $40

The total cost comes out to be around $125 - $260. The upper bound exceeds the limit per person, but considering that the price is for the worst-case situation, it is fair to assume that the budget will still be in the appropriate range.

Issues

1) Implementing SIP and RTP communication
2) Maintaining continuous real-time audio
3) Preventing speaker-to-microphone echo and feedback
4) Integrating Ethernet and sensitive audio circuitry on the PCB

The first and second issues would likely require the most software development, but existing SIP/RTP standards and processors make them slightly more reasonable. The third issue would initially be addressed through speaker/microphone placement and volume control. The fourth issue would require PCB layout and grounding. Overall, each issue appears manageable within the project scope, especially with more members.

Additional Stretch Goal

If time allows, a web-based configuration interface allowing for parameters such as network settings and audio levels to be changed remotely could be added.

Project Pre-Proposal Homework — HW #0

Robert Conforti

Proposal 1: Intelligent Metronome

High-Level Description

One of the hardest parts about learning any instrument is practicing. Metronome practice is one of the best ways to improve technique and memory of the pieces you play, but it is often utilized the least. Listening back to yourself playing is extremely useful but also very tedious, and logging practice hours can help students and teachers get a better sense of progress. To help musicians stay more on top of their practicing, I propose a new device which solves all three of these issues, and more.

At its core, the device is just a simple metronome, about the size of a deck of cards, with a microphone, speaker, and OLED display. The OLED display will be used as a visual metronome (much like many apps), and also used to display some device settings. The device can also be used to track how long you play, simply by turning it on. The metronome does not need to be on for this, and once you have finished your practice session, simply turning it off logs the time and sends it to an app on your phone. This can help musicians and teachers better track progress by seeing how long students are practicing. While the device is being used, the installed microphone also listens to what is being played and can give feedback to the player, anything from pointing out rushing or dragging, to basic mistakes made in the music. This works by having the user upload the music that they are practicing onto the app, which is then used to monitor the player's accuracy. Since the microphone is always recording during use, the player can also use it to record and listen to themselves play. This makes it much easier than pulling out your phone and playing parts again to spot mistakes. This passively does all this for you.

The device basically becomes your own personal teacher; it can help you track progress across multiple sessions, while also improving practicing efficiency. With access to the sheet music, the device can also help flag parts that the player struggles with, telling them they miss certain notes, or are not playing on tempo. The user can also configure tempo for the song on the display, with additional practice features like speeding the tempo up gradually to performance speed if the player continually gets through practice without mistakes.

The app itself will be very minimal, basically just a UI to view and control most of the features on the device. There will be an option to upload music to the device, and also a timeline to show practice hours. The data can be used to show trends in practice consistency, which can be sent to teachers automatically.

The device is marketed towards students and teachers: teachers who would like to know better what their student struggles with outside of lessons so they can give more accurate and appropriate advice, and students who would like to improve their practicing efficiency. Lessons can be very short, and there is sometimes not enough time for teachers to adequately assess the needs of their student, so this device helps solve that issue by providing much more data on practicing trends.

Schedule

1. 9/10 – 9/23, Planning: Research parts including the microphone, speakers, OLED display, Bluetooth module, and microcontroller. Make sure that the microphone supports the full frequency range needed with precise measurements so that notes can be distinguished. The microcontroller must be fast enough to support all of the above features, including listening to the player and referencing sheet music for accuracy. Finish proposal.

2. 9/24 – 10/6: Choose parts from the list above and order them. Begin working on the app, basically just the UI and music upload feature. If devices arrive quickly, prototype on breadboards, and once peripherals arrive, start testing them. Get all peripherals working on breadboards; most importantly, get the microphone to detect and differentiate between notes played (for now, from a piano). Start designing PCBs. The speaker should be able to play rhythmic sounds mimicking a metronome.

3. 10/7 – 10/25: Continue designing PCBs. Start programming the microcontroller. Be able to send data from the device to the app (something simple like duration it was on). Once everything is working, order PCBs. Work on software that will detect the specific notes played. Finish the sheet-music-reading software. Metronome fully

working, with the ability to set tempo manually. OLED should be able to display a rough UI of what the metronome will look like.

4. 10/26 – 11/10: Get a mostly working prototype with breadboards and the app. The prototype will be able to detect which notes are being played (displayed on the OLED for now). Metronome fully functional. OLED fully functional. Work on getting the metronome and OLED synced. The device should recognize when a note is played on tempo and when it is not, and should also recognize when the correct notes are played (referencing the sheet music).

5. 11/11 – 11/25: Move everything from breadboards to the PCB and complete full integration. Aim to have everything working by the end of 11/25. All additional time is for adding more features or allowing room for unforeseen issues. Additional features include selecting practice measures and incrementing tempo upon successful execution of a section (an on/off feature). Get the chassis modeled and printed.

6. 11/26 – 12/9: Final testing. Complete poster and final report. Fix any remaining minor bugs.

Budget

The budget below is based on sourced part pricing for a team of four; 3D-printing costs for the enclosure are not included.

What	Qty	Cost per	Cost total
ESP32-S3 dev board (MCU, BLE, FPU)	4	$10.00	$40.00
MEMS I2S mic breakout (ICS-43434 + SPH0645, for A/B testing + final builds)	5	$8.00	$40.00
OLED display, 128x64 I2C	3	$6.00	$18.00
Small speaker (8Ω) + PAM8302 class-D amp	3	$5.00	$15.00
Rotary encoder + tactile buttons (tempo/mode controls)	3	$3.00	$9.00
LiPo battery (500 mAh) + TP4056 charge/protection module	2	$10.00	$20.00
PCB fabrication, 2-layer (2 small-batch orders: prototype + revision), shipped	2	$25.00	$50.00
Misc. parts (headers, connectors, passives, wiring, breadboards)	1	$80.00	$80.00
Shipping (multiple vendor orders)	1	$75.00	$75.00
Contingency / spares	1	$100.00	$100.00
		Total	**$447.00**

Implementation Issues

At a high level, the hardest parts will be choosing the microphone and designing the software algorithms for music reading and interpretation.

One of the biggest issues for this project will probably be the microphone and the real-time note reading and checking. The microphone will limit how easily it is to distinguish specific notes, and if multiple notes are played at once, each note has to be accounted for. The range of notes on an 88-key piano is from about 27.5–4,186 Hz, which means our microphone has to be able to detect and record at least these frequencies. It also has to have very little noise so that it doesn't pick up notes that were never played. It also needs to be able to have a small enough step size so that the difference between a natural and a sharp or flat note can be measured, which is not a flat number, but a

constant ratio. This will probably have to be tuned manually for each sharp and flat, but so will every note anyway. The smallest difference is around 0.97 Hz, so the microphone has to be that precise to measure that small of a difference so it can be recorded and interacted with by software. Technology like this already exists, I have a guitar tuner app on my phone that uses my iPhone's microphone, and it is able to differentiate notes just fine.

Another issue is that this device could also be set up to be used by different instruments, so depending on if there is time, this might or might not be possible to add. For now we will plan on just doing piano, but if we wanted, say, guitar, we would need different pitch classifications. This can be a reach goal done at the end.

Another issue might be classifying different musical markings, for example, rubato. This is when the player intentionally plays out of tempo for a moment, which could get flagged as a mistake. This would make it much harder to implement the mistake check as simply playing too far from the tempo, since it might be purposeful. This might be a setting that we could tweak, but having the program interpret sheet-music notation instead of just the notes might be extremely complicated and might not fit in the time frame for this project. The amount of computing power needed might also increase exponentially, so it really depends on how we implement the sheet-music reading. If we chose an approach that uses a small ML model, it might be too difficult to complete. If we instead translate sheet music into an easier format to read, then send the more easily read data to the device for interpretation, this would make our lives a lot easier. The frontend for the app might also be a lot of work; it will be a lot of extra code and it will work separately from the actual device. This means we will have to understand how Bluetooth works on iPhone, Android, and other OSes.

Eval and Testing

In order to determine if the device is working correctly, we will have someone perform the entire suite of features.

The full flow is:

1. Download the app and pair with the device.
2. Upload sheet music to the device and have it stored on the app.
3. Turn on the device and turn it off, and check the app to see if the time on is sent.
4. Use the metronome on the device normally (set tempo, start and stop, and so on).
5. Select the uploaded song and start playing, making a mistake on purpose. The device should record this.
6. Use the record-myself feature to listen back to this mistake.
7. Check the app and see if the mistake is logged there.

Beyond running through this flow once, success will be judged against a few concrete measures: the metronome should hold tempo accurately to within a couple of BPM, and the note-detection algorithm should correctly identify a clear majority of notes played on a piano in a quiet room (we'll set a target percentage once we know what the microphone can realistically resolve). We'll also run a short multi-day test using the device for several separate practice sessions in a row to confirm that practice-hour logging and trend data accumulate correctly in the app over time, rather than only working in a single isolated session. Finally, we'll deliberately test edge cases like stopping mid-song or losing Bluetooth connection partway through, to make sure a session's data isn't silently lost.

Proposal 2: Post-Refurb GPU Triage Station

High-Level Description

Refurbished GPUs are everywhere eBay, Facebook Marketplace, Amazon, etc. The problem is that there isn't really a cleaner way to test these devices. Most people will do it by plugging it into a gaming PC for ten minutes and seeing that nothing catches fire. Real GPU failures VRAM errors, thermal throttling from a bad repaste, unstable core clocks, dead display outputs, a failing VRM often only show up under sustained load or specific memory patterns, and most refurbishers don't have the time or software know-how to run a proper stress-test suite on every card that comes through the door.

This device turns GPU triage into something closer to a drive-through than a diagnostic ordeal. It's a standalone board with a PCIe x16 slot and supplemental power, built to receive a "patient" GPU, power it up in isolation, and run it through a battery of automated tests like, memory pattern checks, core/memory clock stability sweeps, a thermal ramp test, fan-control verification, and a basic display-output check (does it produce a signal at all, on every port). No host PC, no OS, no drivers to install: plug the card in, hit a button, and a small LCD (or row of LEDs) reports which subsystems pass and which fail.

Schedule

1. 9/10 – 9/23: Figure out how we'd even talk to a bare GPU with no OS underneath it, and scope out whether a cheap, low-end used card is realistic to source. Figure out how strong of a CPU we need to even test the higher end GPUs. Finish proposal.
2. 9/24 – 10/6: Pick a host microcontroller with enough smarts to sequence PCIe power-on. Start looking into what "known-bad" test units we could actually get our hands on.
3. 10/7 – 10/25: Design and order the power-delivery + PCIe slot board (PCB fab delay window follows).
4. 10/26 – 11/10: While waiting on PCBs, prototype the LED/LCD status panel and basic test logic on the bench with whatever hardware we can scrounge.
5. 11/11 – 11/25: Bring up the real board and get at least one test working end to end.
6. 11/26 – 12/9: Add more tests if budget allows, debug, poster, final report.

Budget / Implementation Issues

This project is basically impossible in the given budget. From what I know about GPUs most of them require very strong CPUs in order to bring out their full performance, as such it this forces us to put a lot of the budget into one of these CPUs. Even without considering CPUs, the GPU market is currently very expensive. Buying a midrange GPU would cost around $500 and maybe a used one $350, which is a lot of money on basically a test bench. For this project to work we would probably have to build a full computer, CPU, RAM and everything, or design an OS that would be able to trick the GPU that those things are there, in order to get it working how we want. We cannot interact with the GPU with only a microcontroller. They need PCIe connection and drivers which turns this into a completely different project. There is also the problem of designing a PCB that supports a PCIe connection. That is an extremely complex problem in itself and would eat up a lot of time thinking about and designing. In that case we would basically be forced to buy a small motherboard that supports PCIe connections. I could keep going on about the issues that this would have but most of them stem from time and money.

The biggest practical problem is simply sourcing the test subjects. We can't manufacture a broken GPUs on demand, at least not without our budget and even then, that seems unethical. The best we could do is scavenge dead cards, which are unpredictable, hard to find on a student timeline, and each only demonstrates one specific failure mode. Worse, testing a genuinely unstable card carries real risk of finishing it off entirely, which means our small pool of test units could keep shrinking over the semester, quietly eating further into a budget that's already tight.

Eval and Testing

Success would mean: given a known-good GPU, the device reports every test passing; given a known-bad GPU, it correctly flags the right subsystem as failing.

Logic Foundry: A Physical Digital Logic Construction Game

Background and High-level Description

Digital logic is usually taught using circuit simulators or breadboards with real components. Simulators provide convenient feedback, but the circuit exists only on a screen and is usually abstract. Breadboards use real components, but beginners often spend more time checking loose wires, fixing potential wrong wiring, and rebuilding duplicated logic circuits than thinking about the logic itself. My proposed game, Logic Foundry, solves this problem by combining physical circuit construction with automatic testing and reusable modules.

In general, players build circuits from enclosed basic logic gate modules such as AND, OR and NOT using jumper wires. These modules use real 74HC-series logic chips and include labeled ports and LEDs. A central terminal supplies inputs, measures outputs, and checks the circuit against the truth table for the selected challenge. For example, after a player builds XOR from AND, OR, and NOT gates, the terminal tests all four input combinations. If a case fails, it reports the input, expected output, and measured output without revealing the correct circuit.

The main feature of out gam is Capture. There are some programmable modules that have some I/O ports and can be programmed to output different outputs given different inputs. After a circuit passes its tests, the player can place a blank programmable module on a pogo-pin dock. The terminal sends every possible input combination with corresponding outputs to the module. The original circuit can then be replaced by this new reusable block. A player could capture an XOR circuit, use it to build a half adder, and later combine captured blocks into larger circuits. This turns abstraction from a diagram into a physical process.

The project is intended mainly for students/educational purposes. It preserves the experience of wiring real components while providing immediate feedback, the same as a simulator. It also allows students to reuse circuits they have already built, so larger projects can focus on higher-level logic instead of repeatedly reconstructing the same subcircuits.

Implementation Details

The electronics are divided into four parts: primitive logic modules, splitter modules, reusable programmable modules, and the central terminal. Primitive modules use real logic ICs with visible output LEDs, while splitter modules distribute one signal to several inputs. The terminal outputs test signals, measures the circuit outputs, checks them against stored answers, and programs the programming module.

Each reusable module contains a small microcontroller with 8 inputs and 8 outputs. During Capture, the terminal records the completed circuit's truth table and writes it to the module. The module then uses its inputs as a lookup-table index and produces the stored outputs. Its firmware remains unchanged, so Capture only transfers configuration data.

The first version will support combinational logic and include enough modules to build circuits ranging from basic gates to a four-bit adder. Each programmable module will support up to 8 inputs and 8 outputs. Sequential circuits, clocks, registers, memory, and processors will remain outside the initial scope.

Schedule

9/10 - 9/24	***Planning*** Form groups, brainstorm to determine the final framework and implementation details of the project. Research possible parts to use. Estimate costs. Study Turing Complete to get some ideas for designing game levels. Write the draft proposal before 9/18. Have the project proposal meetings on 9/24. Refine the plan if needed.
9/25 – 10/4	***High-level Design & Part Selection*** Once the proposal is confirmed, search for suitable parts, processors and get the BOM table. Think about the power stuff. Think about UI/UX. Make a prototype/minimal demo for feasibility verification. (e.g. basic Terminal + basic logic gates + one programmable module) Confirm game levels so that max pins/modules can be determined. Write the final proposal before 10/1.
10/5 – 10/22	***Hardware Design & Prototyping, Thinking about Software Structure + UI/UX*** Design the PCD for 1. the Terminal, 2. programmable modules, 3. basic logic gate modules, 4. the splitter module. 5. Connectors. Think about the magnetic pogo-pin dock. Do some prototype work to verify designs. Order PCBs after all these have done, ASAP. (Need redundancy for ordering the PCB a second time in case of any design errors. 10/22 is the hard deadline.) Think about the UI. Make the game experience more detailed (SWE level). Submit Milestone 1 before 10/15. Prepare for Midterm (10/21)
10/23 – 11/12	***Hardware Test, Software Development, Testing, Mechanical Structure Design*** Before receiving the PCB, (two or three people) write code for the Terminal & programmable modules. Most code should be done before the PCB arrives. Meanwhile, (one or two people) design the mechanical structure of the game. As soon as the PCB arrives, assemble & test. Hopefully there are no design errors. If errors are found, fix them immediately. Send the second version of PCB before 11/3. Submit the Milestone 2 report before 11/10. Attend Milestone 2 meeting on 11/12. Bring an almost-complete demo with most functions implemented (if no need to order another PCB). Boxes not required.
11/13 – 12/6	***Software + Hardware Development & Testing, Refine UI/UX, Game Design*** System-level hardware-software co-development. Improve system stability. Refine user experience. Fun game level design. Implement everything I haven't thought of now. Have the mechanical shell done. Assemble. Make the system "pretty". Prepare for the Design Expo
12/7 – 12-10	***Final Wrap-up*** Prepare for the Design Expo @ 12/10. Write the final report @ 12/11. Finish HW2 @ 12/13. Start preparing to Final @ 12/16

Budget

Item	Unit Cost ($)	Quantity	Total Cost ($)
Strawberry PI 5	50	1	20
ATmega328P	2	10	20
PCB & Components: Terminal	40	1	40
PCB & Components: Prog. Modules	10	10	100
PCB & Components: Logic Gate Modules	10	10	100
Jumpers, Connectors, Splitter Modules	2	10	20
Misc Parts	50	1	50
Mechanical Parts	30	1	30
Shipping	50	1	50
Total			430

Issues

The main concerns of the system are:

1. Design a robust circuit and program for the Terminal and the Programmable modules
2. Programmable module power supply
3. Solid connecting ports
4. Game content design
5. Reliability issues, as users may connect components incorrectly

We only need access to 3D printers to manufacture the mechanical components. Since the power of our system is relatively low (5V), no safety issues are expected. An energy budget problem is also not anticipated.

Evaluation and Testing

1. Test the electrical correctness of all PCB/circuits
2. Test all functions of the Terminal, including game control, level selection, logic correctness check, “Capture” function, etc.
3. Test all functions of the Programmable/Logic Gate Modules
4. Test UI fluency and overall game experience
5. Test electrical robustness: resistance to wrong connections, solid connectors, etc.

A Mechanical Keyboard with a Context-Aware Touch Bar

Background and High-level Description

A keyboard is necessary for computer use, as its keys are physical and tactile, allowing users to type without looking at it. However, it is much less effective for some controls like dragging or scrolling. A touch bar, first introduced by Apple Inc. on the MacBook Pro, offers users the freedom to control software in different ways. However, Apple replaced the function keys on the keyboard with the touch bar, which creates conflicts when the function keys are needed and greatly degrades the experience of using a normal keyboard. For example, programmers cannot use the F5 key to start a debugging process when the touch bar is displaying something else. Meanwhile, Apple limited how programs could utilize and communicate with the touch bar, sometimes making it useless by only displaying the function keys.

As a result, I would like to combine the touch bar with a normal 84-key keyboard. The touch bar lies above the function-key row. It would be context-aware and would provide different controls under different circumstances. On the Windows desktop, it could show a volume slider, microphone state, and media controls. When Visual Studio Code becomes active, it could change to Run and Test controls, coding agent status, and build or test progress. In CAD or video-editing software, it could provide smooth control of zoom, brush size, opacity, or the timeline. A linear resonant actuator would provide a short haptic response when a button is activated or a meaningful slider position is crossed.

Companion software running on the PC provides the context-aware capability of the keyboard. In design mode, users design the components they want on the PC using this software and upload their design to the keyboard. In monitor mode, the software collects operating-system-level or application-level information that the keyboard cannot obtain on its own and transfers this information to the keyboard. The keyboard parses the information and visualizes it according to the user's design. The keyboard also sends the user's interactions with the touch bar to the PC, and the companion software performs the corresponding actions.

Basic functions of a product should always be available even when the fancy part of that product fails. Therefore, the keyboard will still function as a normal wired keyboard if the touch bar's processor, display, firmware, or PC companion software fails. This requires the keyboard and the touch bar to have their own MCUs. A USB hub is then needed to connect both subsystems through a single USB cable.

This idea is useful for programmers, designers, and other users who have special controls that depend on the software they are using. Programmable keyboards can send shortcuts, but they cannot display information or provide natural continuous control. The novelty is the combination of a conventional full physical layout with a narrow, locally rendered, context-aware input/output surface. Physical keys handle typing, while the touch strip handles tasks for which fixed keycaps are a poor interface.

Implementation Details & Issues

Major implementation issues are:

1. **Touch bar screen selection.** It is difficult to find a bare display that is as narrow and long as the Touch Bar on a Mac and therefore conforms to the project's concept. Most currently available screens are either too wide or too short. It is also not practical to assemble the touch bar from several smaller screens because the unavoidable black seams would break the continuous display surface. A suitable panel may need to be sourced directly from a display supplier.
2. **PC companion software development.** We need a configuration format to describe the touch bar's control components, as well as a communication protocol between the software and the touch bar. The software should also provide a UI for users to design control components and their corresponding actions. It must then receive touch input and perform the configured actions on the PC.
3. **Haptic feedback.** We need to find a suitable haptic actuator and driver that provide clear feedback when the user presses a touch-bar button or crosses a meaningful slider position. Selecting and mounting an actuator that produces noticeable feedback without vibrating the entire keyboard may be difficult.

Other potential issues include power delivery, system integration, and enclosure design.

Evaluation

1. The keyboard is recognized as a standard keyboard with all physical keys operating correctly.
2. The keyboard continues to function if the touch bar is down or the PC software is stopped.
3. The touch bar displays different layouts and automatically switches between them when the active PC application changes.
4. The companion software allows the user to create and upload layouts, send correct information to the touch bar, and successfully conduct assigned PC actions when invoked by the touch bar.
5. The haptic actuator responds when the user presses the screen.
6. The system works without conflict.

Main Project Idea: StrideX (Continuation Project)

Name: Ryan Beaudoin (rbeaudoi@umich.edu)

Problem Overview

Hundreds of thousands of people undergo reconstructive or replacement knee surgeries every year due to injury and degenerative aging [links]. While a large percentage of younger patients are able to recover on a standard timeline with few issues, many young athletes face lengthy and frustrating setbacks. Additionally, those who are older and require knee replacements for degenerative conditions struggle to regain proper gait and knee function at a high rate [link].

While a variety of IoT wearable devices have been developed for a flurry of different applications ranging from convenience to health tracking, knee recovery remains largely dependent on live expert physical therapist observation or incredibly expensive research equipment for recovery status metrics. Rather than claiming to replace the need for these avenues, I propose a device that aids in these efforts. The average person simply cannot get to physical therapy more than a couple of times a week and often even this is impossible due to time and cost restraints. Thus, a wearable device that ingests data from the movement of the knees could allow people to get constant feedback whether live and haptic or by seeing progressions of aggregated data analysis for different movements.

Details

My device combines the use of microcontrollers and a suite of simple sensors along with bluetooth connectivity for pairing with a phone. Putting all of this together, patients can get a good enough reading of their knee function in a way that results in haptic buzzing with irregular gaits or phone symmetry and loading analysis of plyometric movements such as jumping and cutting, which are crucial to improving one's recovery trajectory.

There are multiple ways to approach getting the best possible data for loading and knee flexion including the use of IMU's, accelerometers, protractor sensors, strain gauges, etc. and possibly a combination of several to achieve accurate data ingestion of the knee's state in different exercises and movements. On the feedback and processing side, on board compute, fpga boards, or even offloaded DSP analysis through the bluetooth connected mobile device can do the necessary real-time and post-processing of this data for meaningful results. Finally, haptic motors or something like a watch notification can be triggered to subconsciously train the patient to correct their movements in the absence of expert assistance like one could receive in a therapy or research setting.

User

This knee device aims to tackle several distinct issues in knee recovery: gait retraining, knee loading forces, and plyometric symmetry. An important note is that I have already built an

ESP-32/IMU/haptic sensor prototype of this device as a passion project due to my own knee recovery struggles. I went overboard on a simple “use a sensor” class project as a proof of concept for its viability. After showing the device I made to the Orthopedic Rehabilitation & Biomechanics Laboratory, with whom I had been a test subject, I received interest from them in possibly exploring this device more concretely for this fall. I have reached out to them for confirmation if the interest in collaborating on this project is still there and if so, consultative feedback from knee research specialists and the necessary equipment to compare a polished version of the device’s readings to would possibly be available. Their initial expression of interest serves to me as evidence that this device could serve a real purpose

Schedule

9/10-9/23	Planning 1) Consult researchers on the core user base to focus on to decide what features to prioritize 2) Analyze and start to prototype the pros and cons of each sensing method for flexion, force, etc.
9/24-10/6	Design and part selection 1) Decide a 1.0 sensor platform to go in on for the design a) Flexion sensing method or combined b) Haptic feedback method c) Battery setup d) Force and load sensing method 2) Physical Sleeve Design 1.0 a) Solve issue of slippage without restricting movement b) Version that works with bulky breadboarded prototype and look forward to sleeker design for final product
10/7-10/25	Breadboard prototype and PCB design 1) Confirm selected 1.0 sensor platform works together to achieve a usable and reliable data output. 2) Work out embedded software and firmware issues to get sensor stack to integrate with initial data analysis iteration 3) Have the bluetooth link working and sensors properly connected and have the outputs of the data processed and in a state where the output is noisy but decipherable gait data by 10/15 and clean for a walking pace by 10/25 on breadboard 4) Decide Microprocessor and if no native BLE support add module to design 5) Design dual PCB’s for order based on confirmed sensor verification

10/26-11/10	Software development and testing on prototype 1) Get data processing and filtering good enough to accurately detect plyometrics and loading. Loading will be experimental if it is useful information based on the sensors available. 2) Experiment with FPGA vs on-board vs offloaded to iOS DSP data analysis 3) Design sleeker knee sleeve for use with condensed PCB sensor setup
11/11-11/25	Software development and testing on PCB-based system 1) Decide on the final algorithms for data ingestion and processing and map to the now PCB version 2) Setup inside finalized knee sleeve and ensure security, rigidity, and stability 3) Decide on the final data processing algorithms for post-processing of knee data that create the most accurate results
11/26-12/9	Final testing, additional features, poster and expo 1) Additional features may include things like pure software to make app communicate with PT with automatic alerts based on patient progression and other interactive features to make the device more useful for PT and patient users 2) Finalize all analysis with researchers and get final feedback and tweaks
12/10-12/11	Finish final report

Budget

What	Quantity	Cost	Total
(Chip) Xtensa dual-core 32-bit LX7	2	$20	$40
PCB for each knee module	2	$15	$30
MPU-6050s, IMUs (4 packs)	2	$15	$30
Haptic Feedback motors	2	$4	$8
Strain gauges (load measuring)	10	$2	$20
Knee sleeve material/manufacture	2	$50	$100
1000mAh Battery 2-pack	1	$15	$15
Scratch supplies breakouts, velcro, etc	1	$150	$150
FPGA?			
Multiple sets?			
Total			$393

Implementation Issues

1) Keeping the knee-sleeve from slipping and the sensors in the correct orientation. My prototype was okay with this and used velcro to tighten the sleeve to form fit any knee and place the sensors in their proper positions. However, there is a significant difference between the flexed and unflexed, bent and unbent positions of peoples' knees with these gaps depending especially on the individual's level of muscle definition. Thus, a complete overhaul of the sleeve enclosure is necessary to keep the sensors properly oriented in all situations and the sleeve from slipping by leveraging some more stretchy yet still tightenable vessel to ensure the individual can receive accurate readings after calibration.
2) DSP filtering challenge due to reliance on accelerometer based sensing. This can be summed up by considering how in a walking pattern a leg is off the ground, only gravity, and then strikes the ground at the same flexion as right before striking but now feels a normal force that creates noise in the acceleration reading. This is exacerbated in running and jumping movements even more. As such, processing these signals will be a major hurdle to the efficacy of this device. The bright side of this is that since it is a bilateral device, many of these filtering issues are made less consequential since they would happen at the same time in gait patterns. This can also be resolved if another sensing method is determined to be sufficient such as protracting sensors with the caveat that the device must remain non-invasive so that the user does not have their motion significantly restricted to the point of introducing non-injury related movement issues.
3) Verification of accuracy of the readings. If I receive the green light from the ORB lab as mentioned above we would likely be able to collaborate with them on ways to verify the accuracy of the readings by running trials and cross analyzing with the advanced equipment.

Evaluation and Testing for Completion

To expand on point 3 in implementation issues, collaboration with the lab would allow us to not only verify the accuracy of our data but also to stay in sync with the experts on exactly what use cases this device could be helpful for. For instance, they might see that the device gets pretty good at analyzing plyometrics to the point it could be used as an accelerated performance device for athletes versus leaning more into more static-based gait issues that would be relevant to those recovering from knee replacement or strength and subconscious related gait abnormalities.

Secondary Project Idea: SmartPack

Name: Ryan Beaudoin (rbeaudoi@umich.edu)

Problem Overview

People have regular schedules and forget stuff all the time. While there are items you never want to forget like your wallet, keys, and laptop, there are other times when you need to remember your EECS 216 lab kit, your workout clothes, or a secondary work laptop. Airtags do a great job with those everyday items if you're willing to shell out $30 a piece and be bombarded with notifications every time they leave your presence, but there exists a desire to have a more tailored organizational system.

Details

The idea here is a smart backpack base station that has an onboard control module with bluetooth capability to interact with a cluster of bluetooth tags sensed in proximity. The backpack connects to your phone on which you make choices about what devices you need when and where. For instance you can make sure your phone, wallet, water bottle, and keys are always on your person at all times, while your EECS 216 tag will only trigger an alert if you forget it for your 9 AM lab when leaving the house. In addition to these dedicated bluetooth tags that can be placed on devices or items you will need regularly, you can designate buttons for things like workout clothes to make sure you pack your bag and click the button confirming you have what you need for the day.

Depending on your designated schedule you will receive a notification on your phone and a haptic feedback buzz in the backpack straps if you forgot one of the items you want to have on your person based on the day's schedule. A usb-c charging port could be added as well for the user to always have a full charge on their devices.

User

The biggest user would be college students but anyone with busy schedules and uses a backpack will be able to benefit from this organizational device.

Schedule

9/24-10/6	Design and part selection: have BLE sensors, GPS, and board models chosen and ordered
10/7-10/25	Breadboard prototype and PCB design: Get beacons working with control module to be detected in presence or lack of and haptic buzzer buzzes when leaving designated home gps proximity. Order PCB

10/26-11/26	Software development and testing on prototype: Migrate to PCB design platform and into actual backpack environment. Get the embedded side working again. Expand feature set to communicate with iOS over bluetooth for user's phone to do logic to send commands that allow for scheduling user movements and patterns requiring different items to have different notifications/buzzing feedback.
11/26-12/9	Final testing, additional features, poster and expo: Clean up user interface and app experience. More features for designated zones for certain devices and scheduling etc.
12/10-12/11	Finish final report

Budget

What	Quantity	Cost	Total
(Chip) Xtensa dual-core 32-bit LX7	1	$20	$20
EBYTE EWD104-BT58P BLE 5.3 iBeacon	10	$6	$60
Haptic Feedback motors	2	$4	$8
Button Module	1	$10	$10
Backpack	1	$50	$50
1000mAh Battery 2-pack	1	$15	$15
Puck Batteries for BLE beacons	10	$2	$20
Or use beefier 5V battery pack enable usbc charge port for phone	1	$30	$30
usbc charge port	1	$5	$5
PCB for backpack control board	1	$15	$15
Miscellaneous materials to make backpack look polished	1	$100	$100
GPS Sensor	1	$20	$20
Total			$353

Implementation Issues

1) Making a user-friendly interface that allows for easy pairing of bluetooth beacons or have them all presetup and they just match them to desired device in the app (making the user do the setup itself would make it not very user friendly)
2) Making the backpack not look like a complete mess
3) Reliability of beacons pairing always

Testing

Very easy to do user trials

Apple Orchard Frost Risk Sensor Network

High-level description

Apples go through various growth stages, from budding to flowering to fruiting and maturation. Apples in each of these stages have different levels of resistance to the cold. In order to protect their crops, farmers could utilize a distributed network of temperature and humidity sensors throughout their apple orchards to receive warnings for particular areas that are at risk of frost-related damage. They could then make informed decisions for modifications to their orchard to prevent any further kills of their apples.

The system would contain several field nodes and a main gateway. Each field node would be deployed in the orchard and send packets of information to the gateway using wireless radio. The farmer can plug his/her laptop into the gateway and see a dashboard containing the sensor information and any warnings for the nodes. This dashboard will also show the functional status of each node, such as online, low battery, calibration error or offline.

Each field node would contain:

- MCU
- Temperature sensor(s) (multiple for redundancy)
- Humidity sensor
- LoRa transceiver
- Battery and battery measurement/management
- LED for status

The gateway would have:

- MCU
- LoRa transceiver

Each field node is not responsible for interpreting the data it senses, rather it will simply build a packet from its sensors and send it to the gateway for interpretation. The frequency of packets can be configured by a reverse communication from the gateway. Once the gateway is active, the farmer can set individual thresholds for which each area will present a warning and a critical status. The dashboard will have a simple interface for adding and removing nodes in the system.

As described above, this network of sensors is worth building (and using) in order to minimize frost-damage to apples throughout various growth stages, which will in turn produce a greater turnout at the end of the year.

Schedule

September	Brainstorm project architecture and research best-fits for MCUs, radios/wireless comms, batteries and sensors. Determine expected battery life for field nodes and tailor part selection accordingly. Come up with a solid standard for how the devices will interface with each other over the network.
October	Develop firmware for the field nodes and gateway using breadboards and any parts already owned. Parts should be ordered as soon as possible. PCBs should be designed and ordered. Quantity of field nodes should be determined by budget. Create an initial dashboard program which displays gateway info.
November	Build out PCBs and necessary enclosures. Rigorously test and debug node behavior in single-node and multi-node configurations. Polish the dashboard program and continue adding software features.
December	Stress test project in various environments and conditions to catch any hidden bugs. Finalize the project and create documentation/poster.

Budget

Field Node (1)	
RFM95W LoRa Radio Transceiver	19.95
STM32	6
PCB	10
Battery	5
Enclosure	5
Antenna	1
Temp & humidity sensors	5
LED	1
Misc	5
Total	57.95

Gateway	
RFM95W LoRa Radio Transceiver	19.95
STM32	6
PCB	10
Antenna	1
Enclosure	5
Total	41.95

Implementation Issues

Current concerns:

- Design and manufacturing of field node enclosures. This could be done using CAD and 3D printing. They should be weatherproof
- How to prevent dropped packets / collisions at the gateway as the number of field nodes increases

Evaluation/Outcomes

A number of objective tests can be used for the system, which will also be used to demonstrate its functionality. The nodes will be put in independent “chambers” that can be manually temperature controlled. A separate known-good thermometer will be placed in the chambers for comparison. The user should set thresholds for each node, then watch the dashboard for expected warnings while increasing/decreasing the chamber temperature. Nodes can also be intentionally disconnected from power in order to demonstrate changes in the functional status reporting. This should show the node as offline on the dashboard. The user should be able to easily commission and decommission new/old field nodes.

A successful project would run seamlessly and present any important changes in the system as they come up.

Stretch Goals

1. Use solar power for field nodes
2. Allow remote/wireless login to the gateway and dashboard

Smart Ski Goggles with Live Telemetry and iPhone Message and Call Notifications

High-level description

Anyone who has gone skiing or snowboarding before knows the feeling of not being able to find your friends or family while hitting the slopes. Maybe you were the first one down the hill and your buddy wiped out behind you, or you took a right while your buddy went left. It's in these situations that tracking down your skiing partner becomes difficult, especially when your iPhone is zipped up in a jacket pocket and you're wearing thick gloves.

This project aims at solving this issue by providing a Heads-Up Display (HUD) on ski goggles which shows:

- Incoming and missed phone calls
- New text messages (sender, message content, time)
- Current time
- Compass
- Speed

The HUD will be created using a tiny OLED display attached to the outside of a pair of ski goggles. A separate device enclosure will be strapped to the user's helmet and will hold an ESP32, battery, accelerometer, magnetometer and buttons. The user can press a button to switch between different screens within the system.

iPhone notifications can be received via Apple Notification Center Service (ANCS). The goggles will pair with the phone using bluetooth, so an MCU like ESP32 with native BLE support is preferred.

A critical implementation issue for this project would be setting up the display so that it is readable and not too close to the user's eye. It is possible to set up focal lenses and mirrors so that the display is not straight in front of the user's head, however this remains a large implementation risk. Additionally, this project would require a battery to be in close proximity to the users head while they have the possibility to take a rough

fall while skiing. This poses a safety risk and requires further evaluation to ensure the system is strong enough to withstand the impact of a ski crash.

The project will be straightforward to test and demonstrate. An evaluator could wear the goggles, walk around and pair their phone. This would allow them to see the live velocity and compass readings, as well as any new iPhone texts/calls.

Careful evaluation of the power requirements should be done so as to select the proper battery size. The battery needs to be light enough to be on the skiers head while also keeping the system functional for long (5+ hour) ski sessions.

Low-Power Miniature Device for Wildfire Monitoring and Tracking in Remote Areas

---

Thousands of wildfires occur every year in the Canadian boreal forests burning millions of hectares and increasing the carbon footprint of the world. In 2023, record shattering wildfires occurred in which over 15 million hectares of land were burned from more than 6000 fires[1]. To give an idea of how large 15 million hectares is, that is larger than the country of England being burnt. From May to September of 2023, in just one summer, this fire let out 647 Teragrams (10^{12}) of carbon emissions[2]. This number is comparable to carbon emissions from countries such as the United States, China, and India. The reasoning as to why this fire was so large is attributed to the fact that this was the highest recorded temperature in over 80 years by about 0.8℃[1].

As temperatures become worse each year due to climate change from nonrenewable resources, wildfires will escalate and cause even greater devastation to the land. It is a cyclical effect in which these forests burn, temperatures increase, and lead to even more fires. From 2020-2060, it is estimated that there will be an increase of 56% fire occurrences and 59% increase in land destroyed[3] in the United States alone.

While finding a solution to climate change is a massive undertaking that will need the entire globe to cooperate, a small monitoring device could lead to immediate action. One of the most effective ways to extinguish wildfires is by finding them as soon as possible and determining their risk-level. There could be hundreds to thousands of devices equipped with satellite and LoRa communication alongside having IR cameras/carbon dioxide sensors. This would be a low power device operating off a battery and solar panels in extremely remote forests.

Details

When speaking about the Canadian boreal forest, it is hard to truly understand the vastness of the land and the absence of people within it. Sometimes, no one is around to see when and where a fire starts, but with a fire finder device, no one has to be. Rather, I propose a small monitoring collective system in which these devices are equipped with a low power microcontroller to stay alive as long as possible as well as having a battery and solar panels attached. These devices will be of a small form factor, (roughly the size of an average human hand) and would be supplied with IR cameras and carbon dioxide detectors. The detectors would awake the device from a sleep state when triggered and this would then make the camera capture an image. The device would then save temperature and time information along with the image.

---

1 https://natural-resources.canada.ca/stories/simply-science/canada-s-record-breaking-wildfires-2023-fiery-wake-call

2 https://www.nature.com/articles/s41586-024-07878-z

3 https://www.mdpi.com/2571-6255/9/5/208

After the device now has this important information of when the fire started and the situation of it, it would begin the process of transmitting this data back to a hub. The device would be equipped with satellite communication in which it could transfer the fire information as well as GPS data. This device would also have LoRa communication. When the fire initially starts, the first device will store this information and try to first communicate with a satellite. If no satellite is present, then it will use its LoRa communication to transmit data to the nearest fire finder device and try transmitting to a satellite again. This process will continue over and over again until it finally transmits to a satellite, or in the worst case, has to travel hundreds of km back to the hub. In the case that it initially does not find a satellite and GPS location cannot be established, the devices will determine the distance between each other so a map could be built of where it started. This map and other information could then be displayed on a screen back at the central hub.

Schedule

9/10-9/23	Planning After project groups are formed, we will start planning different approaches to how this fire finder device could work. We will also start researching what would be the best form of satellite and wireless communication in a dense forest situation. How the device to device communication could work and how a map could be drawn and displayed from this. Decide what would be the best low power options when it comes to the cameras, wireless communication, smoke/temperature sensors, display, and microprocessor. Write the draft proposal by 9/18.
9/24-10/6	Design and part selection After our initial proposal meetings have completed and the research has been done, we will start ordering the selected parts in this stage. We would see if the chosen processor has its own board with I/O pins so we could start testing it with the components. The final proposal would be submitted and all the final parts would be ordered.
10/7-10/25	Breadboard prototype and PCB design Starting connecting parts together on the breadboard as an initial prototype and programming the microcontroller to test out wireless communication and GPS functionality. Have the temperature sensor, smoke sensor, and camera intake data. Order the PCBs as soon as possible in this period.
10/26-11/10	Software development and testing on prototype Get the wireless communication and GPS to fully work on the breadboard prototype. This means that the prototype will be able to initially detect the smoke, wake up, provide information of the fire, and have this information be sent up to satellite to go back to a “central hub” and showcase information on

	screen. In the case there is no satellite, make sure the prototype can send data wirelessly hopping in between devices. Start assembling PCBs and testing for any faulty equipment and identify any issues in the design.
11/26-12/9	Final testing and expo Have the project completed by 12/4 and fix any minor issues we still have. Design and finish a poster for the expo and begin writing the final project report.
12/10-12/11	Final report completed

Budget

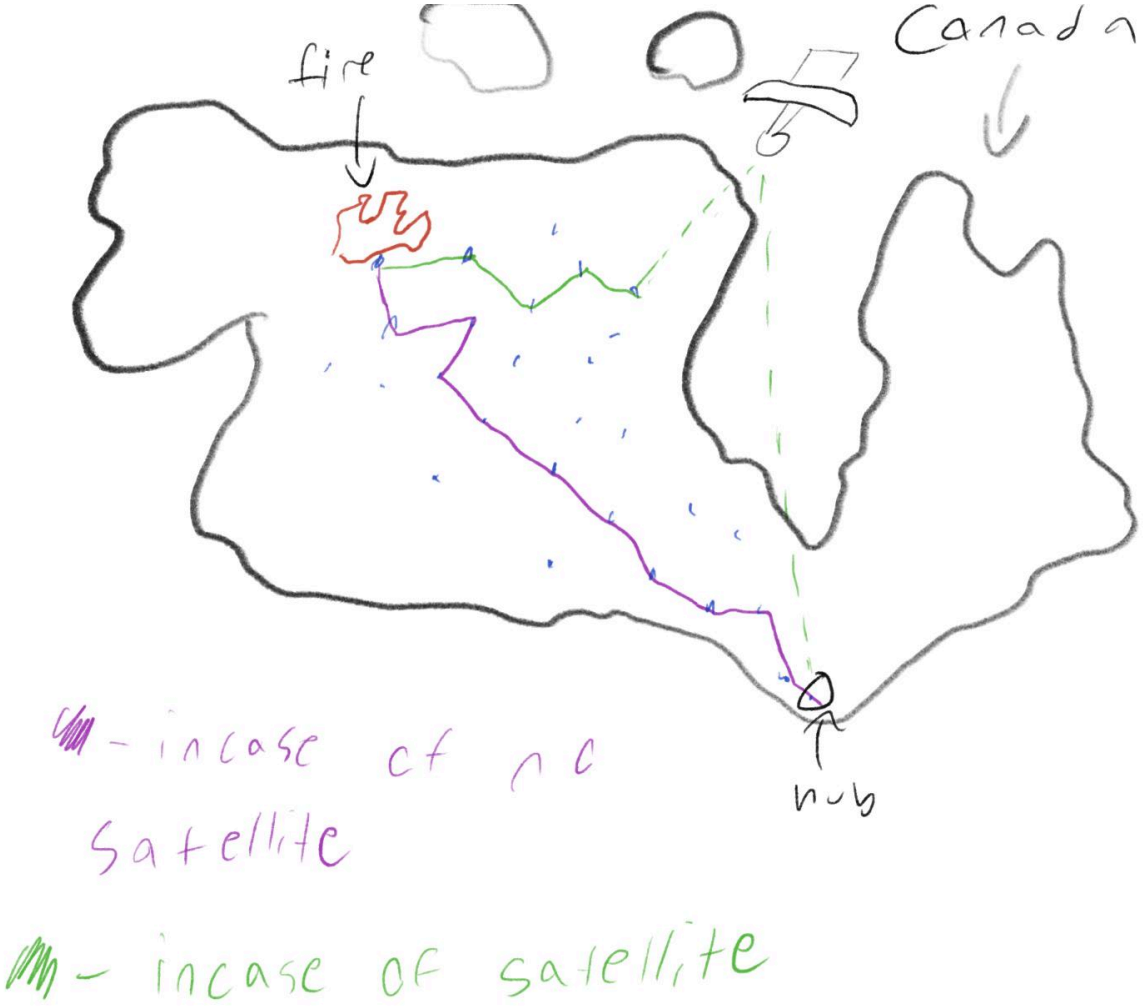
It is not easy to say how much the budget should be at this point but we can generally estimate the average cost for each component on digikey and assume we are building 3 devices in total for the project. Doing this, we can see that this project should be under budget.

What	quantity	Cost per	Total cost
STM32L4	4	~$5	~$15
LoRa device	4	~$7	~$28
GPS/Satellite communication	4	~$30	~$120
IR camera	6	~$40	~$240 (maybe cut down to 1 camera per device instead of 2)
Smoke sensor	3	~$20	~$60
Temperature sensor	3	~$2	~$6
Display	1	~$25	~$25
Total			~$494

Issues

1. Communication between devices in dense forest
2. Communication to satellite for gps location
3. Building a map to figure out location in case of no GPS
4. sensors/camera being low power and data is transmitted in reasonable time

Issue 1: This might prove to be difficult but should be possible using LoRa devices and distance restrictions. Issue 2: might be confusing but should be doable, there should be documentation online for it. Issue 3: This could be difficult since it isn't entirely clear to me how a map could be built using only LoRa devices to figure out position. Issue 4: This depends on part selection in terms of how much memory they take up and how fast the satellite/LoRa can transmit.



Assisting Blind People with LiDAR Interface for Daily Navigation

---

Blind people face the challenge of navigating in a world with constant moving parts and new structures being built or changed everyday without vision. Every new design choice that changes the environment is not too much of a concern for those of us with vision since we can see it and go around or through it. However, those who cannot see cannot do this so easily and rely on tools such as walking sticks to traverse. These sticks mainly aid with knowing the layout of the floor a foot ahead of you but what if the surface is uneven with no clear markers for blind people? What if the space is crowded? What if blind people wanted to generally know the 3D space around them?

I propose a wearable LiDAR navigation system to have a better understanding of the environment in 3D. This system would replace traditional walking sticks with instead lasers constantly spinning and scanning the world around them. The wearable device would also be equipped with vibrational motors that would notify them of how close they are to something or someone with varying vibrational intensity. This clothing would have a portable battery that could be stored in some type of bag so it wouldn't be too noticeable. The user would have buttons to control intensity and distance of what the LiDAR should intake data of.

Schedule

9/10-9/23	### Planning After project groups are formed, we will start planning different approaches to how this fire finder device could work. We will also start researching how LiDAR fully functions and what kind of system we could make wearable. How the LiDAR would decide how much the motors should vibrate. Decide what would be the best low power options when it comes to the microprocessor and motors since the LiDAR would take up most of the power. Write the draft proposal by 9/18.
9/24-10/6	### Design and part selection After our initial proposal meetings have completed and the research has been done, we will start ordering the selected parts in this stage. We would see if the chosen processor has its own board with I/O pins so we could start testing it with the components. The final proposal would be submitted and all the final parts would be ordered.
10/7-10/25	### Breadboard prototype and PCB design Starting connecting parts together on the breadboard as an initial prototype and programming the microcontroller to test LiDAR system. Have the vibrational motors start working. Have buttons control intensity and distance. Order the

	PCBs as soon as possible in this period. Start working on sewing motors onto some type of clothing (most likely jacket).
10/26-11/10	Software development and testing on prototype Get the LiDAR to fully work by spinning and intaking data of the environment. Have this data be evaluated by the microprocessor and sent to the vibrational motors to start vibrating specific motors relative to what the LiDAR is sensing. Have buttons be able to increase or decrease intensity and change the distance of how much the LiDAR should care to intake data of. Start assembling PCBs and testing for any faulty equipment and identify any issues in the design.
11/26-12/9	Final testing and expo Have the project completed by 12/4 and fix any minor issues we still have. Clothing should be finished with parts sewed on. Design and finish a poster for the expo and begin writing the final project report.
12/10-12/11	Final report completed



Motor going off

Motor

Proposal #1

High Level Description

This project would create a weather station that measures local environmental conditions and uses that information to provide data about the surrounding area. The station would measure conditions such as temperature, humidity, atmospheric pressure, light levels, wind speed and direction, and rainfall. It would collect and process these measurements and display the current conditions to the user either on a display or on their phone. The system could also detect significant changes in the environment over time like the beginning of rainfall or unusually strong winds and provide an alert to the user.

Many personal weather stations already provide similar measurements, but stations with more advanced instrumentation and higher accuracy can become expensive. Additionally, many personal weather stations lack the capabilities to track conditions over time. Our goal would be to develop a low-cost, portable weather station that provides useful and reliable local measurements while remaining accessible to a wider range of users. The station could be particularly useful for gardeners and farmers who want the information to help make decisions to protect their plants and weather enthusiasts who are interested in monitoring conditions around their homes.

Budget

This budget includes the major costs associated with the project.

	Cost
BME280 pressure, temperature, and humidity sensor	$14.95
Anemometer	65
Wind vane	20
Rain gauge	29.9
Weatherproof enclosure	$50
PCB	$100
Misc	$150
Total	$430

Schedule

Week	
9/14 - 9/27	Research sensors, define requirements, and decide on main components for the project. Complete final project proposal
9/28 - 10/11	Develop sensor prototype for data collection and communication.

10/12 - 10/25	Design and order the custom PCB while continuing development on the prototype.
10/26 - 11/8	Test PCB and integrate all sensors.
11/9 - 11/22	Build the weatherproof enclosure and perform outdoor testing.
11/23 - 12/10	Final testing, debugging, and Design Expo preparation.

Implementation Concerns

1. Outdoor reliability
2. Communication with the weather station

Issue 1 can be addressed by designing the weather station enclosure to be fully weatherproof and able to withstand exposure to rain and wind. Issue 2 will require further research into different communication methods once the intended use case and communication range have been narrowed down.

Evaluation & Testing

The station would be tested based on sensor accuracy and reliability, The measurements could be compared against a nearby commercial weather station or known reference instruments. A successful system would correctly collect local weather data, survive outdoor operation, and provide useful information or alerts based on the measurements.

Proposal #2
High Level Description

This project would create a water purification system that uses several sensors to monitor water before and after it passes through a filter. The system would collect measurements such as turbidity, temperature, and pH to assess changes in water quality and determine whether the water should be filtered. When filtration is needed, the system would automatically run the water through the filter and notify the user when the filter needs to be replaced. Although existing water purifiers that simply run water through a filter can improve water quality, they do not provide much information about the condition of the water or the filter itself. This project would create a smart, embedded water purification system that both filters water and monitors its quality in real time.

Budget

This budget includes the major costs associated with the project.

	Cost
pH sensor	$29.50
Temperature sensor	$9.95
Turbidity sensor	$9.90
Pump/valve + tubing	$40
Filter + filter housing	$50
Display / LEDs	$20
PCB	$100
Enclosure	$50
Misc.	$100
Total	$409.35

Schedule

9/14 - 9/27	Determine which water-quality measurements are practical and research candidate sensors, pumps, filters, microcontrollers, and displays.
9/28 - 10/11	Build a prototype using development boards. Develop basic pump/valve control and establish communication between the sensors and microcontroller.
10/12 - 10/25	Design and order the custom PCB while continuing development on the prototype.

10/26 - 11/8	Test PCB and integrate all sensors.
11/9 - 11/22	Build the enclosure and perform testing for leaks, sensor reliability, and correct system behavior.
11/23 - 12/10	Final testing, debugging, and Design Expo preparation.

Implementation Concerns

1. Protecting electronics from water
2. Accuracy of inexpensive water-quality sensors

Issue 1 can be addressed by housing the PCB and other electronics in a sealed enclosure, while only exposing the sensors, pump, and other necessary components to the water. Issue 2 will require additional testing to determine how accurately inexpensive water-quality sensors can measure meaningful changes in the water.

Evaluation & Testing

The system could be tested using different water samples and measuring the sensor output before and after the water goes through the filter. The system would be successful if it detects the difference between the filtered and unfiltered water samples.

Proposal 1: Keyboard Key-Light Tutor

High-Level Description

Learning piano could be the most regularly abandoned hobby of all time. The amount of people I've met who 'learned piano as a kid' (me included), only to forget all of it 2 years later is astronomical. How many times I've heard 'Oh I'm learning piano' only to follow up and see that they've abandoned it altogether only a month after. Advertisements for software solutions to this issue have plagued my phone for years, with the animation of the falling notes onto the screen synced with a MIDI keyboard. But this solution requires a tablet propped up next to the instrument, splitting the attention between a screen and the hands. This is where a different solution comes into play. A physical light guide that sits directly on the keys will remove the split. The learner can look at their hands the whole time, the same way they eventually will without any aid at all.

Some commercial products already exist, like Piano LED Shop selling a similar add-on, but they can run into the hundreds of dollars, which is not approachable for the target audience of casual and beginner learning. This hundred dollar difference could be the difference between abandoning or actually learning the piano. A low-cost, DIY version built around more available parts (addressable LED strips, a small microcontroller, and a standard MIDI-in circuit) will deliver most of the same functionality for a fraction of the price.

Details

The system will be a strip of individually addressable LEDs mounted onto a rail that will rest on top of the keyboard keys, aligned so that every LED sits above / right behind the corresponding key. A microcontroller will drive the strip and read live MIDI data from the keyboard's MIDI-out port (5-pin DIN or USB-MIDI, depending on the instrument). The controller will load a target song ahead of time from a MIDI file, then convert it into a sequence of expected notes, and will light the LEDs for the next notes in that sequence. When it receives a matching MIDI note, it will mark that note completed and move onto the next one in the sequence. If it is a wrong note the light will stay on the correct key, and not advance, and flash a red light above the wrong key. This is so the learner cannot skip ahead on accident, and will immediately know they were wrong.

The proposal envisions building one working unit: an LED bar of enough length to span the keyboard, a PCB housing the microcontroller and the MIDI circuitry, and a physical mount that can attach onto the keyboard.

Implementation Issues

Schedule

9/10-9/23	**Planning** Confirm which keyboard MIDI output format we are targeting (5-pin DIN vs. USB-MIDI). Prototype a basic MIDI-in circuit on a breadboard and confirm reading note-on/note-off bytes over UART. Order a short test LED strip and confirm addressable LED control, as well as try to size how much external power we will need. Write the draft proposal by 9/18.
9/24-10/6	**Design and part selection** Select a microcontroller with a UART for MIDI input and enough PWM/DMA capability to drive a long LED strip without flicker or lag. Order the full-length LED strip, MIDI DIN jack, and external 5V supply. Get a breadboard prototype that can receive MIDI messages, light up the LED on the test strip. Submit the final proposal by 10/1. All prototype parts ordered by 10/4.
10/7-10/25	**Breadboard prototype and PCB design** Get a full breadboard prototype working end to end for a short note sequence .Test the full length strip and make sure the brightness is consistent and strong across the whole strip, not just near power. Begin the PCB layout combining the microcontroller, MIDI-in circuit, and the LED driver output. Order PCBs by 10/26.
10/26-11/10	**Software development while waiting on PCB** Build the MIDI-file parser that converts .mid files into a note sequence and start to test it against real song files on the breadboard prototype. Design and fabricate the full-length mounting rail, check for rigidity across the full keyboard. Get the PCB's assembled once they arrive.
11/11-11/25	**Software and testing on PCB** Port the breadboard logic onto the PCB, which should now be driving the full length strip. Test with multiple real songs spanning the full keyboard range, along with multiple testers

	of different skill levels (make sure one is a complete beginner). Fix any bugs in MIDI parsing we might run into, and any LED strip issues.
11/26 - 12/9	**Final testing, additional features, poster and expo** Targeting project completion by 12/3. Build the poster, begin the final report, fix remaining bugs, add any stretch features if time allows.
12/10 - 12/11	**Finish final report**

Budget

What	Quantity	Cost per	Cost total
Microcontroller, RP2040?	3	$6.00	$18.00
LED strip, full 61-key length (~3 ft, 61 LEDs)	2	$18.00	$36.00
MIDI DIN jack	3	$4.00	$12.00
External 5V power supply	2	$10.00	$20.00
PCB manufacture	5	$15.00	$75.00
Mounting rail / enclosure materials	1	$40.00	$40.00
Misc parts (connectors, wire, fasteners)	1	$40.00	$40.00
Shipping	1	$30.00	$30.00
Total			$271.00

Rough estimate budget from sources found online. Cost is very doable.

Other Implementation Issues

- Parsing real-world MIDI files, with chords and overlapping notes, into a clean sequence of ‘next keys’
- Physically mounting the full length LED bar without sagging or blocking any keys
- Not all keyboards implement MIDI identically, so confirming the formatting for that
- Making sure that the POWER draw isn’t too much, likely will need external supply

Parsing the MIDI files should be okay, it is a documented problem so the work should have been partly done. Not just an us issue. Mounting shouldn't be too bad with a rigid rail. We can target our exact keyboard before doing any more work for the MIDI format. And the power supply will need an external 5V for the worst case.

Evaluation and/or testing

Success will be measured whether or not a user with no piano experience can successfully follow the lit up keys across the full keyboard correctly for a pre-loaded song, using only the visual guidance. The system must reliably detect if the key presses were correct and not falsely advancing or missing a valid press. This will include songs that use the full keyboard as well, not just one octave. We will have testers of multiple skill levels and time, how long it takes to complete a song, and if the system is responsive enough to react to a more skilled, quicker player.

Proposal 2: Closet Monkey

High-Level Description

I have a pretty large closet, and tracking which items I have, and how often I've worn them, can be an exceedingly manual and irritating task. There are about 1 thousand vibe coded closet apps, of which I've used many, none of which have been successful. We need a method that's less obtrusive, and more passive. Something that can track this for you without you doing anything. Something that will tell you which items you have, and maybe even suggest which you should wear. Like a Roomba, but for your closet. Introducing the **closet monkey**.

Details

The Closet Monkey is a small robot that rides along the top of a closet rod and keeps track of your clothes for you. It scans and photographs every piece hanging in the closet, builds a simple database of what you own, and logs how often each item actually gets worn versus just sitting there. It will periodically scan the closet, notice when something's missing (because you're wearing it), and log that as a "wear" when the item reappears. The idea being when you open the app you can actually see what you've worn without manual intervention! He will also have a little charging dock that it can charge in automatically so you never have to charge it. How charming!

Implementation Issues

Schedule

9/10-9/23	Figure out how the robot actually moves along the rod (biggest unknown). Look into camera + basic image capture.
9/24-10/6	Pick a microcontroller and camera module. Attempt to get a basic movement mechanism working.
10/7-10/25	Get movement reliable enough to do a full lap of the rod without disturbing clothes. Start on image classification.
10/26-11/10	Build out the "notice something's missing" logic and basic app/database to log wears.

11/11-11/25	Test on an actual closet. Fix whatever breaks (likely a lot)
11/26 - 12/9	Poster, final report, patch remaining bugs.
12/10 - 12/11	Finish final report

Budget

What	Quantity	Cost per	Cost total
Microcontroller	1	$8.00	$8.00
Camera module	1	$15.00	$15.00
Motor/drive mechanism (TBD design)	1	$40.00	$40.00
Charging dock components	1	$25.00	$25.00
Rod/rail hardware or track	1	$20.00	$20.00
Misc parts (wiring, connectors, fasteners)	1	$20.00	$20.00
Shipping	1	$15.00	$15.00
Total			$143.00

Other Implementation Issues

- How does this physically move around the rod?
- How will it charge?
- How will it take pictures? Will it have to move the clothes?

Issue 1 is definitely the most difficult . Getting a robot to stay balanced on top of a rod is a task on its own. No plan for that yet. Issue 2 is solvable, via probably an inductive charging platform like magsafe for Iphones. This will be most reliable. Issue 3 is also difficult. Not sure how to get good photos of clothing from a top POV without having the move other clothes around it.

Evaluation and/or testing

Success would mean it completes a full lap of the rod, counts something as missing, and logs it as a wear when it reappears, and gives a picture of the worn item. Will charge on its own as well!

Wearable IMU System for Baseball Swing Sequencing

Description

Baseball training now involves more data, both biomechanical and batted ball data. This type of data is easy to get if trainers have proper equipment. Consumer tools such as Blast Motion can help players and coaches know bat-related information such as bat speed and attack angle, and systems such as HitTrax can provide them with batted-ball data and video feedback. These tools are useful, however, none of them are regarding biomechanical data. For more detailed biometric information, such as how the pelvis, torso, and lead arm rotate relative to each other, is usually measured by tools like K-Vest/ K-Motion or equivalent full-motion capture system [3].

Coaches can leverage this kind of rotational sequence data because it offers an objective method to compare a hitter's pattern instead of relying on visual observation. For example, K-Vest measures rotational velocity of different parts of the body, finding out the timing of their peaks and comparing them [3]. Previous baseball research used IMU to examine timing and rotational sequence during hitters' swing motion [1], [2]. Current measuring systems, such as K-BASEBALL can cost several thousand dollars. Therefore, It is challenging for users such as high school teams, travel-ball programs or private hitting coaches to access biomechanical information because they cannot afford the system.

Our project tries to focus on a narrower goal: providing cheaper rotational sequence feedback without using a full 3-D biomechanics system. The finished system should use wearable motion sensors to track the timing of relevant body parts in a swing motion and show the sequence and consistency of a series of swing motion to coaches. The goal is not to replace existing professional biomechanical systems or use some algorithm to decide the quality of each swing , but to give smaller teams and coaches an objective measurement. Additionally, they can use it alongside videos, drills to refine trainers hitting motion.

The system would consist of four wearable sensing nodes placed on pelvis, torso , lead upper arm and lead hand, together with a central receiver. This placement is similar to the arrangement of K-Vest/K-Motion for swing analysis [3]. For each sensing node, it consists of an IMU, a microcontroller, a wireless connecting module and a battery.

During a swing, each node records angular velocity data and uses a shared time reference so that measurements from different body parts can be compared. After a swing, all data would be sent to the central receiver. Then the receiver determines the timing of peak angular velocity of each body part and calculates the sequence and timing difference between each segment. Results would be displayed on an application in either laptop or smartphone for coaches to review.

Schedule

9/10-9/28	**System design and start prototyping** Finalize architecture, how to communicate between nodes, select which IMUs, microcontrollers, and wireless modules fit the situation most. Try to test functions in a single node.
9/29-10/12	**Multi-node sensing and synchronization prototype** Expand single node functions to multiple-node functions. Verify IMU sampling, data timestamping, synchronization between nodes, and data transfer. Start building peak-detection software. The goal is to demo sensing and communication functions at Milestone 1
10/13-10/30	**Wearable prototype and PCB design** Try successful functions on different parts of the body, pelvis, torso, lead upper arm and hand. Test the attachment methods and collect swing data. Finalizing PCB design and sending the board for fabrication before the deadline.
10/31-11/9	**Signal processing and interface development** Continue to develop on prototype hardware. Implement an algorithm to decide the timing of peak rotational velocity at each segment and timing differences for segments. Build interface for displaying sequence. It is expected to produce a full function prototype at Milestone 2
11/10-11/27	**Final hardware integration and validation** Assemble and debug our custom PCBs, integrate them into wearable enclosures and repeat the synchronization and sensing tests on final hardware. Use high-frame-rate video to evaluate our hardware.
11/28 -12/10	**Final testing and Design Expo preparation** Keep testing and evaluate consistency between each swing. Evaluate if hardware can distinguish different swing sequences. Fix hardware/ software problems and prepare for the final demonstration and poster for the Design Expo.

Rough Budget:

At this stage, the numbers are estimates because the IMU, microcontroller are not yet determined. In the development process, six units are more desirable because we could potentially break our equipment down. The available budget for this project would be 1000-1250 at most for a four/five-person group. There is margin for replacement of PCB boards.

Item	Quantity	Estimated Cost
High-rate / high-range IMUs	6	$40–60
Microcontroller / wireless hardware	6	$60–120

Batteries, charging and power components	6	$40–60
Custom PCBs and components	6	$120–180
Wearable straps, mounts, and enclosures	—	$60–100
Development boards, receiver, connectors, and spare parts	—	$100–180
Estimated total		$420–700

Suitable high-rate IMUs with gyroscope ±4000 deg/s are available for under approximately $10 per device. For wider gyroscopes, the price of IMUs can jump up to 100+ for each. Ideally, we want to keep it at a low price, but different body segments require different gyroscopes so we need to do some detailed research on it.

Issues

Potential implementation issues

1. Synchronizing and sampling data from multiple IMU accurately enough to compare peak timing.
2. Avoiding gyroscope saturation during swings.
3. Keeping sensors attached and consistently calibrated on different players.
4. The device should not interfere with the swing motion of hitters.

Issue 1 can be addressed by locally timestamping data on each node with a shared sync reference. Then each node transmits buffered data to the central receiver after each swing. For issue 2, we need to test ranges for different body segments and decide a proper IMU before finalizing the PCB. Issue 3 requires tight straps or glove-mounted attachments together with a short static or functional calibration [4]. Issue 4 is solvable, just keep the PCB and battery small enough and test the prototype before enclosure.

Evaluation

We can test in two stages. At first, we test multiple sensing nodes to measure the same event with sufficient consistent timing using controlled motion. The final wearable system would then be tested in a real baseball swing. One way to verify the correctness is to use high-frame-rate video as independent reference for swing events and segment motion. Repeated swing can be used to evaluate the consistency of the measured rotational sequence and peak timing.

A successful system should result in repeatable segment timing measurements without sensor saturation or noticeable sensor movement during the swing. We may need to test some different motions, for example, a swing in which the upper body begins rotating earlier. In this case, the system should be able to distinguish between different swing motions. The final goal is for a

coach to be able to use the displayed sequencing and timing information to compare swings or evaluate changes before and after a drill.

Stretch goal:
If time permits, an additional IMU could be mounted on the bat to connect body-segment sequencing with bat motion and measure hand-to-bat timing or bat rotational speed.

References

---

[1] T. A. Punchihewa et al., "Efficacy of inertial measurement units in the evaluation of trunk and hand kinematics in baseball hitting," Sensors, vol. 20, no. 24, 2020.
[2] Y.-S. Lee et al., "Utilization of inertial measurement units for determining the sequential chain of baseball strike posture," Sensors, vol. 21, no. 9, 2021.
[3] Driveline Baseball, "An Introduction to K-Vest," [Online]. Available: https://drivelinebaseball.com/blogs/blog/introduction-k-vest?srsltid=AfmBOoqqzQ7OsTkHiyvMVIHIQVImFA28YM8LyV9q47qS0oaqhjx5Puzw
[4] M. Pacher et al., "Sensor-to-segment calibration methodologies for lower-body kinematic analysis with inertial sensors: A systematic review," Sensors, vol. 20, no. 11, 2020.

Drone Payload Swing Suppression System

Description

Small drones are now utilized to carry payloads and perform tasks like delivery, inspection and many other tasks. But when payloads are suspended in the air, they can swing like a pendulum. When a drone accelerates, slows down, or turns, the oscillating motion can make a drone hard to control, reduce its positioning accuracy, or even lose its payload during tasks.

Our project tries to solve this problem. We want to develop an add-on system that measures payload swing and reduces oscillation by commanding small lateral motions of the drone. The goal is not to redesign the flight controller, but to add a damping system to make the flight with payload smoother.

Basic System Idea

We can measure payload cable angle using an angle sensor near attachment point, such as a potentiometer or a magnetic encoder. A microcontroller or companion processor reads the sensor and calculates the swing angle and velocity. Then, it determines if a corrective lateral movement is necessary to make the payload stop swinging. Previous work have investigated active swing suppression for quadrotors with suspended payloads [1], [2].

Schedule

1. Select the suspension geometry, angle sensor, processor, and drone interface, and build a simple prototype.

2. Implement a system to detect swing and estimate swing state. Develop a simple damping controller on a simple setup.
3. Finalize sensor, controllers and other components PCB design and order board. Continue controller design and DroneKit software development to control drone movement.
4. Assemble the final hardware and use the damping system on a drone.
5. Perform flight tests with suspended payloads to test and compare the performance of damping enabled/disabled.
6. Refine the controller and packaging for final demonstration.

Rough Budget

Our system should have an angle sensor, a microcontroller for the damping system, custom PCB, power circuitry, mechanical mounting hardware, and a test payload. If an existing drone can be used, our budget can be well below course budget, around $250-400 for prototyping. However, if a drone is needed, we need to select properly to not exceed budget but also perform tasks well.

Issues

The main concerns at this time are:

1. Measure the payload swing accurately enough for feedback control
2. Avoid controller delay or bad tuning so that the damping system increases oscillation rather than reducing it.
3. Make sure the damping controller would not affect existing flight-control loops.

Issue 1 can be addressed by using a direct cable-angle sensor and filtering the signal before estimating swing velocity. Issue 2 can be tested first on a simple pendulum setup to tune the damping controller before flight. For Issue 3, the damping system would only send small high-level position or velocity corrections through DroneKit rather than controlling the motors directly.

Issue 3 is fairly important because a drone might crash if the damping controller makes a flight-control loop fail. Flight testing would also need to be done carefully using a lightweight payload, low altitude, and a controlled testing area.

Evaluation

We can test the system by giving a drone the same lateral movement or a disturbance, and compare the result under enabling or disabling the damping controller. Several metrics are worth considering, including maximum swing angle, average swing magnitude, and settling time. The system should reduce payload oscillations without interfering with the drone. The drone should not be unstable or have worse flight performance.

Stretch Goal

If time permits, the controller could be extended from one-dimensional swing suppression to two-dimensional payload motion, or adjusted automatically for different cable lengths and payload masses.

References

---

[1] M. E. Guerrero-Sánchez et al., “Swing-attenuation for a quadrotor transporting a cable-suspended payload,” ISA Transactions, vol. 68, pp. 433–449, 2017.

[2] G. Yang et al., “Anti-swing control and trajectory planning of quadrotor suspended payload system with variable length cable,” Asian Journal of Control, 2022.

DriveLock Park for Parking Lot Payment

High level Overview

[1] Across the University of Michigan Ann Arbor Campus there are more than 27,000 parking lots, and access to these parking lots is run through parking permits, central pay stations, meters, gates or cashier-attendants. A recurring issue is dealing with parking tickets and unpaid parking.

Often people forget to buy a pass or make payment for the parking they are utilizing. This leads to parking enforcement having to deal with simple issues with complicated processes like towing vehicles or ticketing individuals. [2] Over 8000 vehicles were towed in just Detroit to target illegal parking.

A driver who mistakenly parked without paying often has to deal with the hassle of paying a higher fee, or even worse has to deal with their vehicle being towed away. This leads to unnecessary distress and serious waste of time and money on everyone's part.

As a specific system, instead of doing towing or compounding parking fees, you simply lock the car in place inside the parking lot until the driver pays for the time they utilized in the parking lot. This sort of system would address the issue without the unnecessary distress, loss of time and money that people have to deal with currently.

Details

As soon as the car parks into a parking lot rather than a payment collection booth or parking enforcement, a lock barrier activates and locks the car in the parking lot. The driver has the convenience to pay now or pay later when they return if running late. At their convenience they can provide their parking permit or payment online through nearby QR codes or websites leading to the lock barrier deactivating.

The system would work with a breakdown of vehicle detection through an inductive loop or a camera sensor to establish vehicle detection. Charging of the system should be done

through a cable to an open outlet. It will utilize a servo or motor system to put up and lock a barrier to lock the vehicle, and communicate via bluetooth and internet to unlock the vehicle and pull down the barrier through motor deactivation.

---

[1]https://ltp.umich.edu/parking-/

[2]https://www.bridgedetroit.com/detroit-targets-8k-abandoned-vehicles-in-crackdown-do-residents-know-the-rules/

Schedule

The dates shown are “tentative” and will most likely change but they should be maintained as closely as possible.

Dates:	Milestones:
9/10 - 9/22	**Planning** The first two weeks should be about researching different sensors, servo/motor, and deciding on the best processor for fitting our requirements. Hoping to complete the draft proposal.
9/24 - 10/6	**Design and part selection** Once the proposal is done we should order our sensors, servo, and CPU. We should utilize these to make a minimal prototype design of the parking locking system, and submit our final proposal. All of our parts we needed/selected by the previous milestone

	should be ordered.
10/7 - 10/25	**Prototype and PCB design** We should be able to achieve WiFi connectivity with our prototype and order our PCB design.
10/26 - 11/10	**Software development and testing on prototype** During our time waiting for the PCB design to come back we should test the wifi phone based commands to reach the prototype device, and have all basic functionality completed and start on exploring the Solar Panel based subsystem if possible.
11/11 - 11/25	**Software development and testing on PCB-based system** Testing our PCB to see if all traces and connections are correct. Testing our basic functionality code on our PCB based design and fixing any bugs whether hardware or software based in the system. Additional stretch goals of solar power will be worked on or added during this time.
11/26 - 12/9	**Final testing, additional features,**

	poster and expo During this point, we should have completed our project and should work on the other deliverables like poster, final project, and fix any minor issues found last minute.
12/10 - 12/11	**Finish final report**

Budget

To get an exact budget is difficult during brainstorming. Here is a tentative one. Click Links to see the exact shopping cost.

Microcontroller: ESP32 DEVKIT V1: $8.99

Arduino: $0 (Can utilize same Arduino as lab)

Display: 0.96" OLED Display : $7.99

Barrier Motor: MG996 Servo : $9.99

Motor Driver: TB6612FNG : $14.50

Limited Switches: ESE-22MV21T : 25 Switches : $8.85

Sensors: Camera-OV7670 : $6.23

Power Supply: WSU120-0700-R13: $7.58

Prototype materials: 3D printed + Hardware: $20 - $30 (Estimated Expense)

PCB: $15 - 20 (Estimated Expense)

Wires/Connectors/Breadboard: $0 (Can utilize the ones in lab)

Total: $99.13 - $114.13

Issues

The largest concerns for this project at the time

1. Developing an algorithm for accurate Car detection

2. Communicating effectively with Wifi and other devices
3. Soldering the Camera/Inductive Loop detection mechanism
4. Having to see if the mechanical aspects of the design work appropriate and have enough to withstand the vehicle

Issue 1 is unknown but seems reasonable as there are many ways to go about it. Issue 2 might be a challenge but bluetooth connection is something that even many others have figured out before. Issue 3 will be quite difficult as we will need to be careful not to cause any harm to the system while soldering and not sure if the inductive loop will be affected by the soldering/close proximity to other components. Issue 4 is probably the most difficult as my mechanical background isn't as strong and figuring out how to make it so the barrier can withstand any vehicle will be difficult.

Additional stretch goal

Our stretch goal would be to try to get some sort of rechargeable solar powered system to work instead of an outlet cable as that would conserve energy, and make the unit independent of power outages.

Evaluation/Testing in Regard to Realworld Issues

The project should be evaluated in regard to how the system can detect a real car and lock the vehicle and then activate when paid through an online method, and deactivate the locking mechanism.

Trash Talker

Overview

[1]In 2018 in the USA, a total of 292.4 million tons of waste were generated per day, and approximately 32.1% of the waste goes towards recycling and compost. A large part of this issue is due to people not understanding what goes into recycling, compost and landfill or in a rush putting the trash in landfill trash. [1]Per person waste is also increasing from 4.5 pounds per person in 2017 to 4.9 pounds per person in 2018.

---

[1]https://www.epa.gov/facts-and-figures-about-materials-waste-and-recycling/national-overview-facts-and-figures-materials

Details

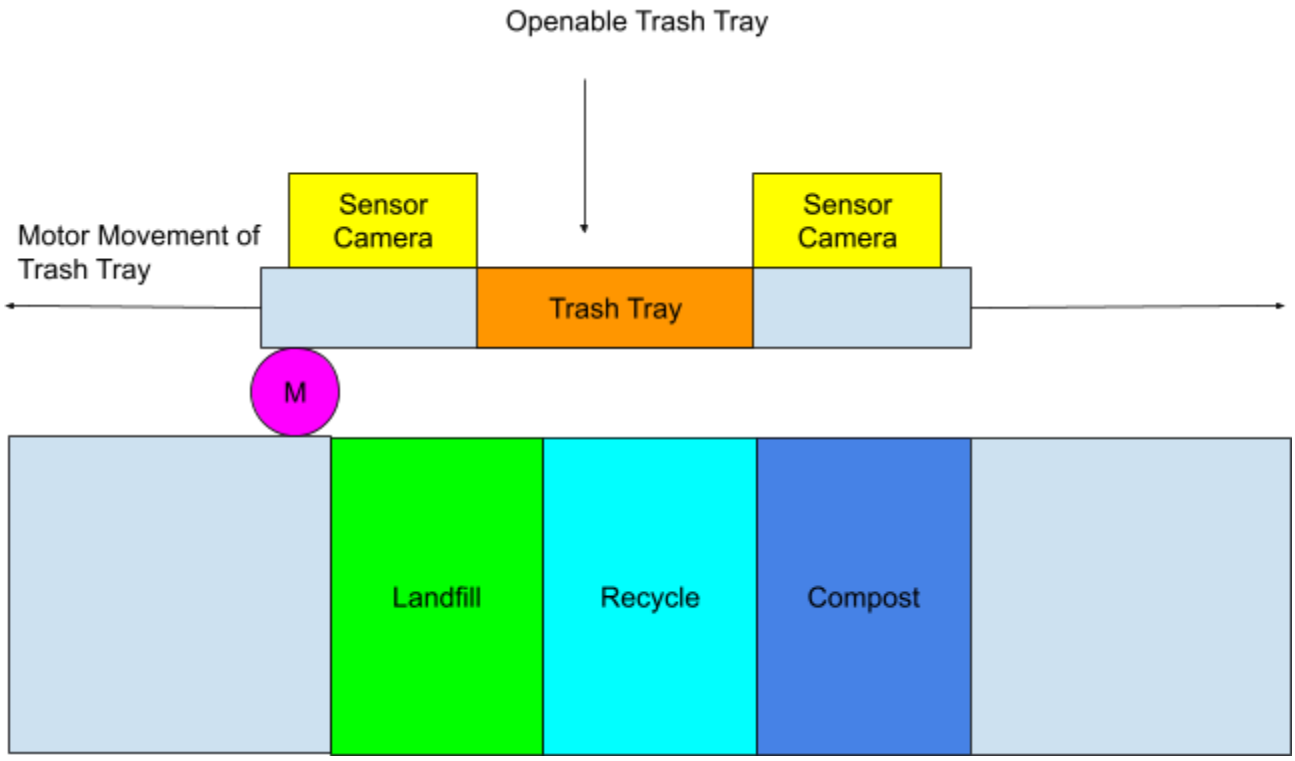
To address this problem, a system can be implemented where:

1. Trash can is closed until trash is placed on top
2. Camera/ML Algorithm decided which type of trash is there recyclable, compostable, or landfill
3. Then a motor comes and moves the trash to the correct type of trash type
4. The System tells you why and which type of trash you deposited
5. Tells you a random statistic of how climate was helped by your efforts

Necessary Components

- FPGA needed to handle the heavy ML workloads to identify the trash type
- Motor/Motor Driver System where correct classifications can trigger motor movement allowing you to move the trash based on the classification
- Camera Sensors to provide feed you can utilize to identify the trash type
- Speaker to communicate random statistics and explain what type of trash you deposited

Diagram


Openable Trash Tray
Sensor Camera
Sensor Camera
Motor Movement of Trash Tray
Trash Tray
M
Landfill
Recycle
Compost

Low-cost Smart-ring Wearable Device for Health Monitoring and Medical Alert

---

Introduction

By September of 2025, about 5.5 million Oura Rings have been sold worldwide, with more than half of those sales being from that past year[1]. However, with a price point of nearly $400[2], Oura Rings prove to be on the expensive end of the market that is out of reach for the wide majority of consumers. On the other hand, ultra-budget friendly smart-rings, such as Colmi and Unigthro, have less comprehensive health monitoring capabilities often providing simple data logs of heart rate, blood oxygen, and step counts amongst other measurements.

Real-time health and motion metrics, including hand tremors, fall detection, and sleep profiling, are also features that tend to be locked behind a monthly subscription. Overall, the market for smart-rings seems to be highly polarized with premium, high-cost solutions on one end and highly simplified, budget-friendly solutions on the other. Therefore, low-cost health monitoring smart-ring wearable devices that allow customers to receive all the benefits of a premium smart-ring without breaking the bank is a niche that has yet to be serviced.

While the existing smart-ring wearables are marketed for more active, fitness-focused customers, senior and elderly customers prove to gain a tremendous advantage from the less bulky form factor of smart-rings. When combined with the capabilities of traditional medial alert devices, this low cost smart-ring with health monitoring, fall detection, and panic gestures can be the sleek, life-saving solution perfectly suited for elderly care.

Details

For meaningful health tracking for the elderly, the smart-ring would include baseline health monitoring statistics as well as several medical alert features ranging from severe concern to more minor concern.

In terms of health monitoring, the smart-ring would be capable of tracking heart rate, blood oxygen levels, skin temperature, and step count. These serve as the basic diagnostics that help keep track of a customer's daily health, specifically in relation to heart health and physical activity. This can be accomplished through the use of multiple sensors that communicate with a microcontroller.

In terms of medical alert, the smart-ring has three main capabilities that serve as useful tools for elderly care: real-time fall and impact detection, an SOS gesture trigger, and nighttime movement and wandering detection. By processing accelerometry data directly on the microprocessor, any spikes in g-forces that indicate a fall by the customer can be alerted quickly using an emergency bluetooth signal to a phone or other device. Furthermore, panic buttons are

---

[1] https://en.wikipedia.org/wiki/Oura_Health
[2] https://ouraring.com/store

commonly used in elderly medical alert devices and often require users to reach down and manually press a button. In place of that, the smart-ring would have a programmed gesture that serves as an SOS alert to a caregiver. For example, three firm taps with a finger or a specific flicking gesture. Lastly, this smart-ring would have the ability to log micro-movements during sleep overnight. Any detection of excessive movement would be alerted to ensure overnight safety and prevent nighttime wandering.

Schedule

This is the proposed schedule for working on this project. This is tentative and will most likely change throughout the duration of the project.

9/10-9/23	**Research and Planning** ● Develop a list of all possible features the smart-ring could incorporate in terms of health tracking ● Identify all necessary sensors and components needed for the health-monitoring features ● Determine if we need to eliminate some features based on spacing concerns since a ring is a pretty small surface area ● Get a nearly final list of features and sensors necessary ● Submit draft proposal
9/24-10/6	**Design and Part Finalization** ● Final all the necessary sensors and components and order them ● Identify and order the specific microprocessor needed ● Start working out communication protocols we want to use ● Determine what algorithms we want to develop ● Develop an electrical schematic for all the components ● Create ideas for a ring shaped housing for everything
10/7-10/25	**Prototyping and PCB Design** ● Start prototyping with the ordered parts on a breadboard or protoboard ● Try to get all the sensors to record data without issues and communicate with microprocessor as desired using the communication protocols ● Change any design features that can't be accommodated ● Finalize the overall prototype and start designing the PCB ● Start CAD-ing a mechanical housing for the PCB in the shape of a ring ● 3D-print various iteration of the ring housing ● Aim to submit the PCB for manufacturing and maybe assembly as soon as possible
10/26-11/10	**Testing/Validation & Software Implementation** ● Have a final prototype of the 3D-printed ring based on the dimensions of the PCB ● Have a fully functional breadboard prototype that works almost identically to how the PCB prototype would ● Once PCB arrives get started on assembling if needed and start validating the PCB

	• Debug any issues with the PCB and make any changes to the design if needed • Start working on the detection algorithms • Submit a 2nd PCB order if necessary
11/11-11/25	**Final Testing and Software Tweaks** • Test and finalize the PCB, especially if a second one was ordered • The overall algorithms should be done by now, whether they were done by hand or found online • Start developing a simple app for use with the smart-ring, if we want to incorporate that
11/26-12/9	**Wrapping Up, Poster & Expo**
12/10-12/11	**Final Report**

Budget

This is just a basic list of items, not including stuff like capacitors, linear regulators, and etc.

Item	Price
Processor: ESP32-C3-MINI-1	$3.26
Accelerometer: LIS3DH	$4.95
Pulse Ox & Heart Rate: MAX14595ETA+T	$4.05
Battery: 3.7V LiPo	~ $4.00
PCB Manufacturing & Assembly (maybe)	~ $100
Total	~ $120

Given the overall budget of $250 per person, we will comfortably be below the budget even with any other components that will be added later on.

Issues

The biggest issues that I can foresee at this time are:

1. Miniaturizing the PCB enough so that it can reasonably fit inside a ring
2. Soldering/Assembling the PCB
3. CAD-ing and 3D printing a ring shaped housing that allows the sensors to function as needed

Issue #1 seems to be the biggest problem that has the potential to cause this project to not work as intended. Since the goal of a smart-ring wearable is to be sleek and jewelry-like, not being able to make the PCB small enough might not allow us to accomplish our goal. Issue #2 isn't too much of a problem since we could get the PCB assembled by the manufacturer depending on the price, but otherwise we could intentionally procure a soldering expert as a team member for this project. Issue #3 also seems to be a pretty big problem since the ring housing is a major part of the finished product. However, this can also be resolved by finding a team member that is good at CAD.

Additional Stretch Goal

There are various stretch goals of this project, including adding more health-monitoring features, such as sleep detection, blood pressure detection, and so on. However, due to the size constraints of our project, these goals might not be very realistic to add on. It might be more prudent to focus on making a simple, reliable product rather than to add a stretch goal since this is a more challenging project size-wise. On the other hand, creating an optional app that can interface with the smart-ring and display all of the health metrics is another stretch goal. If we want to add more features to that app, that would be a better avenue to take than to mess with our hardware design.

Smart Water Bottle for Electrolyte Monitoring

Introduction

With a new trendy water bottle entering the market every year and becoming the new fashion “fad,” water bottles are a very profitable business opportunity. While the current lineup of popular water bottles, like Owalas and Stanley cups, are largely popular due to their colors and dual sipping-straw technology, electronic advancements in water bottle technology are not yet popular amongst the general population.

Apart from tracking the amount of water that one drinks, tracking the balance of electrolytes is also key for preventing dehydration, especially amongst athletes and people that do high-impact exercises[3]. Therefore, by creating a water bottle that can detect the overall amount of electrolytes that have been consumed based on the liquid that has entered and drunk out of the water bottle, we can track overall electrolyte balance preventing electrolyte imbalance.

Details

This new water bottle would contain an electrolyte measurement module as one of its many functions along with an OLED screen that can be used for interfacing with the users. Apart from this, the water bottle would also include various other features such as GPS tracking for preventing losing the bottle, remote locking mechanism to prevent tampering, and bluetooth capabilities for getting health insights on an accompanying app.

Schedule

9/10-9/23	**Research and Planning** ● Research sensors and parts that would allow for these features ● Order dev kits and sensors for prototyping
9/24-10/6	**Design and Part Finalization** ● Start experimenting with the sensors that we ordered ● Try connecting components together
10/7-10/25	**Prototyping and PCB Design** ● Get things breadboarded together ● Get a final schematic together and start PCB design
10/26-11/10	**PCB Design & Software Implementation** ● Continue PCB design and get it ordered

[3] https://my.clevelandclinic.org/health/symptoms/24019-electrolyte-imbalance

	• Get a finished breadboarded prototype working • Start working on code
11/11-11/25	**Final Testing and Software Tweaks** • Start testing PCB and make sure it works • Finalize the code
11/26-12/9	**Wrapping Up, Poster & Expo**
12/10-12/11	**Final Report**

Budget

This is just a basic list of items, not including stuff like capacitors, linear regulators, and etc.

Item	Price
Processor: ESP32-S3-DevKitC-1	~ $10.00
GPS: u-blox NEO-M8N Module	$34.05
Electrolyte: Analog TDS Sensor Kit	$11.80
DS18B20 Waterproof Probe	$9.95
0.96" I2C OLED	$17.50
Total	~ $80

Given the overall budget of $250 per person, we will comfortably be below the budget even with any other components that will be added later on.

Issues

As of now, I don't really see many potential issues for this project. One challenge might be adding an auto-locking mechanism, especially on a pre-existing water bottle. Another issue might be deciding if we want to 3D print our own custom water bottle or use a generic one available on the market.

Edge-Tensor Accelerator With Linux DMA

Inference models utilizing tensors, such as real-time computer vision systems, are notorious for being computationally heavy. Enough so that running complex neural networks usually requires compute power and memory bandwidth to be located in local data centers, severely limiting the capabilities of both locally hosted and real time systems. Furthermore, modern AI acceleration is challenged by the memory bottleneck: Even with CPU compute growing, the memory bandwidth of these processors often limits real-time inference applications.

By prioritizing memory bandwidth and data reuse over simply bolstering compute, I propose a system that demonstrates spatial computing and memory optimizations that are being studied and applied at frontier AI labs such as Nvidia, Google, and Meta. This project aims to create an edge-based neural network accelerator optimized for real-time computer vision that operates independently of cloud computing resources.

Details

This project is built around an FPGA-based systolic array fed by a host CPU (from our personal devices) over a Linux DMA pipeline. This architecture optimizes the inference path from the camera to the silicon multiplier. The processing engine itself features an *N-by-N* array of processing elements designed to perform 8-bit integer quantized multiply-accumulate operations. To prevent mathematical overflow, the results are collected in 32-bit registers.

To eliminate unnecessary CPU overhead, part of this project will be to develop a custom Linux device driver which will use the *mmap* system call to map DMA ring buffers directly to the user space, allowing the hardware to pull image data without consuming CPU cycles. The hardware will also feature a comprehensive power management system, providing an extra feature for our project group to work on. The group should design a custom PCB that routes a camera interface with decoupling capacitors and other power modules to handle voltage spikes and unstable power.

Schedule

We will use the schedule below as a rough guide for how the project should progress:

Dates	Phase
9/10 - 9/23	**Planning** We should figure out how large we can make the systolic array, which camera interface to use, and determine which FPGA board works within our budget and use case. After drafting the architecture, the proposal should be ready by 9/18.

9/24 - 10/6	**Design and Sourcing Parts** We should order the FPGA board, camera, and any power modules required to stabilize voltage. During this phase, we can also draft the PCB schematic, and finalize our proposal by 10/1.
10/7 - 10/25	**Prototype and PCB Design** Test the basic FPGA capabilities (approaching it like one would a Lab session). We should also finalize the routing PCB and order fabrication a week before the PCB deadline. We should also start researching how to write the Linux driver.
10/26 - 11/10	**Software and Testing** Implement the INT8 systolic array logic onto the FPGA using Verilog/VHDL. This will enable us to test the DMA ring buffer data transfer. While completing the software portion, we can also assemble the FPGA/PCB/Camera system by 11/8, and begin testing parts in unison.
11/11 - 11/25	**Integration Testing** After assembly and our software work, we should start ironing out the inevitable kinks in the fully integrated system. Comprehensively test the full inference pipeline, and monitor and resolve the voltage spikes, bounce, or other current problems that might appear when the systolic array is under load.
11/26 - 12/9	**Final Testing and Expo** Start benchmarking the bandwidth efficiency of our systolic array compared against any benchmarks we can find in similar items online, or against pure software solutions. Use benchmarks, and info on our procedure to prepare the final poster and demo for the expo.
12/10 - 12/11	**Documentation** Self-explanatory: Document our process, ensure our code is readable and reusable, and emphasize the importance/utility of our product.

Budget

The target budget for a four-person group should be towards $1000. The amounts below are incredibly rough estimates:

Item	Quantity	Cost
FPGA Development Board	1	~$550
Custom Camera PCB (plus any iterations needed to be fabricated)	3	$120
MIPI High-speed shutter camera	3	$150

Capacitors, and various ICs for power	x	$80
High-speed cables and other misc. hardware	x	$100
Total		**$1000**

Issues

The most significant challenges for this project include:

1. **Power Delivery:** Supplying a lot of current to the systolic array without inducing feedback, voltage drops, or ground bounce.
2. **Hardware Timing:** Closing timing on the INT8 systolic array across the FPGA layout.
3. **Software/Hardware Interface:** Developing a custom Linux driver that properly manages the DMA pipeline without crashing the CPU.

Target Scope

The ultimate goal of the project is to deliver an end-to-end edge inference pipeline, meant to be accelerated compared to a software-only solution, and ideally competitive with similar hardware out in the world. A successful project should demonstrate that our hardware-accelerated solution is more efficient and powerful than software-only or un-accelerated solutions.

Ultra-Low Latency Order Book Parser and Executor

Introduction

Quantitative finance is one of today's most competitive industries, where seemingly non-technical firms are engineering and managing novel hardware in the hopes that the performance jump over existing technology will ensure market advantage. It has already been proven that there is a performance gap between software-only algorithmic trading and hardware-accelerated traders. By designing a hybrid CPU-FPGA node designed to ingest ultra-high speed time-series financial data, we hope to replicate the hardware used in industry with a much smaller budget.

The hardware product for this project will parse network packets in real-time to maintain a local Limit Order Book and trigger trade execution, which mimics what firms do in-industry. The ultimate goal is to achieve nanosecond-scale tick-to-trade latency, demonstrating how chip-level networking can greatly optimize finance operations.

Details

The core of this system will be a device to parse AXI4-Stream, which is a high-speed protocol used to transfer data unidirectionally without using memory addresses. This parser exists as a SystemVerilog finite state machine, intended to decode UDP packets as they arrive. If we can achieve a clock frequency of 322 MHz, this parser can achieve a pipeline latency of 8 clock cycles, or about 24.8 nanoseconds. To ensure logical hardware execution, parameter updates will be handled by a custom Linux character driver using the *ioctl* system call over the AXI port.

Given the sensitivity of how we take in the data stream, power integrity will be emphasized in the project design. We should design a custom PCB featuring a step-down converter to reliably maintain any power input to 3.3V at 3A. The PCB's layout should be designed to minimize the high-current switch path and use solid ground planes to try and prevent electromagnetic interference.

Schedule

9/10 - 9/23	**Planning** Design the system architecture for the CPU-FPGA hybrid system. Prepare for the networking work by researching SFP+ interface requirements, and research which optical receivers are in-budget. Submit initial proposal by 9/18.
9/24 - 10/6	**Design and Sourcing Parts** Draft the PCB schematic, especially the step-down converter layout. Order SFP+ cages, optical transceivers, and power regulation components. Submit final proposal by 10/1.
10/7 - 10/25	**Prototype and PCB Design**

	Finalize the PCB layout, making sure the ground planes and other power regulation ICs effectively mitigate EMI, then order the PCB. We should also begin writing the custom Linux driver.
10/26 - 11/10	**Software and Testing** Develop and simulate the Verilog AXI4-Stream Parser FSM. Look into and implement how to handle clock domain crossings. Assemble the fabricated carrier PCB.
11/11 - 11/25	**Integration Testing** Integrate the PCB with the FPGA SoC, and test whether the 10Gbps market feed ingestion is successful and UDP packets are being parsed. Troubleshoot any EMI, parasitic inductance, or communication errors that may occur. Make sure the information stream is pristine!
11/26 - 12/9	**Final Testing and Expo** Verify that the top-of-book prices are correctly maintained in the local RAM memory. Benchmark the pipeline to confirm the ~25ns latency. Prepare the poster and demo for the 12/3 expo.
12/10 - 12/11	**Documentation** Document our process, make code readable and reusable, and finish our final report.

Budget

Item	Quantity	Cost
FPGA SoC Development Board	1	~$600
Impedance-controlled custom PCB	3	$150
10G optical transceivers (for Ethernet) and SFP+ cages	2	$80
Power regulation	x	$70
High-speed cables and other misc. hardware	x	$100
Total		**$1000**

Issues

The most significant engineering challenges at this time should be:

1. **Clock Domain Crossings:** Safely managing CDC and parallel data pipelining across the FPGA.

2. **Signal Integrity:** Preventing EMI and parasitic inductance originating from the power delivery from corrupting the high-speed differential Ethernet pairs.

Target Scope

This project should deliver a fully functional network-attached hardware accelerator for order book parsing and trade execution. A successful implementation should be able to process 10Gbps market feeds and accurately maintain top-of-book prices within local RAM. If benchmarks for industry hardware are available online, comparing against these benchmarks would make for some interesting data.

Would like this not to be shown to the class please. I don't mind snow cleaners.

Wireless Drone Battery Health Telemetry System

Moving from making drones as a hobbyist to running an actual company requires considering battery health. A battery that is discharged too far, draws unusually high current, or experiences a large voltage drop could cause reduced flight time or, in the worst case, cause the drone to lose power during operation. For the hobbyist, this could prove inconvenient, but for a company operating or testing a fleet of drones, this could prove detrimental. A method to monitor this information remotely could be much more useful.

Our project proposes a wireless battery health telemetry system for a drone. The goal is for the system to collect information about the battery while the drone is operating and periodically send packets of battery information to a base station. From here, a laptop or script could parse the information, allowing developers to create applications to analyze the battery packet information.

The clear customer for this system would be a drone company operating a fleet of drones. We want to create a reliable measurement tool for engineers to test with so they can monitor how the battery performs under different loads without needing to connect measurement devices to the drone physically. In a similar vein, the same idea could also be useful for maintenance, since unusual battery behavior could be explored before it becomes a larger problem.

For the course project, we would love to test with an actual flying drone, but that would not necessarily be required. Instead, we can focus on a mock-up drone power system built using a battery, motors, and other loads to simulate changes in power consumption. This makes testing much easier and safer while keeping things doable within the semester.

Details:

The main component the project would be built around would be the Holybro PM02D digital power module. The module is intended for drone power systems and would measure the battery voltage and current digitally through I2C. The module supports 2S through 12S battery systems and includes a 5.2-volt regulator capable of supplying up to 3 A.

The idea is to hook up the PM02D to a microcontroller on our own PCB. The responsibility for the current scope of the project is to have the microcontroller periodically make measurements and create a battery telemetry packet. Some packet information contained should include the following: packet number, timestamp, battery voltage, battery current, power consumption, estimated energy used, estimated battery percentage, peak current, and fault flags. These are subject to change based on stakeholder needs and professor input.

Would like this not to be shown to the class please. I don't mind snow cleaners.

Values such as voltage and current would come directly from the sensor. We would then take other values and do the calculations on the microcontroller. For instance, electrical power is calculated by multiplying the voltage by the current, and energy consumption is calculated by taking the integral over time.

At the end of the day, the PM02D is not a complete BMS. It provides pack voltage and current information, so I want to describe this project as a battery telemetry and monitoring system. I would like to explore more sensors to gather information for our data packets, such as adding a sensor for temperature.

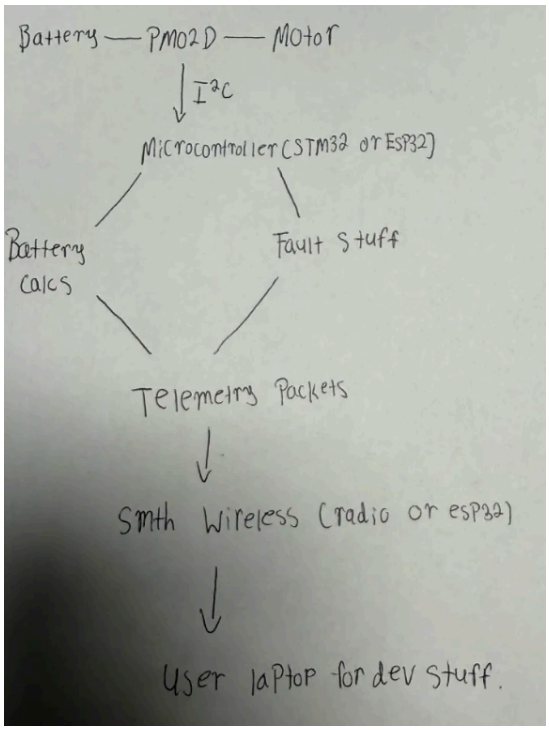


Figure 1: Starting chart

On initial inspection, the ESP32 processor family supports 2.4 GHz Wi-Fi and Bluetooth. It also provides UART, SPI, and I2C to provide a nice starting point for prototyping. The final radio choice depends on the desired range, power consumption, the amount of work required for the semester, and what stakeholders want.

Once the receiver gets the data, we can convert it to a JSON file and either create a CLI interface to read telemetry data or create something more comprehensive with Python. This allows for warning signs to be raised if something deviates from the expected behavior. Warning flags could be raised, and an interrupt could be transmitted instead of waiting for the next telemetry packet.

Would like this not to be shown to the class please. I don't mind snow cleaners.

All in all, this is where the project becomes more useful than simply making a wireless voltmeter. A drone company could run a test flight or bench test, save the telemetry, and compare battery performance between different batteries, motors, payloads, or operating conditions.

Budget:

Everything should fit within.

Parts:

- Holybro PM02D ~$25
- MCU / Dev boards ~ $40
- Batteries (should be in lab)
- Wireless modules (should be in lab)
- Motors (should be in lab)
- Temperature/other sensors (unsure at this time)
- PCB components (unsure at this time)
- PCB fab (unsure at this time)
- Drone might be provided by a groupmate

Implementation issues:

Main concern is reliable communication, potentially getting an actual drone to test on, safely testing the system, and ensuring we read correct data. An example problem is: if we get the drone, how do I verify in flight that we are truly reading the right data without finding a flying oscilloscope?

Additionally, if a drone cannot be acquired, we would test it with motors and sensors that would go into a drone.

Would like this not to be shown to the class please. I don't mind snow cleaners.

Schedule:

Dates	Phase
9/14 – 9/27	**Planning and System Requirements**
9/28 – 10/11	**Component Selection and Initial Design**
10/12 – 10/25	**Prototype Development and PCB Design**
10/26 – 11/8	**Wireless Communication and Software Development**
11/9 – 11/24	**PCB Integration and System Testing**
11/30 – 12/11	**Final Testing, Demo Preparation, and Project Completion**

Stretch goal:

Create an application that reads data from a fleet and gives users at the receiving end a pretty way to view said data.

Would like this not to be shown to the class please. I don't mind snow cleaners.

Snow cleaner

High-level Description.

My father recently had me shovel snow in the front yard, and it really reminded me what it meant to be working class. It was at that moment the gears in my head started to turn, and I thought to myself that I wanted nothing more than to use my degree in computer engineering to make a little robot clear the snow out for me.

I propose to create a small autonomous robot designed to clear snow from a driveway or sidewalk. The user would define an area from their laptop that needs to be cleared, and the robot would travel through that area while pushing or torching snow out of the way. Ideally, this would allow me to remain indoors while the robot experiences the wonders of winter.

Details
The robot would use a microcontroller and a similar program as we use in the lab. We need to drive two motors and control a snow-removal mechanism. Some sort of distance sensors or keyboard inputs from the laptop would be helpful to detect and avoid obstacles. Wheel encoders and an IMU should also help follow intended paths as well.

The goal for the semester is to not build a full blown snow plower from scratch. Instead, a small prototype could be built to clear the early snows that might appear in the Michigan winter. The aim is to clear the snow without using an unreasonable amount of power.

Issues
Problems I could think of would be mechanical problems. For starters, we need to ensure that the vehicle can move with a certain load. We need to ensure that it is waterproof, can detect obstacles, and move the snow without taking up an ungodly amount of power.

Testing
We can successfully clear out a predefined rectangular area without leaving the area or running into an obstacle. Testing definitely should be a controlled environment we create to see how well we are cleaning out snow before we put it in the Michigan winters.

HW #0 - Project proposals

Traction Inverter based on Silicon Carbide MOSFETs for a Formula SAE Electric Vehicle

Background

Formula SAE or Formula Student is an internationally recognized category of small open-wheel race cars built by college students across the world. Students form teams within their schools and build a racecar according to an extensive rulebook, optimizing for the conditions and goals of the competition. The main North American competiton sanction by SAE International is held every year at Michigan International Speedway in Brooklyn, MI. University of Michigan's team, MRacing FSAE, is attending this competition every year.



MR26: MRacing's 2026 vehicle

The competition has both electric and combustion categories, with the former recently overtaking the latter in the number of teams competing and prestige. MRacing FSAE competes in the electric category.

The vehicle has 4 synchronous AC motors located in the wheel hubs, thus needing 4 traction inverters housed within the vehicle. The project is proposing to build a 3-phase AC traction inverter for driving the motors on the MR27, MRacing's upcoming 2027 vehicle.

Details

There are 4 traction inverters in the vehicle; this makes their weight a significant part of the overall vehicle mass (a very sensitive parameter). Off-the-shelf inverters currently used by MRacing are significantly more massive than newer options on the market, while costing tens of thousands of dollars. Additionally, the inverters currently used are highly susceptible to EMI and have poor overall fault handling.

This project aims to cut the inverter weight, drastically reduce the cost, and improve reliability and efficiency.

The Design

The design is going to be based around a pre-packaged Infineon 6-pack Silicon Carbide MOSFET module (Part #: FS02MR12A8MA2B). Compared to the IGBTs used in the MR26's inverters, SiC MOSFETs can achieve higher switching frequencies and lower losses in the power range that the task demands.

A decrease in volume and weight is going to be achieved by a gate driver module directly on top of the MOSFET module, a tiny control module (compared to the off-the-shelf inverters), and more volume-efficient DC-link capacitors.

The main challenge of the project is going to be designing the high-speed control loop to be able to drive a 10-pole IPM AC motor at 20k+ RPM with field weakening capabilities.

Schedule

Dates	**Milestone**
9/13	Main parts locked - MOSFETs, gate drivers, microcontroller, current sense, DC-link capacitors
9/25	PCB design ready and ordered; parts ordered. Firmware work starting while waiting for shipping.
10/9	PCBs have arrived (hopefully), assembly starting. Firmware skeleton ready.
10/15	First encoder read complete
10/30	First motor spin complete
Per facility availability	Dyno testing and tuning

Budget

Item	**Estimate**
6xSiC MOSFET Module	$700-800 (sponsored by MRacing FSAE)

Microcontroller, gate driver, other ICs	$300
Passives	$50
Current sense, capacitors, busbars	$100
Misc	$100
Total	$550

Main hurdles

- Testing facilities:
    - Need a 600V power supply capable of supplying 40kW+ of power
    - Need a 20k RPM and 35+ Nm capable dynamometer.
- Safety implications
    - The project is working with high voltage at high power. Appropriate design and testing measures need to be taken for the safety of the students and the hardware.
- EMI
    - Electromagnetic interference can become an issue for control electronics. Appropriate shielding and grounding techniques need to be implemented

Stretch firmware goal

Achieve fully stable operation with field weakening in the entire torque/speed range.

- Requires precise tuning and calibration of the control loop.

Optical ground speed sensor for a Formula SAE Electric Vehicle

Background

Formula SAE or Formula Student is an internationally recognized category of small open-wheel race cars built by college students across the world. Students form teams within their schools and build a racecar according to an extensive rulebook, optimizing for the conditions and goals of the competition. The main North American competiton sanction by SAE International is held every year at Michigan International Speedway in Brooklyn, MI. University of Michigan's team, MRacing FSAE, is attending this competition every year.



MR26: MRacing's 2026 vehicle

The competition has both electric and combustion categories, with the former recently overtaking the latter in the number of teams competing and prestige. MRacing FSAE competes in the electric category.

The vehicle has 4 synchronous AC motors located in the wheel hubs, thus having the capability of spinning all 4 tires at the same time, which leads to a loss of a true ground speed reference. The project aims to restore that reference with an optical ground speed sensor, improving the vehicle control algorithms drastically.

Details

Since the vehicle can actuate all 4 wheels independently, there are various control techniques that allow the team to extract significantly more pace out of the vehicle compared to its mechanical baseline capabilities. These techniques include torque vectoring, traction control, launch control, yaw rate control, etc.

For all of these algorithms, the knowledge of precise ground speed is essential for various feedback loops. And since all 4 wheels are driven, none of them are free-rolling; this means that under acceleration, more often than not, all 4 wheels spin faster than the ground is moving underneath them (slipping). This makes it impossible to directly derive ground speed based on wheel speed data. Currently, MRacing FSAE is using a sensor fusion algorithm that combines wheel speeds and IMU data to estimate ground speed; however, this algorithm is not as robust as a direct sensor reading could be. The project aims to build a sensor that will solve this problem by computing ground speed using computer vision.

The idea is to point a high-speed camera at the ground and compute the displacement in the asphalt's pattern, thus computing both V_x and V_y in the car's reference frame.

Proposal 1



Figure 1: Ray, a similar wrist-worn fitness tracker, which inspired Vitanklet

Vitanklet (VITal + ANKLET)

Active and resting heart rate, body temperature, and steps are integral **vitals** of metabolism, and thus health and fitness. There are wearable devices that regularly record them, but they also try to do too much, have to be charged frequently, and are bulky.

Therefore, Vitanklet is proposed: an **anklet**-worn fitness tracker. Vitanklet would be grounded, recording only one's active and resting heart rate, body temperature, and steps; solar-powered; and in a tasteful form factor.

By being ankle-worn instead of wrist-worn, Vitanklet could be more accurate in its heart-rate measurement and pedometry: one study found photoplethysmography (PPG)

produced more accurate heart-rate measurement on the ankle than on the wrist, and walking requires leg–not arm–swing.

Details

An integrated PPG sensor would be used for heart-rate measurement, a chest-worn CORE 2 thermal sensor would be used for thermometry, a low-power three-axis accelerometer would be used for pedometry, a solar cell; solar-charging capable power-management IC with buck boost and MPPT controls; and rechargeable battery would be used for power, a BLE IC would be used for wireless communication with the chest-worn thermal sensor and companion device, and a flash memory IC would be used for offline data storage.

Schedule

The dates selected are mostly “lines in the sand” and will almost certainly not be exact, though they should be close.

Sept. 10–23	*Planning* In the two weeks after group formation, start sensor research: PPG sensor heart-rate measurement, thermal sensor interfacing, and accelerometer pedometry. Do something similar for solar charging, battery, and flash memory. Begin estimation of MCU requirements, and hopefully identify an MCU. Write the draft proposal by the 18th.
Sept. 24–Oct. 6	*Design and part selection* Once the proposal meetings are done, order the sensors and the selected MCU. Bring up a minimally-fashioned prototype (ignoring form-factor requirements). Submit the final proposal by Oct. 1. All parts we *think* we need for the prototype will have been ordered by the 4th.
Oct. 7–25	*Breadboard prototype and PCBA design* Get solar charging, battery, and flash memory working. Order PCBAs by the 26th.

Oct. 26–Nov. 10	*Software development and testing on prototype* While waiting for the PCBAs to come back, further develop software. Get solar charging and battery 100% working on the prototype. Have all functionality written and tested on the prototype by Oct. 10. Identify solutions to all non-electrical issues (chassis, band, etc.) and get all things ordered on that front.
Nov. 11–25	*Software development and testing on PCBA-based system* Testing PCBA. Once basics working on PCBA, testing will be done to ensure system integrity. Accuracy of heart-rate measurement and pedometry will be tested. Bugs will be found and fixed. PCBA issues resolved. If needed, PCBA revisions will be made. Any additional features or stretch goals will be researched and decided upon by this time.
Nov 26.–Dec. 9	*Final testing, additional features, poster and expo* Targeting project completion Dec. 3. At this point we'll get our poster done and start on the final report. Continue to fix any remaining minor bugs.
Dec. 10–11	*Finish final report*

Figure 2

Budget

While a realistic budget is difficult at this time, the numbers below are a reasonable first pass. PCBA manufacturing costs are likely underestimated. With a team of 5 we would be under budget. Several Vitanklets would be built (we would wear them at the expo and a screen would display our live heart-rate and steps), but only one Vitanklet would have thermometry at a time.

What	Quantity	Cost per	Cost total
Nordic nRF52832-QFAA	5	$8.00	$40.00
CORE 2 thermal sensor	1	$309.95	$309.95
PCBA	5	$30.00	$150.00
Leather band	5	$25.00	$100.00

Chassis	5	$20.00	$80.00
Misc. parts	5	$50.00	$250.00
Shipping	1	$75.00	$75.00
Total			$1,004.95

Figure 3

Issues

The largest concerns at this time are:

1. Desired design originality (I'm quite a fan of the design in Figure 1)
2. Desired form factor
3. Budget

The first issue seems minor (Vitanklet is noncommercial; its intended users are only friends and family of the developers as well as the developers themselves), but it's important to disclose. The second issue is the primary challenge, but seems to be only that—not a showstopper. The third issue is corollary: the desired form-factor means the PCBA and other parts will be more expensive. This seems manageable, however.

Additional Stretch Goal

The desired form factor is already a stretch goal; if time allows we could develop our own integrated PPG sensor and/or thermal sensor.

Proposal 2



Figure 1: AccuVein© Vein Finder, a similar handheld device, which inspired Proposal 2

Low-Cost Vein Finding and Blood Oxygen (SpO_2) Measurement

Low-cost vein finding is often difficult for healthcare professionals, and SpO_2 measurement devices are relatively inaccurate. Fortunately, near-infrared (NIR) imaging can find veins, and dual-wavelength photoplethysmography (PPG) using NIR and red-light sensing can measure SpO_2.

Therefore, Proposal 2 is proposed: a handheld vein finder and pulse oximeter. By being handheld instead of finger-clipped, the device would be swept across the body, taking many samples of SpO_2 across the body, instead of only on a digit.

Details

An integrated dual-wavelength PPG sensor would be used for SpO_2 measurement, an NIR LED array and NIR-sensitive camera would be used for vein imaging, and a solar cell; solar-charging capable power-management IC with buck boost and MPPT controls; and rechargeable battery would be used for power.

Schedule

The dates selected are mostly “lines in the sand” and will almost certainly not be exact, though they should be close.

Sept. 10–23	*Planning* In the two weeks after group formation, start sensor research: Dual-wavelength PPG sensor SpO_2 measurement and NIR imaging. Do something similar for solar charging and battery. Begin estimation of MCU requirements, and hopefully identify an MCU. Write the draft proposal by the 18th.
Sept. 24–Oct. 6	*Design and part selection* Once the proposal meetings are done, order the sensors and the selected MCU. Bring up a minimally-fashioned prototype (ignoring form-factor requirements). Submit the final proposal by Oct. 1. All parts we *think* we need for the prototype will have been ordered by the 4th.
Oct. 7–25	*Breadboard prototype and PCB design* Get solar charging and battery working. Order PCBs by the 26th.
Oct. 26–Nov. 10	*Software development and testing on prototype* While waiting for the PCBs to come back, further develop software. Get solar charging and battery 100% working on the prototype. Have all functionality written and tested on the prototype by Oct. 10. Identify solutions to all non-electrical issues (chassis, etc.) and get all things ordered on that front.

Nov. 11–25	*Software development and testing on PCB-based system* Testing PCB. Once basics working on PCB, testing will be done to ensure system integrity. Accuracy of SpO_2 measurement and vein finding will be tested. Bugs will be found and fixed. PCB issues resolved. If needed, PCB revisions will be made. Any additional features or stretch goals will be researched and decided upon by this time.
Nov 26.–Dec. 9	*Final testing, additional features, poster and expo* Targeting project completion Dec. 3. At this point we'll get our poster done and start on the final report. Continue to fix any remaining minor bugs.
Dec. 10–11	*Finish final report*

Figure 2

Budget

While a realistic budget is difficult at this time, the numbers below are a reasonable first pass. PCB manufacturing costs are likely underestimated. With a team of 5 we would be under budget. Two devices would be built, so that we could demo more than one device at a time.

What	Quantity	Cost per	Cost total
Nordic nRF52832-QFAA	2	$8.00	$16.00
Dual-wavelength PPG sensor	2	$29.75	$59.50
PCB	2	$15.00	$30.00
NIR LED array	2	$25.00	$50.00
Chassis	2	$20.00	$40.00
Misc. parts	2	$50.00	$100.00
Shipping	1	$75.00	$75.00
Total			$370.50

Figure 3

Issues

The largest concerns at this time are:

4. Sensor data-processing algorithms
5. Vein projection

The first issue seems major (Vein imaging and dual-wavelength PPG would be especially difficult), but consulting LLMs and hard work could keep the show going. The second issue is also relatively major, but the previous solution applies.

Additional Stretch Goal

The complexity of the sensors as well as vein projection are already stretch goals; if time allows we could develop our own integrated dual-wavelength PPG sensor.

Autonomous Plant-Watering Robot

Tiffany Cheung, EECS 473

Houseplants and small gardens fail for a simple, universal reason: nobody waters them consistently. Improper watering, whether too much, too little, or simply inconsistent, is responsible for the large majority of houseplant deaths[1]. This is a common experience, as one survey found the average plant parent has killed seven houseplants[2], and 32 percent of people who took up plant-keeping during the pandemic had already quit within about a year, with roughly a third of losses blamed on overwatering and another third on underwatering in about equal measure[3]. Existing “smart” solutions (moisture-sensing stakes, timer-based drip systems) are either single-plant, non-mobile, or unable to distinguish between different plants with different watering needs, and finding a plant sitter. At the same time, traveling is itself a commonly cited source of plant-keeping stress[4].

Not only does plant neglect harm the plants themselves, but it also harms the owners' mental well-being. Surveys report that around 80 percent of plant owners say keeping houseplants has a positive effect on their mental health, and about half of new plant owners took up the hobby specifically to relieve stress[5]. However, that benefit is conditional on the plant staying healthy: multiple studies found that neglected, dying plants can leave people feeling worse than having no plants at all, and researchers recommend removing unhealthy plants rather than leaving them visible[6]. A robot that reliably keeps plants healthy is therefore not just a convenience; it protects a real, measured wellbeing benefit that a forgotten, dying plant would otherwise undo, while also solving the very common problem of plants dying from inconsistent watering.

---

[1] “Overwatering vs. Underwatering: Why Understanding Your Plant’s Needs is Key to Helping Them Thrive,” Early’s Garden Center.
https://www.earlysgarden.com/resources/gardening-tips-a-tricks/item/646-overwatering-vs-underwatering-why-understanding-your-plant-s-needs-is-key-to-helping-them-thrive

[2] "This Is How Many Houseplants the Average Plant Parent Has Killed," Apartment Therapy.
https://www.apartmenttherapy.com/dead-houseplants-average-37076429

[3] "Trees.com Survey Finds Half of New Plant Owners Started Plant-Keeping to Relieve Stress During the Pandemic, but 32 Percent Have Already Quit," PRWeb.
https://www.prweb.com/releases/trees-com-survey-finds-half-of-new-plant-owners-started-plant-keeping-to-relieve-stress-during-the-pandemic-but-32-percent-have-already-quit-824256904.html

[4] "This Is How Many Houseplants the Average Plant Parent Has Killed," Apartment Therapy (plant-sitter/travel-stress statistic).
https://www.apartmenttherapy.com/dead-houseplants-average-37076429

[5] "If You’re a Plant Person, You’re Not Alone," New York Botanical Garden Plant Talk.
https://www.nybg.org/planttalk/if-youre-a-plant-person-youre-not-alone/

[6] "Attractive Houseplants Give Biggest Well-Being Boost," University of Reading.
https://www.reading.ac.uk/news/2022/Research-News/Attractive-houseplants-give-biggest-wellbeing-boost

Details

As aforementioned, existing “smart” plant watering techniques are stationary and aren’t specific to the plant’s needs. Because there are many factors in how to properly keep plants healthy, I propose an autonomous robot carrying a water reservoir, a metered pump, and a downward/forward-facing camera. A base station (a Raspberry Pi or similar single-board computer) handles AprilTag detection and path planning; a separate microcontroller handles real-time motor control and pump actuation. Each pot in the deployment gets a printed, laminated AprilTag and a low-cost soil moisture probe wired or wirelessly linked back to the base station. The owner registers each tag ID once (plant type, watering profile), and from then on the robot runs its rounds autonomously, navigating to a pot by driving toward the tag using simple proportional control on the tag's detected pose.

Plant identification and dosing are handled without any hard computer-vision or mechanical problems. Rather than requireing the owner to know and enter a precise watering volume, registration works by species: when a tag is registered, the owner picks the plant’s species from a small built-in list (roughly 10-15 common houseplants, covering most cases), and the robot looks up a sensible default does and frequency from a species reference table compiled ahead of time from exisitn ghouseplant care guides. The owner can manually override the dose/frequency for an unusual plant or a personal preference. The AprilTag ID is simply a lookup key into a plant profile table (species, dose, frequency) set once at registration, and moisture sensing uses a fixed, pre-placed probe per pot rather than having the robot insert anything, sidestepping the pot-size and soil-depth variability that would otherwise make automated probe insertion difficult. A combined temperature/humidity sensor is read at the same time as the moisture check, riding on the same per-pot wiring and tag-keyed profile lookup, and is used to flag frost/heat risk for outdoor pots and unusually dry or humid conditions rather than driving any actuationWatering history, tank/battery level, calculated plant health, and any failures (tag not found, low tank) are reported to the owner over a small local web dashboard hosted on the base station.

Schedule

Weeks 1–2: Finalize system architecture (SBC for AprilTag detection/navigation, MCU for motor + pump control). Select and order chassis/motor kit, camera, moisture sensors, pump, and processor dev boards. Print and laminate test AprilTags. Draft proposal.
Weeks 3–4: Get AprilTag detection running on the SBC against a static tag (library-based, no navigation yet). Bring up the MCU on a devkit driving a motor and the pump in isolation. Begin PCB layout for the motor-driver / pump-driver board. Order PCB by end of week 4 to absorb the ~10-day fab delay.
Weeks 5–6: While waiting on the PCB, get closed-loop “drive toward a tag” working on the breadboard prototype. Assemble and bring up the PCB once it arrives; get basic motor/pump control working on real hardware.
Weeks 7–9: Integrate multiple tags/plant profiles, wire in moisture sensors and a combined temperature/humidity sensor at each pot (read at the same time as moisture, over the same wiring), and get the full loop working: drive to tag → read that plant's moisture sensor → dose if needed → log the event. Build the local web dashboard.

Weeks 10–11: Multi-plant testing across different pot sizes/spacing, tune navigation and dosing, add failure-notification logic. Fix bugs found during repeated test runs.
Weeks 12–13: Final polish, poster, stretch goals if time allows, final report.

Budget

Rough first-pass budget (team of 4, $250/student = $1000 team budget):

Item	Qty	Cost per	Cost total
SBC (Raspberry Pi or similar) + camera	1	$80.00	$80.00
Motor driver / drivetrain kit + motors	1	$60.00	$60.00
MCU dev board	2	$15.00	$30.00
Custom PCB (motor + pump driver)	5	$15.00	$75.00
Soil moisture sensors	6	$6.00	$36.00
Temperature/humidity sensors	6	$4.00	$24.00
Small metered pump + tubing	2	$20.00	$40.00
Water reservoir + chassis materials	1	$50.00	$50.00
Battery + charging	1	$40.00	$40.00
AprilTags (printed/laminated) + misc mounting hardware	1	$20.00	$20.00
Misc parts / connectors / wiring	1	$60.00	$60.00
Shipping	1	$75.00	$75.00
		Total	**$590.00**

This leaves meaningful headroom under the $1000 team budget for a second pump or a spare SBC in case of failure, without needing the full amount.

Issues

The largest concerns at this time are

- Real-time AprilTag detection on modest hardware
- Splitting perception and real-time control across two processors
- Depth/insertion of moisture sensing
- Plant-type identification
- Water spillage near electronics

Issue 1 seems manageable because detection runs on an SBC rather than the microcontroller, using the well-tested, actively maintained open-source AprilTag library rather than custom vision code[7]. Issue 2 follows a standard robotics pattern (SBC for planning, MCU for motor timing) connected over a simple serial/UART protocol, which keeps the two halves independently testable.

---

[7] AprilRobotics, "apriltag: A robust and flexible visual fiducial system."
https://github.com/AprilRobotics/apriltag

Issue 3 is avoided entirely by pre-placing a fixed moisture sensor in each pot rather than having the robot insert a probe, a deliberate scope decision rather than an oversight. Issue 4 is avoided by using the AprilTag ID as a lookup key into a plant profile table rather than attempting visual plant classification.

Issue 5 is addressed by keeping the pump and reservoir physically separated from the electronics bay, with tubing routing tested well before the Expo. No major safety or special-facility issues are anticipated; the system runs on battery power at low voltage and small water volumes, and each issue seems manageable, though it is likely something unforeseen at this time will be the real challenge.

Additional Stretch Goal

While the autonomous robot is already a stretch goal, if time allows, we will see if the moisture sensor could be on an arm of the robot. This way, the probe can be inserted into the pot with current-sensing “touch detection,” similar to how 3D printers do bed leveling.

Tactile Communication Wearable for Deafblind Users

Tiffany Cheung, EECS 473

Most assistive communication technology assumes at least one of sight or hearing still works: screen readers and audio feedback for blind users, captions and visual alerts for deaf users. There are an estimated 70,000 deafblind people in the United States[8], and communication for this population, by contrast, still mostly relies on tactile fingerspelling or Pro-Tactile ASL, a touch-based language performed hand-to-hand or hand-to-body with another person physically present[9]. When no interpreter or communication partner trained in tactile methods is around, there is very little affordable technology bridging that gap.

This proposal is a hand-worn device that converts text typed by any untrained communication partner (on a phone app) into a tactile pulse pattern the wearer can feel and read, without requiring a trained interpreter to be present. It targets a real, underserved gap rather than a reskin of an existing product: most consumer assistive tech splits cleanly into “for blind users” or “for deaf users,” and there is very little affordable, non-interpreter-dependent tech serving people who are both. It is also a strong technical showcase: wireless communication, real-time pattern generation, and a genuine human-factors design problem (what pulse encoding is actually learnable and readable), built on a hardware base that is already proven to work.

Details

The device is a glove that has an output of vibration, as well as sending data from the user to the other person in the conversation. A microcontroller on the glove or wristband drives a small array of vibration motors, pulsing out a simple, learnable pattern per letter or common phrase, sent wirelessly (Bluetooth) from a companion phone app where the communication partner types. To make the conversation two-way, a small set of physical buttons on the glove/wristband lets the wearer select from a short list of pre-set replies (“yes,” “no,” “wait,” “help,” “repeat that”), which are sent back to the partner's phone as text or a simple tone — a low-risk addition since it only needs a few momentary switches and simple button-press logic, not new sensing or recognition. The finished device is a lightweight glove or wristband with embedded motors, reply buttons, and a small battery, paired with a minimal phone app for message entry — usable by anyone as a communication partner, with no special training or equipment on their end.

---

[8] "Helen Keller National Center for Deaf-Blind Youths and Adults," Wikipedia. https://en.wikipedia.org/wiki/Helen_Keller_National_Center_for_Deaf-Blind_Youths_and_Adults

[9] "Protactile," Wikipedia. https://en.wikipedia.org/wiki/Protactile

Schedule

- Weeks 1–2: Finalize motor layout (glove vs. wristband, number of motors) and pulse-encoding approach for the MVP (single/few-motor Morse-like pulses). Select microcontroller, Bluetooth module, and vibration motors. Order parts. Draft proposal.
- Weeks 3–4: Bring up a single vibration motor on a breadboard, driven by the MCU, and confirm pulse timing is perceptible and controllable. Begin PCB layout for the motor driver + MCU + Bluetooth board. Order PCB by end of week 4 to absorb the ~10-day fab delay.
- Weeks 5–6: While waiting on the PCB, build the phone app's text-entry-to-Bluetooth pipeline and get the full text → encoded pulse sequence → motor output loop working on the breadboard. Assemble and bring up the PCB once it arrives.
- Weeks 7–9: Mount the driver board, motors, and a small set of reply buttons into a glove/wristband form factor. Wire buttons to the MCU and confirm button presses reliably send the corresponding pre-set reply back to the phone app. Refine pulse timing/spacing based on early informal testing. Add a message-history/repeat button.
- Weeks 10–11: Run structured accuracy/speed tests with blindfolded, earplugged volunteers after a short training period. Iterate on encoding based on results.
- Weeks 12–13: Final polish, poster, stretch goal (expanded multi-motor pattern language) if time allows. Finish final report.

Budget

Rough first-pass budget (team of 4, $250/student = $1000 team budget):

Item	Qty	Cost per	Cost total
MCU dev board w/ Bluetooth	2	$15.00	$30.00
Vibration motors (assorted, for testing layouts)	20	$1.50	$30.00
Motor driver ICs	4	$3.00	$12.00
Custom PCB (driver + MCU carrier)	5	$15.00	$75.00
Glove / wristband material + fabric motor mounts	3	$15.00	$45.00
Momentary push buttons (reply set	15	$1.00	$15.00
Battery + charging circuit	2	$12.00	$24.00
Blindfold + earplugs (testing supplies)	1	$10.00	$10.00
Misc wiring/connectors	1	$40.00	$40.00
Shipping	1	$50.00	$50.00
		Total	**$331.00**

This leaves substantial headroom under the $1000 team budget, most of which could go toward a second, richer motor array if the multi-motor stretch goal is pursued.

Issues

The largest concerns at this time are

- Designing a learnable, readable pulse encoding
- No access to actual Deafblind testers in a class setting
- Motor placement and skin sensitivity vary by person
- Reading speed vs. learnability tradeoff

Issue 1 is the largest scope of the project; the proposal scopes to a single working encoding (simple Morse-like pulses on one or two motors) as the core deliverable, with a richer multi-motor spatial pattern language framed as a stretch goal rather than a promised outcome. Issue 2 is addressed directly by using blindfolded/earplugged sighted-and-hearing volunteers to simulate the sensory channel, a legitimate and commonly used testing method for this class of device. Issue 3 is addressed by choosing a wrist/palm form factor and making motor spacing adjustable during testing. Issue 4 will be measured directly during testing rather than assumed, and the encoding tuned based on real results. No major safety issues are anticipated; low-voltage vibration motors worn against skin are benign, and the device is fully passive from the wearer's side.

Additional Stretch Goal

The reply buttons cover the most common, time-critical responses, but they are limited to a fixed phrase set. If time allows, we will explore ricer return-channel options, roughly in order of difficulty: first, a tap-pattern input mirroring the existing pulse encoding, the wearer taps out their own Mosre-like pulses on a small touch surface, which get decoded into free text on the partner’s phone. This resuses the same encoding/decofing logic already built for the outgoing channel, just reversed, so it is the more tractable of the two stretch options. The other option uses flex sensors and IMU to let the wearer fingerspell, with teh microcontroller recognizing letter shapes and sending them as text, this ia harder, open-ended gesture-recognition problem, and would only be attempted if the core system and the tap-pattern input are both solid well ahead of the Expo.

Standalone Audio Sampler and Beat Sequencer

Intro and Need

Professional beatmaking hardware tends to be sold at a high price and is made as a closed system. Usual hardware devices like the Akai MPCs cost anywhere from $500-$4000 and entry level devices like Korg Volca Sampler 2 (~$150) do not have the “standalone” features of recording sounds so they require laptops or smartphones as a tether. Additionally, most beatmaking hardware is designed to be a closed system meaning you cannot modify effects or algorithms shipped with the device.

The global market for hardware-based production equipment is expanding rapidly with a compound annual growth rate of 7.88% from 2025 to 2032[1]. The major driver of this growth is an increased demand for live performance hardware and on-the-go music production tools to showcase creativity in an ever growing AI landscape[2]. Software based production environments have numerous options which may affect focus when creating music[3].

This project tries to bring portability and tactileness of the Akai Professional MPC Sample while lowering the cost. It will create a “scoped” beat production environment by limiting features but allowing for swapping out/editing of effects through a programming interface uploaded on the device. This creates a device that is approachable for beginners while offering back-door functionality for advanced sound engineers.

Envisioned Product Details

To achieve a competitive product in the current market, a few functional requirements such as IO connectivity, a microphone, a speaker, sequencing of samples, modifications of samples via DSP techniques, and a battery life for live performance sets needs to be achieved. These implementations will likely be similar to existing products such that a user is able to quickly transition from their existing product.

All of these functions would need to be completed on device so a very capable microcontroller processor will be needed to handle digital signal processing, memory read/writes, and IO processes simultaneously.

To take this product to the next level, some more functions like microSD importing and exporting of sound effects will need to be added and a designed programming interface for editing functions on device. This idea will culminate into a device that can sample, modify, and then play beats by itself.

---

1. https://www.360iresearch.com/library/intelligence/music-production-equipment
2. https://www.fortunebusinessinsights.com/musical-instrument-market-108706
3. https://www.sweetwater.com/insync/beyond-the-daw-dawless-jamming-and-the-tactility-renaissance/

Rough Schedule

9/10-9/25	***Understanding and Planning*** The team should research existing products and understand: (a) pros and cons of samplers and beat sequencers, (b) generally what users expect in samplers and beat sequencers, and (c) the basic usage of a sampler and a beat sequencer. Decide on MCU, IO connectivity, and how the programming interface works.
9/26-10/8	***Prototyping I build*** Choose a test board and develop the demonstrable functional requirements for sampling and beat sequencing.
10/9-10/25	***Prototype II and PCB Design*** Have a functioning prototype that has all features from the planning stage included and implemented (data pipeline from Prototype I and IO connectivity included). Order PCB by 10/26.
10/26-11/10	***Software Development and testing Prototype II*** While waiting for the PCBs to arrive, transition the prototype code into a properly interfaced codebase and start working on the programming interface for the device.
11/11-11/25	***Software Development and PCB testing*** Develop a comprehensive testing procedure for the product. Start testing for the PCB using testing procedure. Fix any bugs that occur. Conduct PCB revisions if necessary. Make decisions on additional features and stretch goals.
11/26-12/9	***Final testing, additional features, project expo*** Project completion 12/3. Get the poster completed and start the final report. Fix software only bugs.
12/10-12/11	**Submit Final Report**

Rough Budget

What	Quantity	Cost per	Cost total
STM32H7 MCU	**5**	**$18.00**	**$90.00**
PCB	**5**	**$15.00**	**$75.00**
Audio Codec (ADC/DAC)	**5**	**$8.00**	**$40.00**
Tactile Sequencing Pads	**80**	**$1.00**	**$80.00**
Li-Ion Battery & Power IC	**5**	**$15.00**	**$75.00**
MEMS Microphone & Speaker	**5**	**$6.00**	**$30.00**
1/4" TRS Inputs	**5**	**$1.50**	**$7.50**
1/4" TRS Outputs	**5**	**$3.00**	**$15.00**
1/8" (3.5mm) TRS Jack	**5**	**$0.50**	**$2.50**

5-Pin DIN Jacks	**5**	**$2.00**	**$10.00**
1/8" (3.5mm) TS Jacks	**5**	**$2.00**	**$10.00**
MicroSD Socket	**5**	**$2.00**	**$10.00**
Enclosure & Knobs	**5**	**$20.00**	**$100.00**
Screen (maybe)	**5**	**$16.00**	**$80.00**
Misc parts	**1**	**$75.00**	**$75.00**
Shipping	**1**	**$60.00**	**$60.00**
Total			**$760.00**

Issues

The largest concerns for what needs to be thought about:

1. Designing the programming interface to be easy-but-with depth for the user
2. What effects should be included in base product (non programmed)
3. Sequencing the beats
4. Are DSP algorithms sourced or written from scratch?
5. Resolution and quality of the sound input and output

Issue 1 is the differentiator between the market and this project. It needs to be executed well so we know we are providing value to the user. *Issue 2* can most likely pull from existing products. *Issue 3 & 4* are figure-outable but will take significant time. *Issue 5* will likely determine the price point of the product. We want to make sure there is a balance between quality and cost.

Stretch Goals

Some cool ideas that may not make it to production in a semester:

1. Interchangeable IO ports for extra modifications
2. MIDI connection to Laptops and/or Smartphones

Evaluation

The evaluation of the audio sampler will focus on validating its reliability as a distraction-free performance tool to ensure it meets the specific real-world needs described in the project's introduction. The primary measure of success will be testing whether a user can easily sample live audio, sequence a multi-track beat, and perform a live set (5 hours) continuously on battery power without encountering software crashes or perceptible audio latency. The user should also understand how to program the board without the assistance of a team member.

Baseball Pitch Analysis

Intro and Need
Professional baseball analytics rely heavily on expensive, external tracking systems like the Rapsodo PRO 2.0 or TrackMan radar systems, which cost upwards of $3,500 to $20,000 per unit[1]. This creates a massive barrier to entry for amateur leagues and collegiate athletic programs. Furthermore, external tracking can suffer from line-of-sight occlusion and struggles to capture the micro-dynamics of a ball's rotation (true spin rate, gyro degree, and exact release velocity) without relying on predictive algorithms.

This project aims to address the high overhead of baseball pitch design analytics by moving the tracking hardware inside the projectile itself. By embedding high-G inertial measurement units and wireless transmitters inside a ruggedized core, the device will capture spin rate, angular velocity, and impact force directly from the source. This creates an accessible, highly accurate biomechanical analysis tool that costs a fraction of external optical systems and can be used in any environment, without needing a dedicated radar alignment.

Envisioned Product Details
To achieve a competitive product, the primary functional requirement is hardware survivability. The internal electronics must survive extreme impact forces while capturing the necessary projectile data and transmitting wirelessly at a high data rate.

The device must also fit the baseball regulation meaning that there will likely be no exposed ports. The device should employ aggressive low-power sleep modes to preserve battery life between pitches. The core processing will require a high-speed, power-efficient microcontroller to rapidly pull data from the IMU over an SPI bus and write it to onboard flash memory before the physical impact with the catcher's mitt occurs.

To ensure the PCB traces and IC solder joints do not shear upon impact, the internal PCB assembly will be suspended in a specialized potting compound or shock-absorbent resin. There is strong industry precedent for this engineering approach: Kookaburra and tech firm SportCor successfully developed a "SmartBall" for cricket, proving that embedded microchips can survive repeated impacts upwards of 150 km/h (93 mph) without sustaining physical damage or interrupting telemetry transmission[2].

---

1. https://rapsodo.com/products/pro-2-ball-flight-monitor
2. https://indianexpress.com/article/sports/cricket/smartball-a-microchipped-cricket-ball-that-may-soon-hit-bbl-5899322/

Rough Schedule

9/10-9/25	***Understanding and Planning*** The team should research: (a) the physical constraints, center of mass, and weight distribution of regulation baseballs, (b) the specs of high-G accelerometers, and (c) potting and shock-absorption techniques for PCBs. The team will decide on the MCU and wireless protocol.
9/26-10/8	***Prototyping I build***

	Choose a development board and develop the demonstrable functional requirements for high-speed data logging (i.e. reading from an IMU and writing to SPI Flash memory) and BLE transmission.
10/9-10/25	***Prototype II and PCB Design*** Have a functioning prototype that has all features from the planning stage included (sensor polling, memory management, and wireless data offloading). Complete the custom circular PCB layout designed to fit inside the projectile core.
10/26-11/10	***Software Development and testing Prototype II*** Transition the prototype code into a robust, event-driven codebase that handles sleep modes and impact-trigger events. When PCBs arrive, assemble and start testing. Make decisions on the potting compound, 3D printed molds, and wireless charging receivers.
11/11-11/25	***Software Development and PCB testing*** Final testing, additional features, project expo Conduct field testing (throwing the projectile into a net or mitt) and evaluate data accuracy.
11/26-12/9	***Final testing, additional features, project expo*** Project completion 12/3. Get the poster completed and start the final report. Fix software only bugs.
12/10-12/11	**Submit Final Report**

Rough Budget

What	Quantity	Cost per	Cost total
STM32 / NRF52 MCU (w/ BLE)	**5**	**$12.00**	**$60.00**
PCB Fabrication	**5**	**$15.00**	**$75.00**
High-G Accelerometer (e.g., ADXL375)	**5**	**$18.00**	**$90.00**
High-Speed SPI Flash Memory	**5**	**$3.00**	**$15.00**
Small Li-Po Battery (Coin/Pouch)	**5**	**$8.00**	**$40.00**
Wireless Charging Receiver IC & Coil	**5**	**$10.00**	**$50.00**
3D Printing Materials & Potting Resin	**1**	**$60.00**	**$60.00**
Misc passive components	**1**	**$65.00**	**$65.00**
Shipping	**1**	**$60.00**	**$60.00**
Total			**$515.00**

Issues

The largest concerns for what needs to be thought about:

1. **Hardware Survivability:** Ensuring the PCB traces, battery connections, and IC solder joints do not shear off during high-velocity impacts with a catcher's mitt or a bat.
2. **Weight Distribution:** Centering the mass of the electronics perfectly so the ball does not wobble or fly with a lopsided trajectory, maintaining the exact 5.0 to 5.25 ounce weight of a regulation baseball.
3. **Antenna Detuning:** Designing the BLE antenna so the signal isn't completely absorbed by the pitcher's hand, the leather casing, or the dense potting resin.
4. **Power Management:** Because there are no physical buttons to turn the device on or off, it must rely on a flawless accelerometer wake-up interrupt to preserve battery over weeks of standby time.
5. **Testing:** To actually test if the ball works, we are going to have to find where we can use very expensive ball tracking Radar equipment and collect data

Issue 1 is the primary engineering challenge and differentiator. Issue 2 requires careful CAD design and balancing of the internal battery and PCB layout. Issue 3 will require some RF tuning during the PCB layout phase. Issue 4 is a critical firmware state-machine problem that dictates the usability of the final product. Issue 5 is a big problem too because building this product and not being able to test it is a waste of time and money.

Stretch Goals

Some cool ideas that may not make it to production in a semester:

- Developing a basic Python or web app interface to visualize the trajectory and spin axis in 3D.
- Implementing an inductive charging dock that doubles as a data-sync hub, eliminating the need for BLE offloading.

Evaluation

The instrumented baseball will be evaluated by comparing its internal tracking data against ground-truth metrics to verify it serves as a viable, low-cost alternative to commercial radar systems. The measure of success involves throwing the regulation-weight projectile alongside a standard tracking unit (will need to find) to confirm that the recorded pitch velocity, spin rate, and spin axis remain highly accurate after repeated, high-velocity impacts, clearly defining how the project works and meets its core real-world application.

EECS 473 – Project Pre-Proposal Homework

Tsung Lin Hsieh

Fall 2026

Project 1: Smart Bicycle Collision Warning System

High-Level Description

Cycling is an efficient and inexpensive form of transportation, but a cyclist has limited awareness of vehicles approaching from behind. Looking backward takes attention away from the road ahead, while mirrors provide useful information only when the rider actively checks them. This project proposes a compact, bicycle-mounted collision-warning system that continuously monitors the area behind the bicycle and warns the rider when another road user is approaching in a potentially dangerous way.

The finished device would consist of a rear-facing sensing unit mounted near the seat post, a custom processor-based PCB, and a simple rider interface mounted near the handlebars. The rear unit would measure the distance to objects behind the bicycle and estimate whether an object is getting closer. The embedded processor would filter the sensor measurements, estimate closing behavior, and classify the situation into several warning levels. The rider interface would then communicate the warning without requiring the rider to look backward. For example, LEDs could indicate that an object is present, while a vibration motor or buzzer could provide a stronger warning when an object is rapidly approaching.

The intended users are cyclists traveling on roads or shared streets, especially commuters who regularly ride near motor vehicles. The goal is not to replace normal situational awareness or guarantee collision avoidance. Instead, the device would act as an additional source of information, similar in spirit to a vehicle blind-spot warning system, while remaining inexpensive and small enough to mount on an ordinary bicycle.

This project is attractive for an embedded-systems course because it combines sensing, real-time processing, user feedback, power management, mechanical packaging, and PCB design into a complete physical device. It also has clear real-world performance requirements that can be tested quantitatively. A successful prototype should reliably detect relevant approaching objects while avoiding excessive false warnings caused by stationary roadside objects or noisy sensor readings.

Implementation Issues

The system would be divided into three major functions: rear sensing, embedded decision making, and rider notification. A rear-facing distance sensor would periodically measure objects behind the bicycle. Possible technologies include time-of-flight, ultrasonic, or compact radar sensors; the final choice would be made after comparing usable range, field of view, update rate, outdoor reliability,

power consumption, cost, and ease of interfacing. A microcontroller on the custom PCB would collect the measurements and maintain a short history of distance estimates. From this history, the software could estimate whether an object is approaching and how quickly the separation is changing.

The warning algorithm does not need to identify the exact type of vehicle. A practical first version could classify risk using distance, persistence, and estimated closing rate. For example, a distant object moving at approximately the same speed as the bicycle would not require an urgent warning, while an object whose measured distance is rapidly decreasing would generate a stronger alert. Filtering and hysteresis would be used so that a single noisy measurement does not immediately trigger a warning.

The custom PCB would contain the microcontroller, sensor interface, power regulation, and outputs for the warning devices. Depending on the selected architecture, the handlebar warning unit could be connected by wire or through a low-power wireless link. A wired prototype would reduce communication risk, while a wireless interface could be treated as a later enhancement if the schedule permits. The electronics would be powered from a rechargeable battery and placed in a weather-resistant enclosure suitable for mounting on a bicycle.

Rough Schedule

Stage	**Planned Work**
1. Requirements and sensor research	Define target detection range, update rate, warning behavior, size, and battery goals. Compare candidate sensing technologies and select a processor family.
2. Development-board prototype	Obtain candidate sensor hardware and build a bench prototype. Verify sensor range and repeatability, collect sample data, and implement initial distance filtering and closing-rate estimation.
3. System design and PCB	Finalize the electrical architecture, power system, user interface, and communication method. Design and review the processor-based PCB and submit it for fabrication.
4. Software development during PCB fabrication	Use the approximately 10-day PCB fabrication period to improve the warning algorithm, build a test-data logging system, develop the rider interface, and test the system on development boards.
5. PCB bring-up and bicycle integration	Assemble and bring up the PCB, verify power rails and interfaces, integrate the sensor and warning hardware, design the enclosure/mount, and begin bicycle-mounted testing.
6. Final testing and Design Expo preparation	Measure detection performance and false-warning behavior, test battery operation and mechanical stability, fix remaining problems, and prepare a repeatable demonstration.

A major scheduling goal would be to prove the sensing concept before the PCB is ordered. This reduces the chance of designing a board around a sensor that cannot provide useful outdoor measurements. Software development and mechanical work can continue while the PCB is being fabricated, so the fabrication delay should not stop progress.

Rough Budget

The exact components would be selected later, but the following is a reasonable first estimate for one or more prototypes.

Item	Estimated Cost
Distance/ranging sensors and evaluation hardware	$100
Microcontroller development boards and debug hardware	$60
Custom PCB fabrication and assembly parts	$180
Battery, charging, and power components	$70
LEDs, vibration motors/buzzer, switches, and connectors	$50
Enclosure, bicycle mounts, and mechanical hardware	$90
Spare parts and miscellaneous components	$100
Estimated total	**$650**

For a group of approximately five students, this estimate is comfortably below the course allowance of $250 per student and leaves margin for replacement sensors, a PCB revision, or a more capable sensing module if early testing shows that it is necessary.

Potential Implementation Challenges

Outdoor sensing reliability. The most important technical uncertainty is whether a low-cost sensor can reliably observe objects at a useful distance behind a moving bicycle. Different surfaces, sunlight, weather, road vibration, and sensor field of view may affect measurements. This risk would be addressed early by testing candidate sensors outdoors before finalizing the PCB.

False warnings. A wide field of view could detect parked cars, signs, walls, or other objects that are not collision threats. The software would therefore use a sequence of measurements rather than a single distance threshold. Filtering, persistence requirements, and closing-rate estimates should reduce false positives. The goal is not perfect traffic understanding, but a useful and explainable warning rule.

Mechanical mounting and vibration. The rear sensor must maintain a useful orientation even when the bicycle travels over bumps. The enclosure and mount should be rigid, adjustable, and easy to remove for debugging. Early prototypes can use 3D-printed or off-the-shelf mounting hardware before a final enclosure is produced.

Power consumption. Continuous sensing and wireless communication, if used, could reduce battery life. The processor can operate at only the update rate needed by the application, and the design can avoid unnecessary wireless communication. Battery life will be measured rather than assumed.

Safety. This device would be an experimental warning aid, not a safety-certified collision-prevention system. Testing would initially be performed on a bench and in controlled low-risk environments. The project would avoid tests that require a rider to intentionally create a dangerous traffic situation.

Stretch Goals

If the basic collision-warning system is working reliably, several extensions would be possible. A wireless handlebar module could eliminate a cable between the rear sensor and rider interface. A small display could show approximate distance or warning level. Data logging over Bluetooth could record detected passing events for later analysis. A more advanced algorithm could distinguish ordinary objects from consistent vehicle-like approach trajectories. These features would be secondary to producing a reliable basic warning system.

Evaluation and Testing with Regard to Real-World Issues

The project should be evaluated as a complete warning device rather than only by demonstrating that the sensor returns numbers. Testing would therefore focus on whether the system provides useful warnings under repeatable conditions.

First, the sensing subsystem would be tested at known distances using several representative targets. Measurements would be collected while targets are stationary and while they approach the sensor at controlled speeds. These tests would determine usable range, update rate, measurement variation, and the accuracy of the estimated closing rate.

Second, the complete warning algorithm would be evaluated using repeatable approach scenarios. A second bicycle or other safe moving target could approach the instrumented bicycle in a controlled parking-lot environment. The system would be tested at different approach speeds and lateral offsets. For each trial, the team would record whether a warning occurred, when it occurred, and whether the warning level matched the intended behavior. Stationary roadside objects and non-approaching targets would also be used to measure false warnings.

Third, system-level properties would be measured, including warning latency, continuous battery operating time, stability of the bicycle mount, and whether the rider can perceive the warning under normal riding conditions. The project would be considered successful if it can repeatedly detect meaningful approaching targets within the experimentally established operating range, issue a warning with sufficiently low latency to be useful, and avoid frequent warnings when no object is actually closing on the bicycle. Exact numerical thresholds would be finalized after early sensor characterization so that they are based on realistic hardware performance rather than arbitrary assumptions.

The Design Expo demonstration could use a controlled miniature roadway or a full-size stationary bicycle setup in which a target moves toward the rear sensor. A display or laptop could show measured distance and estimated closing rate while the physical LEDs or vibration device demonstrate the warning delivered to the rider.

Project 2: Autonomous Runaway Alarm Clock

High-Level Description

Conventional alarm clocks are easy to defeat: a sleepy user can press the snooze button without fully waking up and immediately return to sleep. This project proposes an intentionally inconvenient alarm clock that attempts to escape when its alarm is ignored. Instead of allowing repeated snoozing from bed, the clock would drive away, avoid nearby obstacles, and force the user to physically get up and catch it.

The completed device would look like a small wheeled robot containing a clock, speaker, display, sensors, motors, battery, and custom processor-based PCB. Before bedtime, the user would set an alarm using buttons or a simple interface. At the alarm time, the device would first behave like a normal alarm clock. If the user does not successfully dismiss it within a short period, the robot would begin moving around the room while continuing to make noise. To turn it off, the user would need to catch the device and perform a deliberate action such as holding a button for several seconds or entering a short button sequence.

The project is deliberately somewhat "wacky," but it is still a complete embedded system with sensing, motor control, real-time behavior, power management, and human interaction. It also creates an entertaining Design Expo demonstration: the device can visibly react to its surroundings while a user tries to catch it.

Implementation and Feasibility

The core hardware would include a microcontroller, real-time clock function, small display, speaker or buzzer, motor drivers, two drive motors, wheels, a rechargeable battery, and short-range obstacle sensors. Time-of-flight, infrared, ultrasonic, or bumper sensors could be used to detect walls and furniture. The final PCB would integrate the processor, motor-control interface, power circuitry, buttons, alarm output, and sensor connections.

The navigation algorithm would intentionally remain simple. The robot does not need to map a room or perform sophisticated path planning. When escape mode begins, it can drive in a semi-random direction, detect obstacles, back up or turn when necessary, and periodically change direction so that its movement is difficult to predict. This keeps the project achievable while still producing interesting autonomous behavior.

An accelerometer or other motion sensor could optionally detect when the clock has been picked up. This could be used to change the alarm pattern or require the user to complete the dismissal sequence after physically catching the robot. However, pickup detection is not necessary for the minimum viable system.

A rough development plan would be:

1. Define the alarm behavior, robot size, user controls, and minimum navigation requirements; select motors, sensors, and processor architecture.

2. Build a development-board prototype and test the clock interface, alarm output, motor control, and obstacle sensing independently.

3. Integrate the subsystems on a temporary chassis, implement basic obstacle avoidance, and verify that the robot can move reliably on common indoor floors.

4. Design and order the custom processor-based PCB. During PCB fabrication, continue software development and mechanical work using the prototype electronics.

5. Bring up the PCB, integrate it into the chassis, test battery operation and motor noise effects, and debug the complete alarm-to-escape sequence.

6. Perform repeated room-scale tests and prepare the final enclosure and Design Expo demonstration.

A preliminary budget is approximately $350–$500 for the complete prototype and spare parts. Motors, wheels, chassis materials, and motor drivers may cost roughly $100–$150; sensors and user-interface components approximately $50–$100; batteries and power electronics approximately $50; and PCB fabrication, processor components, connectors, and spare parts approximately $150–$200. This should remain well within the available budget for a normal project group.

The main implementation issue is mechanical reliability. A robot that cannot move consistently on carpet, cross small floor transitions, or survive being caught would not be convincing. Motor selection and chassis design would therefore be tested early. A second issue is obstacle detection: the clock should not repeatedly collide with walls or become trapped in corners. The minimum requirement would be reactive obstacle avoidance rather than full autonomous navigation. A third concern is power integrity because motor current and electrical noise could interfere with the processor or sensors. Appropriate motor drivers, decoupling, and separation of power paths would be considered during PCB design.

For safety, the robot would operate at low speed and use a lightweight chassis so that collisions with furniture or people are unlikely to cause injury or damage. The motors would be limited to the torque and speed necessary for indoor movement.

Evaluation and Testing

The project would be considered successful if the device can maintain time, activate at the programmed alarm time, transition into escape mode when the alarm is not dismissed, move autonomously through a typical indoor area, avoid common obstacles, and require the user to physically interact with the device before the alarm stops.

Testing would include repeated alarm-time trials to verify timing and state transitions, obstacle-avoidance trials using walls and boxes at different approach angles, and mobility tests on representative indoor surfaces. The team could define a simple test area containing several obstacles and run the clock repeatedly for a fixed amount of time. Useful measurements would include the fraction of trials with no collision, the number of times the robot becomes stuck, operating time per battery charge, and the time required for a user to catch and dismiss the alarm.

Possible stretch goals include a docking station for automatic charging, configurable escape difficulty, an accelerometer-based pickup detector, Bluetooth configuration from a phone, or an adaptive behavior in which the robot changes its motion pattern after repeated catches. These features would only be attempted after the basic clock, motion, obstacle avoidance, and custom PCB are working reliably.

Stall Status: Wireless Bathroom Occupancy & Queueing System

Anyone who has lived in a house with more roommates than bathrooms knows the routine: knock, wait, knock again or just waiting outside the door hoping the door opens soon. In a house of five or six people sharing only one or two bathrooms, this adds up to real wasted time every single day, along with an occasional awkward standoff over who's next to use it. This device is applicable to a wide range of people, whether it be large families, shared houses such as college students and could even scale to more commercial applications.

I propose a small embedded system that removes the guesswork and leads to less time wasted. A sensor on the bathroom door would report whether it's open or closed, a control panel inside the bathroom lets the current occupant flag what they're doing (shower, toilet) and roughly how long they expect to be, and also a display on the outside of the bathroom to show real time status plus a time remaining. If more than one person wants the bathroom, they can join a simple queue from the outside panel and get notified when it's their turn, removing the need to physically wait outside the door. There could also be a phone app that connect to the system to more easily get notifications outside of the bathroom but maybe even includes the inside bathroom controls as everyone is on their phone.

Details

The system would be built around three cooperating nodes, each designed as a custom PCB built around a microcontroller with integrated or add-on wireless communication, this could be an ESP32 chip which has WiFi build in or an STM32 pared with a wireless module if we want finer control over digital design. All three nodes would communicate over the house's exisitng WiFi network using a lightweight protocol.

- **Door sensor node** - A magnetic switch or hall-effect sensor mounted on the door and door frame to report open/closed state. This is the simplest board and mainly serves as a sanity check against the occupant's manual input, might have an auto clearing status if door is open and stays open. Have to also determine what is the best sensor to have for this application and ensure we don't get any false positives like if the door is cracked it shouldn't count it as closed necessary.
- **Control panel (inside the bathroom)** - A small touchscreen plus a few physical buttoms to let the occupant select an activity (shower/toilet/other) and an estimated durations. This board publishes the current status and countdown to the network. We might also want to show the active queue for the occupant to know, maybe even a notification if its been a while. Idealy we would want to make this water/moisture resistant if possible or just conformal coat everything to limit moisture.
- **External display (outside the bathroom)** - A low power e-ink if running on a battery or a OLED screen if running on external power, this would be mounted next to the door to show current status, estimated time remaining, and queue position for anyone waiting. Might have the ability to send notifications to a particular user if they are waiting and don't want to wait at the door. If the external unit runs on wall power then this might be the brains of the nodes.

Schedule

Dates	Phase	What happens
9/10 – 9/23	Planning	Finalize node architecure, decide on microcontroller so either ESP32 or STM32 + seperate wireless module, select communication protocol, select door sensor and display hardware. Draft schematics in more detail.
9/24 – 10/6	Design & part selection	Order chosen MCU/wireless chips, door switch, touchscreen/display for internal and external nodes. Bring up chosen MCU on breakout board/eval board ti vakudate firmware.
10/7 – 10/25	Prototype & PCB design	Get all three node types communicating over the chosen communication protocol. Design and order PCBS for all three nodes.
10/26 – 11/10	Software dev. & testing	Build out the software so logic, firmware, timers on prototpe hardware while PCBs are on the way. Get touchscreen/buttons UI working. Also being enclosure designs as size is determined by PCBs.
11/11 – 11/25	PCB bring-up & integration	Assemble and test the PCB nodes, add software over to fix issues, fix hardware issues, finish enclosure design and printing.
11/26 – 12/9	Final testing, poster, expo	Full system testing with multiple users/queue scenarios, actually put into a bathroom to have real world testing. Prepare poster and final report.

Budget

The budget below assumes one full bathroom setup (door node, control panel, and external display). With a team of four to five students this fits well within our budget and might even be able to produce a second set for spares or a second bathroom unit if time allows. Most of the cost would be prototyping and the end user cost would be much lower.

What	Quantity	Cost per	Cost total
ESP32 modules (or STM32 + wireless module)	3	$10.00	$30.00
MCU eval/breakout board (firmware bring-up)	2	$25.00	$50.00
Capacitive touch LCD (control panel)	1	$40.00	$40.00
E-ink display (hallway status)	1	$30.00	$30.00
Door sensor	1	$15.00	$15.00
Custom PCBs (3 node designs, 5 each)	3	$25.00	$75.00
3D-printed enclosures (almost free on campus)	3	$12.00	$36.00
Misc. wiring, connectors, batteries	1	$100.00	$100.00
Shipping	1	$75.00	$75.00
		Total	$441.00

Issues

The largest concerns at this time are

1. Battery life for all the nodes, especially door sensor so making something very efficient. Also figuring out if all battery power, wall power only or ability to charge.
2. Bringing up a WiFi capable MCU on our own PCB (antenna layout or external?)
3. WiFi range and reliability throughout the house.
4. Having all three nodes intercommunicate.

Issue 1 is addressable by keeping each node in a low-power sleep mode between events and only waking to transmit on a state schange, which is espencially important for the door sensor since it has the least opportunity to be near a convenient power source. We would still need to decide per-node whether battery only, wall powered, or a rechargeable with wall backup design makes the most sense; the control panel and external display being mounted near the bathroom are good candidates for wall power.

Issue 2 is the most novel part of the hardware design, but is de risked by validiting firmware on an eval board first so that by the time we lay out our own PCB, the only new risk is the board design itself rather than the firmware. We'll also need to decide between a module with an onboard antenna vs a diesign using an external antenna, depending on enclosure placement and range needs.

Issue 3 seems manageable in most houses since bathrooms are well within WiFi range on a router.

Issue 4 is the least well understood part of the project but with some research and intial prototyping I am confident that we can get the three nodes talking reliably.

Additional stretch goal

If time allows, I would like to add a companion phone app that shows live bathroom status and queue position, and sends a push notification when it's almost someone's turn, so people don't need to be standing near the external display to know when to head over.

Trumpet Harmonizer

Ever wish you could play a trumpet duet by yourself? This project is a small unit that clips directly onto the trumpet itself, listens to the live sound, figures out what note is being played, and generates one or more harmoney notes in real time, then mixing them with the original sound out to a speaker. Since it's mounted on the instrument rather than sitting on the floor, the player can change the harmoney on the fly, thus switching intervals or toggling the effect mid phrase without breaking their playing position where other solutions like a floor pedal would.

Details

A small microphone or equivalent feeds the trumpet's sound into an amp/filtering then into a ADC to be digital for the microcontroller. The microcontroller would likely be an STM32 with enough bandwidth for real time signal processing or a dedicated audio codec chip along with a microcontroller, this would run a pitch detection algorithm to identify the fundamental frequency being played, then synthesizes a harmoney note at a musical interval above or below it. The original and harmonized signals are mixed and sent out through a DAC to a small onboard speaker, or a line out for a jack, we could also try to do bluetooth or a wireless comms to an external speaker/amp. The whole unit which would be the MCU PCB, battery, a few small buttoms/slider control would clip onto the trumpets body in a 3D printed housing. It would be positioned so the player can reach the controls with a thumb without needing to stop playing, letting them change the harmoney interval or toggle the effect on/off in real time while performing. A small OLED or LED indicator on the unit itself could show the currently selected interval so the player has a quick visual confirmation of what mode they're in without needing to look away from their music for long.

Implementation Notes

The riskiest part of this project would be doing the pitch detection and harmoney synthesis with a low enough latency that it still feels live to the player, this is a similar problem commercial guitar harmonizers solve so it can be solved, but the trumpet might take some iteration. Mounting everything on the instrument itself adds contraints a floor pedal wouldnt have: the whole thing needs to run on battery (could have an external power cord worst case), stay small and light to not change the players balance or valve technique, and survive an entire performance. Budget wise this would be much smaller to build than my project 1, an MCU pcb, mic, speaker, battery, maybe wireless comms, 3D printed housing should come under $ 200. If real time pitch tracking proves too difficult to get solid by the PCB deadline, a reasonable fallback is to only support a single fixed harmony interval rather than be switchable, but would defeat the purpose of being able to select what you want. This would be a smaller scope as well so we would want the finished product to be very polished and be easily integrated onto any trumpet, we could also poll artists to see if we should add additional options. There is a market for this as there isn't anything for trumpets that exist on the market.

Evaluation

Success would look like a trumpet player being able to play a simple melody and hear a correctly pitched harmoney come out in real time with no noticeable lag and no wrong notes as the melody moves around. We'd test this across a few players and a range of tempos and note ranges, and verify that the interval switching control is usable if a player can change it mid phrase without needing to stop.

PROJECT ONE (Realistic): Portable Autonomous eDNA Sampler for Remote Field Sites

High Level Description

Environmental DNA (eDNA) is DNA organisms shed into their environment through sources such as skin cells, mucus, and feces [1]. By analyzing these traces, researchers can study wildlife without directly observing or capturing an organism. eDNA has been used to track endangered species, monitor invasive species, measure biodiversity, and study disease spread [2]. In the Florida Everglades, for example, eDNA increased detection of invasive Burmese python clusters from about 1% with traditional search-and-capture methods to over 90% [3].

However, eDNA tracking is still limited by various factors. Currently in aquatic environments, researchers must pump water through a filter, bring that filter back to a lab, and analyze the trapped DNA. This still requires someone to physically travel to the sampling location when necessary. This makes frequent sampling difficult at remote sites or during sudden, inconvenient, or dangerous events. A 2024 USGS study [4] found that eDNA concentrations can change substantially over short time periods and can correlate with conditions such as river discharge and turbidity, highlighting the limitations of manual sampling.

I propose a small, portable autonomous eDNA sampler that could be carried to a field site and left beside a stream for several days. The device would contain four independent filter cartridges and collect samples either on a preset schedule or when it detects environmental changes such as increased turbidity or water level. It would not analyze DNA itself; instead, it would collect the physical samples and log relevant environmental data for later laboratory analysis.

Existing autonomous eDNA samplers are expensive and difficult to transport. The Smith-Root system, for example, weighs about 100 pounds and costs roughly $30,000. Rather than matching its full capabilities, the goal would be to build a four-sample, backpackable system within the EECS 473 budget.

Potential users include researchers sampling remote or hazardous locations, ecological nonprofits and smaller communities/nonprofits that cannot afford commercial systems, and Great Lakes researchers monitoring biodiversity and invasive species.

Implementation Details

Technical Product Description:
The proposed device would be a portable, battery-powered autonomous eDNA sampler capable of collecting four separate water samples during a multi-day field deployment. The entire sampler would fit inside a weather-resistant enclosure small enough to be carried by one person. Four intake tubes would be placed in the stream, while the electronics, pump, filters, battery, and valve mechanism would remain above the water line.

Each sample would use its own intake tube and disposable eDNA filter in order to reduce cross-contamination. Downstream of the four filters, the lines would connect through individually controlled pinch valves to a shared flow sensor, pressure sensor, and peristaltic pump. Only one valve would be opened during a sample. Because all shared tubing and instrumentation are located after the filter, water from one sample does not pass through another sample's filter. The pinch valves would be mechanically actuated using four TowerPro SG92R micro servos.

Water would be pulled through the selected filter with a Kamoer KPK400 12 V peristaltic pump. A potential model provides up to roughly 400 mL/min at 12 V. Actual filtered volume would be measured rather than estimated only from pump run time. A DFRobot SEN0216 YF-S401 flow sensor is a reasonable option because its 0.3–6 L/min range includes the expected pump operating point.

A second measurement would monitor the pressure drop across the filter. The proposed Adafruit MPRLS pressure sensor measures 0–25 PSI absolute and costs $29.95. The controller would record the pressure before starting the pump and monitor the decrease in pressure while water is being pulled through the filter. If the pump is running but flow decreases while vacuum increases, the firmware can identify the filter as clogged and terminate the sample rather than incorrectly reporting a complete collection.

The system would also independently monitor stream conditions to decide when sampling is useful. Water level would be measured using a DFRobot KIT0139 industrial submersible level sensor, which measures 0–5 m of water depth and currently costs $43.90. This output would be converted to a voltage on the custom PCB and read by the MCU's ADC.

It may also include measuring turbidity, as it is a useful tool to determine when sampling is useful. Turbidity would be measured with a DFRobot SEN0710 industrial turbidity probe. The sensor measures 0–1000 NTU, and communicates over RS-485 using Modbus-RTU.

The system would be controlled by an STM32L476RGT6 microcontroller mounted directly on the team's custom PCB. The custom PCB would also contain the 12 V pump MOSFET driver, four servo outputs, an RS-485 transceiver for the turbidity sensor, a 4–20 mA receiver for the level sensor, battery-voltage sensing, 12-to-5 V and 5-to-3.3 V regulators, the microSD interface, and connectors for all external sensors.

An RFM95W 915 MHz LoRa radio would provide low-data-rate status communication. The module costs approximately $19.95. The radio would not transmit raw scientific data continuously; instead, it would periodically send a short status message containing battery level, remaining filters, last successful sample, and fault state. This would allow a researcher to confirm that a remote sampler is still functioning without physically reaching it.

The electronics and fluid hardware would be mounted in a Polycase ZQ-100804 polycarbonate outdoor enclosure. The enclosure is approximately 10 × 8.4 × 5.2 inches and currently costs $58.14.

Combining all this information, each sample record would contain the filter number, timestamp, requested and actual filtered volume, average flow rate, maximum vacuum, water level, turbidity, water temperature, battery voltage, and any detected errors.

Validation Plan & Stretch Goal

Validation would begin at the subsystem level before testing the complete sampler outdoors. The fluid system would first be tested by commanding a known sample volume and comparing the measured flow-sensor volume against the actual collected volume. The four sampling channels would then be tested for isolation by running differently colored or different-conductivity water through each intake and checking that inactive channels do not leak or backflow. Clogging would be simulated by restricting a filter or increasing sediment concentration: the system should recognize the combination of decreasing flow and increasing vacuum pressure and terminate the sample rather than marking it complete.

The integrated device would then be tested for event triggering, data logging, communications, and unattended operation. Water-level and turbidity changes could be artificially introduced in a tank or stream-like setup to verify that the correct filter is selected and a sample is collected. Final validation would consist of at least a 24–72 hour outdoor deployment in which the system autonomously collects multiple samples and records their timestamps, volumes, environmental conditions, and fault states. True DNA-level cross-contamination testing would require access to an eDNA lab; without that, fluid isolation can still be validated using dye or conductivity, with PCR-based testing treated as a stretch goal.

Reliability

Because the device may be left unattended for several days, it should fail in a way that preserves information rather than silently producing bad samples. Each filter would therefore have a persistent state such as unused, sampling, complete, or invalid. If power is lost while Filter 2 is running, for example, the system should mark that cartridge as invalid after reboot and continue with the next unused filter instead of attempting to reuse it. The firmware would also detect conditions such as low flow, excessive vacuum, low battery, failed sensors, and unsuccessful valve actuation and record these faults alongside the sample data.

The hardware would be designed so that a single sensor failure does not damage the rest of the system. The pump would have current limiting or fuse protection and the MCU would use a watchdog. The LoRa link would only be used for status monitoring, so loss of radio communication would not prevent the sampler from continuing its scheduled or event-triggered operation.

Safety

The system is relatively low risk because it operates from a low-voltage battery and does not perform DNA extraction or contain hazardous biological reagents. The battery, PCB, pump electronics, and actuators would remain inside the weather-resistant enclosure above the water line, with only tubing and externally rated sensors entering the stream. The pump output would be fused, exposed electrical connections would be avoided, and cable glands would provide strain relief and water resistance.

The main field risks are leaking water, tubing becoming detached, and the enclosure or intake becoming a physical obstruction. The sampler would therefore be securely anchored above the expected high-water line, with tubing positioned so that it does not create a hazard to people or wildlife. Field testing would initially be conducted at safe, easily accessible sites rather than during actual flooding or severe weather.

Timeline

Weeks 1–2 — Requirements and single-channel proof of concept	Finalize target sample volume, deployment duration, and sensing requirements. Purchase pump, filters, tubing, flow sensor, and pressure sensor. Demonstrate that the Kamoer pump can pull water through one eDNA filter and accurately measure collected volume. Characterize normal flow and pressure
Weeks 3–4 — Four-channel fluid system	Build the four-channel tubing setup and prototype the SG92R servo pinch valves. Verify that one channel can be selected while the other three remain closed. Begin clog-detection testing and determine practical pressure/flow thresholds.
Weeks 5–6 — Electronics prototype and PCB design	Bring up the STM32 development platform and interface the flow sensor, pressure sensor, water-level sensor, servos, pump driver, microSD, and optionally turbidity/RS-485. Finalize schematic and layout of the custom PCB and submit it for fabrication.
Weeks 7–8 — Firmware while PCB is manufactured	Implement the sampling state machine, filter bookkeeping, SD logging, scheduled triggering, sensor monitoring, watchdog behavior, and reset recovery using the development-board prototype. Develop event-trigger logic for water level and turbidity. Begin enclosure and mechanical packaging.
Weeks 9–10 — PCB bring-up and system integration	Assemble and test the custom PCB. Integrate the completed electronics with the four-channel fluid assembly and enclosure. Add LoRa status transmission once the autonomous sampling path works reliably.
Weeks 11–12 — Reliability and outdoor testing	Run repeated four-filter cycles, induce clogged filters and power interruptions, measure sample-volume accuracy, test radio range, and

	measure battery consumption. Perform an initial overnight or multi-day outdoor deployment.
Final weeks — Field validation and Expo preparation	Conduct a 24–72 hour autonomous field test, analyze logged data and failures, make final mechanical/firmware revisions, and prepare the demonstration. If time and access permit, use laboratory controls to test DNA-level cross-contamination.

Budget

- STM32L476RGT6 + custom PCB electronics: $130
- Kamoer KPK400 peristaltic pump: $129
- 4× TowerPro SG92R servos: $24
- DFRobot SEN0216 flow sensor: $9
- Adafruit MPRLS pressure sensor: $30
- DFRobot KIT0139 water-level sensor: $44
- RFM95W LoRa radio: $20
- RTC/microSD components: $30
- Polycase ZQ-100804 enclosure: $58
- Battery and power hardware: ~$80
- eDNA filters/consumables: ~$175
- Tubing, fittings, general mechanical parts: ~$75
- PCB fabrication/spares/contingency: ~$100
- **Core Capabilities = ~$900**
- Optional DFRobot SEN0710 turbidity probe: +$229
- **System with turbidity = ~$1,130**

Given a 5 person team, this is $226/person in the most expensive case and below the $250/max

Possible Implementation Issues

The largest technical risk is likely the four-channel fluid system. The servo-driven pinch valves must completely close three tubes while allowing reliable flow through the selected fourth tube. This should therefore be prototyped early using the SG92R servos and silicone tubing before committing to the final mechanical design. If the servo pinch mechanism proves unreliable, commercially available pinch valves could be substituted at higher cost.

A second concern is filter clogging and contamination. Natural water containing sediment may clog a filter before the target volume is reached, which is why the design includes both flow and vacuum-pressure measurements. Preventing sample-to-sample contamination is harder to prove

without laboratory analysis, but separate upstream tubing and filters minimize shared wetted surfaces. The DFRobot turbidity sensor is also relatively expensive, so turbidity sensing can be treated as an optional feature if budget or integration time becomes limiting. Another implementation issue is packaging: tubing, four filters, a pump, battery, and electronics must all fit into a backpackable enclosure while remaining accessible enough for researchers to replace cartridges between deployments.

Resources

[1] https://www.nhm.ac.uk/discover/what-is-environmental-dna-edna.html
[2] https://inhs.illinois.edu/research/by-topic/environmental-dna-edna/
[3] https://iucn.org/resources/issues-brief/environmental-dna
[4] https://www.usgs.gov/publications/field-trials-autonomous-edna-sampler-lotic-waters

PROJECT TWO (Less Realistic): Autonomous Self-Repositioning Camera

High Level Description

Camera traps allow researchers to monitor wildlife for weeks without a person continuously present. However, their usefulness depends heavily on maintaining a clear and consistent field of view. Vegetation can grow or move in front of the camera, creating false triggers or completely obscuring animals, while snow and physical movement of the camera can alter the view during a long deployment. Camera-trapping guidance specifically recommends clearing vegetation because it can block the field of view or cause large numbers of empty images, and examples of unsecured cameras show that even relatively small changes in orientation can make a station less effective. [1][2]

This project proposes a self-maintaining camera-trap mount that can detect when its view has shifted or become obstructed and make small pan/tilt corrections without requiring a researcher to revisit the site. Rather than designing another wildlife camera, the system would act as a motorized platform compatible with an existing commercial trail camera. A small secondary camera mounted alongside the wildlife camera would periodically inspect the scene. The system would compare the current image against a reference view recorded during installation and determine whether the camera has shifted or whether a new nearby obstruction, such as a branch or accumulated vegetation, is blocking the target corridor.

If a problem is detected, the mount would scan several nearby pan/tilt positions and select the position that preserves as much of the original target area as possible while reducing the obstruction. If the system cannot confidently find a better view, it would remain in its current position and flag the camera for human inspection rather than risk making the data worse.

This addresses a different problem from most “smart” camera traps. Instead of improving animal recognition after an image has already been captured, the goal is to keep the camera station itself usable throughout a long unattended deployment. This could be useful for wildlife researchers monitoring remote areas where checking every camera frequently requires significant hiking and field time. NPS monitoring projects, for example, commonly leave motion-triggered cameras deployed for roughly six weeks before retrieval, after which false-trigger images must be removed from the dataset.

Implementation Details

<u>Technical Product Description</u>:

The prototype would consist of an off-the-shelf wildlife camera attached to a two-axis motorized mount. A Bushnell Prime Low Glow trail camera, currently approximately $100, could serve as the test camera. Alongside it, an Arducam Mega 5 MP SPI camera would provide images to the embedded controller. The Arducam module costs approximately $40, communicates over SPI, and can capture images as small as 320 × 240.

An STM32H7-class microcontroller on a custom PCB would periodically wake the secondary camera and acquire a low-resolution image. During installation, the user would aim the trail camera normally and press a button to store the initial image as the reference view. Future images could then be compared using image-processing techniques, though the goal would not be sophisticated computer vision or species recognition. It would just see if the entire image has shifted relative to the reference, whether a large object is now occupying the field of view, and whether moving the camera would make it more usable.

Two ROBOTIS DYNAMIXEL XL430-W250-T smart actuators could provide pan and tilt. Each actuator contains its own motor controller and absolute position sensor, provides position feedback, and currently costs about $24. Using actuators with built-in position sensing would allow the STM32 to command a specific orientation and verify that the mechanism actually reached it.

The PCB would contain the STM32, DYNAMIXEL half-duplex UART interface, microSD storage, RTC, battery monitoring, voltage regulation, watchdog circuitry, and optionally an RFM95W LoRa radio for transmitting status messages such as “CAMERA OK”, “VIEW CORRECTED”, or “HUMAN CHECK REQUIRED”. The entire control system and battery would be mounted in a weather-resistant enclosure, while the pan/tilt mechanism would be attached to a tree or post using straps or a clamp. A Polycase ZQ-100804 outdoor enclosure currently costs about $58 and is large enough for the controller and battery.

Validation Plan

The system would first be tested indoors by deliberately rotating the mount by known angles and determining whether it can restore the original field of view. Obstruction testing would then place branches, leaves, or artificial snow-like barriers in different parts of the view and verify that the system selects a nearby orientation with less obstruction while remaining inside its permitted pan/tilt region.

Reliability testing would expose the system to changing daylight, moving vegetation, and repeated disturbances to measure false repositioning. The final test would be a multi-day outdoor deployment in which the camera periodically checks itself while several controlled changes are introduced. Success would mean the system maintains a usable view for the duration of the test, correctly fixes intentional orientation changes or major obstructions, and does not unnecessarily move in response to normal lighting or wildlife activity.

Reliability, Safety, & Implementation Issues

The largest technical risk is determining when the camera should actually move. Lighting, shadows, leaves, animals, rain, and snow may all make the current image look different from the reference even though the camera is still positioned correctly. Therefore, the system should be deliberately conservative. Repositioning would require a sustained problem across multiple checks, movement would be restricted to a predefined angular window, and a low-confidence result would cause the system to log an error rather than move. The original pan/tilt coordinates and reference image would remain stored so the camera can always return to its initial orientation.

Timeline

Weeks 1–2	Define what counts as a usable field of view. Purchase secondary camera and actuators. Build a one-axis prototype and demonstrate repeatable positioning.
Weeks 3–4	Build the complete pan/tilt mechanism. Interface the Arducam with an STM32 development board and store/capture reference images.
Weeks 5–6	Develop basic reference-image comparison and camera-shift detection. Add candidate-position scanning and scoring. Complete PCB schematic/layout and order the board.
Weeks 7–8	While PCB is manufactured, test the algorithm under changing lighting, vegetation, and artificial obstructions. Develop enclosure and tree/post mounting hardware.
Weeks 9–10	Bring up the custom PCB and integrate motors, camera, storage, battery monitoring, and watchdog behavior. Add LoRa if time permits.
Weeks 11–12	Perform outdoor reliability tests, tune false-reposition thresholds, measure power consumption, and validate recovery from intentional camera displacement.
Final weeks	Conduct a multi-day unattended field test, fix remaining mechanical/software issues, and prepare the Design Expo demonstration.

Budget

- Bushnell Prime Low Glow trail camera: ~$100
- Arducam Mega 5 MP SPI secondary camera: ~$40
- 2× DYNAMIXEL XL430 actuators: ~$48
- STM32H7 + custom PCB components: ~$140
- PCB fabrication: ~$75
- Battery that can last a long time: ~$80
- Polycase outdoor electronics enclosure: ~$58
- RTC, microSD, battery monitor, connectors: ~$50
- Optional LoRa radio: ~$20

- Bearings, brackets, shafts, straps, fasteners, and 3D-printed parts: ~$125
- Weatherproof cable glands and wiring: ~$40
- Spares/contingency: ~$100
- **Estimated total: ~$876**

Resources

[1] https://www.nps.gov/goga/learn/management/upload/GGNRA186477_GGNRA194722-PT-8_Subpart-2.pdf

[2] https://nordensark.se/media/179889/picamonitoring-the-manul-guidelines-for-practitioners.pdf

[3] https://www.nps.gov/articles/000/sodn-wildlife-monitoring-sagu-tmd-2023.htm

Supermileage: Inverter and DAQ board project

High-level description

"Cars and trucks account for nearly one-fifth of all US emissions, emitting over 23 pounds of carbon dioxide and other global-warming gases for every gallon of gas" [1].This remains one of our most pressing environmental challenges as we strive to preserve our planet for future generations. The University of Michigan Supermileage (UMSM) team's mission is **to work towards mitigating the environmental impact of transportation by engineering some of the world's most energy-efficient vehicles.**

The University of Michigan Supermileage (UMSM) project team has been attempting to create and compete with an electric urban concept vehicle for the last four years. The last successful electric vehicle that the team produced was in 2023 and since then there have been multiple attempts to create and compete with an electric vehicle, but each has failed to do so.

The goal of this project is to create a system which is able to produce enough torque to move a 85 pound vehicle and 150 pound driver at 17 MPH for 40 minutes.The team will operate on an Minimum Viable Product (MVP) mindset by partially integrating with every step of the design process to ensure that we always have a product that works and is precise and accurate throughout the entire project.

Timeline

EECS 473 Inverter Project Timeline	Week
Planning ● The 473 group will meet with the Supermileage Electrical Systems team and the executive board to develop a set of requirements, must-haves and nice-to-have features. ● Begin working on the proposal ● Create Design Log to help with final report	Pre-proposal Meeting (9/24) 9/10 - 9/24
Part selection ● Decide which parts the UMSM currently has will be provided by the team to the 473 team and which parts the 473 will have to purchase ● Once the proposal has been approved, begin working on planning which parts to purchase	Post proposal 9/24 - 10/1

○ Look at multiple different processors (different for DAQ and Inverter), H-bridges, mosfets, encoder, sensors (gps, hall/shunt, temp, imu) and choose best based on efficiency and integratability. ○ Research communication methods LoRa and beacon strategy for Comms board stretch goal ● Decide on modular PCBs with hot-swappable H-bridges and MOSFETs or one board design model ● Begin working on developing basic functionalities with UMSM supplied motor ● Update Design Log & continue weekly updates with UMSM Electrical Systems Team	
Prototype Have basic functionalities on benchtop: ● Spinning motors with basic motor control ● Connect peripherals to second processor and send CAN messages between different devices→ fallback is I2C and SPI ● Begin working on PCBs for Inverter and DAQ ● Update Design Log & continue weekly updates with UMSM Electrical Systems Team	10/1 - 10/22 10/22 Milestone 1
● PCBs ordered early in case we need revisions (week of 10/22) ● Trapezoidal motor control on bench spinning motors – develop easy to use interface to interact with motor control code ● GPS, and other DAQ peripherals working inside the car with proto-board ● Hall sensors and current sensors work, wheel sensors ● Update Design Log & continue weekly updates with UMSM Electrical Systems Team	10/22 - 11/12 11/12 Milestone 2
● Test PCBs → debug problems as they arise – reorder PCBs if needed BEFORE Thanksgiving break. ● Begin Integrated set-up ● Update Design Log & continue weekly updates with UMSM Electrical Systems Team	11/29 - post thanksgiving break

• Buffer for integration issues • [Stretch] On vehicle testing and final in vehicle integration • [Stretch] Use LoRa radio to live transmit DAQ information to a Grafana dashboard • Update Design Log & continue weekly updates with UMSM Electrical Systems Team	11/29 - 12/10
• Convert Design Log into Final Design report and have a knowledge transfer meeting with UMSM • Study for exams!!!	Post Showcase 12/11 - 12/16

Costs

Koford Motor and a 48V battery will be provided by the UMSM team. The following budget is subject to change after discussions with 473 team and UMSM team.

Inverter:

Item	Cost
DC-DC Converters (high efficiency)	$70
HBridges	$15
Mosfets (lots of various categories)	$50
Current Sensors (shunt/hall)	$5-$20
Encoder (single vs multi turn, incremental vs absolute)	$50-100
STM32 family processor/RPI(?) (low latency, multi thread able, power efficient)	$100-200
Extra (capacitors, resistors, diodes, fuses)	$20
PCB Order + shipping and tariffs	$120
Total	$595

DAQ:

CAN transceivers and other CAN parts	$80

GPS & IMU, high accuracy and sampling speed	$20
Driver Display, good in sun and energy efficient	$10
Arduino or STM processors, cheap and small	$20
PCB order + tariffs and shipping	$55
Total	$165

Stretch: LoRa chip, Antenna, Receiver on a Protoboard: $80

Total cost: $840 + 20% buffer = $1,008

Implementation Issues

There are a few possible implementation issues that we might run into:
Mechanical Issues: UMSM is currently upgrading various parts of the Cedar mechanical subsystem. There is still currently the aerobody with the wheel hubs attached which can be used as a test bed. However when the Mechanical Systems subteam will be integrating the new upgrades, there will be a few weeks where we will not have complete access to a functioning vehicle.
Safety Issues: Our testing stretch goal is to use the vehicle and run it on MCity. This comes with its own inherent risks of having a human inside the vehicle. UMSM has safety procedures that we follow at every test. The 473 team will also come up with a safety guideline for when we have a human in the vehicle test.

Testing Issues: UMSM works with MCity and DPSS when conducting tests. We will have to make sure that we have access to parking lots or the MCity facility.

Energy budget – The goal is to use as little battery power as possible to move our vehicle. This will require the team to select and operate sensors, computing hardware, and other electrical components efficiently, as well as by managing power.

Measures of Success:

The measure of success for the project is if the vehicle is able to compete and complete a valid run at the Shell Eco-Marathon in spring 2027. This final test and competition will be conducted at the Indianapolis Motor Speedway by completing a successful competition run. This larger success metric involves the entire UMSM team to pass a rigorous technical inspection and requires all other parts of the vehicle to be working which is not in scope for the 473 project.

A more attainable measure of success is to be able to create as much torque as needed to accelerate the vehicle to 17 MPH while minimizing power draw. This will be tested by

integrating our system with the existing UMSM vehicle, Cedar. This will allow us to have a testing environment and will allow us to integrate and test without relying on UMSM to complete anything. The initial integration tests will initially be with the vehicle suspended in the air and future tests will be conducted with the vehicle on the ground at the orange lot near the FRB or at the MCity testing facility with cooperation of the UMSM team.

References

[1] Union of Concerned Scientists. (2025). *Car Emissions and Global Warming*. Union of Concerned Scientists. https://www.ucs.org/resources/car-emissions-global-warming

Proposal 2:

Shower Operator

Have you ever forgotten to shower because you're too busy working on an EECS project? Have you gotten water damage on your phone for bringing it into the shower? Have you ever wanted to scroll reels while in the shower? Well wait no more, the solution to your problems is here: The Shower Operator. This product will make showering the most enjoyable experience you've ever had.

Project Needs:
- The ShowerOperator will be able to accurately sense the water temperature and display it to you on a digital display

- The ShowerOperator will have the ability to wirelessly connect to phones and play music
- The ShowerOperator will have a way for users to program in "shower flows" and upload them to the device i.e.
    - Cold for 5 mins and then slowly increase temperature
    - Ice Shower for 3 minutes with Mo Bamba playing from the speakers
    - Hot shower for 15 minutes with temperature fluctuating and playing a specific user defined playlist or podcast
    - Start shower while away and wait for it to reach desired temperature
- Ability to scroll instagram reels on the HUD.

Target Audience:

- High performance athletes that don't have the money or space to spend money on a dedicated ice bath
    - Allow them to use their shower as an ice bath without changing the temperature excessively to aid with rest and recovery after strenuous training
- People who use their phones in the shower
    - All the things you would want your phone in the shower to do, the shower operator is able to do – and better

PCB:

- Temp sensor, bluetooth module, haptic sensors, display, cheap and simple processor

App:

- Ability to set and upload shower schedule, ability to upload music

Requirements:

- Waterproof
- Easy to use UI/App
- Lightweight
- Small and mass produceable

PROPOSAL 1: Drone Detector

High-level description

Drone warfare has become an indispensable part of modern warfare and modern military structures. A drone's advantage lies primarily in its low risk and cost efficiency compared to drone defense systems, therefore a crucial demand exists for a low-cost drone defense system. Military-grade counter-drone platforms cost tens of thousands to hundreds of thousands of dollars per unit and still suffer from high false-positive rates and degraded performance in bad weather (1)(2). The cost mismatch can be extreme: attack drones can cost as little as $20,000 to $50,000 to build, while a single interceptor missile used against them can run $4 to $15 million, a cost exchange ratio sometimes over 100 to 1 (3)(4).

My solution involves working with the open source Flipper Zero, which can be used to detect RFID signals, sub-GHz radio protocols, and other data. The Flipper Zero runs on an STM32WB55 microcontroller with FreeRTOS and reads sub-1GHz radio (300 to 928 MHz via a CC1101 transceiver) along with 125 kHz RFID and other short range wireless protocols (5)(6). Rather than buying a full Flipper Zero for every node, we would port the relevant open source sub-GHz signal reading logic from its firmware onto a cheaper STM32 or ESP32 based board, since the full device (about $199 retail) is overkill for a fixed sensor node (6).

The embedded aspect involves extending this with a "mesh" or "nexus" of peripheral nodes that communicate with each other to cover a wide range and process incoming signals to alert users to drones in range. This is the same "many cheap sensors plus software filtering" philosophy already used in the real world. Ukraine's "Zvook" acoustic sensor network, for example, uses roughly 10,000 low-cost sensors (about $500 each) and machine learning to cut its false-positive rate from 50 percent down to 1.6 percent (7). We're aiming for something similar but using RF sensing instead of acoustic, since RF hardware is cheaper to source at hobbyist scale. Solar batteries could be used to extend the run time and expand the areas this system could be used in.

This proposal comes with the advantage of being in a new field (drone technology) and using open source resources. The main benefit of this project would be to protect and give warning to communities from unwanted drone activity, without ever transmitting or interfering with anything ourselves. This is strictly a detection and alert system.

Implementation issues

What is your rough plan for getting this working by the Design Expo?

Sept 10 to Sept 23: Group formation and protocol research. Evaluate two or three sub-GHz receiver modules and microcontroller options (STM32 vs ESP32) for cost, power, and range tradeoffs. Begin porting relevant open source sub-GHz decode logic. Draft proposal due.

Sept 24 to Oct 6: Order parts (CC1101 modules, MCUs, RFID readers, batteries, solar panels). Get a single node prototype reading a basic RF signal and printing it to a screen. Final proposal due.

Oct 7 to Oct 25: Build a two or three node mesh. Get inter-node communication and a shared alert protocol working on a breadboard. Order PCBs, keeping in mind the roughly 10-day delay from order to arrival.

Oct 26 to Nov 10: While PCBs are in transit, finish firmware including detection logic, alerting, and (stretch goal) camera snapshot capture. Begin weatherproof enclosure design.

Nov 11 to Nov 25: Assemble and test PCB-based nodes, debug hardware issues, run outdoor range and reliability tests.

Nov 26 to Dec 9: Final integration testing, poster, and Expo prep.

What is your rough budget?

Team of 4 students, $250 per student, $1000 total budget.

What	Quantity	Cost per	Cost total
CC1101 sub-GHz module	4	$8	$32
STM32 "Blue Pill" MCU board	4	$6	$24
125 kHz RFID reader module	4	$10	$40
ESP32 board (mesh/comm)	4	$12	$48
Custom PCB fabrication	4	$15	$60
Camera module (stretch goal)	1	$15	$15
Solar panel + LiPo + charger	4	$25	$100
Enclosures, antennas, misc	1	$100	$100
Shipping	1	$50	$50
Total			$469

This leaves headroom under the $1000 budget for a revised PCB order or better weatherproof enclosures if the first pass has issues.

Are there any other obvious implementation issues?

The main implementation issues would be maintaining cross communication with a notable amount of node modules and being able to process and send/receive data quickly enough to alert users of drones in the area. We may fall back to a simple star topology (all nodes report to one hub) as a stretch goal descope if a full mesh proves too time consuming.

Weatherproofing is a must to ensure reliable outdoor operation. We plan to use off the shelf IP65 enclosures rather than designing custom ones, to keep this from becoming its own subproject.

Testing will need to be scheduled in advance to avoid legal issues, since drone tech and related fields are highly regulated. Our hardware only listens, it does not transmit, so any bench testing of our own test signals will be done at low power indoors or in a controlled area.

False positives are a real concern too. Distinguishing an actual drone RF signature from other 300 to 928 MHz traffic (garage doors, weather stations, etc.) is a genuine unknown. Industry sources note this is one of the most underreported challenges in real counter-drone systems, so we are not expecting to solve it perfectly, only to reduce it meaningfully (8).

Evaluation and/or testing with regard to real world issues

We will evaluate the system two ways. First, a controlled bench test using a signal generator or a second radio transmitting known sub-GHz patterns at low power indoors, to confirm each node reliably detects and logs the event. Second, an outdoor field test placing two or three nodes around a small open area and using a personally owned hobby drone (flown legally, in an approved area, at a safe altitude) to see whether the system detects its approach with a low false-positive rate compared to a control period with no drone present. Success means consistent detection within a set range, alert latency under a few seconds, and a false-positive rate low enough that a non-expert could rely on it.

PROPOSAL 2: Nutty-Putty-Buddy!

Cave divers are always doing silly things like getting stuck in a crevice. The 2009 death of John Jones in Utah's Nutty Putty Cave is the grim real-world version of this, he became wedged in a passage as narrow as 10 by 18 inches and could not be extracted after about 27 hours, which is why the cave is now permanently sealed with concrete (9)(10). Why don't we help future cavers out by building a robot that listens for their calls and navigates toward them, creating a path that rescuers can follow?

Search and rescue teams already use acoustic sensing, sound tracking, and small tracked robots to find people trapped in confined spaces after earthquakes and mine collapses, since a person's voice or knocking is often the only way to locate them when visual and radio contact fail (11)(12). Nutty-Putty-Buddy takes that same idea underground: a small, low-profile robot with a microphone array that listens for a trapped spelunker's voice or knocking and crawls toward the strongest signal, leaving a trail rescuers can follow through the maze of passages.

We can install a camera and some sensors that detect how stable the rock or ground is around the robot so rescuers don't also get stuck. We can also program in some stock phrases or sounds to soothe the trapped spelunker, similar to how existing search and rescue robots use onboard speakers for two-way reassurance with victims (13).

The main risk here is mechanical, not electrical. Building something that can actually squeeze through cave sized gaps (often under 18 inches) is a much harder design problem than the sensing and software involved, and we would not realistically be able to test this in an actual cave given access, safety, and liability concerns. Evaluation would be limited to a mocked up narrow passage test rig on campus, tracking a hidden "victim" making noise, rather than a real cave trial.

SOURCES

1. https://oem.flir.com/learn/discover/thermal-infrared-sensor-design-considerations-for-counter-uas-defense/
2. https://www.quickset.com/how-counter-uas-systems-work/
3. https://www.epc.eu/publication/the-new-economics-of-warfare/
4. https://missilestrikes.com/blog/asymmetric-warfare-cheap-drones-expensive-interceptors
5. https://en.wikipedia.org/wiki/Flipper_Zero
6. https://docs.flipper.net/zero/development/hardware/tech-specs
7. https://drone-warfare.com/counter-uas/acoustic-detection/
8. https://www.robinradar.com/resources/10-counter-drone-technologies-to-detect-and-stop-drones-today
9. https://en.wikipedia.org/wiki/Nutty_Putty_Cave
10. https://www.sltrib.com/news/2018/07/09/nutty-putty-i-really/
11. https://www.sciencebuddies.org/science-fair-projects/project-ideas/Robotics_p048/robotics/sound-tracking-robot
12. https://www.nature.com/articles/s41598-025-16003-7
13. https://amprius.com/rescue-robots/

Advanced 3-Phase Motor Inverter + Integrated Data Acquisition for Electric UM Supermileage Vehicle

High Level Description

The Shell Eco marathon is a competition in which many schools come together to compete and race their vehicles to see who can achieve the highest efficiency. Schools choose to either run an internal combustion engine or a motor based on their desires and what category they think they can win in. The University of Michigan Supermileage team is competing in April 2027 of the shell eco marathon with an electric powertrain and needs a high efficiency, variable speed controller along with a real time vehicle monitoring system. We propose designing and fabricating an integrated system combining a 3-phase inverter with closed loop motor control and a full data acquisition (DAQ) system that will replace the teams current motor control and data logging.

The Problem

Most supermileage vehicles operate with significant limitations: 1) simple motor controllers lacking efficiency 2) slow manual data logging without real time diagnostics 3) motor control and data systems lack unification 4) lack of adjustments during the race to no real time monitoring/limit analysis. For an electric car to be utilized to the most of is capability and at efficient manner, these problems need to be addressed

The Solution: Two Integrated Systems

Motor Inverter and Controller: 3-phase MOSFET bridge with trapezoidal commutation. Hall-effect encoder feedback, current sensing capabilities, temperature monitoring with emergency auto shutdown, on board power supply step down (48V,12V,5V,.3V), CAN communication, and throttle information from the DAQ.

Data Acquisition System(DAQ): Real time driver display LCD, GPS for position and pace, wheel encoders for speed measurement, SD card data logging, CAN hub, receiving inverter data, and wiper control. Other nice to haves include: information on steering angle, brake pressure, and LoRA.

These systems will be integrated via CAN bus as one platform, providing closed loop motor control optimized for efficiency plus comprehensive vehicle monitoring.

Timeline

Phase 1 Planning: 9/10-9/23, Define CAN architecture, select MOSFET/GPS/MCU/LCD components, design power budget. Deliverable: Architecture Diagram and component list

Phase 2 PCB Design 1: 9/24-10/15, First schematic design for inverter and DAQ PCBs, high-current routing, EMI shielding, order PCBs by 10/15. Deliverable: First iteration of PCB designs

Phase 3 Build and Integrate: 10/16-10/23, Assemble both PCBs, test power supplies/converters, verify CAN communication, prototype subsystems. Deliverable: Both systems working in benchtop testing

Phase 4 PCB Design 2(if needed): 10/23-10/31, Buffer period allowed if a 2nd revision of PCBs if needed. Deliverable: Second iteration of PCB designs

Phase 5: Software and Control: 9/24-11/07, Can be worked on simultaneously as other members are designing PCBs, Implement closed-loop motor control, data logging firmware, LCD interface, CAN protocol, efficiency, optimization. Deliverable: Fully functional firmware

Phase 6: Vehicle Integration: 11/08-11/15, Design enclosures, install systems in vehicle, route CAN/power, thermal verification, efficiency, characterization. Deliverable: Both systems installed and verified

Buffer Period: 11/15-12/4, Grace period to allow for any extra time that would be needed throughout the process or unforeseen circumstances

Phase 7: Final Testing: 12/4-12/9: Finish all the testing/reports and be prepared to present and demo to the expo. Deliverable: Final report and competition ready system

Budget

A super accurate budget is not realistic at this time, due to the components not being finalized and honed in with the team yet. We will also need to buy spare parts, how much will be determined later, there is plenty of room in our budget to add this later. Our group contains 5 people so our total budget. The numbers are from a first rough passover before group discussions

Category	**Item**	**Quantity**	**Cost/Unit**	**Total**

Inverter: Power	MOSFET (Sic 48v)	8	$8	$64
	Gate Drivers	3	$2.50	$7.50
	Hall Sensors	3	$3	$9
	Current Sensor	2	$5	$10
Inverter: Control	STM32L4R5ZI T6 MCU	2	$18.60	$37.20
	CAN Transceiver	2	$3	$6
Inverter: Power Supply	DC DC converters	3	Varies	$30
	Temp Sensors	2	$1.50	$3
DAQ: Sensors	GPS/IMU module	1	$25	$25
	SD card module	1	$12	$12
	LCD display	1	$15	$15
	Encoder Interface	1	$10	$10
	STM32L4R5ZI T6 MCU	2	$18.60	$37.20
	CAN transceiver	1	$3	$3
PCB and Assembly	PCB Manufacture	2 Designs + Revisions	-	$300
	Passive and Connectors	-	-	$40
	Heatsinks and thermal	-	-	$20
	Enclosures (2x)	-	-	$35
	Wiring and Hardware	-	-	$20

Total: $683.90 out of $1250, plenty of rooms for spares and extra revisions

Other Implementation Issues

Thermal Management: The MOSFETs will handle peak currents of ~200A at 48V, dissipating approximately 40-60W during continuous full power operation. To manage the heat, we will mount the MOSFETs on an aluminum extrusion heatsink with thermal interface material (total cost around $20). The inverter enclosure will be designed with ventilation holes to allow airflow and prevent thermal buildup. If testing reveals inefficient cooling, we have the budget remaining to add a small cooling fan/cpu cooling fan if needed

Safety Considerations: Our 48V system presents several safety considerations that we will address in the design. The inverter will include input fuses to protect against short circuits and a capacitor bank to suppress voltage spikes that could damage components. We are using isolated gate drivers to ensure clean and safe MOSFET switching even under fault conditions. Most critically we will implement temperature monitoring firmware that automatically shuts down the system if MOSFET temperature exceeds 100°C, and we will add CAN based alerts before thermal shutdown occurs. All high voltage nodes will be clearly labeled and the enclosure will be designed to prevent accidental contact with live circuits.

Mechanical Design & Facilities: The inverter and DAQ systems use surface mount components , which require careful soldering. The UM Supermileage workspace has the necessary equipment including a reflow station for this work. Both PCB systems will be mounted in aluminum enclosures with proper connector placement for vehicle integration. The inverter enclosure will include heatsink mounting brackets and cooling vents: the DAQ enclosure houses the microcontroller, display connections, and power entry points. No extreme facilities will be needed beyond standard equipment

Component Selection Justification: Our component choices are based on proven industry standards and cost effectiveness. The STM32L4R5ZIT6 microcontroller is widely used in embedded motor control applications and provides sufficient GPIO, ADC, and PWM channels for our requirements It is also the chip the UM Supermileage team has experience with previously. As of now we have selected SiC (Silicon Carbide) MOSFETs for their low on resistance and switching losses at 48V reducing thermal dissipation compared to traditional silicon devices, which is critical for efficiency in a supermileage vehicle. The GPS/IMU module and LCD display are standard components used in automotive and robotics applications. Similar 3-phase controller designs exist on GitHub and in open source commercial product designs, demonstrating that this approach is proven and feasible.

Evaluation and Testing

Criteria for Success will be as follows:

Motor Control: Variable speed operation (5%-100% throttle), speed error < 5%, efficiency > 90%, response time <100ms, MOSFET cooperating temperature <80 degrees celsius.
DAQ System: Accurate LCD display, GPS +- 5m accuracy, wheel encoders +-3% speed error, SD card logging, without corruption, zero missed CAN messages
Integration: 30+ min of continuous operation without faults or autostops, testing all protection features to ensure full operation

We will measure system efficiency by recording both input power and mechanical output power. Input power is calculated from battery voltage (measured with an analog input) x current drawn (measured with a shunt resistor sent to the ADC). Motor mechanical output power will be measured using a dynamometer, which provides a torque reading at various speeds. By comparing electrical input electrical power to output mechanical power across the full throttle range, we can calculate end to end system efficiency and validate our >90% efficiency target. This data will also help us optimize the control algorithm for different operating points in the vehicle

Testing plan will be as follows:
Bench: Power Supply verification, PWM & gate driver testing, CAN loopback, GPS/LCD functionality, smaller scale motor validation
Vehicle: Install systems, test CAN integration, verify throttle response, LCD accuracy, conduct range test, compare efficiency vs previous years and targets.

Suspension Dynamics Data Logger and Street Performance Analyzer

High Level Description

While the Shell Eco marathon takes place on a smooth, paved circuit, the UM supermileage vehicles are ultimately designed to be street legal and fully operational on real roads. Real world street driving presents completely different challenges: potholes, speed bumps, uneven payment, and rough terrain that the competition track never encounters. We propose building a real time suspension dynamics monitoring system consisting of four wireless accelerometers (one per wheel) plus a central gyroscope and data logging unit to record suspension motion during real world street testing

The real world application: As the team develops and tests the vehicle on public roads and varied urban terrain, the vehicle will encounter: smooth highways, bumpy city streets, potholes, railroad tracks , speed bumps, etc. Each impact will create a vibration signature that reveals: 1) road hazards 2) suspension health (oscillating, damping) 3) tire condition 4) driver feedback 5) reliability issues. By logging this data during street testing, the team validates that their suspension design is robust enough for real world operation, not just the paved marathon track.

Implementation Overview

Hardware: 4 wireless accelerometer nodes (ADXL335 + nRF24L01 per wheel, powered with a coin cell) and a central STM32L4 with 9-axis IMU, nRF24L01 receiver, and SD card logging. Sampling at 200 Hz per accelerometer. Weatherproof 3D printed enclosures for wheel nodes; main logger mounted in vehicle.

Software: Wireless mesh protocol via nRF24L01 with automatic retry. Real-time SD card logging through DMA. Post-drive python analysis script filters acceleration data, detects road hazards via peak detection, visualizes suspension dynamics. Comparative analysis supporting suspension response across different roast types/scenarios.

Budget:

Component	Cost
4x Accelerometer + wireless nodes	$40
Batteries + holder	$10

9-axis IMU	$8
SD Card module	$5
STM32L4 chip	$15
3D printed enclosures	$12
Wiring, misc	$10
PCBS + spares	$150

Evaluation and Testing

Success Criteria:

- 4 wireless nodes transmit reliably for 30+ minutes while driving with less than 5% packet loss
- 200 Hz acceleration data logged to SD card without corruption
- System detects real world hazards (potholes, speed bumps) during street testing
- Post drive analysis produces readable CSV with suspension comparisons

Testing:

- Vibration testing
- Bench wireless range and battery life
- Multiple street tests drive on smooth, rough, and hazardous roads
- Post drive data analysis comparing suspension response

Why this is important: For a street legal vehicle, suspension performance, especially in a state like Michigan is critical. The marathon track does not test suspension robustness. This project provides real world data for vehicle validation and bridges competition design with actual street operation.

Stretch goal ideas: Real time FFT analysis for resonance detection. GPS time stamping for hazards. Machine learning classifier for road type identification.

Proposal 1: Low-Cost Blind-Spot and Rear Collision Warning System for Bicycles and Electric Scooters – Wei Wen Wang

Bicycles and electric scooters are increasingly used for short-distance transportation, particularly around college campuses. However, riders of these vehicles have significantly less protection and situational awareness than driver of automobiles. A rider generally has no mirrors, blind-spot monitoring, or active safety system to warn them when a vehicle is approaching rapidly form behind. Looking backward also requires the rider to temporarily take their attention away from the road ahead and may make it more difficult to maintain control of the bicycle or scooter.

We propose a low-cost blind-spot and rear collision warning system for bicycles and electric scooters. The system would use one or more rear-facing sensors to detect vehicles or other large moving objects approaching from behind. A microcontroller would process the sensor measurements in real time to estimate the distance and relative motion of an approaching object. If the system determines that the object presents a potential danger to the rider, it would provide a warning using haptic and/or visual feedback.

Details

The primary vehicle-mounted system would contain a rear-facing ranging sensor, a microcontroller, and a method of communicating warnings to the rider.

A radar-based sensor appears to be a promising option because radar can potentially provide both range and relative-velocity information and is less dependent on lighting conditions than a camera. However, other ranging technologies such as compact LiDAR or Time-of-Flight sensors could also be considered during the initial feasibility study.

A simplified system would operate as follows: The processor would periodically obtain measurement from the rear-facing sensor. At minimum, the system would attempt to estimate the distance between the rider and an approaching object. If the selected sensor can directly measure relative velocity, that information could also be used. Otherwise, relative velocity could potentially be estimated from consecutive distance measurements.

For example,

$$v = -\frac{d(t) - d(t + \Delta t)}{\Delta^t}$$

The system could then calculate an approximate **Time-to-Collision(TTC):**

$$TTC = \frac{d}{v}$$

The exact warning thresholds would be determined experimentally. One possible scheme would use multiple levels:

Condition	Rider Feedback
Object detected but low closing risk	No warning/low-intensity LED
Vehicle approaching	Visual indication
Low estimated TTC	Haptic warning
Very low TTC	Stronger or repeated haptic warning

A haptic interface is particularly attractive because a bicycle or scooter rider should not need to look at a display while moving. Small vibration motors could potentially be mounted near the handlebars, where the rider can easily perceive them.

If the sensing system can determine whether an approaching vehicle is primarily on the left or right side, separate left/right haptic motors or LEDs could indicate the direction of the threat.

Schedule

Date	Task
9/10–9/23	Planning and sensor research. Define required detection range, field of view, update rate , and warning time. Research two or three radar/ranging platforms and compare them based on range, relative-velocity capacity, power, cost, size, and interface requirements.
9/24–10/15	Part selection and working prototype. Order parts after the staff meeting. Bring up the sensor, haptic/visual warning devices, and their hardware interfaces on a development board.
10/16–11/3	PCB design and software development. Complete and order the custom processor PCB.
11/4–11/10	PCB assembly and integration. Assemble and bring up the PCB, verify sensor and warning interfaces, and transfer the software from the development-board prototype.
11/11–11/23	System integration and testing. Complete the enclosure and bicycle/e-scooter mounting system. Test detection range, warning timing, and overall system reliability in controlled scenarios.
11/24–12/9	Final testing and Design Expo preparation. Debug the completed system, evaluate performance metrics, complete stretch goals if time permits, and prepare the final demonstration and poster.

Budget

Component	Approximate Cost
2-3 candidate radar/ranging sensors	150-250
MCU development board	50-75
Custom PCB fabrication	75-125
PCB processors and components	75-125
Haptic motors, LEDs	30-50
Battery and power electronics	40-60
Enclosure and mounting hardware	50-100
Connectors, cables, miscellaneous parts	50
Spare/replacement components	75
Estimated total	$595-910

Issues

1. Reliably detecting vehicles approaching from behind.
2. Distinguish dangerous approaching vehicles from stationary or irrelevant objects.
3. Determining whether the system can provide useful left/right blind-spot information.
4. Providing sufficient warning time at realistic closing speeds.
5. Mounting the sensor reliably on both bicycle and electric scooters.
6. Safety testing the system with moving vehicles.

Issue 1 is the most important question. The sensor must detect approaching vehicles at a sufficient distance to provide a useful warning. Issue 2 can potentially be addressed using relative motion. Issue 3 depends on the selected sensor. If a single sensor cannot reliably determine the direction of an approaching vehicle, two sensors with different viewing directions or a sensor with angular information could be considered. Issue 4 requires sufficiently fast sensing and processing. Issue 5 is primarily mechanical. Issue 6 will be addressed through controlled testing. Initial experiments can use a stationary bicycle or scooter while another bicycle, cart, or vehicle approaches from behind.

Evaluation and Testing

The system will be tested in a controlled environment by placing the device on a bicycle or electric scooter while another bicycle or vehicle approaches from behind at different speeds and positions.

The primary measures of success will be like:

Metric	Target
Approaching vehicle detection rate	>90%

Warning latency	<100m
Warning time before potential collision	>= 2s

The system will be considered successful if it can reliably detect an approaching vehicle and provide the rider with a warning early enough to react.

Proposal 2: Road hazard detection and warning system for bicycles and electric scooters

Poor road conditions can be dangerous for bicycles and electric scooters because their small wheels are more vulnerable to potholes, bumps, and uneven pavement. Riders must monitor the road surface while also paying attention to surrounding traffic and pedestrians, making some hazards difficult to notice in time.

Detail

Here I propose a real-time road hazard detection and warning system that detects significant road irregularities ahead of a bicycle or electric scooter and warns the rider before reaching them. A forward/downward-facing Time-of-Flight sensor of compact LiDAR would measure the road surface, while an Inertial Measurement Unit would estimate the vehicle's orientation and help distinguish actual road irregularities from changes caused by vehicle pitch and vibration.

The sensor measurements would be processed by a microcontroller to estimate the road profile ahead. If a significant pothole, depression, or bump is detected, the rider would received a haptic, visual, or audible warning. Vehicle speed could also be considered so that hazards receive more urgent warnings when the rider is approaching them quickly.

Development could begin with an Arduino or similar development board to evaluate the sensors and detection algorithms. If successful, the processor, IMU, power circuitry, sensor interfaces, and warning-device drivers would be integrated into a custom PCB and packaged as a compact bicycle/scooter-mounted device.

Issues

1. Whether a low-cost distance sensor can detect road hazards far enough ahead.
2. Distinguishing potholes and bumps from vehicle pitch and vibration.
3. Finding a sensor mounting angle that provides useful road measurements.
4. Providing enough warning time at realistic riding speeds.

The first issue would be investigated early by testing candidates ToF/LiDAR sensors on road irregularities of different sizes. IMU measurement and filtering could compensate for vehicle motion. If sensing range is limited, the intended operating speed of the system could be reduced.

Evaluation

The system would be tested on a controlled course containing flat pavement and artificial potholes or bumps of known sizes.

Metric	Target
Hazard detection rate	>90%
Warning latency	<100 ms
Warning time	>=0.5s

The system would be considered successful if it can reliably detect significant road hazards and warn the rider before reaching them.

Strech Goal

As a stretch goal, detected hazards could be associated with GPS location through a connected smartphone and stored on a shared road-hazard map. This could eventually allow other riders to receive warnings about known potholes before reaching them.

EECS 473 | HW #0 | PROJECT PRE-PROPOSAL 1

Contactless Laser Keyboard for Interactive Music Learning

High-level description

A first encounter with an electronic instrument should make the relationship between movement and sound easy to explore. I propose a tabletop keyboard whose keys are invisible optical paths: placing a finger into a path plays a note, and lifting the finger releases it. The intended users are beginning musicians and visitors to classroom or public demonstrations. Labeled note positions and indicator lights would help them play short melodies and experiment with chords without pressing mechanical keys.

The finished device would contain eight parallel laser paths corresponding to C4, D4, E4, F4, G4, A4, B4, and C5. Each emitter would face a receiver across a rigid frame, with the paths approximately 3 cm apart and 15-20 cm long. Fingers would enter from above, a few centimeters above the base. The enclosure would include a speaker, volume control, power switch, and note LEDs. An external 5 V supply would let the instrument operate without a computer. These dimensions are starting points to be refined through user testing.

Laser instruments already exist. Beamz demonstrates laser-triggered music control, while an Adafruit laser harp uses distance sensors to produce MIDI events [1,2]. This proposal therefore focuses on a specific engineering contribution: a compact, self-contained instrument with reliable beam-break detection, simultaneous notes, and measured playing latency. It would be an interactive learning and performance device; it would not attempt to reproduce the pressure sensitivity or full range of a piano.

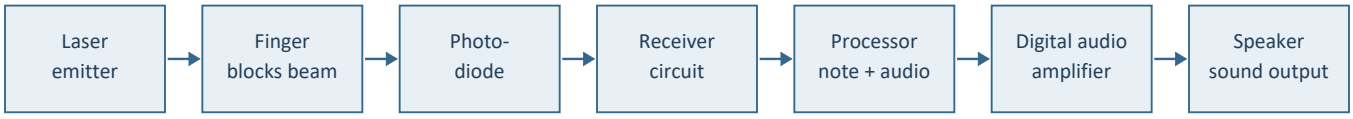
Implementation issues

Detection and sound generation

Each channel would use a photodiode, a current-to-voltage amplifier, and a comparator with hysteresis. A BPW34 photodiode is a plausible candidate [3]. The controller would accept a transition only after it remains stable for approximately 2-3 ms, then generate a note-on or note-off event. A held finger would sustain one note rather than repeatedly restarting it. Shading around the receivers and adjustable thresholds would address ordinary room lighting. If needed, synchronized emitter modulation and receiver detection would be investigated after selecting a compatible driver.

An RP2040 is a candidate processor: its programmable I/O can implement the digital audio interface [4]. Firmware written in C/C++ would mix independent waveform voices at a proposed 48 kHz sample rate, with attack and release envelopes to avoid clicks and sufficient headroom to avoid clipping. An I2S amplifier such as the MAX98357A would drive the speaker [5]. The required result is eight selectable notes and at least four-note polyphony. Eight-note polyphony, USB MIDI, and LED-guided melodies are stretch goals.

Signal path



Schedule

The following six stages begin after group formation. Week ranges are planning estimates, not course deadlines; they would be adjusted to the announced PCB and Design Expo dates. The target is a fully playable device at least one week before the Expo.

Stage	Work and completion milestone
Weeks 1-2 Feasibility	Verify one optical channel under several indoor lighting conditions. Review emitter safety documentation, compare receivers, and measure detection delay. Select a feasible optical arrangement before scaling up.
Weeks 3-4 System prototype	Build eight channels on development hardware. Demonstrate four simultaneous notes, clean note release, and adjustable volume. Test spacing with users and freeze the core feature set.
Weeks 5-6 Design and order	Design and review a processor-based main PCB and receiver boards. Include programming access, test points, and adjustable optical mounts. Order PCBs and remaining enclosure parts.
Weeks 7-8 PCB transit	Allow approximately 10 days for PCB delivery. Continue firmware and latency tests on the prototype, build the enclosure, and prepare assembly and test procedures.
Weeks 9-11 Integration	Assemble and bring up the custom boards. Align optics, resolve audio interference, and run multi-channel tests. Reserve time and funds for one PCB correction if needed.
Weeks 12-13 Validation and Expo	Complete reliability and user trials, fix remaining defects, and freeze the build. Prepare the poster and demonstration; add stretch features only if core acceptance tests pass.

Budget

For an assumed five-person team, the course allowance is $1,250 ($250 per student). The following $660 planning estimate covers one instrument and limited spares. These are allowances, not vendor quotes; safety-qualified optics and PCB assembly costs require early confirmation. Existing laboratory instruments are assumed available.

Item	Estimated cost
Eight emitters and compatible drivers	$100
Photodiodes, amplifiers, comparators, and passives	$60
Processor, Flash, power circuitry, and connectors	$55
Audio amplifier and speaker	$35
PCB fabrication and partial assembly	$130
Frame, adjustable mounts, and light shields	$70
Supply, wiring, buttons, and LEDs	$40
Shipping, spare parts, and PCB correction reserve	$170
Total	**$660**

The estimate leaves $590 below the assumed team ceiling. Optional displays and advanced sound features would be dropped before reducing the core detection, enclosure, or testing work.

Mechanical, safety, power, and production risks

Keeping all eight beams aligned will probably be the most difficult part of the frame design. If an emitter moves, the receiver may interpret the missing beam as a held note. I would use mounts that can be adjusted during setup and then tightened in place. Small boards next to the photodiodes would handle the weak receiver signals, while the processor and audio output would sit on the main PCB. Separating these circuits should help keep speaker noise out of the detection signals. The frame could be 3D printed or made with basic workshop tools, and the electronics could be assembled and tested with the soldering equipment, oscilloscope, and logic analyzer available in the lab.

The lasers need to be chosen carefully because players will put their hands into the beams. I would look for a documented Class 1 option [6] and check the complete arrangement with the course staff before testing it with users. The beams would end inside recessed receivers, and the frame would have no shiny surfaces near the light paths. Opening a rail for adjustment would turn the lasers off. Note labels and LEDs would show where to play, so visible beams in the air would not be necessary.

For power, I would start with a regulated 5 V, 2 A adapter and measure how much current the instrument draws at full volume. This gives a 10 W supply limit without adding a battery or charging circuit. Room lighting will also need testing: if a receiver saturates, I would first improve its shading or reduce the amplifier gain. At startup, the player would leave all paths clear while the controller checks the receivers. This should help catch a blocked path, a failed emitter, or poor alignment before playing.

Evaluation and testing with regard to real world issues

Success requires both correct electronics and usable musical interaction. The following are proposed acceptance criteria, not measured results. Tests would use the final enclosure and custom PCB.

Question	Test and proposed success criterion
Does it respond quickly?	Measure optical interruption and electrical audio onset on an oscilloscope across at least 100 events. Target 95th-percentile latency at or below 15 ms; report the maximum and include envelope/buffer delay.
Are notes reliable?	For each channel, perform 100 block/release cycles under each of three indoor lighting conditions. Target at least 99% correct note-on/note-off pairs and no stuck notes; record missed and duplicate events.
Can it play music stably?	Verify pitch within 1% of each target frequency, four-note chords without clipping, and no audio dropouts during a 30-minute session. Require no false notes during 10 idle minutes in each lighting condition.
Can beginners use it?	Ask five volunteers to play a labeled eight-note sequence after a two-minute introduction. Target at least four completing it with no more than one wrong or missed note within three attempts. Record errors and comments to refine spacing.

References

[1] Beamz, User Manual (controller features).

[2] Adafruit, MIDI Laser Harp with Time of Flight Distance Sensors.

[3] Vishay, BPW34 / BPW34S photodiode datasheet.

[4] Raspberry Pi, RP2040 datasheet.

[5] Analog Devices, MAX98357A product documentation.

[6] FDA, Frequently Asked Questions About Lasers (classification).

EECS 473 | HW #0 | PROJECT PRE-PROPOSAL 2

Defuse Together:
A Cooperative Puzzle Box

High-level description

Imagine trying to stop a countdown when the person who understands the instructions cannot see your controls. I propose a tabletop puzzle box for two players: an operator who sees the puzzles and handles the buttons, and a decoder who sees the rules. They would sit on opposite sides of a divider and explain what they see to each other. Neither side would show enough information to solve the puzzles alone. The "bomb" would be a game prop, with lights and sounds marking success or failure.

The players would have three minutes to finish three puzzles, with three incorrect submissions ending the round. In the first puzzle, the operator would describe a symbol and number, and the decoder would use a rule table to explain which button sequence to enter. The second would require translating a code into a numbered rotary-dial setting. In the third, the operator would describe a flashing light sequence, while the decoder would explain how to convert it into button presses. Completing all three would stop the clock.

The finished box would have two small displays, a divider, labeled buttons, a rotary encoder, indicator lights, and a speaker. It would run from an external low-voltage supply without a computer. This would suit a winter Design Expo: visitors could play indoors at a table, and a shorter demonstration mode would keep the queue moving. All interaction would be between people and the device. The appeal is communicating clearly while an impatient box counts down.

Implementation issues

One microcontroller on a custom PCB would manage both panels, the timer, and the game state. An RP2040 is a possible choice, with C/C++ firmware handling inputs and display updates. A checked question bank would keep each operator prompt, decoder instruction, and correct answer together. Completed puzzles would remain solved until the next round.

The main challenge would be making the instructions understandable without giving away the answer. Early playtests would help adjust the wording and time limit. Button debouncing and reliable encoder counting would prevent accidental extra inputs; an undo button would let players correct an entry before submitting it. Lights and text would accompany sound cues, and lights would be labeled by position as well as color. The enclosure could be made with 3D printing or basic workshop tools. The buttons and wiring would be secured for repeated use.

Schedule

The plan has five stages, with dates adjusted to the announced Expo and PCB deadlines:

Stage	Work
1. Game prototype	Try paper rules and one electronic puzzle with classmates.
2. Full prototype	Build all three puzzles on a development board; test both player interfaces.
3. PCB and enclosure	Design and order the processor PCB; finalize the panel layout.
4. Transit and assembly	Use the roughly 10-day PCB wait for firmware and enclosure work; then assemble and debug.
5. Final testing	Reserve several weeks for repairs and playtests before preparing the Expo demo.

Budget

I expect one complete box to cost about **$420**. The breakdown below is a first estimate; I would check prices once the parts and panel layout are chosen. A five-person team would have $1,250 available at $250 per student, leaving room for a board revision if needed. I would use the lab's existing tools for assembly and testing.

Item	Estimated cost
Controller and power circuitry	$60
Two displays	$60
Buttons, lights, rotary encoder, and sound components	$60
PCB fabrication and partial assembly	$120
Enclosure, divider, and panels	$60
Cables, shipping, and spare parts	$60
Total	**$420**

Randomized rules, extra puzzle modules, and score tracking would be stretch goals. The priority would be three dependable puzzles and a quick reset between players. Keeping the device wired and externally powered would avoid adding wireless setup or battery charging to the project.

Evaluation and testing with regard to real world issues

I would test the finished box myself before asking five pairs of volunteers to try it. The first checks would cover wrong answers and restarts as well as normal play. During the trials, I would watch for confusing instructions and whether both players take part. The goals below are starting points to check on the final PCB and enclosure.

Question	Test and expected result
Does it judge answers correctly?	Try every question with its correct answer and several wrong answers. Each should be accepted or rejected as intended.
Do the controls and timer work?	Check rapid button presses, dial changes, undo, timeout, and the third mistake. Inputs should register once, and the round should end at the right time.
Can it handle repeated demos?	Run 20 rounds in a row without a crash or old answers carrying over. A new round should be ready in less than 10 seconds.
Can new players learn it?	Give five pairs a short introduction. Aim for at least four pairs to finish within two attempts without help. Record their times, mistakes, and questions.
Does it encourage cooperation?	Watch whether both players describe information and make decisions. Ask what was confusing afterward, then revise rules that repeatedly caused misunderstandings.

EECS 473
Project Homework 0

Name: Xinyu Hu
Uniqname: aureoleh
Date: September 6, 2026

PROPOSAL 1: SMART MEDICATION DISPENSER

High-Level Description

People who take several medications or supplements may need to follow different schedules throughout the day. Conventional pill organizers help separate doses, but they are passive: they cannot remind the user, control which compartment is accessed, or tell whether a scheduled dose was removed.

This project proposes a Smart Medication Dispenser, a tabletop embedded system that stores medication in approximately six to eight compartments and automatically provides access to the correct compartment at the scheduled time. The current concept uses a rotating carousel with a single access opening. At a scheduled time, the device would alert the user, rotate the correct compartment into position, verify its position with a sensor, and monitor whether medication likely passed through the dispensing opening.

The device would not make medical decisions or determine what medication a user should take. Instead, the user would enter a schedule in advance, and the dispenser would organize access to the corresponding compartments and record whether an interaction occurred.

The finished system would include a custom microcontroller-based PCB, a real-time clock, motor-control electronics, position and dispensing sensors, a small display, buttons, LEDs, a buzzer, and power-management circuitry. The carousel and enclosure would likely be 3D printed so that the mechanical design can be revised quickly during development.

The main advantage over a conventional pill organizer is that the system combines scheduled access, physical actuation, sensing, and user feedback in one standalone embedded device. It is also a good embedded-systems project because its success depends on the interaction between mechanical design, sensing, firmware, PCB design, and real-world reliability.

Implementation Issues

The largest implementation challenge is likely to be the mechanical dispenser. A rotating carousel is attractive because a single motor can serve several medication compartments, reducing wiring and actuator count. A stepper motor or similar geared motor could rotate the carousel to fixed positions corresponding to each compartment.

The system should not rely only on counting motor steps because a jam, slip, or manual movement could cause the controller to lose track of the carousel position. A physical home-position sensor, such as a Hall-effect or optical sensor, would allow the device to establish a known reference after startup or after detecting a fault. The most important safety behavior

would be to fail closed: if the controller cannot verify the carousel position, it should alert the user rather than expose an uncertain compartment.

Medication removal could initially be detected with an optical break-beam or reflective sensor near the dispensing opening. This would not prove that medication was swallowed, but it could determine whether an object likely passed through the dispenser. If this approach proves unreliable with different pill sizes or shapes, a weight-based sensor could be investigated later.

A microcontroller would coordinate the RTC, schedule storage, motor, sensors, alerts, and user interface. The schedule should remain in nonvolatile memory after power loss, while the RTC maintains the correct time. After a reboot, the device should re-home the carousel before continuing normal operation.

The final custom PCB would integrate the microcontroller, RTC, motor driver, sensor interfaces, user-interface connections, power regulation, and programming/debug interface. Early testing would use development boards and breakout modules so that the mechanical system and sensors can be validated before the PCB design is finalized.

Because the unit is intended for tabletop use, the simplest primary power source would be an external low-voltage adapter. A small battery backup could be a later feature, but the core design would prioritize reliable operation over full portability.

Schedule

9/10–9/24 — Mechanical Prototype and Part Selection

Build a basic carousel prototype and determine compartment size, motor requirements, and a suitable position-sensing method. Select candidate MCU, RTC, dispensing sensor, motor driver, display, and other major components.

9/25–10/8 — Module Testing

Test the motor and carousel for repeatable positioning. Bring up the home sensor, dispensing sensor, RTC, display, buttons, and buzzer individually using a development board.

10/9–10/25 — Integrated Prototype and PCB Design

Combine the mechanical and electronic modules into a working prototype. Implement basic scheduling, carousel movement, position verification, and dispensing detection. Design and order the custom PCB once the interfaces are stable.

10/26–11/8 — Firmware and Mechanical Refinement

While waiting for PCB fabrication, continue firmware development and improve the carousel and enclosure. Add power-loss recovery, error handling, and more robust dispensing logic.

11/9–11/23 — PCB Bring-Up and Final Integration

Test power, processor, RTC, motor driver, sensors, and user interface on the custom PCB. Replace the development-board prototype and integrate the PCB into the enclosure.

11/24–12/9 — Reliability Testing and Design Expo Preparation

Run repeated dispensing cycles and fault tests, fix remaining mechanical and firmware issues, and prepare the final enclosure, demonstration, and poster.

Rough Budget

Microcontroller, RTC, display, and UI electronics — $80

Motor, motor driver, and position sensing — $100

Dispensing sensors and experimental sensor hardware — $60

PCB fabrication and spare boards — $100

3D printing and mechanical hardware — $100

Power components, connectors, wiring, and spares — $120

Estimated Total — $560

For a team of approximately five students, this estimate is well below the course budget of $250 per student. The actual cost could be lower depending on the number of enclosure and PCB revisions required.

Evaluation and Testing with Regard to Real-World Issues

The project should be evaluated primarily on whether it provides the correct compartment reliably and detects user interaction consistently.

First, the carousel would be commanded to move between randomly selected compartments for several hundred cycles. The system should consistently align the requested compartment and should detect faults such as a failed home search or blocked motion rather than silently exposing the wrong compartment.

Second, the dispensing sensor would be tested using test objects with different sizes and shapes. Testing would measure both missed detections and false detections. The goal is not to determine whether medication was actually taken, but whether a dispensing event can be detected reliably.

Finally, the full device would be tested through accelerated schedules and intentional power interruptions. After power is restored, the device should recover the stored schedule, maintain the correct time, re-establish carousel position, and continue normal operation without exposing an incorrect compartment.

A successful final prototype would therefore demonstrate repeated correct positioning, reliable dispensing-event detection, and safe recovery from common faults.

Stretch Goals

If the core dispenser is reliable before the end of the semester, possible stretch goals include Bluetooth-based schedule configuration, medication-removal history logging, missed-dose phone notifications, or a rechargeable backup battery.

PROPOSAL 2: RESCUELINK

A Multi-Sensor Wilderness Emergency Device

High-Level Description

Hiking, camping, climbing, and other backcountry activities can place users in locations where cellular service is unreliable or unavailable. Smartphones can still provide navigation and emergency tools, but continuous GPS, wireless communication, and screen use can drain the battery quickly.

This project proposes RescueLink, a handheld embedded device that combines basic navigation, environmental sensing, emergency detection, and low-data-rate long-range communication. The device could include GPS, a magnetometer, barometer, temperature and humidity sensing, an IMU, a small display, a rechargeable battery, and a sub-GHz radio.

The main engineering idea is not simply to combine many sensors. RescueLink would manage them according to the user's situation and available energy. In normal operation, GPS and radio activity could be limited to conserve battery. In an emergency state, the device could prioritize location updates and emergency transmission even if that reduces the remaining operating time.

A user could save a campsite or trailhead and later view the approximate direction and distance back toward it. Environmental sensors could provide basic information such as pressure trend, temperature, and humidity. The IMU could also help detect a possible impact followed by prolonged inactivity. Before automatically entering emergency mode, the device would ask the user to cancel the alert to reduce false alarms.

The first prototype would demonstrate emergency messages between RescueLink and a separate base receiver using a long-range, low-data-rate radio. A message could contain GPS coordinates, time of the last fix, emergency status, and battery level. Satellite communication would be outside the required scope because of its additional cost and complexity.

Implementation and Feasibility

The largest technical issue is power consumption. GPS, the display, and radio transmission are expected to consume much more energy than the environmental sensors, so they should not remain active continuously. Early prototypes would measure the current required by each subsystem and use those measurements to define operating modes.

GPS could wake periodically to obtain a new position, while low-power environmental sensors and the IMU could remain active more often. During low-battery operation, GPS updates could become less frequent and the display could remain off for longer periods.

The IMU could identify large impacts and subsequent inactivity, but a single impact should not trigger an emergency message because dropping the device could create the same signal. The firmware would therefore combine impact detection, inactivity, and a user-cancel period.

Environmental sensing is relatively straightforward, although sensor placement will matter. For example, the temperature sensor should be separated from heat-producing components such

as voltage regulators or the battery. GPS and radio antenna placement would also need to be considered during enclosure and PCB design.

The final system would use a custom PCB containing the microcontroller, power-management circuitry, sensor interfaces, GPS, radio connections, display interface, and battery-charging circuitry.

Schedule

1. Part Selection and Architecture

Choose the MCU, GPS, IMU, environmental sensors, radio, display, and battery system.

2. Module Testing

Bring up sensors individually and characterize GPS, radio, and power consumption.

3. Integrated Prototype

Implement navigation, emergency detection, and operating modes on a development-board prototype.

4. PCB and Enclosure Design

Finalize interfaces, antenna placement, power architecture, and mechanical design.

5. PCB Bring-Up and Outdoor Testing

Test GPS, compass, radio range, emergency logic, and power use in realistic outdoor conditions.

6. Final Reliability Testing

Refine firmware and enclosure, repeat outdoor tests, and prepare the Design Expo demonstration.

Rough Budget

GPS, IMU, and environmental sensors — $150

Long-range radios and antennas — $120

MCU, display, and controls — $80

Battery and power electronics — $90

PCB fabrication and enclosure — $170

Spare parts and contingency — $110

Estimated Total — $720

For a five-person team, this remains below the available course budget.

Evaluation and Key Risks

GPS would be evaluated using known outdoor locations and by measuring acquisition time after startup and low-power wakeup. Radio communication would be tested outdoors at increasing distances while recording successful and missed messages.

Emergency detection would be tested using controlled normal movement, drops onto a padded surface, sudden motion, and simulated impact followed by inactivity. The main goal would be to detect meaningful emergency-like events without generating frequent false alarms.

Power consumption would be measured in normal, low-battery, and emergency modes. Battery-life estimates would then be based on measured current rather than only datasheet values.

The largest project risks are GPS and radio power consumption, radio range in obstructed terrain, magnetometer interference from nearby electronics, and false emergency detection. These issues are testable early enough that the project scope can be reduced if necessary.

Stretch Goals

If the main system is completed early, possible additions include breadcrumb route logging, communication between multiple RescueLink devices, or solar trickle charging.

Project Pre-Proposal Homework #0
Yagho Franco Grossi Mota
August 17th, 2026

Proposal 1: HealthWear, Elderly real-time health tracker and fall detection wearable

Project description

With the uncontested increase in the percentage of elderly population, together with the sad statistics of elderly abandonement, increasing their quality of life requires, most of all, better automatic health monitoring for disease prevention and quicker automatic activation of emergency services. Most current solutions ride atop of expensive all-purpose solutions (e.g. an Apple Watch), or have only either the health monitoring or the emergency service activation, thus a narrow-focused solution which does both has a good market.

Current developments have shown the power of continuous health data monitoring for prediction of illnesses, as well as for informing doctors' decisions, and maybe in the future even feeding into agentic doctors for a maximally-personalized health experience. As an example, there is an open source algorithm called CardioPPG (Ding, Hu et al., *npj Digital Medicine*, 2026; DOI 10.1038/s41746-025-02228-3) which screens for over 10 cardiovascular conditions from PPG alone. Therefore, having a cheap sensor which tracks these could offer easy access to those health advancements.

Another important point is that elders have increased mortality risk due to untimely arrival of health services, thus some algorithms which could trigger such services upon certain detected health events are most valuable.

My vision is a low power wrist wearable which tracks heart rate and PPG characteristics, is able to detect sudden falls, and can contact the health services upon any heart or fall emergency.This wearable must have a very long lifespan, be water resistant, and fall resistant. An extended goal would be the NFC contactless transmission of the data to a HealthTotem, which is plugged to the wall and is connected to wifi, and in this totem all the algorithmical processing of the health data would happen, as well as the eventual contacting of emergency services.

Schedule

9/10 – 9/23	**Planning and part selection** Group forms 9/9. Lock the system partition (wrist senses and alerts, totem computes) and write the interface contract between the two: packet formats for the BLE bulk sync and the sub-GHz SOS path, and a numerical definition of a "health event." Compare two or three candidates each for the PPG front end, IMU, and sub-GHz radio on power, cost, and driver maturity; nRF52840 + SX1262 is the baseline. Build the energy budget spreadsheet (PPG duty cycle, sync interval, SOS burst) before committing to a battery footprint. Also, do clinical outreach so that we know what evidence is necessary to prove that our sensors meet the real health needs and resolution.Contact MICHR and a Michigan Medicine cardiology or digital-health.
9/24 – 10/6	**Devkit bring-up and algorithm baseline** Proposal meetings 9/24. Order all long-lead hardware by 9/25: nRF52840 DK, SX1262 breakouts, sensor breakouts, Raspberry Pi, LoRa HAT. Get a breadboard wearable talking to a breadboard totem over both radios, even with meaningless payloads. Fall detection development starts on the public datasets in parallel. Measure real current draw on the prototype to validate the energy budget. Schematic capture proceeds alongside, with the RF sections reviewed first since 2.4 GHz and 915 MHz matching and keepouts are what you cannot fix in firmware later.
10/7 – 10/16	**Layout, review, and PCB order** Layout the wearable board, then get it reviewed by staff and by a second pair of eyes on the team. Meanwhile the prototype reaches meaningful function: PPG waveform captured and heart rate extracted, IMU plus barometer fusion separating a sit-down from a fall, an SOS packet crossing sub-GHz and triggering a placeholder alert on the totem. HW1 due 10/14 and Milestone 1 report 10/15 fall inside this block, so protect layout time deliberately.
10/17 – 11/3	**Assembly, bring-up, and respin decision** Fall Break 10/20, midterm 10/21, Milestone 1 meetings 10/22. While boards are in transit, finish the power management state machine, the BLE sync protocol, and the totem-side ingestion and risk computation on the Pi. Tune fall detection to a target false-positive rate on the public datasets, then on staged falls with the prototype. Resolve every mechanical issue in parallel: enclosure CAD, resin print iterations, strap, gaskets, O-rings, and the ePTFE vent membrane if the barometer survived September.
11/4 – 11/24	**Integration and validation testing** Milestone 2 report 11/10, MS2 meetings 11/12. Port full firmware to the real board and run the tests that constitute your actual evidence: PPG heart rate and waveform morphology against the reference pulse oximeter and chest strap, fall detection scored on scripted falls and near-falls (hard sitting, device drop, bending over), end-to-end SOS latency from impact to totem alert, sub-GHz range through interior walls, and measured current draw

	extrapolated to battery runtime. Destructive and high-repetition drop testing uses the breadboard prototype or a weighted dummy in an identical enclosure, never the single real unit. Stretch goals decided by 11/24. Thanksgiving 11/26.
11/25 – 12/10	**Final validation, poster, and Expo** Target project completion 12/1, which is also the in-class poster session. Waterproofing validated by submersion or spray test on a sealed unit. Poster built around measured numbers rather than claims. Final report drafted through the 12/8 lab day. **Design Expo 12/10.**
12/11	**Final report due** *Stretch goals, in attempt order:* ECG electrodes on the case back feeding an open-source screening algorithm; a totem carrier PCB replacing the Pi HAT stack; supercapacitor or solar backup so the totem survives a power cut, which matters because a totem that dies in an outage fails exactly when its user is most at risk.

Budget

The following costs would be a first draft which should be bound to change when we account for things like getting extra components in case some get fried or break. I have made the price distinction between the wearable components, the totem components, and the devboards which would be used strictly for testing and developing the firmware while the PCB is out in production.

wearable	totem	devboard and testbench	TOTAL
239	130	97	$556
Part name	Description	qty	price unit
STM32WB15CC	(MCU+BLE)	1	$5.00
SX1262	Sub GHz module	1	$9.00
MAX30102	PPG sensor	1	$6.00
LSM6DSOX	IMU	1	$6.00
DPS310	barometer	1	$3.00
MAX30208	skin temperature	1	$3.00
GRP604543-1C-3.7	LiPo rechargeable 1000mAh	1	$10.00
ANT162442DT-200	BLE antenna (2.4 GHz)	2	$0.50
	Sub-GHz antenna (915 MHz)	2	$3.00
	passives, ESD diodes, button, haptic buzzer	1	$50
	wearable pcb+stencil	1	$50
	enclosure+aerogel	1	$90
	raspi 4	1	$60
	LoRa hat for raspi	1	$30
	pi psu, sd card, totem	1	$40
	NUCLEO-WB15CC, mcu+ble devboard	1	$22
	SX1262 breakout, subghz	1	$30
	MAX30102 breakout, ppg	1	$10
	LSM6DSOX breakout, imu	1	$12
	MAX30208 breakout, skin temp	1	$10
	DRV2605L breakout, haptic	1	$13
	shipping	1	$90

Implementation Issues

1. Tight power budget. The goal is to minimize the number of charges, but the telecommunications in case of health events, and the PPG component itself are quite power hungry. Passing the compute-heavy tasks to the HealTotem is a way of accounting for this.
2. Distinguishing bumps from real falls using only the IMU is tricky. Either have a better algorithm, or using a barometer is necessary.
3. Waterproof. Falls can happen during any time of the day, specially during showering, so having the wearable be waterproof is quintessential.
4. If we use the barometer+IMU (BIMU), the hole used for pressure might make waterproofing tricky. We could use a cheap aerosol mesh to cover it up while maintaining it hydrophobic.
5. Mechanical constraint, since it must fit in the wrist (or maybe be a necklace), be lightweight, and be waterproof. Might need to find someone with ME background to CAD this up.
6. How to account for the range of the HealthTotem? For occasional datasync, opportunistic BLE might be fine (small distance big payload), but for the emergency service activation some subGHz (big distance low small payload) might be the best option.
7. How to place the sub GHz antenna? We could do a chip antenna, but lose range, a PCB antenna but risky design, or a standard big one and wrap around wrist but also risky.

Evaluation and/or testing with regard to real world issues

The following would be the main benchmarks to be used:

1. For the heart disease identifier, the best we can do without clinical trials in a couple of months is to prove that our collected PPG data is good enough in resolution and feature wise compared to professionally collected data in clinics, then we can use the open source algorithm with highest success rate and “borrow” the validity from the original publishers.
2. For the fall detection, specially if we end up using the BIMU approach (barometer + IMU), we need to test falling from different angles and different heights (a mattress in the ground should be enough) mainly to catch false negatives, as well as falling in different wind conditions, so we can test against false positives.
3. For the range of both the sub-GHz and the BLE, these also should be tested, and a simple “how far away can be and still send the data packages or send the SOS?” should be enough.
4. Waterproofing should be tested accordingly.

Stretch goals

The primary goal to be reached is the fall detection. If we end up having enough people in the group, adding the PPG with the algorithm might be a great idea, but it bears the challenges of having to transfer a big amount of data via bluetooth, thus probably introducing the challenges of making mesh nodes that would relay the package to the totem (imagining our customers would be either retirement houses or elderly with considerably big houses), or maybe even communication to phones; the PPG sensor and its algorithm requires heavy computation and it would need to be done out of the wearable, thus why I assume the totem to be a must have in this case.

Proposal 2: SafeNet, fall detection and person tracking network for retirement houses

If the idea of proposal 1 gets torn apart due to the difficulties, another project proposal would be to focus on the fall detection, and expand to a multi-user system (customer mainly being a retirement home), with the wearable and the totem. Then with the added bandwidth of not having the PPG nor the computing-heavy algorithms, we could add a speaker on the wearable so that the elder could receive and send voice messages walkie-talkie style (specially on the occasion of a fall), and we could add mesh nodes around which the wearable could connect to, to be connected with the totem, expanding the working range of the connection. We could add BLE so it could interact with a phone app for the case of when the user goes out of the mesh grid but still with their phone. And as a stretch, we could also add location within the grid using the mesh nodes.

The vision is basically a elderly home with one single totem, and many elders all wearing the wristband. They know roughly their location through the mesh, and when one of them falls, the fall is detected, and is reported to the totem, being relayed from the wristband to the mesh nodes and downstream to it. The facility talks to the elder to calm him down, and if needed, they can also have him explain where exactly he is, as well as how serious the fall was, so the aid team can act accordingly.

Schedule

The schedule should be basically the same as with the HealthWear, but we would need to focus on developing and testing the mesh nodes as well as the totem and the wearable.

Budget

We would subtract the PPG related cost from the HealthWear budget, probably dial down the computing power of the totem to something lesser, like a Pi Zero instead of a Pi4, but add the costs of the mesh nodes. On top of that, we would need to produce at least 2 wearables rather than only one, if we aim at having the map across mesh network idea implemented.

These mesh nodes could be made using a USB wall adapter, and a ESP32-C3 SuperMini dev board, which can do both BLE and WiFi but more importantly, it can communicate using ESP-NOW, which has no access point and no router, so the nodes can talk to each other and relay the message to the totem without IT connectivity issues of being logged to the home's WiFi.

Cost per node

Item	Cost
ESP32-C3 SuperMini dev board	$2.50
UL-listed 5 V USB wall adapter	$1.50
3D-printed shell	$0.50
Per node	**$4.50**

If we account for about 6 nodes (one of which would be acting as ESP-NOW-to-serial bridge to connect to the Pi), this brings the cost to $27.
The overall budget would look in total like this, thus:

wearable	totem	mesh	devboard and te	TOTAL
368	78		105	677
Part name	Description	qty	price unit	line total
STM32WB55CG	MCU + BLE (I2S for audio)	2	$6.00	$12.00
SX1262	Sub GHz module	2	$9.00	$18.00
LSM6DSOX	IMU	2	$6.00	$12.00
DPS310	barometer	2	$3.00	$6.00
W25Q128	SPI NOR flash, 16MB	2	$1.50	$3.00
GRP604543-1C-3.7	LiPo rechargeable 1000mAh	2	$10.00	$20.00
ANT162442DT-200	BLE antenna (2.4 GHz)	2	$0.50	$1.00
ACAG1204-915-T	Sub-GHz antenna (915 MHz)	2	$3.00	$6.00
SPH0645LM4H	I2S MEMS microphone	2	$3.00	$6.00
MAX98357A	I2S class-D amplifier	2	$3.00	$6.00
CDM-20008	waterproof mylar speaker	2	$4.00	$8.00
GAW332	Gore ePTFE vent, pressure + acoustic	1	$20.00	$20.00
SSD1306	OLED display 0.96in	2	$5.00	$10.00
	bulk caps + separate MCU regulator	1	$4.00	$4.00
	passives, ESD diodes, button, haptic buzzer	1	$60.00	$60.00
	wearable pcb+stencil (5pc min covers both)	1	$50.00	$50.00
	enclosure, 2 shells, straps, seals	1	$120.00	$120.00
Raspberry Pi Zero 2	totem compute	1	$15.00	$15.00
	LoRa hat for raspi	1	$30.00	$30.00
	pi psu, sd card, totem enclosure	1	$30.00	$30.00
ESP32-C3 SuperM	ESP-NOW to serial gateway for totem	1	$3.00	$3.00
ESP32-C3 SuperM	mesh node MCU	8	$2.50	$20.00
	UL-listed 5V USB wall adapter	8	$1.50	$12.00
	node 3D printed shell	8	$0.50	$4.00
NUCLEO-WB55	mcu+ble devboard	1	$35.00	$35.00
	SX1262 breakout, subghz	1	$30.00	$30.00
	LSM6DSOX breakout, imu	1	$12.00	$12.00
	DRV2605L breakout, haptic	1	$13.00	$13.00
	I2S mic + amp breakouts	1	$15.00	$15.00
	shipping	1	$90.00	$90.00

Implementation Issues

1. Waterproofing gets much harder here, since we now have three holes in the case (speaker, mic, button) instead of one. Gore and Donaldson make ePTFE vents exactly for this and sample them for student projects, and the button can go capacitive to avoid a fourth hole.
2. The audio should be push-to-talk, half-duplex. With the speaker and mic two centimeters apart on a wrist, full duplex would need echo cancellation, which is a semester on its own. Voice messages sidestep the problem entirely instead of solving it.
3. The mesh should never be in the safety path. Fall alerts go straight to the totem over sub-GHz, and the nodes only add voice, localization and bulk data, so help still gets called even if every node is unplugged.
4. No mains voltage on anything we design. The nodes plug into UL-listed USB wall adapters, which keeps line voltage as someone else's certified problem.
5. ESP-NOW is Espressif proprietary, so the Pi cannot hear it on its own. A $3 ESP32-C3 wired over USB works as the gateway.
6. On the nodes, BLE scanning and ESP-NOW share one radio and have to time-slice. This should be tested early with two nodes and a beacon, and if it gets ugly we can split listener and forwarder roles across different nodes.
7. The speaker draws enough current in bursts to sag a small LiPo and brown out the MCU. Bulk capacitance at the amp and a separate regulator for the MCU fixes this for pennies.
8. Residents should not be trackable by any stranger with a phone, so the wearable should use rotating BLE addresses with the totem holding the key.

Evaluation and/or testing with regard to real world issues

The following would be the main benchmarks to be used:

1. Fall detection is the whole product here, so it needs proper scoring: falls from different angles and heights onto a mattress to catch false negatives, and hard negatives like sitting down heavily, dropping the watch or swinging the arms to catch false positives. We can start on open datasets like SisFall and MobiAct so the algorithm exists before the board does, and use a weighted dummy for the destructive drops.
2. Alert latency measured end to end, from impact to something a nurse can see on the totem, tested through walls and floors and with the body between the watch and the nearest node.
3. Localization scored as "how often did it name the right room", not in meters. Trilateration from RSSI is a trap indoors, so nearest-node is the honest baseline and fingerprinting the stretch.
4. Voice tested for intelligibility rather than fidelity, meaning word recognition through the sealed case with the acoustic vent in, since the vent costs us a few dB.
5. Mesh degradation on purpose: unplug nodes one by one and confirm coverage drops gracefully and that the SOS still arrives with every node dark.

6. Battery runtime measured rather than extrapolated, with the accelerometer running 24/7 and a realistic number of messages a day.
7. Waterproofing should be tested accordingly.

Portable Multi-Frequency Battery Health & Impedance Analyzer

High-Level Description

Lithium ion cells degrade through two separate mechanisms: electrolyte drying, which increases the series resistance, and electrode passivation, which causes the charge transfer impedance to rise. A standard multimeter or a single point internal resistance tester that costs between $30 and $80 gives only one DC reading and is therefore unable to distinguish between these two effects. On the other hand, the Electrochemical Impedance Spectroscopy setups used in industry are expensive, ranging from $3,000 to $15,000, and are designed for use on a factory floor rather than in a service van or on a workbench. This project presents a portable device that addresses the gap between the two approaches. First, it applies a stepped AC signal over a frequency range from 10 Hz to 2 kHz to a lithium-ion cell or pack with a capacity between 1S and 4S and at a voltage level from 3.0 V to 16.8 V, then calculates the complex impedance at each frequency, providing the technician with a real diagnostic signature rather than just a single resistance value.

The device is aimed at technicians carrying out maintenance on EV battery modules, at solar or home-storage system installers who are assessing cells for second-life use, and at operators of drone or robotics fleets who need a rapid pre-flight health check. When it is completed, it will be in the form of a handheld unit similar in size to a large multimeter, featuring four wire Kelvin clip leads, a small built-in text display which gives a quick numerical reading or a pass-fail result, and a complementary Bluetooth LE mobile app that displays the full Nyquist and Bode plots as well as maintaining a log record for each cell.

It is a genuine application project since there is at the moment no cheap and portable alternative available between simple resistance testers and laboratory-level EIS equipment. By looking at the actual impedance spectrum rather than just a single resistance value, a technician is able to tell the difference between a battery that only needs the cell downgraded to a less demanding use and one that is actually degrading and therefore unsafe to keep in service. The situation presents both an interesting challenge in signal processing and analog design and one that has clear practical value. What is novel here isn't EIS itself, which is already well established in laboratory settings, but rather the ability to carry out synchronous digital lock-in detection in a cheap way using a single microcontroller rather than having to rely on FFT hardware or a benchtop instrument that costs more than $3,000, this is what makes it possible to create a truly portable and affordable version of this tool.

Implementation Issues

Schedule

9/10 to 9/23: Planning and core-technique validation
Check the AD9833 DDS together with the STM32F446 12-bit ADC, which means use a known test RC network using a lab function generator to ensure that the phase-resolution method is realistic before proceeding with it. Choose the instrumentation amplifier, INA826, and the BLE module. Prepare the design of the analog front end. Place orders for the evaluation boards of the MCU, DDS, INA826, and BLE module.

9/24 to 10/6: Breadboard analog front-end
Connect the signal chain on the breadboard: from the DDS to the current pulse driver to the cell being tested to the 4-wire Kelvin sense to the INA826 and then to the ADC. Make sure that we are able to obtain the magnitude and phase at a number of test frequencies using the test RC network. The written proposal should be finalized by October 1st.

10/7 to 10/20: Firmware lock-in detection and PCB layout
In the firmware, implement synchronous multiply and integrate lock-in detection with synchronization from the DDS output using DMA, and check its performance over the entire sweep range from 10 Hz to 2 kHz on the test network. When starting to draw up the schematic and layout for the 4-layer PCB, begin with an isolated analog ground plane. Place an order for the PCBs at the 10/20 date in order to cover the approximately 10-day fabrication delay.

10/21 to 11/3: App development while PCBs are in fab
Construct the skeleton for the BLE mobile app, implement Nyquist or Bode plotting, establish a BLE link with the device, and improve the calibration procedure using the breadboard prototype and the reference cells. Once the PCBs have arrived, start assembling them and get the system up and running.

11/4 to 11/20: PCB bring-up and real-cell testing
Make sure the entire signal chain is working on the PCB, carry out sweeps on real 1S to 4S batteries at various states of health, which is new one as compared to cycled one, and then compare the resulting curve shapes with the published equivalent-circuit models. Debug any systematic phase and offset errors; if we have the time, we can extend the sweeps down to below 10 Hz.

11/21 to 12/9: Calibration, enclosure, poster, and report
Finish the calibration using the reference cells, 3D-print the enclosure and complete it, polish both the BLE app and the firmware for the onboard display, finalize the test matrix, and get the poster and the final report ready.

Budget

Total prototype budget target: $180 to $280, leaving some budget for shipping, enclosure printing, and a second board revision. A rough breakdown for the project breakdown:

What	Qty	Cost of each	Total cost
STM32F446 MCU/dev board	2	$12	$24
AD9833 DDS breakout	2	$9	$18
INA826 instrumentation amp	4	$6	$24
BLE module	2	$10	$20
PCB fabrication (proto + rev-2)	2	$45	$90
precision resistors, DC-blocking caps, Kelvin leads and connectors	1	$30	$30
Onboard text display	2	$8	$16
Enclosure (3D printing + hardware)	1	$20	$20
Total			$242

We have $250 per student, a four-person team has $1000 available, so this $267 estimate leaves a comfortable budget if PCB fab or the instrumentation-amp samples cost higher than expected.

Other Implementation Issues

- Safety: 16.8 V is not dangerous in the way that might cause a shock, but if the Kelvin leads are connected to and disconnected from the live pack repeatedly, transient spikes as high as 3.3 V can occur in the analog front-end. This issue can be handled by using the DC-blocking capacitors that are already in the signal path together with input clamping diodes on the side facing the ADC. Moreover, the evaluation plan mentioned below involves deliberately ageing one or two cells by subjecting them to repeated deep-discharge cycles; since over-discharge cycling does carry a certain risk of a thermal event, this will be carried out in a fireproof LiPo-safe bag on a monitored bench and will not be left unattended.

- reliability: The greatest risk is calibration drift that occurs with repeated use. To deal with this, a rapid self-calibration sweep is carried out against an onboard known reference resistor before each measurement, rather than assuming that the factory calibration remains valid indefinitely.
- Production needs: This project aims only at producing one working prototype and possibly a small number of spare parts for the most difficult components, with no intention of carrying out large-scale manufacturing, so concerns about production requirements such as the need for test fixturing, stable processes, or assembly line procedures do not apply to this project.
- The mechanical aspect of designing the enclosure and handling the leads for the Kelvin clips is a small job which can be addressed with using a simple 3D-printed enclosure and does not require any unusual mechanical engineering.
- In terms of energy consumption, the combination of the DDS, MCU and BLE draws a current of about tens of mA, meaning that the device can be powered either by USB or by a small LiPo cell without encountering any significant energy budget issues. The main unanswered question is whether the current-injection driver stage needs a small boost or a negative rail in order to switch cleanly around the pack voltage; this aspect is included in the analog front-end design in the first schedule block.

At this stage, no other important implementation problems are expected; the main genuine uncertainties are the analog noise floor when the signal level is at the micro-volt level and the accuracy of firmware phase alignment, both of which are mentioned below.

Evaluation and Testing

Success consists in the device providing impedance measurements that are not only accurate when compared with a known reference but also useful for distinguishing between different battery health states in practice. Specifically:

- Carry out an accuracy check by sweeping a range of known test RC/RLC networks using the computationally determined theoretical impedance at each test frequency and verifying that the measured magnitude lies within about 5% and the phase within a few degrees of the theoretical value throughout the entire 10 Hz to 2 kHz range.
- For the repeatability check, carry out multiple sweeps using the same cell and make sure that the variance remains low each time, since a noisy measurement is of no use to a technician even if it is unbiased.
- Real-world validation involves testing a small sample of lithium ion cells with capacities ranging from 1S to 4S that are in new condition or have obviously degraded or have been cycled before, and verifying that the Nyquist plots obtained show the expected qualitative features as described in the literature, such as a growing high-frequency semicircle or an increasing resistance at the real-axis intercept with age, rather than merely ensuring that the numbers calculated are correct.

If the device can reliably separate a healthy cell from a degraded one using repeatable spectra rather than a single resistance number, it will have met the core real-world need.

Low-Cost Industrial Motor Vibration & Health Monitor

High-Level Description

It is a compact vibration-sensing unit which attaches directly to the housing of a motor, for example shop tools, HVAC blowers, and pool pumps. Instead of simply indicating that vibration has increased, it is able to identify the type of mechanical fault that is developing. Smart-plug sensors available to consumers for between $20 and $50 only monitor the total amount of movement, while industrial vibration analyzers, which cost between $1,500 and $5,000, do provide real diagnostics but require a certified analyst to interpret the results. This product is aimed at filling the gap between the two, focusing on small businesses, HVAC technicians, and hobbyist machinists who would like to have automated fault classification at a price of about $150. A three-axis accelerometer supplies a real-time FFT pipeline on a microcontroller, and the firmware compares the resulting spectrum with known fault patterns, a peak at one times the rotational speed indicating motor unbalance, peaks at two or three times indicating a misalignment, an integer harmonic series indicating loose mounting, and high-frequency sidebands indicating bearing wear, with the findings being sent to a phone dashboard via BLE.

Implementation Issues

Schedule

Weeks 1 to 2:

The accelerometer together with the DMA and FFT pipeline should be bench-tested against a known vibration source, for example an intentionally unbalanced fan, before adopting the method.

Weeks 3 to 5:
the PCB was designed and ordered, including the analog anti-aliasing filter, and the FFT and peak detection firmware were simultaneously developed on the eval board.

Weeks 6 to 9:
raise the PCB, verify that there is sustained SPI and DMA throughput without any lost frames, and check the fault classification using a small number of motors that have either actual faults or faults that have been deliberately induced.

Weeks 10 to 12:
The BLE dashboard should be completed, the mount and enclosure finalized, and the final report and poster prepared.

Budget

Verified per-unit cost is approximately $88.50. Building two working units brings the estimated prototype total to $150 to $250:

What	Qty	Cost/ea	Cost total
KX134-1211 accelerometer	2	$12	$24
STM32F446 MCU + PCB fab (2 orders)	2	$35	$70
BLE module + analog front-end	1	$40	$40
Enclosure + mounting hardware	1	$20	$20
Total			$154

Other Implementation Issues

- Regarding to EMI and noise, the precision analog traces and SPI lines must be isolated from motor EMI through a careful arrangement of the ground plane, something which can be achieved by proper PCB design.
- When mounting the accelerometer, it must be securely mechanically attached to the motor housing using either a magnetic or a screw mount so that it accurately detects vibrations of the housing rather than the rattling of the mount.
- In terms of safety, the only potential issues are those involving testing when fitting a sensor close to function powered equipment that rotation very fast and some other potential mechanical hazard.

Evaluation and Testing

Success is measured by whether the device can correctly identify the type of fault, not just detect that vibration increased:

- Carry out an accuracy check by comparing the accuracy of the FFT bins with a known frequency source, such as a calibrated shaker or a motor with known RPM.
- For the fault classification test, carry out the test on motors with faults deliberately caused, such as having an unbalanced blade, a loose mount, and a worn bearing, and make sure that the correct fault category is indicated, with the aim of reliably distinguishing at least two or three fault classes while keeping the false-positive rate low.

Precision Fermentation Container

Fermentation is a common household food preparation method, but since it relies on microbial activity, there are precise conditions for optimal fermentation and thus optimal food. Two common foods that are easy to ferment are sourdough and yogurt. Having incorrect conditions with regard to humidity, temperature, and light can cause incorrect fermentation that ruins the taste and can cause a large change in taste. Common methods for fermented foods are typically not power efficient, nor are they able to maintain a consistent environment, such as leaving your starter food in the oven for an extended period of time. This allows for external conditions (weather, temperature of the house, etc.) to be factors that can change the final product and leave people frustrated, as their food quality is dependent on conditions they have little to no control over.

Other existing applications may run north of $300 dollars and are typically limited to only alcoholic beverages, as this is the most common case of fermentation in people's lives. However, this does not work for other foods such as yogurt and sourdough that have different requirements. In a lot of Asian cultures, fermentation is done regularly, and there simply is no existing way of controlling this process. This precision chamber would be much cheaper and more applicable to foods that are not alcoholic beverages, and allow for better food in homes.

Details

Since we are controlling humidity, temperature, and light, we will need sources and sensors for each of these, along with an MCU for interfacing with all of these, and lastly a method for putting in desired inputs. The system should be able to maintain the conditions requested and have live reporting in the form of an interactive display.

An ideal size for this project would be 1 cu. foot, or realistically large enough to hold a large mixing bowl for sourdough or multiple containers for yogurt.

This proposal will go in depth of talking about the hardware structure, how everything should be roughly connected, along with some possible component ideas to see what the cost of the project will be, and lastly a schedule that is realistic at this point of development where we are still uncertain but know about some roadblocks, such as the PCB ordering time and what we can do in the mean time.

Schedule (tentative)

9/10-9/23	Planning Group formation, 9/18 Draft Proposal Due, Define the system requirements and identify the pins needed on the MCU of choice.
9/24-10/6	Design & Part Selection Finalization 9/24 Proposal meetings in class, 10/1 Final proposal due, Order components ideally by 10/4 while under budget

Precision Fermentation Container

10/7-10/25	Breadboarding + PCB Design 10/15 Milestone 1 due, 10/20 Fall Break, 10/22 Milestone 1 meetings Have breadboarding prototype done along with PCB's to order for first iteration, build the actual chamber to our liking
10/26-11/10	Hardware Interfacing 11/3 PCB Deadline, 11/10 Milestone 2 Due Write out the software needed to interface with the hardware (can be done on breadboards) while we wait for the PCB's to be made and sent back to us
11/11-11/25	Development with the PCB's 11/2 Milestone 2 Meeting Migrate from breadboard onto PCB, ensuring functionality still
11/26-12/9	Final Testing Also buffer time
12/10-12/11	Expo + Finish Final Report

Budget

Since we do not know the final parts yet, it is not possible at this moment to find an accurate cost, but we can estimate with parts that should in theory work together. If not, similar cost parts should be available on the market for us to use.

PCB (Not entirely sure how to price)	$60
ESP32-S3 MCU	$10
TPS5430 Buck Regulator (Converting 24V to 5V)	$10
Power Supply (24V Barrel Jack)	$20
Sensors SHT31, BH1750	$20
Actuators (Silicone heating pad, misting disk, circulation fan, LED light)	$35
Enclosure	$50
User Interface LED Display	$15
Total:	$220

Precision Fermentation Container

While these are rough costs, I do not foresee us running into budget issues and will have plenty of money to iterate through other components we feel may be better if we have already made purchases that did not work.

Issues

- Heat & Safety: Need to ensure voltage hazards are minimized and thermal runaway is not possible with the heating pad, and ensure heat is being distributed evenly
- Mechanical Design: Ensuring the container is able to maintain temperature and moisture at a reasonable level, ensure water for mister is kept separate from all other electronic components
- Energy Budget: Ensure we are able to use energy optimally, otherwise simply using an oven would defeat our purpose in most scenarios.
- Cooling: Unsure if this would need to be an issue, but extra components for cooling the chamber down may be needed.

Testing

- Thermal precision can be tested as long as we maintain within an acceptable range for temperature fluctuation, such as 0.5°C as a standard.
- Humidity precision should be kept at a standard too, though unsure what this value should be
- Can we actually ferment food inside of it? Will the final product be better than conventional methods?
- Does the sensor readings show accurately on the display without consuming too much energy?

Stretch Goals

One stretch goal for this would be to have some mobile indicators for when the machine is done, when it needs water, or if there happens to be some malfunction or if food spills out for some reason.

Smart Tablet Tray

As people age, they tend to need more pills in their life and their memory and time become fainter and scarcer. People forget, double-dose, and thus their health takes an unnecessary toll as their medication sits in their drawer. It is difficult for healthcare professionals to ensure their clients are properly taking their medications, and current pill boxes rely too much on trust in their users and take time to fill up.

Since every pill takes up some amount of mass, if we can measure the weight of the container against the pill, we will be able to determine if a pill has left the container and be able to indicate this to the consumer.

Details

As indicated above, the idea would be to have a way to measure the weight of the existing container against an individual pill, so we can track when a pill is able to leave the container from which we assume it was consumed by the individual. We would need a precise weight sensor for this, and an easy way for users to add their pills to the system so it would be able to learn the weights properly.

The size of the device should be about the size of an iPad Mini (8.5in x 6.0in), though this would be entirely dependent on how the power source is achieved here. It would be plausible to use USB-C power or batteries, and it should not change the device's overall functionality. There would be dedicated slots for the medications, and a low-power sensor to detect when one is removed and put back, so we know when to actually measure weights without wasting power. There will also be a screen with user guidance indicating of a dose was taken and if the machine was able to recognize it.

Schedule

9/10-9/23	Planning Group formation, 9/18 Draft Proposal Due,
9/24-10/6	Design & Part Selection Finalization 9/24 Proposal meetings in class, 10/1 Final proposal
10/7-10/25	Breadboarding + PCB Design 10/15 Milestone 1 due, 10/20 Fall Break, 10/22 Milestone 1 meetings
10/26-11/10	Hardware Interfacing 11/3 PCB Deadline, 11/10 Milestone 2 Due
11/11-11/25	Development with the PCB’s 11/2 Milestone 2 Meeting
11/26-12/9	Final Testing
12/10-12/11	Expo + Finish Final Report

Smart Tablet Tray

Budget

Due to the size of the project, I highly doubt we would be approaching even half of the allocated budget for teams, and would have lots of money left over to experiment with different parts or reordering components as we need. A worst case estimate would be $150 dollars for the entire thing leaving us with over a $1000 dollars for emergency.

Issues

- It is possible over time that the scales for the weights measure differently over time and would record false positives for double-dosing or not accounting for a missed dose

Testing

- Testing should be fairly straightforward, as we can test with candies in some small hard plastic container to see if we can pick up slight changes in mass (smaller candies for more extreme testing) and see if it is recorded and understood by the device

Stretch Goals

- Being able to have a mobile interface to see which medicines have or have not been taken for caretakers would be ideal for convenience and as a secondary measure to ensure people are taking their medications, and also be able to see logs of missed doses or accidental double doses.

RadarCPR: A Remote Radar-Based CPR Depth and Rate Monitoring System



INTRODUCTION

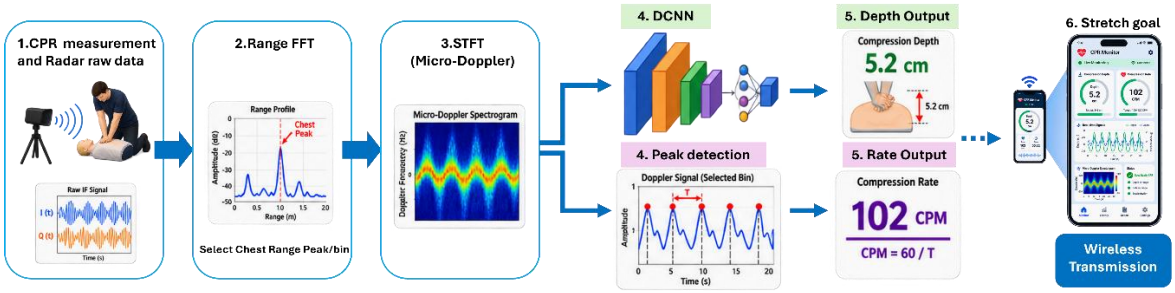
Cardiac arrest is a life-threatening emergency, and survival depends strongly on the rapid delivery of high-quality cardiopulmonary resuscitation (CPR). Although telecommunication CPR (T-CPR) can provide real-time guidance on proper compression posture and hand position, accurately achieving and maintaining the appropriate chest compression depth and rate remain major challenges. Both compression depth and rate are essential parameters for delivering high-quality CPR and improving patient outcomes [1].

This project proposes a remote frequency-modulated continuous-wave (FMCW) radar-based CPR monitoring system that estimates and displays chest compression depth and rate in real time. The system is intended to assist rescuers, especially in out-of-hospital situations where dedicated medical equipment may not be readily available.

By enabling remote measurement of small chest movements through Doppler and micro-Doppler information, FMCW radar can monitor CPR from a distance without requiring physical contact with the patient or rescuer. This approach overcomes the limitations of conventional contact-based methods, such as attached sensors, smartphones placed between the hands, and smartwatches, which must be attached, held, or worn by the patient or rescuer and may require repositioning when rescuers change, potentially interfering with CPR performance [2].

Unlike previous approaches that have largely relied on signal-processing-based estimation [3, 4], this system incorporates AI-based analysis and provides continuous, non-contact measurement of chest compression depth and rate with real-time feedback. Since some smartphones already include UWB or radar-related hardware for spatial sensing, future versions of this system could potentially be integrated into smartphones for more compact CPR assistance.

MAIN PIPELINE



Radar Device
(FMCW Radar + Processor + Display + Wi-Fi/BLE + Rechargeable Battery + Power Management)

↓ Wi-Fi / Bluetooth

Smartphone
(CPR Monitoring App)

↓ Wi-Fi / Cellular

Cloud (Stretch Goal)
(Backend + Database + Data Storage / Remote Monitoring)

DETAILS

1. Signal Processing and DCNN Model

- Process FMCW radar signals using Range FFT to identify the chest range bin.
- Generate micro-Doppler spectrograms using STFT from chest motion.
- Use the spectrogram as input to a DCNN regression model to estimate CPR compression depth.
- The DCNN consists of convolution, batch normalization, ReLU, pooling, and fully connected layers, ending with a single regression output for compression depth.

2. Ground-Truth Data Collection

- Record CPR using a 60-fps smartphone camera.
- Attach a small marker to the rescuer's hand and place a calibration grid in the same motion plane.
- Use Kinovea motion tracking to obtain vertical displacement of the marker.
- Use the measured displacement as the ground-truth compression depth, and cycle timing as the ground-truth compression rate for DCNN model training. (This follows the video-reference method used in the smartwatch CPR study)

3. Required Computational Capability

Need to determine whether the embedded processor can support real-time signal processing and DCNN inference.

- Processing capability: computation required for Range FFT, STFT/WVD, and spectrogram generation.
- Inference speed: DCNN inference must be fast enough to provide real-time CPR feedback.
- Memory: sufficient RAM/flash/storage for the trained DCNN parameters and intermediate signal/spectrogram data.

SCHEDULE

Period	Phase	Plan
9/10–9/25	**Planning and Part Selection**	Determine the required radar, MCU, battery, and power-management specifications before selecting and ordering major components. Set up the Infineon radar development environment and plan the data collection procedure, radar measurement setup, and FMCW parameters.
9/26–10/11	**Data Collection and AI Development**	Establish the ground-truth data collection method using the Little Anne QCPR and collect radar measurements. Generate training datasets and train and evaluate the DCNN model, refining the model until acceptable depth-estimation performance is achieved.
10/12–10/31	**MCU Prototype and PCB Design**	Implement and benchmark FFT/STFT processing and AI inference on the STM32 development board and develop the display interface. Design the custom processor PCB, then order the PCB for fabrication and assembly.
11/1–11/15	**PCB Fabrication and Prototype Development**	While the custom PCB is being fabricated and assembled, continue software development and system testing using the STM32 development board. Begin smartphone communication and app development for wireless CPR monitoring.

11/16–11/30	**Custom PCB and Radar Integration**	Bring up the assembled custom PCB and integrate the display. Test battery-powered operation and measure system power consumption and operating time. Transfer the signal-processing and AI inference pipeline to the custom hardware and resolve hardware/software integration issues. Make a PCB revision if necessary.
12/1–Design Expo	**Final Packaging and Stretch Goals**	Complete the final system packaging and resolve any remaining issues. Finalize the project report and prepare for the Design Expo. Complete smartphone monitoring and, if time permits, implement cloud-based session storage as a stretch goal.

BUDGET

Component		Budget	Why
FMCW Radar	Infineon DEMO BGT60TR13C	~$300	60-GHz FMCW radar, raw ADC access without DCA1000
CPR Reference	Little Anne QCPR	$324	Compression depth/rate reference
MCU Option 1	STM32H743 (LQFP)	$50 dev board / ~$20 chip	480-MHz Cortex-M7, suitable for FFT/STFT and lightweight inference testing
MCU Option 2	STM32H7S3 (BGA)	$55 dev board / ~$12 chip	Higher-performance newer STM32H7 family, better suited for heavier DSP/AI workloads
Custom Processor PCB fabrication	2 PCB revisions + assembly	~$150	including one possible revision
Custom PCB components	MCU + regulators + memory + USB + connectors	$70–90	Components for the custom PCB
Display	3–3.5" SPI LCD	$30	Real-time depth/rate display
Battery & Power System	Rechargeable battery	~$40–60	Portable operation and stable power for radar, MCU, and display
Shipping / contingency		$100	

STM32H743 option: Estimated Total Budget: $1,064–1,104
STM32H7S3 option: Estimated Total Budget: $1,069–1,109

IMPLEMENTATION ISSUES

-Radar operation
FMCW radar setup and parameter tuning may take significant effort and time. / However, the radar provides a GUI, and I have previous experience researching radar systems.

-Ground-truth data generation
Generating sufficient and accurate ground-truth data may be time-consuming. / However, a method for generating ground-truth CPR data has already been demonstrated in previous research.

-MCU performance
The MCU may have limited performance for signal processing, AI inference, and real-time display. / However, we can benchmark these tasks on a development board before finalizing the PCB.

SUCCESS CRITERIA

-Depth estimation accuracy: Achieve a depth RMSE below 0.5 cm, comparable to the performance reported in previous research.

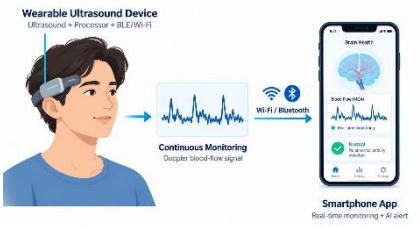
-Real-time estimation and display: Estimate and display the compression depth and rate in real time, with the display updated for every compression.

STRETCH GOAL

Cloud connectivity: Add cloud-based CPR session storage and remote monitoring.

CerebroGuard: Wearable Ultrasound-Based Cerebral Hemodynamic Monitoring for Early Detection and Prediction of Imminent Stroke

INTRODUCTION



Stroke is a leading cause of death and long-term disability, causing more than 7 million deaths worldwide each year. Current technologies are effective for long-term risk monitoring such as blood pressure, ECG/PPG, and carotid imaging, as well as post-onset diagnosis and treatment using CT/MRI.

However, there is still a critical gap in continuously monitoring physiological changes before obvious stroke symptoms appear. Early prediction during **this pre-symptomatic stage** is particularly important because stroke treatment is highly time-sensitive; an untreated ischemic stroke can result in the loss of about 1.9 million neurons per minute.

Recent studies suggest that continuous wearable monitoring can capture dynamic physiological changes that intermittent clinical visits may miss [5]. AI is also increasingly being adopted in clinical practice, including FDA-authorized stroke-related AI systems.

Therefore, we propose an AI-driven wearable ultrasound system for continuous cerebral hemodynamic monitoring and early prediction of imminent stroke.

Stroke risk factors
↓
Long-term progression
↓
Possible cerebrovascular physiological changes
↓
Critical monitoring gap before obvious symptoms
↓
TIA / early stroke
↓
B.E. F.A.S.T. symptoms
↓
Established stroke

MAIN PIPELINE

Primary Goal:

Continuous cerebrovascular anomaly detection

Future Goal:

Extend to imminent stroke prediction as sufficient clinical data become available.

Wearable Ultrasound Device
(Ultrasound Transducer + AFE + Processor + IMU/Contact Sensor + BLE/Wi-Fi + Rechargeable Battery + Power Management)
↓
Continuous Cerebrovascular Monitoring
(Doppler Blood-Flow Signal + Feature Extraction)
↓ Wi-Fi / Bluetooth
Smartphone
(Real-Time Monitoring + AI-Based Anomaly Detection + Patient Alert)
↓ Wi-Fi / Cellular
Cloud / Clinician Dashboard (Stretch Goal)
(Data Storage + Remote Monitoring + Future Stroke Prediction)

DETAILS

1. What the Ultrasound Sensor Measures

- Continuously measure cerebral blood-flow velocity using Doppler ultrasound.
- Extract key hemodynamic parameters such as PSV, EDV, MFV, PI, and RI.
- Monitor changes in blood-flow patterns that may indicate abnormal cerebrovascular conditions.

2. Signal Processing and AI Model

- Apply band-pass filtering, Doppler FFT/spectral analysis, and envelope/peak tracking to extract blood-flow waveforms and hemodynamic features.
- Use temporal features and personalized baseline data as inputs to the AI model.
- Use models such as LSTM, 1D-CNN, or an Autoencoder for temporal anomaly detection and risk-score generation.

3. Improve Measurement Fidelity Using IMU and Contact Sensor

- Use an IMU to identify motion artifacts caused by head or device movement.
- Use a contact/pressure sensor to verify proper ultrasound probe contact.

IMPLEMENTATION ISSUES

-Ultrasound probe placement and parameter tuning
Determining the optimal probe location and ultrasound parameters to reliably measure cerebral blood flow through the skin and skull.

-Clinical ground-truth data
Collecting sufficient labeled clinical data and reliable ground-truth data for AI training and validation.

-Wearable comfort and form factor
Designing a stable wearable that maintains probe contact without causing discomfort during long-term use.

STRETCH GOAL

Cloud connectivity: Add cloud-based storage of cerebrovascular monitoring data and enable remote clinician access for long-term monitoring and future stroke-risk prediction.

REFERENCES

[1] Y. Kim, B. G. Yu, J. H. Oh, and T. W. Kim, "Novel Chest Compression Depth Measurement Sensor Using IR-UWB for Improving Quality of Cardiopulmonary Resuscitation," *IEEE Sensors Journal,* vol. 17, no. 10, pp. 3174–3183, 2017, doi: 10.1109/jsen.2017.2680454.

[2] I. Choi, S. G. W. Lee, H. J. Kong, K. J. Hong, and Y. Kim, "Estimation of Compression Depth During CPR Using FMCW Radar with Deep Convolutional Neural Network," *Sensors (Basel),* vol. 25, no. 19, Sep 24 2025, doi: 10.3390/s25195947.

[3] E. F. de Borba *et al.*, "Smartwatch assessment of CPR performance administered to an instrumented mannequin," *Med Biol Eng Comput,* vol. 64, no. 4, pp. 1561–1572, Apr 2026, doi:

10.1007/s11517-026-03533-z.
[4] Y. Song, Y. Chee, J. Oh, C. Ahn, and T. H. Lim, "Smartwatches as chest compression feedback devices: A feasibility study," *Resuscitation,* vol. 103, pp. 20–23, 2016/06/01/ 2016, doi: https://doi.org/10.1016/j.resuscitation.2016.03.014.
[5] D. B. Olawade *et al.*, "Integrating AI-driven wearable devices and biometric data into stroke risk assessment: A review of opportunities and challenges," *Clin Neurol Neurosurg,* vol. 249, p. 108689, Feb 2025, doi: 10.1016/j.clineuro.2024.108689.

Wireless Wearable for Climbing Performance Analysis

Climbing, especially indoor bouldering, has seen a huge increase in popularity over the recent years. The sport entails following up a set route of holds and surfaces to reach the top, and requires a blend of both strength and technique. Climbers rarely ascend to the top of the route in their first attempt (such a success would be called a flash), and regularly require multiple attempts to improve their body positioning and tension.

However, climbers currently have to rely on their own subjective feel of the route, vague feedback from others, or watch a recording of themselves to find areas to improve on. Integrating wearable technology to track muscle activity patterns and limb orientation would greatly increase the data available for analysis and pattern recognition across multiple attempts.

Details

In climbing, body positioning and muscle activation are vital for setting up their action for the next hold. These areas can be tracked via a gyroscopic sensor and accelerometer for limb orientation, and an electromyography sensor for muscle tension. We propose having these sensors be worn on each limb of a climber, and the telemetry be sent over Bluetooth to a central processor for data analysis. Ideally, the whole process should also be synced to a paired phone's recording for better visual feedback and clarity.

Schedule

Date	Details
9/10 - 9/23	**Planning** The first two weeks would be for brainstorming ideas on how to keep the wearable compact and energy-efficient for long climbing sessions. For more advanced data analysis, we would also need to find an ideal microprocessor at low cost. The draft proposal would be written on 9/20.
9/24 - 10/2	**Part Selection** Select the wireless communication components, the gyroscope + accelerometer, and EMG sensors. Have a basic data transmission test to the chosen microprocessor.
10/3 - 10/15	**Initial Breadboarding** From initial estimates, select a battery of sufficient capacity and create a breadboard prototype of four sensor batches with one main processor. Communication protocols to arbitrate or handle the 4 batches are chosen. Have Milestone 1 report done.

10/16 - 10/23	**PCB Design** Start first batch of PCB designs and start sensor noise calibration testing using breadboarded components. Order first batch by 10/23.
10/24 - 11/3	**Reliability Testing & PCB Revision** All functionality will be written during wait for first PCBs. Once PCBs arrive, start reliability testing with full prototype and design for second batch of PCBs if needed. Revised PCBs done by 11/3 latest.
11/4 - 11/13	**Bug fixes & Hardware Manufacturing** Catch any final software bugs for core implementation, and design casings needed for the wearables. Stretch goals can be pursued after. Milestone 2 report done by 11/9.
11/14 - 12/11	**Final Feature Implementation, Report & Expo** Finish bug fixing for any stretch goals and revise any hardware casings before final expo on 12/10. Final report draft will be done by 12/7.

Budget

The budget will be based on the cost for one climber, this includes all the body sensors and main processor board, complete with PCB manufacturing and any hardware costs. Shipping costs are estimates. Initial rough budget estimations consider ESP32 based architecture and will be revised. All sensor and battery components are also subject to change.

No.	Quantity	Item	Cost / Quantity	Total Cost
1	4	Analog EMG Sensor	49.50	198.00
2	4	Inertial Measurement Unit	40.00	160.00
3	3	ESP32 x 3	16.80	16.80
4	4	BLE module	10.99	43.96
5	8	Li-Ion Battery x 4	23.99	47.98
6	1	Misc Parts	50.00	50.00
7	6	PCB	100.00	100.00
8	1	Shipping	50.00	50.00
Total				666.74

Issues

There are 4 main issues that have to be addressed at this time:

1. DSP processing of noisy signals coming from IMU and EMG
2. Calibration software needed for customisation to each climber
3. BLE communication between all the wearables and the main processor
4. Keeping the footprint small and safe

Issue 1 and 2 are solvable depending on the hardware components we choose and the degree of accuracy we plan to allow. Human variance is likely much higher than the system. A more capable microcontroller would have to be chosen, or multiple feeder systems would have to be developed for issue 3. Issue 4 could be resolved should the energy requirements be minimized.

Stretch Goals

Once core functionality has been done, communication of the main microprocessor with a phone recording would greatly boost the visual clarity and feedback of the system for the climber. Furthermore, modification of the kit for easy attachment to climbing shoes and clothing would greatly improve convenience and ease of use of the devices.

Cheap Handheld 3D Scanner for 3D Printing

The prevalent rise in 3D printing technology has greatly increased the need for better modelling software and hardware. Replicating parts or creating 3D maps of areas that need work is currently done by hand measurement, if not by expensive, industrial-grade 3D scanners. The current market consists of models ranging from $230 (Creality CR-Ferret SE 3D Scanner) to $2300 (All-in-one 3D Scanner for Professional Applications - Revopoint MIRACO). With the best value sitting at around $499.

Details

We propose a cheaper, open-source alternative to market solutions that can produce a basic 3D scan. We propose using current 3D scanning theory and off-the-shelf components to create a scanner to lower the bar of entry for hobbyists, similar to Arduino for embedded projects.

We propose the solution of using a Raspberry Pi-based structured-light scanner. The package would likely include a camera, a laser line, and a turntable. These would produce a rough scan of the object to be used as a starting base for easier 3D object replication in a user's 3D printing software.

Schedule

Date	Details
9/10 - 9/23	**Planning** The first two weeks would be for brainstorming ideas on how to create the laser line and turntable using common components. A check on the Pi's capability is also done. The draft proposal would be written on 9/20.
9/24 - 10/2	**Part Selection** Select the camera, Raspberry Pi, laser, and turning components. Also choose a wireless communication protocol to a user's computer.
10/3 - 10/15	**Initial Breadboarding** From initial estimates, select a battery of sufficient capacity and create a breadboard prototype of the main laser line. Check math needed for 3D distance estimation. Have Milestone 1 report done.
10/16 - 10/23	**PCB Design & Hardware Design** Start first batch of PCB designs and mechanical prototype to fix camera and laser placements. Order first batch by 10/23.

10/24 - 11/3	**Software Calibration & PCB Revision** All component placements are fixed and software tuning is done to improve scans. Once PCBs arrive, start distance testing and scanning with full prototype and design for second batch of PCBs if needed. Revised PCBs done by 11/3 latest.
11/4 - 11/13	**Bug fixes & Hardware Manufacturing** Catch any final software bugs for core implementation, and compare initial scans to true models to improve accuracy. Stretch goals can be pursued after. Milestone 2 report done by 11/9.
11/14 - 12/11	**Final Feature Implementation, Report & Expo** Finish bug fixing for any stretch goals and revise any software before final expo on 12/10. Final report draft will be done by 12/7.

Budget

The budget will be based on the cost for one scanner, this includes all the lasers and main processor board, complete with PCB manufacturing and any hardware costs. Shipping costs are estimates. Initial rough budget estimations consider Raspberry Pi-based architecture and can be revised. All sensor and battery components are also subject to change.

No.	Quantity	Item	Cost / Quantity	Total Cost
1	2	Laser Module x 2	12.99	23.98
2	1	Inertial Measurement Unit	20.00	20.00
3	1	Raspberry Pi	54.00	54.00
4	4	Li-Ion Battery x 4	23.99	23.99
5	1	Misc Parts	50.00	50.00
6	6	PCB	50.00	50.00
7	1	Shipping	50.00	50.00
Total				271.97

Issues

There are 3 main issues that have to be addressed at this time:

1. Software needed for the math of the 3D estimation
2. High processing speed needed
3. Sensor calibration drift

Issue 1 are solvable depending on the current availability of open-source algorithms that can be adapted to our physical setup. Issue 2 could be resolved b using a faster Raspberry Pi or by decreasing the accuracy needed by the setup. Issue 3 requires choosing better hardware and ensuring the physical setup is very secure.

Stretch Goals

Once core functionality has been done, live communication of the main microprocessor with a computer could be pursued. Improvements of the laser line for better precision could also be done should the math check out.

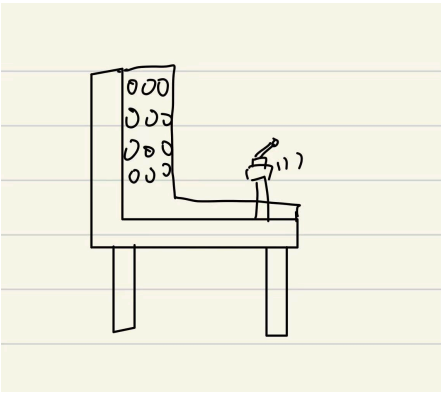
Speaker-Based Vibrotactile Massage Chair

(This project I have discussed with my friend Linglong Qian. However, we finish this document individually.)
People often use devices like massage chairs to relax. Most of these devices rely on mechanical rollers or vibration motors embedded in the chair. These systems usually have fixed massage patterns and generally do not give users much control over where the vibration happens or how it moves. Users cannot specify an arbitrary vibration path.

We want to try out a different massage approach by utilizing acoustic vibration instead of using mechanical components. We want to attach multiple speakers on a chair and play low-frequency sounds through the speakers. Users will be able to feel the vibration through the chair. Since each speaker is controlled separately, we can create personalized patterns that move from one part of the chair to another. Users will be able to adjust the massage pattern in real time through a wireless control interface. They can decide where the vibration starts, the strength of the vibration, and the path the vibration will go using the remote control interface.

The project mainly needs a chair, multiple speakers, a custom PCB, and a wireless control panel. This device is helpful for people with personalized massage needs and our goal is to create a customizable vibrotactile relaxation system that allows users to control where and how vibration moves across the chair. It may provide users with a relaxing or massage-like experience.

Sketch



Schedule

9/10-9/23	*Planning* In the two weeks after the project groups are formed, continue to refine the current project idea and brainstorm approaches.

	Can think about what other features can be added to the product. Research suitable audio speakers and wireless control hardware that fits into the budget. Search for a suitable microprocessor and have a general idea on how the PCB can be designed (including all the needed components such as DACs and amplifiers). Write the draft proposal by 9/18.
9/24-10/6	*Design and part selection* After the proposal meeting, refine the proposal. Decide on the locations of the speakers on the chair to ensure that people can feel the vibration. Order the speakers, the selected processor (eg.ATmega128RFA1), the small electric components, and the wireless control pad. Finish the proposal by 10/1. All parts we think we need for the prototype will have been ordered by 10/4.
10/7-10/25	*Speaker Testing* Use a MCU and an amplifier to test on controlling one speaker first. Extend the testing to multiple speakers and with software programming, see if the program can control the speakers to play sequentially. Design the PCB according to the needs. Finish milestone report by 10/15.
10/26-11/10	*Develop wireless communication* Develop the wireless control hardware to send the path info to the audio controlling part wirelessly. Might use esp32 for building wireless data connection. The processor(probably ATmega) needs to map the path info onto the speakers. Order the PCB before the deadline
11/11-11/25	*Integrate Full Prototype* Check and finish the path mapping. Integrate the multiple speakers and the wireless control pad, producing a prototype. Then integrate it onto the PCB. If needed PCB revisions will be made.
11/26-12/9	*Final testing, poster and expo* Continue to test on the PCB and fix minor bugs. We'll get our poster done and start on the final report.
12/10-12/11	*Finish final report*

Budget

What	Quantity	Cost per	Cost total
ESP32	1	20	20
Speakers	12	8	96
Multi-channel amplifier boards	6	15	90
Wirelss control panel	1	20	20
Custom PCB fabrication			200
Mounting materials and chair			50
Electronics, DACs, power supplies			120
Shipping			75
Total			671

Currently I have an UMC 1820 (an audio interface). This is probably useful for prototyping. The budget is a rough estimate of the components we are going to use. PCB costs are underestimated. Moreover, the budget does not include replacement or miscellaneous components. However, it can be seen that the total cost should be less than $1000.

Issues

The largest concerns at this time are

1. Developing the algorithms needed to map the drawn path onto the speakers
2. Design a relatively complex PCB board that embeds multi-channel amplifiers and DACs.
3. Soldering the surface mount with the correct components onto PCBs
4. Power delivery for multiple speaker/amplifier channels
5. Interfacing with the audio interface which is used for prototyping
6. Wirelessly send the drawn input

Issue 1 is the most difficult part. We know MAXMSP, which is a useful app that can design spatial sound. However, it needs to be operated on the laptop. Ideally, we want the vibration paths to be configured with a microprocessor. We might need to figure out how to do it on a microprocessor, probably without MAXMSP. Moreover, we need to figure out how to control the speakers so that users can feel the pattern they draw. Issue 2 and 3 might not be a challenge if we learn PCB design and make good use of the course's resources. For issue 4, driving multiple speakers may require more power than the processor. The amplifier power supply will be sized based on the measured power requirements. As for issue 5, the audio interface has many buttons, and it may take some time to figure out exactly which button should be used. We might use an esp32 to send the data wirelessly from the touchpad input if we could not find a suitable wireless touchpad.

Evaluation

Our preliminary evaluation will be for users to correctly feel the direction of the vibration paths. When users draw a path such as lower-left to center and then to upper-right, the system maps it onto the nearest speaker sequence. Each speaker vibrates in the correct order and the user identifies and compares the perceived vibration path. This might be helpful for individual back pain and muscle stiffness problems. This project has the potential to help make people relax and the vibration might also help regulate people's breath. Apart from the simple pattern check, we can also ask users whether they feel more relaxed after using the device.

SmartBed for heart rate and respiration detection

(Briefly discussed with Linglong Qian, but did the homework individually)
Heart rate and respiration during sleep can provide useful information about a person's physical state. Nowadays, many existing monitoring systems use wearable devices such as smartwatches, chest bands, or contact sensors.These devices are sometimes uncomfortable when worn during sleep. What if the sensing is not wearable or adhesive but embedded in our beds? In this case, it can monitor our sleep, heart rate and respiration without disturbing our sleep or making us uncomfortable.

We want to design a smart bed with piezoelectric sensors placed inside or under the mattress. These sensors can capture small mechanical vibrations generated by the body. Breathing produces slow and large movements, while heartbeat creates short and smaller vibrations that travel through the body and bed. With these vibrations signals, the system will attempt to separate heartbeat and respiration signals and estimate a user's breathing rate and heart rate. The sensors will be connected to the microprocessor which will then process the signals, make judgements, and then show on the display either wirelessly or through cables.

The main advantage is that the user would not need to wear or attach any sensors while sleeping and the bed can still continuously monitor the user's body situation. This might be useful for people who want to know more about their sleep status and body status. A display or a wireless system can be added to show the result.

Schedule

9/10-9/23	*Planning* Can think about what other features can be added. Research suitable piezoelectric sensors and search for a suitable processor. Write the draft proposal by 9/18.
9/24-10/6	*Design and part selection* Decide and order the corresponding piezoelectric sensors and the display. Prepare the microprocessor. Finish the proposal by 10/1. Draw the PCB design layout and order all the parts before 10/4.
10/7-10/25	*Prototyping* Order the PCB. Make a rough prototype that uses the sensor and display. Develop the algorithms for heartbeat and respiration

	sensing. Find the suitable position to place the sensors.
10/26-11/20	*Software* Develop the heartbeat and respiration conversion algorithms. Develop the program for display (wirelessly ideally).
11/11-11/25	*Testing* Integrate the full system and test on the PCB
11/26-12/9	Final testing, and refine any additional features
12/10-12/11	*Finish final report.*

Budget

What	Quantity	Cost per	Cost total
Piezoelectric sensor	5	5	25
Custom PCB fabrication		150	150
Display	2	10	20
ESP32	1	20	20
bed frame			200
mattress			200
Total			615

The budget is relatively underestimated.

Issues

1. PCB design
2. Conversion program
3. Finding the suitable locations

We might spend some time on searching for the suitable sensor and learning the datasheet of the sensor. Sensor locations also need to be taken care of. Clearly deciphering the sensor's data will be important in our project. PCB design is also an important issue because I don't have much experience in this design and I hope to develop one that connects all the needed components correctly.

Evaluation

For testing, we will ask the users to wear a respiratory belt to check the respiratory rate and a pulse oximeter to check the heart rate. Then users will lie on the bed and see if our data is similar to the data gain from the belt and the oximeter.

A Wearable Hazard Awareness System for Blind and Low-Vision Users

High-Level Description

Although white canes are very effective mobility tools, they mainly provide information about the ground and objects within the cane's physical sweep. This means hazards at chest or head height may not be detected until the user is already very close to them. Furthermore, some blind users may prefer not to use a cane because it can be uncomfortable on the hands and joints, and because of the social stigma sometimes associated with using one.[1]

There are already several technologies that try to address parts of this problem.[2] For example, the WeWALK Smart Cane adds an ultrasonic sensor to a traditional white cane, allowing it to detect obstacles around chest and head height. However, it still cannot detect downward terrain hazards such as stairs, curbs, or holes.[3]

SensAble's Vizion 1 is one of the closest existing systems to this project. It combines ultrasonic sensors, an infrared array, and a gyroscope to provide obstacle detection, surface identification, navigation, and haptic/audio feedback.[4] We would not attempt to reproduce Vizion 1's broader navigation features. Instead, our project would focus specifically on real-time multi-height hazard detection using multizone depth sensing and motion-corrected geometry.

The main technical goal is to determine whether an obstacle or terrain discontinuity actually intersects the user's projected walking volume, allowing the system to ignore nearby but non-threatening objects. This protected-path filtering, together with IMU-corrected terrain sensing and quantitative false-positive testing, would be the main focus of our design.

Details

The project combines multi-height depth sensing, ultrasonic sensing, wearable motion compensation, and walking-path filtering into one standalone embedded system. It would mainly detect three categories of hazards, including terrain hazards, body-level obstacles, and head-level obstacles. The device would resemble a lightweight chest harness or vest, with sensors mounted rigidly on the torso and small haptic motors positioned where the user can easily distinguish their feedback. The final device would contain a custom microcontroller PCB, distance sensors, an IMU, haptic drivers, a rechargeable battery, and a wearable enclosure.

The proposed sensing system would combine Time-of-Flight (ToF) depth sensing, ultrasonic sensing, and an IMU. One multizone ToF sensor would face forward to detect objects in the user's walking space, while another would be angled downward toward the ground several steps ahead of the user. The ultrasonic sensor would provide a redundant forward-range measurement for larger obstacles, particularly when optical ToF measurements are degraded by lighting or target reflectivity. Because it has much poorer spatial resolution, it would not be used for path geometry by itself. To compensate for wearer movement, a six-axis IMU containing an accelerometer and gyroscope would be used to measure the motion and orientation of the wearable and adjust the expected sensor geometry before deciding whether something is actually a hazard.

Schedule

9/10 – 9/23: Planning, sensor research, proposal
During the first two weeks after project groups are formed, we would brainstorm alternative solutions and agree on the exact minimum requirements for the project. Mainly, we would research two or three possible ToF sensors, ultrasonic sensors, IMUs, and microcontroller families, and order development boards early so we can test the sensors before committing them to a PCB. By the end of this period, we should have selected the basic sensor architecture and have a rough idea of our processor, I/O, battery, and mechanical requirements. Also complete the draft proposal by 9/17/2026.

9/24 – 10/6: Part selection and testing, basic software
Using development boards, test the ToF sensors, ultrasonic sensor, and IMU and confirm reliable communication with the microcontroller. Write basic code for sensor initialization and continuous data collection. Once the individual sensors are working, mount them on a temporary chest harness or 3D-printed frame to begin collecting realistic measurements while a person is walking.
By the end of this period, we should have at least a simple version of forward obstacle detection and haptic feedback working. Terrain detection and IMU compensation may still be rough, but we should have enough data to determine whether our proposed sensor layout is reasonable before finalizing the PCB design.

10/7 – 10/25: Breadboard prototype, PCB design
Begin combining ToF, ultrasonic, and IMU measurements and start implementing the main algorithm. Test whether the system can detect and distinguish objects correctly and determine optimal sensor placements and angles. Design and order the custom PCB by 10/25 (deadline 11/3), and begin wearable-mount design.

10/26 – 11/10: PCB fabrication, algorithm development
While waiting for the PCB to arrive, work on the software and algorithm, focusing on the details of terrain-discontinuity detection, IMU-based motion compensation, false-positive filtering, and haptic feedback. Also fabricate the final wearable enclosure, sensor mounts, and straps.

11/11 – 11/25: PCB testing and full system integration
Once the board arrives, test all the basic PCB functionality, then move the software from the development-board prototype to the custom board. Measure actual current consumption and estimate battery life. If needed, we could modify sensor update rates or add simple low-power behavior. Most of this period is for finding and fixing integration problems, and testing the obstacle-detection system while a person is moving and walking with the device.

11/26 – 12/10: Final testing, debugging, additional features, poster, expo
Test in several environments, including hallways, cluttered rooms, doorways, and outdoor sidewalks. Tune thresholds and filtering to reduce false alarms while maintaining good obstacle detection. Attempt stretch goals. Make a poster.

Rough Budget

Item	Estimated Cost
2 multizone ToF sensors/development boards	$50
Ultrasonic sensors and prototype hardware	$5
MCU development boards	$20-35
IMU development board	$20
3 Haptic drivers	$35
3 Vibration motors	$5-10
PCB fabrication and components	$80-100
Battery and charging/power circuitry	$20
Miscellaneous electronics + spares	$100-150
3D-printed enclosure, mounts, and wearable straps	$50
Estimated total	**around $400**

** The exact cost will depend on final component selection.

Implementation Issues

- Sensor reliability: Optical ToF sensors may behave differently in direct sunlight or when looking at dark, reflective, transparent, or very small objects. The ultrasonic sensor gives us an independent measurement when optical sensing is unreliable.
- Wearer motion: Since the sensors are mounted on the torso, every step changes their orientation. Without correction, these movements could easily look like changes in the ground. IMU-based orientation compensation and temporal filtering would be used to reduce this problem.
- False alarms and missed detections: A sensor can correctly detect an object that is not actually dangerous, so we have to explore how to determine whether an object actually intersects the expected walking path.
- Power: Because the device is intended to be worn continuously while walking, the final design must remain lightweight and operate for a useful period from a rechargeable battery. After the sensor and processor choices are finalized, we would measure actual current consumption and select a battery that balances runtime with size and weight.
- Mechanical mounting: The downward-facing sensor's angle relative to the body is important for terrain detection, so the wearable mount must keep the sensors reasonably fixed while remaining comfortable. A simple adjustable chest harness or vest mount should make this manageable.
- Safety: Because missed detections could cause injury, prototype testing would initially be performed in controlled environments with a sighted spotter, and the device would be treated as a supplemental aid rather than a replacement for a white cane or other established mobility aid.

None of these issues appears impossible within the semester, although false-positive reduction and reliable terrain sensing will probably require the most experimentation.

Stretch Goals

Possible additional features include distinguishing ramps from stairs, estimating whether an obstacle is moving toward the user, automatically changing warning distance depending on walking speed, and adding Bluetooth for data logging or configuration.

Evaluation and Testing

The final system would be evaluated in three areas, including hazard-detection accuracy, robustness to wearer motion, and usefulness of the haptic feedback. The obstacle-detection system should be able to correctly identify and distinguish between head-level hazards, frontal obstacles, and different terrain discontinuities placed at different distances and positions relative to the walking path. Motion testing would include walking at different speeds, leaning forward and backward, rotating the torso, and stopping suddenly to determine whether IMU compensation prevents these movements from being interpreted as terrain changes. In terms of user experience, each hazard category should produce the intended haptic warning, and the warning should occur early enough to be useful. False-positive testing would also be performed while walking through various environments to verify that nearby but non-threatening objects do not produce excessive warnings. A successful prototype should detect most test hazards consistently while maintaining a low enough false-alert rate that the feedback remains understandable rather than distracting.

References

[1] reddit
[2] wearable obstacle detection devices
[3] https://wewalk.io/en/
[4] https://www.besensable.com/post/about-vizion1

Wearable Air-Drum Set

High-Level Description

Traditional drum sets are large, loud, expensive, and difficult to transport. The goal of this project is to build a wearable electronic drum set that lets the user play drums almost anywhere by striking the air or nearby surfaces. Instead of carrying physical drums, the user would wear small motion-sensing devices on their wrists or hands and possibly one on their foot. The system would recognize different drumming gestures and convert them into drum sounds such as a snare, hi-hat, tom, cymbal, or bass drum.

Each wearable unit would contain an IMU, such as a 6-axis accelerometer and gyroscope, connected to a microcontroller on a custom PCB. The wrist units would continuously measure acceleration and angular velocity. A sharp acceleration followed by a rapid change in direction could indicate a drum strike. The direction and orientation of each strike could be used to classify different virtual drums. For example, different combinations of wrist orientation, swing direction, and gesture pattern could map to a snare, hi-hat, tom, or cymbal. A sensor worn on the foot could detect a tapping or stomping motion and act as the bass drum pedal.

The wearable devices would transmit detected hits wirelessly to a central receiver. The receiver would map each hit to the appropriate drum sound and play it through a speaker or connected computer. LEDs or vibration motors on the wearables could also provide feedback when a strike is successfully detected.

The finished system would ideally include two wrist-mounted sensor modules, one foot module, and a small base station. Unlike an electronic drum kit, it would not require a large frame or drum pads, making it much easier to carry around. It could also serve as a portable practice tool or a new type of musical controller.

Schedule

9/10 – 9/23: Planning, sensor testing, gesture experiments
Brainstorm alternative solutions and research component options. Decide on an IMU and wireless platform. Use development boards to record accelerometer and gyroscope data from different drum motions. Determine whether different strikes can be reliably distinguished.

9/24 – 10/6: Basic prototype
Build two wrist prototypes. Implement hit detection, orientation estimation, and simple drum classification. Send detected hits wirelessly to a computer or base station and play corresponding sounds.

10/7 – 10/25: PCB and wearable design
Select the final microcontroller, IMU, battery, charging circuit, and radio configuration. Design and order the custom PCB and begin enclosure/wristband design.

10/26 – 11/10: Software development

While waiting for the PCB, improve gesture recognition using the development-board prototypes. Add strike velocity so harder swings produce louder sounds. Develop the foot-controlled bass drum sensor and work on audio playback.

11/11 – 11/25: PCB integration and testing
Move the existing firmware to the final hardware. Measure wireless latency, battery consumption, hit accuracy, and false-trigger rates. Fix mechanical and electrical problems.

11/26 – 12/10: Full-system testing
Integrate all wearable modules, tune the virtual drum layout, and prepare the system for the Design Expo. Stretch goals could include user-configurable drum layouts, haptic feedback, recording performances, or recognizing more complicated gestures such as drum rolls.

Rough budget

Item	**Estimated Cost**
3 IMUs	$55
Wireless MCU development boards/modules	$60-100
Batteries	$25
Audio amplifier	$5
Speaker	$5-10
PCB fabrication and components	$50-70
Miscellaneous electronics + spares	$50
Estimated total	**around $300**

Implementation Issues
The biggest technical challenge would be distinguishing an intentional drum strike from normal hand motion. Simply triggering whenever acceleration exceeds a threshold would probably produce many false hits. This could be solved by finding a combination of acceleration magnitude, gyroscope data, timing, and hand orientation, and fine tuning the device to detect these specific combinations. We could start with simple threshold-based detection and then collect motion data to tune the algorithm.

Another challenge is latency. For the instrument to feel playable, there cannot be a noticeable delay between a user's motion and the resulting sound. Because only a few bytes need to be transmitted for each detected hit, BLE or another short-range low-latency radio appears feasible. Wireless and end-to-end audio latency would be measured early in prototyping to verify that the instrument remains responsive.

Evaluation
Evaluation would measure hit-detection accuracy, incorrect or missed drum classifications, false triggers during normal arm motion, and end-to-end latency between a physical gesture and audible output. Known drum sequences could be repeated multiple times at different speeds to determine whether the generated sounds consistently match the user's intended motions.

Low-Speed Surround Clearance Assistance System for Novice Drivers

Many new or less experienced drivers have difficulty judging how close different parts of their vehicle are to nearby obstacles. This becomes especially challenging when parking between cars, driving through narrow streets, entering tight garages, or passing vehicles parked on both sides of the road. A driver may be able to see an obstacle through a mirror or camera, but it is still difficult to translate that view into a clear understanding of how much space remains around the front corners, sides, or rear of the vehicle. This uncertainty can lead to scratches, minor collisions, or simply make drivers overly nervous in situations that should otherwise be manageable.

We propose a low-speed surround clearance assistance system that gives drivers a simple, real-time view of the space around their vehicle. Distance sensors placed around the front, rear, sides, and corners of the vehicle will continuously measure nearby obstacles. The system will combine these measurements into an easy-to-understand representation of the vehicle's surroundings rather than requiring the driver to interpret several separate sensor readings.

Details

The main driver interface will be a dashboard-mounted HUD-style display positioned close to the driver's normal field of view. A top-down image of the vehicle will be surrounded by different warning regions corresponding to the front, rear, sides, and corners. These regions will change from green to yellow to red as obstacles become closer. When an obstacle becomes particularly important, the display can also show its approximate distance and a short warning such as “Left front clearance low” or “Stop.” An audible alert can provide additional warning without requiring the driver to continuously look at the display.

The finished product would be a self-contained embedded system with several vehicle-mounted sensor modules, a central embedded controller, and a small HUD-style display on the dashboard. It is mainly intended for new or less confident drivers in low-speed situations such as parking, moving through narrow spaces, or entering tight garages. The system would not take control of the vehicle. Instead, it would give the driver a clearer idea of what is around the car and help reduce minor collisions and stressful low-speed maneuvers.

Schedule

9/9–9/24	System planning and architecture Finalize the project scope and requirements. Decide how many distance sensors are needed, where they should be placed, and what sensing technologies are worth testing. Decide what information the HUD should show and investigate practical HUD designs, including the display, transparent combiner, brightness, viewing angle, and mounting position. Choose a processing platform based on processing needs, display support,

	real-time requirements, power use, PCB integration, and software complexity. Also decide how the system will be powered and estimate its power use and operating time. Prepare the draft proposal and update the design based on feedback from the proposal meeting.
9/25–10/15	Early prototyping and implementation Order the development board, sensors, display, HUD materials, and other prototype parts. Test the main components individually and start building the first working prototype. Implement sensor reading, basic distance processing, HUD graphics, warning logic, and the FreeRTOS task structure. By Milestone 1, at least one sensor should work through the full pipeline from distance measurement to a warning shown on the HUD.
10/16–11/3	Full prototype and PCB design Add more sensors and expand the prototype toward the final sensor layout. Improve distance filtering, warning-zone logic, and HUD behavior. Complete the schematic and PCB layout for the main controller, including the processor, sensor and display connections, alert outputs, debugging connections, and power regulation. Check that the software and interfaces used on the development board can transfer to the custom PCB without major changes. Order the PCB before November 3.
11/4–11/13	System refinement during PCB fabrication While waiting for the PCB, continue improving the multi-sensor prototype. Work on sensor calibration, filtering, warning thresholds, HUD responsiveness, and FreeRTOS communication. Test different obstacle positions and distances and fix remaining software issues. Start designing the HUD enclosure and sensor mounts, and prepare the software for the custom PCB. By Milestone 2, most of the required functions should already work on the development-board prototype.
11/14–11/30	PCB bring-up and full-system integration Assemble and test the custom PCB. Check power, programming/debug connections, sensor interfaces, display communication, and alert outputs. Move the existing firmware from the development board to the custom PCB and connect the full sensor system, HUD, and alerts. Install the electronics into the planned enclosure and mounts, then repeatedly test the full system and fix hardware/software integration problems.
12/1–12/09	Final testing, tuning, and Design Expo preparation Finish system testing and tune the sensor calibration, warning thresholds, and HUD behavior. Test distance accuracy, warning latency, obstacle-direction detection, and reliability in situations such as parking and narrow passages. Finish the enclosure and mounting, prepare the poster and Expo demo, and practice the demonstration several times.

	Stretch features will only be added if the core system is already stable.
12/10	Design Expo!!!!!!
12/11	Finish and submit the final report.

Budget

Exact components will not be finalized until the planning and prototyping stages, so the numbers below are only a first estimate. The largest expected costs are the distance sensors, HUD hardware, PCB fabrication and components, and mechanical mounting parts. Extra development boards and sensors are included for parallel development and as spares. The final device only needs one main controller and one HUD, so not all prototype hardware will be used in the final system.

What	Quantity	Estimated Cost per Unit	Estimated Total
Distance sensors	10	$15	$150
Microcontroller development boards	3	$50	$150
High-brightness TFT display	1	$60	$60
HUD combiner / optical materials	1	$40	$40
Custom PCB fabrication, components, and assembly supplies	5 boards	$30	$150
Buzzer / haptic / alert hardware	1 set	$40	$40
power supply / battery hardware	1	$40	$40
Enclosure and sensor mounting materials	1 set	$60	$60
Connectors, wiring, and miscellaneous parts	1 set	$20	$20
Shipping			$70
Estimated Total			$780

Implementation Issues

Sensor reliability and placement. The system needs reliable distance measurements around the front, rear, sides, and corners of the car. Sensor readings may change depending on the angle, shape, or surface of an obstacle. We will test different sensor types and mounting positions and use filtering, calibration, and overlapping coverage in important areas to reduce blind spots and improve reliability.

HUD and mechanical integration. The HUD needs to be easy to read without blocking the driver's view. We plan to use a high-brightness display reflected from a transparent combiner so that the warning information is visible while the driver can still see the road through the display area. The HUD size, brightness, viewing angle, and mounting position will be adjusted during testing. It needs to be large enough to read quickly, but not so large that it blocks the driver's forward view or becomes distracting. The HUD enclosure and sensor mounts can be made with 3D printing and adjusted during prototyping.

PCB, software, and power. Most of the software will first be written and tested on a development board before moving to the custom PCB. This way, PCB delays or board problems will not stop software development. Several PCB copies will be ordered in case one is damaged or assembled incorrectly. The PCB will connect the processor, sensors, display, alerts, debugging hardware, and power system. We will avoid switching regulators and use a regulated power source with simpler linear regulation where needed.

Final demonstration. Bringing a full-size car to the Design Expo would not be practical, so the live demo will use a remotely controlled scale vehicle with the sensor system installed on it. Obstacles can be placed around the vehicle to simulate narrow passages, close parking, and corner-clearance situations while the HUD updates in real time. We can also show recorded video from controlled testing on a real vehicle to demonstrate how the system works in the environment it is actually designed for.

Evaluation and Testing

The system will be tested both at the sensor level and in complete low-speed driving scenarios. Distance accuracy will be measured by placing obstacles at known distances and comparing those distances with the sensor readings. We will also test warning latency, whether the system identifies the correct obstacle direction, and how often it gives false or missed warnings. The HUD will be tested under different lighting conditions and viewing angles to make sure the information is still easy to see.

For more realistic testing, the system will be used in situations such as narrow passages, parking near obstacles, and approaching objects near different corners and sides of the vehicle. A successful system should consistently detect nearby obstacles, show the correct warning region, and give the driver enough time to respond. Controlled testing on a real vehicle can also be recorded to check that the system remains useful outside of the scale-car demo.

Additional Stretch Goals

If the main surround-clearance system is finished early, one stretch goal would be to make the HUD steering-aware. By measuring or estimating the steering angle, the system could predict the approximate path of the car and show when that path may intersect with a nearby obstacle. This would make the HUD more useful than simply showing the current distance to obstacles. Another possible stretch goal would be basic rear-side sensing for blind-spot or lane-change warnings. This would only be attempted after the low-speed clearance system is already working reliably.

UWB-Assisted Hands-Free Autonomous Travel Vlogging Drone

Many people like recording videos while traveling, but filming a vlog usually requires holding a phone, camera, or selfie stick for long periods of time. This becomes inconvenient when hiking, walking on steep trails, cycling, carrying luggage, or doing other activities where having both hands available would be useful. It also limits the types of shots a traveler can capture without asking another person to act as a camera operator.

This project proposes a compact autonomous camera drone for hands-free travel vlogging. The drone would follow one selected traveler and automatically keep that person properly framed while recording. Since a drone flying directly next to someone would be noisy and distracting, the system would try to maintain several meters of distance while still producing useful video. Keeping the drone portable while still getting enough flight time, camera quality, and tracking range will be one of the main design challenges.

Rather than relying only on face recognition, the traveler would carry a small UWB (Ultra-Wideband) beacon that helps the drone estimate the range to the target and provides a coarse cue for locating the correct traveler. A camera and visual tracking system would then provide more precise localization and framing. This combination would be more reliable than vision alone when the traveler turns away from the camera, is briefly partially blocked, or is surrounded by other people.

The finished device would ideally be small enough to fit into a backpack and useful for short autonomous recording sessions while traveling. A stabilized camera would record video, while a wireless microphone worn by the traveler could record speech separately so that rotor noise does not dominate the audio.

Rough Schedule

September	Select the drone platform, flight controller, UWB system, camera, companion computer, and overall architecture.
October	Bring up the drone and UWB tracking system, implement basic visual tracking and communication with the flight controller, and begin schematic and PCB design for the tracking and safety control board.
November	Complete and order the custom PCB, continue UWB and visual tracking development while it is being fabricated, then bring up the board and migrate the embedded software once it arrives. Integrate the tracking system with autonomous following and camera framing.
Early December	Test complete follow-and-film scenarios, improve tracking

	reliability and flight behavior, and prepare the final demonstration.

Implementation Issues

Embedded hardware and flight control. Designing the entire flight controller and stabilization system from scratch would add significant risk, so an existing open-source flight controller would be used for basic flight stabilization. A custom microcontroller-based PCB would handle UWB ranging and filtering, obstacle sensors, and communication with the companion computer and flight controller. Besides running FreeRTOS, the board would also implement watchdog and failsafe logic and check high-level tracking commands before forwarding them to the flight controller. If communication is lost or an obstacle is too close, it could request the drone to hover or stop following the target. A small Linux-based companion computer would run the more computationally expensive visual tracking and framing algorithms.

Weight, battery life, and mechanical design. The drone should remain small enough to carry easily while still supporting the camera, processing hardware, battery, and sensing hardware. Battery capacity, total weight, and flight time will need to be balanced during prototyping, while noise will also affect how closely the drone can follow the traveler. Lightweight 3D-printed parts could be used for sensor mounts, electronics housings, and propeller guards without adding too much weight.

Safety and testing. Autonomous drone testing introduces a greater safety concern than a stationary embedded system. Initial tests would be performed in a controlled area with propeller guards and, when appropriate, a physical tether to reduce the risk of an uncontrolled flight. Outdoor testing would only begin after basic tracking and flight behavior work reliably, and real-world tests would follow applicable flight and campus restrictions.

Rough Budget

The largest costs would likely be the drone frame and propulsion system, flight controller, camera, companion computer, UWB hardware, custom PCB fabrication and components, batteries, and spare parts. Because this is still an early proposal, the exact parts have not been selected yet, so the estimate is intentionally rough. A total budget of about $700–900 should be enough to build a working prototype while still leaving some room for replacement parts or changes during testing. The final cost will depend most heavily on the camera, flight platform, and UWB hardware selected.

Evaluation

The drone could be evaluated by having a person follow a predefined walking route while the system attempts to keep that person in frame. We would measure tracking success rate, how well the traveler stays centered in the frame, following distance, response to direction changes, battery life, and how often tracking is lost. Additional tests could include briefly blocking the camera view or placing several people nearby to see whether the UWB-assisted system continues following the correct traveler.

Additional Wacky Stretch Goal

If the basic follow-me drone works reliably, a more unusual stretch goal would be to design it so that it can dock onto or fold against a traveler's backpack when not flying. In this mode, the camera could act like a normal body-mounted travel camera. When the traveler reaches an open area and wants a third-person shot, the drone could be removed or deployed and switch into autonomous follow mode.

Name:Zheng Lu, Unique name: zhelu

Low-Cost FPGA-Based Portable Logic Analyzer

High-Level Description

Digital communication protocols such as UART, SPI, and I2C are commonly used in embedded systems to connect processors, sensors, memories, and peripheral devices. When developing or debugging these systems, engineers often need to observe the timing and values of multiple digital signals at the same time. An oscilloscope can provide detailed electrical information, but it is often inefficient for decoding long sequences of digital data. Commercial logic analyzers are designed for this purpose, but capable devices can be relatively expensive, while inexpensive alternatives may have limited sampling rates, channel counts, triggering capability, or protocol support.

This project proposes a **low-cost, portable FPGA-based logic analyzer** intended for students, hobbyists, and embedded-system developers. The device will capture several digital input channels simultaneously, store the sampled waveform around a configurable trigger event, and transfer the captured data to a computer through USB. A simple host application will display the digital waveforms and decode at least one common embedded protocol such as UART. The target system will support approximately eight digital input channels and a configurable sampling rate with a target maximum of at least 20 MSamples/s.

The main architectural feature of the project is the use of an FPGA for time-critical data acquisition. A microcontroller could sample GPIO signals in software, but its achievable sampling rate and timing determinism may be limited by instruction execution, interrupt latency, and other firmware activity. An FPGA can instead sample all input channels simultaneously on each sampling clock while also performing trigger detection, timestamp generation, and memory control in parallel. This makes the FPGA well suited for the real-time front end of the logic analyzer.

The proposed system will consist of four major components:

Digital inputs → FPGA acquisition logic → microcontroller → USB → host computer

The FPGA will implement the sampling engine, trigger logic, circular capture buffer, and possibly hardware-assisted protocol decoding. The microcontroller will configure the FPGA, manage capture operations, and communicate with the host computer. The host software will allow the user to configure acquisition settings and visualize the recorded signals.

Physically, the final device is expected to be a small PCB enclosed in a portable case. One side of the device will expose approximately eight digital input channels and a ground connection using header pins or test leads. The other side will contain a USB-C connector for power and host communication. LEDs may indicate power, acquisition state, trigger detection, and communication status.

An initial target specification is:

- 8 digital input channels
- At least 20 MSamples/s maximum sampling rate
- Programmable lower sampling rates
- Rising-edge and falling-edge trigger support
- Pre-trigger and post-trigger waveform capture
- USB host communication
- UART protocol decoding

- Input support for common 3.3 V digital systems

SPI and I2C decoding, more advanced trigger conditions, higher sampling rates, and waveform compression will be treated as stretch goals rather than required functionality.
The final product is intended primarily as an educational and development tool rather than as a replacement for high-end laboratory equipment. The emphasis will be on low cost, portability, reliable digital capture, and providing enough functionality to debug typical embedded hardware.

Implementation Issues

The project will first be developed using FPGA and microcontroller development boards before a custom PCB is designed. This reduces the risk of discovering architectural or firmware problems only after the PCB has been manufactured.
The FPGA portion will contain several RTL blocks. The input sampling block will synchronously capture all digital channels at the selected sampling frequency. A configurable trigger engine will monitor the inputs for conditions such as rising edges, falling edges, or simple bit patterns. Before the trigger occurs, samples will continuously be written into a circular buffer so that activity immediately before the trigger is preserved. After the trigger is detected, the FPGA will continue recording for a programmable number of samples and then stop acquisition.
The capture memory will most likely use the FPGA's internal block RAM during the first implementation. Memory capacity is one important design constraint. For example, continuously recording eight channels at tens of millions of samples per second generates data much faster than a small embedded system can permanently store. Therefore, this project is intended to capture relatively short waveform windows surrounding important events rather than continuously record signals for long periods. Lower programmable sample rates can also be used for slower protocols such as UART.
The microcontroller will communicate with the FPGA through a relatively simple interface such as SPI. It will send configuration information including sample rate, trigger type, trigger channel, and capture length. When acquisition is complete, it will read the stored samples from the FPGA and transmit them to a host computer over USB.
A simple desktop program, likely written in Python, will provide the user interface. It will configure the device, receive captured samples, and display each channel as a digital waveform. The first required protocol decoder will likely be UART because it is simple to generate, verify, and decode. SPI and I2C decoding will be added if schedule permits.
The custom PCB will integrate the processor, FPGA or FPGA module, USB interface, voltage regulation, clock generation, and input circuitry. Rather than using a large BGA FPGA that would make PCB routing and assembly unnecessarily difficult, a small FPGA available in a relatively accessible package or FPGA module will be preferred. The exact devices do not need to be selected at this stage, but a low-cost FPGA in the Lattice iCE40-class together with an STM32-, RP2040-, or similar microcontroller should provide sufficient resources for the target design.
Input protection is another important issue. External systems may operate at different logic voltages, and directly connecting an excessive voltage to an FPGA input could damage the device. The required implementation will therefore initially specify operation with 3.3 V logic, with appropriate series resistance and input protection. If a suitable level-shifting solution can be implemented without significantly increasing complexity, support for 5 V logic may be added.
Signal integrity may also become an issue at high frequencies. Because the goal is a low-cost educational instrument rather than a very-high-performance commercial analyzer, the initial

sampling-rate target will intentionally remain in the tens-of-MSamples/s range. This should make PCB layout, input circuitry, and validation more manageable within one semester.

Proposed Schedule

Phase 1 – Architecture and Component Selection

Define the number of channels, sampling-rate target, capture-memory size, FPGA-to-microcontroller interface, and host communication protocol. Evaluate candidate FPGA and microcontroller devices and obtain development boards. A basic RTL simulation environment will also be created during this stage.

Phase 2 – FPGA Acquisition Prototype

Implement and verify the FPGA sampling engine, programmable clocking, rising/falling-edge trigger logic, circular sample buffer, and pre/post-trigger capture using a development board. Use known digital waveforms from another microcontroller or function generator to verify basic operation.

Phase 3 – MCU and Host Software Integration

Implement communication between the FPGA and microcontroller. Develop a simple USB command protocol and Python host application. The host software should be able to configure a capture, retrieve samples, and display digital waveforms. UART decoding will also be developed during this period.

Phase 4 – Custom PCB Design and Ordering

Complete the schematic and PCB layout, including FPGA/MCU circuitry, power supply, USB, clocking, connectors, and protected digital inputs. Order the PCB and remaining components. During the approximately one- to two-week PCB manufacturing period, continue improving the FPGA RTL, firmware, host software, and protocol decoder rather than waiting for the board.

Phase 5 – PCB Bring-Up and System Integration

Assemble and test the custom PCB. Verify power rails, clocks, FPGA configuration, MCU operation, USB communication, and input channels independently before running full-system captures. Resolve hardware and firmware problems and make a PCB revision only if required.

Phase 6 – Validation and Final Demonstration

Perform quantitative testing of sampling rate, timing accuracy, trigger behavior, communication reliability, and UART decoding. Compare measured waveforms against an oscilloscope or existing logic analyzer. If the required functionality is complete, implement stretch goals such as SPI/I2C decoding, pattern-based triggering, or higher sampling rates.

This schedule intentionally performs most RTL and software development before the custom PCB arrives so that PCB manufacturing delay does not stop overall project progress. This follows the assignment's emphasis on accounting for PCB lead time and leaving substantial time for debugging.

Budget

A preliminary budget is shown below. Exact parts may change after component selection.

Item	Estimated Cost
FPGA / FPGA module	$25–40
Microcontroller	$5–15
FPGA/MCU development hardware	$20–40
Custom PCB manufacturing	$30–50

Item	Estimated Cost
USB/power circuitry	$5–10
Input protection/level shifting	$5–15
Connectors and test leads	$10–15
Enclosure	$10–20
Miscellaneous components/spares	$20
Estimated total	**$130–$180**

Evaluation and Testing

Success will be evaluated using measurable electrical and functional requirements rather than only checking whether digital waveforms appear on a computer screen.

The first metric will be **sampling performance**. A function generator or microcontroller will generate known square waves at several frequencies. The proposed analyzer should reliably capture all channels simultaneously at its specified maximum sampling rate. A target of at least 20 MSamples/s on eight channels will be used for the initial design.

The second metric will be **timing accuracy**. Input pulse periods and widths measured by the FPGA-based analyzer will be compared with measurements from an oscilloscope or commercial logic analyzer. The target is for measured digital timing to be within approximately one sampling-clock period of the reference measurement.

The third metric will be **trigger correctness**. Known rising-edge and falling-edge events will be generated at controlled times. The system should reliably stop acquisition at the requested trigger and preserve both pre-trigger and post-trigger data. Repeated trigger tests will be performed to verify deterministic behavior.

The fourth metric will be **protocol-decoding accuracy**. A microcontroller will generate known UART messages at standard baud rates. Captured traffic will be decoded by the host application and compared automatically with the transmitted bytes. The target is error-free decoding over repeated transmissions under valid electrical and timing conditions.

Finally, the complete system will undergo longer-duration reliability testing. Multiple captures will be repeatedly initiated and transferred over USB to check for corrupted samples, lost captures, or host communication failures.

The project will be considered successful if the final custom PCB can independently capture multiple digital channels, trigger on selected digital events, preserve signal history around the trigger, transfer captured data to a computer, and correctly display and decode at least UART traffic.

Additional Risks and Stretch Goals

The largest technical risks are expected to be PCB bring-up, FPGA memory limitations, digital input protection, and communication bandwidth between the FPGA, microcontroller, and host. These risks are manageable because the system can first be validated using development boards, while the required feature set does not depend on advanced protocol decoding or extremely high sampling rates.

If the required system is completed early, possible stretch goals include SPI and I2C decoding, multi-channel pattern triggering, protocol-aware triggering, 50 MSamples/s or greater sampling, compressed edge-based storage, and support for additional input voltage levels.

The primary goal, however, will remain the construction of a reliable, low-cost embedded debugging tool rather than maximizing the number of features.

FPGA-Based Smart Basketball Shooting Coach

High-Level Description

Basketball players often improve their shooting technique through repetition and feedback from coaches or video recordings. However, recreational players usually do not have access to immediate quantitative feedback about their shooting motion. A shot may miss because of inconsistent wrist motion, release timing, or arm movement, but these differences can be difficult to identify by observation alone.

This project proposes an **FPGA-based wearable basketball shooting coach** that measures a player's shooting motion and provides immediate feedback about shot consistency. A small wearable device attached near the shooting wrist would contain an inertial measurement unit (IMU), an embedded processor, and an FPGA. The IMU would continuously measure acceleration and angular velocity while the player shoots. These signals would be processed in real time to identify individual shooting attempts and extract motion characteristics such as peak wrist angular velocity, acceleration, shot duration, and release timing.

The basic system would operate as:
Wrist IMU → FPGA Processing → Microcontroller → Bluetooth → Phone/Laptop

The FPGA would perform the time-sensitive sensor processing. Rather than having the microcontroller continuously process every IMU sample, the FPGA could implement filtering, shot detection, timestamping, and feature extraction as parallel hardware blocks. Once a shot is detected, only the relevant motion statistics or a short window of sensor data would need to be transferred to the microcontroller. The microcontroller would manage the IMU, configure the FPGA, and transmit results over Bluetooth to a simple phone or laptop interface.

The user's previous successful shots could be used to establish a personal reference motion. After each new shot, the device could compare measurements such as angular velocity, acceleration, and motion duration with this baseline and generate a simple **shot consistency score**. For example, a player could determine that missed shots are frequently associated with a slower wrist rotation or an unusually early release. The goal would not be to determine a universally “perfect” shooting form, but rather to help a player understand how consistent their own shooting mechanics are.

A stretch goal would add a second wireless sensor near the basketball hoop. An infrared, vibration, or acoustic sensor could determine whether the ball entered the basket. The wrist device and hoop node could then correlate shooting mechanics with actual makes and misses:
Wearable motion data + Make/Miss result → Shot history and feedback
This could eventually allow the system to report information such as, “Shots with wrist angular velocity near your normal range were made 70% of the time.”

Implementation Plan and Issues

Development would begin using an FPGA development board, microcontroller board, and commercially available IMU. Known hand movements would first be recorded to determine what characteristics reliably distinguish a basketball shot from normal movement. The FPGA

RTL would then implement sensor-data buffering, digital filtering, threshold/event detection, timestamping, and basic feature extraction.

After the signal-processing pipeline works on development hardware, a compact custom PCB could combine the IMU, processor, FPGA or small FPGA module, power-management circuitry, and rechargeable battery. The board could be placed inside a wrist strap or lightweight enclosure.

The main challenge is determining when a shooting motion starts and ends. Simple thresholds based only on acceleration may incorrectly classify running or other arm movements as shots. A combination of acceleration, angular velocity, duration, and state-machine-based detection would therefore be used. More complicated machine-learning classification could be considered as a stretch goal but would not be required for the initial system.

Another issue is physical size and weight. Because the device is intended to be worn while shooting, the PCB and battery must be small enough that they do not significantly interfere with normal wrist motion.

A rough schedule would consist of sensor characterization and architecture design, FPGA motion-processing development, MCU/Bluetooth integration, PCB design and manufacturing, wearable integration, and final basketball testing.

The expected cost is approximately **$100–$150**, including the FPGA or FPGA module, MCU, IMU, rechargeable battery, PCB, enclosure/wrist strap, and optional hoop sensor. This is within the course's $200-per-student budget.

Evaluation

The system would first be tested using repeatable controlled wrist motions and then during actual basketball shooting. The primary evaluation metric would be **shot-detection accuracy**: the device should correctly distinguish shooting attempts from unrelated arm movements. A reasonable initial target would be greater than 90% detection accuracy during a controlled shooting session.

The consistency of extracted motion features would also be evaluated by recording repeated shots from the same player and determining whether similar shooting motions produce similar measured acceleration and angular-velocity profiles. Communication reliability and feedback latency would also be measured to ensure that results can be presented shortly after each shot.

If the hoop sensor stretch goal is implemented, its make/miss result would be compared manually against recorded shots. The final demonstration could show a player taking a series of shots while a laptop or phone displays the detected shot count, wrist-motion measurements, consistency score, and—if available—make/miss result in real time.

The final goal is to create a wearable embedded system that makes basketball practice more quantitative while demonstrating how FPGA-based real-time processing can be integrated with sensors, wireless communication, and a physical embedded product.

Bike Computer and Power Meter Combo

In recent years, cycling has become one of the most popular activities recorded using platforms such as Strava and Garmin Connect, where riders log their rides, performance, data, and training progress. Currently, many cyclists rely on a phone or a watch to record their rides, which both are limited in collecting bike sensor data and can be inconvenient and distracting while riding. Bike computers solve this problem by consolidating all functions into a display mounted in the middle of the handlebars, from tracking distance traveled via gps, to collecting other data from connected sensors like power, speed, heart rate, and cadence. Unfortunately, many cyclists are priced out or are satisfied with just their phone or watch because such systems that have all of those features such as the Garmin Edge bike computers can cost hundreds of dollars before even considering other sensors.

Apart from recording basic ride statistics, one of the most important measurements for most performance focused cyclists is measuring power. Unlike speed, which can vary significantly depending on road conditions, wind, and other external factors, power provides a more direct measure of a rider's effort and is easy to compare between rides. For this reason, cyclists use a sensor called a power meter to measure power as their main data point to monitor long term progress and planning workouts. However, power meters can also cost several hundred dollars, and the large number of available options and styles can make them intimidating or inaccessible to newer cyclists who aren't sure they want to commit fully to a top of the line model.

This project aims to develop a low cost bike computer and power meter combination as a single integrated system. By combining ride tracking, real-time cycling data, and power measurement in a single package, this could provide new or budget-conscious cyclists with access to features that are typically found in much more expensive equipment. The goal of the project is to create a simpler and affordable entry point into cycling technology without the user needing to purchase multiple separate devices.

Details

The bike computer portion of the project will be mounted near the center of the handlebars, allowing information to be viewed easily without impeding vision too much. It acts as the hub/master of the system, and will contain a display, GPS module, processor, SD card, buttons, and wireless communication hardware. The GPS data will be used to determine the rider's location and calculate speed. When an activity is started using one of the buttons on the device, the computer will periodically record GPS coordinates, timestamps, speed, power, cadence, and other available sensor measurements to a GPX-compatible file stored on an SD card. By pressing the button again and confirming to end the ride, this file can be transferred via the SD card to a computer or phone and uploaded to the cyclist's favorite service such as Strava, Garmin Connect, or intervals.icu.

The power meter will be located near the bicycle crank or pedal area, where forces generated and the revolutions per minute spins of the pedal (cadence) can be measured. A strain gauge will measure the force the rider applies to the drivetrain. Using just one strain gauge on one pedal is usually good enough for most cyclists, where they take the power from one side and multiply by 2, but there are many dual-side true left/right power meters on the market, where we would then need to use two. This could be a flex goal if time allows, since we

would need to sync the two together. A hall-effect sensor will also be used to detect crank rotations and determine cadence. Using the measured force and crank speed, the power meter system will determine the rider's power in watts. Power and cadence data will then be transmitted wirelessly to the bike computer through Bluetooth Low Energy.

Bluetooth Low Energy will be a communication method between the power meter and bike computer. The bike computer will also be designed to support compatible third-party Bluetooth cycling sensors, such as heart-rate monitors and speed sensors. Support for ANT+ could also be explored if time allows, since many sensors communicate through ANT+. Using these standard wireless communication protocols will make the system expandable rather than limiting it to just the sensors in the project. A cyclist could initially just use the bike computer and include a power meter, then later add additional sensors without needing to replace the bike computer. This allows the system to act as a long term platform for collecting and managing cycling data instead of being replaced when the rider wants additional functionality.

Schedule

These are rough time estimates and goals, subject to change.

9/10-9/23	**Planning** After group formation, we will spend the first two weeks planning how to tackle the project and research parts. Research for each component (sensors + processors) will include looking at multiple options and considering the tradeoffs for each, then picking the best for our application (probably low-power + accuracy for sensors and low-power for processor platform). Try to identify if any other smaller parts are needed, like capacitors, resistors, or similar. Have a draft proposal ready by 9/18.
9/24-10/8	**Project Design and Parts, Begin Breadboarding** Once the group goes through the proposal meeting process, try to order two of each major part (LCD, sensors), and see if there's a devkit for the processor selected or multiple devkits of different families if not decided yet and want some initial testing. Start breadboarding parts individually and try to get them to work independently. While doing this, identify any potential issues and how to tackle them, and try to get this done so it can be incorporated into the final proposal for 10/1. Additional parts should be ordered and finalized before the end of this period. Start looking at PCB design.
10/8-10/27	**Prototype + PCB design** If independent parts working is successful, start integrating them together as a full prototype. Get most important features working first (logging gps data to a GPX file, strain gauge measurements can be converted to watts), then work on other features (good-looking screen UI, bluetooth, etc). Make effort in PCB design, and whether there needs to be multiple (one for bike computer, one for power meter). Finalize PCB design and try to order by the end of the period, which is about a week early, in case we need to order a second revision.

10/27-11/10	**Prototype Testing and Software** Once simple integration + functionality is established, the prototype of all connections is mostly finalized, and waiting for PCBs, this is where the refinement of the software should be completed. Have all primary goals for software finished and tested on the prototype before the PCBs come in. Document issues for report. Assemble PCBs and identify any initial issues before the end of the period. If time allows, design a case for the bike computer and print/order as soon as possible. If possible, mount strain gauge and cadence on a bike and see if there's output and accuracy to a real power meter.
11/10-11/23	**PCB Testing and Software** Test the PCB and software. Most things should be working from prototype, debug any issues. If there is a big bug on the PCB and cannot be fixed easily or in software, consider ordering a second revision. Document issues for report. Take the project outside and test GPS accuracy and power meter. Try to connect additional sensors, like a bluetooth heart rate monitor. Take the project for a spin, and see if the GPX file can be written properly and uploaded to Strava. If time allows, consider additional features to the project, like a simple radar/ or lights on the seatpost of the bike for visibility that can be controlled via the bike computer.
11/24-12/10	**Final Testing, Poster, Expo** Ideally begin the final report by 12/3, so make sure the project is complete by that date. Complete and order poster for 12/10 expo, and finish final report for 12/11.

Budget

It is hard to get an accurate budget, but this should be decently reasonable:

Component	Qty	Est Cost	Cost Total	Notes
Processor	2	$10	$20	One for bike computer, one for power meter
GPS	1	$20	$20	
LCD	1	$20	$20	
Strain gauge + amplifier	1	$20	$20	Buy two if we want true left/right power
Hall-effect sensor	1	$5	$5	
PCB (bike computer)	5	$15	$75	
PCB (power meter)	5	$15	$75	
Battery/power parts	2	$15	$30	

SD Card Module	1	$10	$10	
Misc Parts	1	$100	$100	
Cost Total			$375	

Issues

The biggest issues that may arise during the project are:

- **Interfacing with GPS**: need to reliably receive and process GPS data when the rider is moving, and also figure out what to do when there's poor signal (ex. Passing under a tunnel).
- **Logging all data to a GPX file, including various sensor data + learning GPX file format**
- **Algorithms needed to keep track of all sensors and display in real-time**
- **Bluetooth/possible ANT+ communication with third-party sensors + how to display them on the LCD**

Most of these issues should be reasonable, except maybe connecting to third party sensors. This is because it depends on the protocols supported by the external devices, and how they would output data to our bike computer. If this proves more difficult than expected, this can be a flex goal, while maintaining bluetooth communication between the project's power meter and bike computer as a core requirement.

Additional Stretch Goals

One stretch goal could be to implement dual-sided strain gauge sensing to independently measure left/right leg power. This provides true left/right power balance rather than estimating total power from a single crank by multiplying by two. This adds a more advanced feature found in higher end power meters.

Another could be auto-uploading activities to Strava if the bike computer has access to wifi through the Strava API. Moving files through the SD card is tedious and can be annoying, especially if a user cycles frequently. Or, the bike computer sends completed activity to a phone app which then uploads through the Strava API.

Testing

Testing can be completed by comparing results to manually tracking bike activities through the Strava App to check GPS accuracy, and also through a Garmin Edge Bike Computer + Power Meter combo mounted on the same bike to measure accuracy and responsiveness of the system.

Pickleball Training Robot

Pickleball is a rapidly growing sport, and many players want to improve but may not have the time or budget for frequent coaching or practice partners. While commercial pickleball training robots already exist, machines with more advanced features can cost thousands of dollars, which is not practical for newer players, and can even not be programmable, which is not practical for more advanced players. This project proposes a lower-cost, user-programmable pickleball training robot that can feed balls to different locations with adjustable speed, spin, and frequency.

Details

The robot will be stationary while adjusting the ball in two axes. One motor moves the launcher left and right, while another controls the vertical angle of the launcher. Together, these will allow the machine to target different areas of the court without needing to move the whole robot.

The launching mechanism itself could start with two independently controlled spinning wheels positioned above and below the ball. By varying the speed of the wheel, it would be sufficient for controlling launch speed and producing topspin or backspin scenarios. A four wheel design could be explored, with the wheels positioned above, below, left, and right of the wheel to provide more precise control over sidespin and ball trajectory.

A hopper and motorized feeding mechanism will be able to store many pickleballs and release balls one at a time into the launcher. The processor will coordinate motor speed, horizontal and vertical aiming, and ball feeding according to the user program, allowing the user to control shot placement, speed, height, spin, and frequency. Maintaining consistent and repeatable shots may also require feedback sensors to monitor the motors and ensure the correct speed and orientation is met each time the ball is launched.

A touchscreen LCD/tft or companion application could provide access to common preprogrammed drills as well as allow the user to create custom drills. The user could adjust things like the number of balls, their target location, speed, and spin. For example, if a player wants to practice blocking drives at the net, they could program fast shots aimed for the far end of the court, then stand near the net to block them. More advanced drills could cycle between multiple court locations, like dropping the ball short on the left and right for a while and then lobbing the ball to simulate realistic game scenarios. A random game mode could also be implemented to have random but realistic point simulation.

The goal is for the robot to remain fixed while still being able to reach most areas of the court through a combination of launch speed, spin, and two-axis aiming.

Issues

An major implementation issue could be mapping the user selected shot parameters to the physical motor controls of the launcher. From the user's perspective, the interface should expose intuitive settings like location, speed, etc. Internally, these need to be converted into specific values for the motors, wheel speeds, and feeding mechanism. This mapping may require significant calibration and testing because the relationship between the motor commands and the resulting ball trajectory is probably not linear. A lot of factors can affect

where the ball lands. The software would need good control functions or lookup tables that convert the user inputs into the appropriate launcher settings.

Another challenge will be the selection of the motors and making sure that they would be powerful enough for aiming and launching. The motors controlling left/right up/down would need to support the weight of the whole rig while still moving accurately and repeatedly. This may require more geared/higher torque motors. The launcher motors also would be challenging to pick. We need to make sure that they can reach the high rotational speeds to launch the ball in the right location with the right spin while also having enough torque to avoid slowing down when a ball contacts the wheels. Motor selection needs to consider maximum speed and torque, and testing may be required and be expensive.

Another challenge would be to make sure the resulting shots are realistic. From experience using table tennis robots, motorized ball launchers can generate unrealistic trajectories or excessive spin that won't closely resemble shots from a real player, and make the purpose of the robot less useful.

EECS 473 Homework 0 - Project Idea Proposal

Achal Khatri, akkik@umich.edu

Project Idea One: Sleep Cycle Based Alarm Clock

Traditional alarm clocks do not take into account the user's sleeping patterns and what part of the sleep cycle the user is currently on. When users are awoken in the middle of a sleep cycle, they have a harder time waking up and feel less alert and tired as they start their day.
To address this issue, I propose a product which collects data on the user's sleeping patterns to detect what part of the sleep cycle they are on and, after enough cycles have passed and they are at the end of a cycle, an alarm clock will sound to wake them up.

Details:
There are two significant components of this product: a watch-like wearable device which collects information regarding how the user is sleeping and the actual alarm system.
The wearable device will contain an accelerometer, temperature and pulse sensors. These sensors will provide metrics to calculate whether or not the user is asleep and if they are asleep, what part of the sleep cycle they are on. The data will be wirelessly transmitted to the alarm system device.
The alarm system device will receive information from the wearable. If the user is meant to wake up, as they have reached the end of their sleep cycle, the alarm will gradually increase in sound to gently wake them up.
Buttons for a "snooze" mode and for turning the alarm off will be found on the wearable device, as well as on the physical alarm clock. The wearable is the primary device for user operation so the state of the mode of the wearable will be communicated to the alarm system.

Figures:

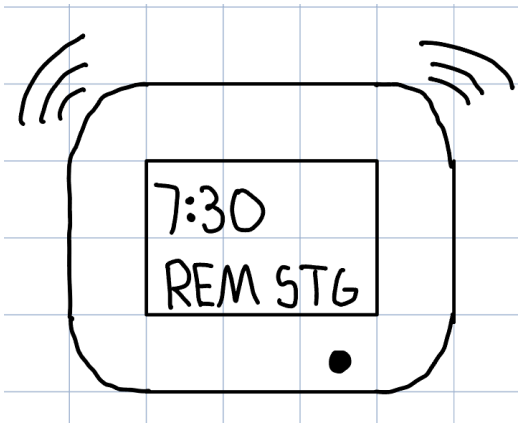


Figure 1: Physical Alarm Clock
Showcases a screen which displays the time and current sleep stage. Button on the bottom right is used to toggle between different modes of operation.

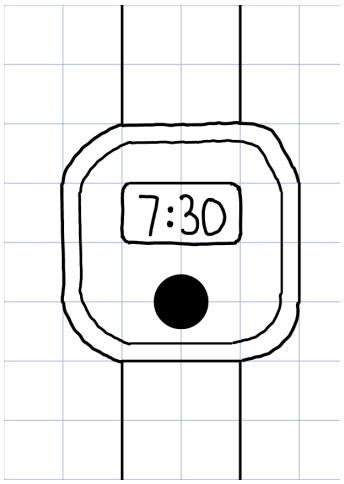


Figure 2: Wearable Device
Strap which is to be worn on the wrist that holds the device that reads user metrics. Displays time And has a button for toggling between modes of operation.

Schedule:

System Architecture and General Planning	Every single component and feature of the product is mapped out. The interaction between the two halves of the system is the main focus, so figuring out all the peripherals and communication between the two devices and planning this is the main priority.	9/14/2026 - 9/28/2026
Finalize Components and Order	With an architecture diagram and general understanding, we will have received and finalized all necessary components for the product.	9/28/2026 - 10/5/2026

Prototype and PCB Design Development	Initially using breadboards and prototype boards, we will interact with all the different sensors, screens, and ensure the connection between the two devices is proper. Parallel to developing a prototype, we will have finalized and sent out our PCB design for manufacturing.	10/5/2026 - 10/28/2026
Prototype Testing	Formal testing and alterations of the product will be completed using research we conduct and information we gain from speaking with different labs and organizations on campus. Another PCB with an updated design will be ordered if necessary.	10/28/2026 - 11/26/2026
Final Implementation	Final testing, documentation, and demonstration will be completed.	11/26/2026 - 12/11/2026

Budget:

Sensors and Peripherals	$200
Wireless Communication Modules	$50
PCB Development	$100
Microcontrollers	$100
Physical Housing	$75

I estimate that the costs for microcontrollers and the PCB to be the main causes of a price increase as these require the most research. The prices above assume we are creating two or more products to showcase.

Issues:

1. Testing: To accurately test our product, we need to be sleeping to see how the user experience is. This is an obvious problem and issue to work around. With connections from labs on campus, we may potentially be able to use devices which accurately measure and determine sleep status, our personal wearable devices, and more metrics to test if the data we are collecting and our product functionality is accurate and what we are looking for.
2. Demonstration: Similar to above, our product is truly tested whenever the user is sleeping. To showcase that our product works as intended, we will need to find a creative way to showcase everything working, other than just showing the health metrics collected.
3. Comfort: At this point, we are unsure what both components of the product look like and how they will work, but it is a given that the device which collects health metrics will need to be worn by the user. We will need to ensure that our final prototype is small enough to not obstruct the user.

Project Idea Two: Auto-Adjusting Solar Panel

Fixed solar panels are placed at a specific angle and cannot be moved upon installation, which means their efficiency will drop at certain times as the sun moves across the horizon. Ideally, a solar panel could be moved to maximize the amount of solar energy they harvest. To address this idea, I propose a system which uses control system logic to automatically adjust the angle the solar panel is positioned to effectively harvest solar energy.

Details:

This system contains a solar panel placed onto a motorized platform which can be manipulated on a single axis by tilting the motor one way, or in the opposite direction. The direction the platform is tilted is based directly on the power consumption of the solar panel, which acts as the feedback for the control system. The product will examine the power consumption at the starting angle, take the reading again at a slightly adjusted angle, and continuously iterate until an ideal angle is found.

The solar panel will charge the battery which powers the microcontroller, motors, and any other components of the mechanism, which allows the system to operate independently without the need of an external power source.

An interface with details regarding the state of the solar panels is also included in the robot-like product.

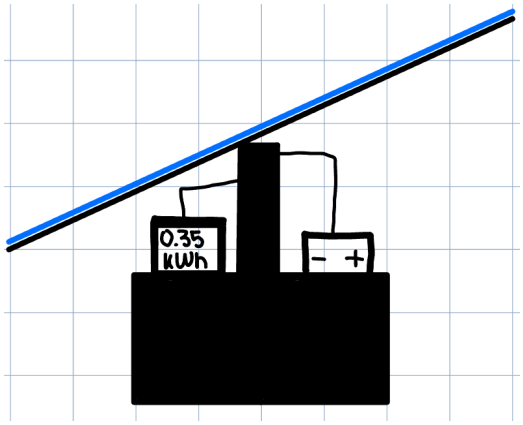


Figure 1: Front-view of the platform. Panel facing upwards at an angle which powers the battery.
Metrics regarding power consumption shown on screen.

Schedule:

System Architecture and General Planning	Every single component and feature of the product is mapped out. The mechanism regarding how to charge the battery using solar energy and power regulation is researched and optimized.	9/14/2026 - 9/28/2026
Finalize Components and Order	With an architecture diagram and general understanding, we will have received and finalized all necessary components for the product. The circuitry required for the solar panels is the bulk of the research.	9/28/2026 - 10/5/2026
Prototype and PCB Design Development	Initially using breadboards and prototype boards, we will construct a robot which operates on a single axis while designing and ordering the PCB.	10/5/2026 - 10/28/2026
Algorithm Development and Testing	The algorithm is adjusted and optimized while finalizing the electrical details regarding power optimization and storage.	10/28/2026 - 11/26/2026
Final Implementation	Final testing, documentation, and demonstration will be completed.	11/26/2026 - 12/11/2026

Budget:

Solar Panels	$80
Batteries	$50
Various Electrical Components	$100
Microcontroller	$40

Mechanical Components	$100
PCB	$70

Much of this is still unknown until additional research is completed. I estimate that the cost of all the components regarding harvesting solar power and storing that power will be more expensive than initially thought.

Issues:

1. Mechanical Power Consumption: As the product is operated independently and does not need an external power source, the motors and all other components of the robot will need to be highly optimized to ensure they draw the least amount of power possible. If they are using the majority of the power the solar panels harvest, the use case of the product is negligible.
2. Mechanical Complexity: Developing a robot which automatically drives motors to control the angle a heavy piece of equipment is located is a significant mechanical task. Initially, this idea was a robot which tilts the panel on an axis, as well as using light sensors to seek out shaded and nonshaded areas and to drive the robot to the sunny spots, but the idea was altered based on the mechanical complexity. Optimizing the motors for torque will be important.
3. Testing: Since we are using solar panels, ideally everything will be tested using the sun and natural light. As we are entering fall in the Midwest and it will be difficult to routinely work outdoors with all the equipment we need, testing our product in natural conditions will be difficult. If the solar panel we use harvests energy using growth lights, that is a method to continue testing after the sun sets.

HW0: Project Pre-Proposal

Main Project: RingMove Before Flight

High-level description

Context

Remove before flight items are commonly found in the aerospace industry, they allow mechanical protection, locking or disabling of some critical parts of an aircraft. For example the Boeing 787 has 5 landing-gear downlock pins (preventing the landing gear to be raised) that are set during maintenance and parking and are removed before takeoff to ensure the plane can retract its landing gears.

These items are tagged with long red streamers with the inscription “Remove Before Flight” (RBF) which eases the visual detection of the RBF items, and their removal by ground personnel before takeoff. However, in a few cases, some tagged items were not removed due to human factors. In some cases, it was unclear who was supposed to remove the items and check the removal. In other cases, the visual detection of the streamer was unsuccessful, because of dirty, grim colors or because of psychological bias (more difficult to see what we expect to be missing). Anyway, these examples demonstrate a failure point in an industry that is obsessed with safety.

While these examples didn’t lead to casualties, they reveal failures in the RBF item detection and removal, and are a call for a complementary layer of safety.

Design

The project idea is to add a layer of safety and reliability to this process and avoid further incidents, by adding sound detection. Indeed, a sound indication is easier to detect for ground personnel and can be tracked even when out of sight. The idea is to create a new design of RBF tags that can be triggered by a remote controller and ring loudly. This way, the ground personnel can use the remote controller to “hear” and locate any forgotten RBF items or verify that none were forgotten.

The product is composed of three systems:

- **5 enhanced Remove Before Flight tags**: They replace the traditional streamers and can produce a sound 100dB ringing if triggered.
- **1 storage/isolation/charging box**: A Faraday cage where the already removed tags will be placed to avoid their triggering.
- **1 triggering remote**: It sends a radiowave signal that triggers all exposed RBF tags.

Let’s take an example. After all tags have been supposedly removed from the aircraft and placed in the isolation box, the triggering remote is activated. It sends a radiowave signal that awakens any forgotten tag which enters the ringing mode (emitting a powerful ringing noise to warn the personnel). If no noise is detected, it means all RBF items have been removed. If a ringing is detected, the personnel can localize it easily and remove the RBF item.

The tags are composed of a radiowave antenna, to receive the signal from the remote controller, a buzzer to emit loud soundwaves, a processor to control the buzzer and the sleep mode, and a small battery to provide power.

The triggering remote is composed of a radio and an antenna to emit the triggering radiowave, a processor to control the radio, a button, and a battery to provide power.

Additional goals

If the time allows it, an additional small solar panel can be added to the tags, to allow for better autonomy, and the possibility to recharge the tags between each iteration, when placed into the isolation box (The box would embed a light system to charge the tags though their solar cells).

Implementation

Schedule

09/10 - 09/24	**Proposal Development** Reflexion on the design and possible alternatives. Definition of the goals, features of the system and testing methods. Research of buzzers that can provide sufficient noise. Research of small integrated antennas and radio that can be used for the tags and the remote. Preliminary study of the achievable autonomy and power requirements. Decide if implementing the solar cells and charging feature in the box. Role attribution.
9/25 - 10/9	**Hardware Design** Selection and ordering of the antennas, buzzer, battery (solar cell). Selection of the processors (same or different processors for the tags and the remote). Prototype on a breadboard, verify the connection and interface between battery, antenna, processor and buzzer. Testing of the communication between the antennas, verify the power performances of the communication. Selection of the material for the isolation box (what metal, what assembly, lights to recharge the tags?).
10/10 - 10/24	**PCB Design and Software** 2 people work on the PCB design for the embedded tags. They also measure the average power of the prototype to verify the expected autonomy and make any necessary change. 2 people work on the PCB design for the remote controler. 1 person works on the

	communication protocol and software. PCB order.
10/25 - 11/5	**Software Development and Box assembly** Waiting on the PCB. Development of the softwares for each PCB. Testing of the software on the breadboards. Design of a box for the remote controller (using 3D-printing). Assembly of the isolation box. Design of the attachment system for the tags' PCB. Second design of PCBs if needed.
11/6 - 11/20	**Software update for the PCBs, hardware assembly, debug** Final assembly of the systems, debug of the code. Verify individually that everything works nominally (radiowave communication, buzzer activation, battery). Verification of the isolation box. Measurement of the average power consumption of the tag system.
11/21 - 12/3	**System-level testing** Extensive test campaign of the combination of the 3 systems together in real condition (see Testing and Validation). Gathering of data and analysis of performances (autonomy, reliability). Development of the poster and final report.
12/3 - 12/10	**Final report and expo training**

Budget

Description	Unit price	Nb of Unit	Total price
Buzzer, >90dB	$3	5	$15
Battery (Tags)	$6	5	$30
Reception Antennas (Tags)	$10	5	$50
PCB (Tags) + components	20$	5	$100
Solar cells	($10)	(5)	($50)
Antenna (Remote Controller)	$10	1	$10
Radio (Remote Controller)	$10	1	$10
Battery (Remote Controller)	$20	1	$20
PCB (Remote Controller) + components	20$	5	$100
Copper sheets (Isolation Box)	$20	1	$20
Light System (Isolation Box)	$10	1	$10
Total			$415

Implementation issues

- The autonomy of the device is too short (they need to be changed often -> unreliable)
- The radio is not powerful enough to trigger all the tags in the complex spatial geometry.
- The final system is too complex: 3 different systems need to be designed, the tags, the remote controller and the isolation box (the latter can be reduced to a custom metal box without any electronics involved)
- The isolation cage doesn't work

Testing and Validation

Validation of the warning system of the tags

A few tags will be secretly set on a small aircraft (available in the FXB building hall) by one person. A second person will trigger the remote 30 feet away, and if they can hear any tags, they will search for them. The success is reached if all the tags are always triggered and localised by the operator, over multiple attempts (>10).
This test validates that the system meets its goal (allow the triggering and localisation of forgotten tags on a complex aircraft). This test campaign will prove the radio waves are always received by the tags despite the complex geometry of the aircraft, and that when the signal is received, each tag is automatically triggered and produces a noise sufficient to warn the operator and allow for a quick localisation of the tag.

Validation of the autonomy

The second important test is the autonomy of the device. Indeed, it is necessary that the tags have sufficient autonomy to prevent any failure (a tag was forgotten but it didn't ring because the battery was empty). Since the desired autonomy is much larger than the period of time for this project, it will be impossible to verify the autonomy experimentally. Therefore simulations will be run to demonstrate the theoretical autonomy of the device. The simulations will be complemented with extensive power measurement on the tag, and calculation of the mean power consumption to deduce the expected autonomy.

Backup Project: Battery Fire Warning System

From 2019 to 2023, fires started by battery-powered mobility devices (e-bikes, scooters, and hoverboards) caused $518.6 million in damage just in New York City. Indeed, the usage of E-bikes and electrical powered mobility devices is surging, and while the risk of battery fire is extremely small, the increase in the number of batteries in cities, transformed battery fires into a significant issue. Moreover, the consequences of battery fire are severe since battery fires produce toxic smoke, can't be stopped, and can propagate to furniture and ignite the room they are stored in. These fires are particularly severe because these batteries are usually stored inside houses and apartments, which increase the risk of spreading the fire. Moreover, they are charged during night (when the mobility device is not used), which reduces the response time of the owner.

The proposal is to design a detection system for thermal runaway (when batteries catch fire) and a warning system for surrounding people and 911.

The solution will be combined with a battery storage box (BatSafe), that will prevent the fire from spreading (the storage box will be bought since it is a mechanical challenge), but the sensor, warning system, and 911 transmission will be designed, implemented and tested.

While the system doesn't prevent the fire from happening, it allows early warning of the people close to the battery fire. Through a possible 911 hotline, it also allows for a quicker response of the firefighters that can secure the fire before it spreads to the surroundings and becomes a much larger issue.

One advantage of this project is that since batteries are stored inside houses, we can rely on wifi connection.

The system will be composed of the following:

- Protection box (bought)
- Battery fire detection system: thermal sensor and gas sensor put inside the box (designed and tested)
- Warning system: buzzer, alarm on the user phone (designed and tested)
- 911 warning (if relevant)

One of the main issues of this project is that the testing of the system will necessitate a battery fire, which can be hazardous, and unsafe. Although it is possible to mitigate many of the safety concerns, it may be difficult to prove the safety of the testing procedure.

The second issue is economic viability. The expected cost of the full system (battery fire proof box + detection/warning system) is of the order of $200. It may be difficult to convince people to buy this device because they may never experience a battery fire (low probability event, but severe consequences). However, the manufacturer may be interested in protecting its clients, to

avoid bad press. They may be the potential buyer. Alternatively, we could omit the casing to focus entirely on the detection system, at an estimated low cost of around $50, this approach is far easier to justify to the consumer.

Aimbot Controller

“Anti-Cheater Beater”

Anti-Cheat in modern videogames is getting better and better as game creators continue to silence cheaters and their valuable expression of skill in their craft. Modern anti-cheat systems like Vanguard, EasyAntiCheat, and VAC now scan kernel memory and process behavior to catch software based cheats, making traditional aimbots easy to flag since they leave a software fingerprint. 11.8 Million players have been banned by VAC alone for this. A purely external system that never touches the game's memory or files bypasses this. If a device watches the screen like a human and moves a physical controller like a human, there's no software signature to detect.

Details

The system has three parts: a camera facing the monitor, a microcontroller running a lightweight object detection model, and an actuation stage that physically drives the controller's sticks and triggers. The camera captures screen frames, which a small ML model processes to output target bounding boxes. The microcontroller converts the offset from screen center into an XY movement vector to aim at the target.

That vector drives small servos or actuators that physically nudge the controller's stick module and a solenoid that depresses the trigger on lock. There will be 2 actuators on each stick to coordinate XY movement along both axes with pushing and pulling of the sticks. Solenoids will be mounted onto specific mapped buttons/triggers such as “reload”, “aim”, and others. More complexity can be added according to the full rubric and along revisions and determining what buttons if any the user would still like to have access to. Since the computer only ever sees a real, unmodified controller reporting physical inputs, it's indistinguishable from a human.

Schedule

The dates in this section are fluid and almost assuredly will change later as the project progresses.

9/10-9/23	**Planning** Research camera and microcontroller options, evaluate ML models for real time object detection, and settle on an approach for physically actuating the controller
9/24-10/6	**Design and part selection** Order the camera, microcontroller, and actuator hardware. Select and begin training or fine tuning a detection model on sample game footage.

	Begin schematic design for the PCB.
10/7-10/25	**Prototype integration** Get a breadboard version working: camera feeding frames into the detection model, model output driving a test servo/solenoid on a controller. Finalize PCB layout based on what the breadboard proves out. Order PCBs by the 26th.
10/26-11/10	**Software development and testing** Refine detection accuracy and latency, tune the correction logic that converts target position into stick movement, and test end to end response time. Get PCBs assembled by 11/8. Address mounting/enclosure design for the actuator rig.
11/11-11/25	**Full system testing** Test the complete system in live gameplay, measure detection reliability and reaction speed, and iterate on model or mechanical adjustments as needed.
11/26-12/9	**Final testing, poster, and expo** Targeting project completion 12/3. Fix remaining bugs, finalize the poster, and start the final report.
12/10-12/11	**Finish final report**

Budget

While a realistic budget is difficult at this time, the numbers below are a reasonable first pass. If costs run over, using a lower-resolution camera or fewer actuators (just sticks, no trigger solenoid) is a way to cut costs. PCB manufacturing costs might be underestimated. Only one full rig will be built, with a couple of spare actuators and microcontrollers on hand in case of failure, since ordering a few extra is nearly the same price as one.

What	Cost total
Camera	$50-70
Microcontroller	$15-25
Servos/actuators (4x)	$40

Solenoids (5x)	$50
PCB manufacturing (5x)	$50-80
Misc. (mounts, wiring, connectors)	$20
Shipping	$50
Possible Extra hardware (controller, monitor, etc)	$0-$500
Total (est.)	**~$280-835**

Issues

The biggest concerns at this time are

1. Achieving low enough detection-to-actuation latency for the system to feel responsive
2. Training or tuning a model accurate enough to avoid false-positive targets
3. Physically mounting actuators to precisely nudge the stick without damaging the controller
4. Generalizing detection across different game visual styles

Issue 1 is the core technical risk but seems achievable with a lightweight model and fast microcontroller. Issue 2 will likely require iterative tuning once real gameplay footage is available. Issue 3 is mostly a mechanical design problem solvable with careful prototyping, but it will likely be difficult for EECS majors. Issue 4 is being treated as a stretch goal. The initial system will target one game only. Each issue seems manageable, and it is almost certainly something unforeseen at this time that will be the real problem.

Additional stretch goal

If time allows, we will explore extending detection to multiple games, or adding another camera angle to improve tracking accuracy when targets move quickly.

https://vac-ban.com/ban-history

Bluetooth Cheese

Manual cheese grating remains one of the last great inefficiencies of the modern kitchen, forcing users to expend valuable calories they could otherwise save for eating cheese. This project proposes a wireless, button activated cheese grating system: a Bluetooth-enabled remote lets a user press a button from across the room, triggering a conveyor belt that retrieves a block of cheese from the fridge and delivers it into a motorized grater without the user ever having to lift more than a finger.

Details

The system has three parts: a Bluetooth remote, a microcontroller driven conveyor inside the fridge, and a grating mechanism at the endpoint. The remote can be placed anywhere around the house to ensure your cheese can be grated at any time. Pressing the remote's button sends a signal to the microcontroller, which activates the conveyor to move a pre-loaded cheese block out of the fridge and along a track to the grater. A motor-driven grating wheel then processes the cheese into a waiting bowl.

Schedule

The dates in this section are fluid and almost assuredly will change later as the project progresses.

9/10-9/23	Select microcontroller, Bluetooth module, and motor for grating mechanism.
9/24-10/6	Design conveyor track and grater housing, and order parts.
10/7-10/25	Prototype breadboard. Get button press reliably triggering conveyor motion.
10/26-11/10	Integrate grating motor. Test full button to grated cheese pipeline.
11/11-11/25	Refine reliability (cheese jams, grate speed). Do stretch goals if time allows.
11/26-12/9	Final testing, poster, and expo prep.
12/10-12/11	**Finish final report**

Budget

While a realistic budget is difficult at this time, the numbers below are a reasonable first pass. If costs run over, using a lower-resolution camera or fewer actuators (just sticks, no trigger solenoid) is a way to cut costs. PCB manufacturing costs might be underestimated. Only one full rig will be built, with a couple of spare actuators and microcontrollers on hand in case of failure, since ordering a few extra is nearly the same price as one.

What	Cost total
Bluetooth remote	**$15**
Microcontroller	**$15**
Conveyor motor and track	**$25**
Grating motor and wheel	**$20**
Conveyor PCB	**$80**
Grating motor PCB	**$80**
Misc	**$15**
Total (est.)	**~$250**

Issues

The biggest concerns at this time are

1. Preventing cheese from jamming the conveyor

2. Keeping the grater from clogging mid-use.

Both are manageable with careful mechanical design, though as always, something unforeseen will likely be the real problem.

Projected Keyboard

Desktop space is often limited as many things need to be kept there. Papers, books, monitor(s), and stationary, among other things. Keyboards are another one of these objects with some particular downsides:

- They must be kept near the monitor or directly in front of the user
- Removing them while not in use is often inconvenient because they are in use frequently
- Things cannot be placed on top of them
- There exists great variation between keyboard layouts (e.g. number pad, escape location), so something that may be familiar to one user may be difficult for another to use

However, not all of these issues are unavoidable. By reducing the footprint and slightly modifying the utility, some of these issues are eliminated and the others are alleviated. To this end, I present the projected keyboard.

Function

The projected keyboard would consist of a box with its footprint being less than that of most keyboards. It would be fairly sturdy and have a flat top, so things could be placed on it without interfering with its functioning. On one side, the front, there would be some number of projecting devices and sensors. On one of the other sides, there would be a number of switches (including a power switch) and potentially other physical controls. It is possible that the device could connect to computers wirelessly or via a cord, which would come out of one of the other sides.

The keyboard layout would be projected from the front of the device. A projector might be too big or too expensive for this purpose, so it would be useful to look into DLP engines and MEMS laser scanning, which would be more difficult to set up, but fit better within the constraints. Various sensors would help identify when keys are pressed and which keys. These could include fairly exact distance sensors or time-of-flight sensors.

There are a number of ways the keyboard layout could be handled. The device could ship with a small selection of default layouts with a physical switch to make it switch between them. Alternatively, the device could ship with a single default layout and other layouts could be arranged using an app on the computer to which the device is connected. This latter method has the benefit of allowing the user more customization and the ability to maintain more layouts at a single time while the former makes the device more plug-and-play.

There are a number of aesthetic choices that could be made to make the device more appealing to users. Some users like the sound keys make when pressed. The device could include a speaker that would play noises when key presses are detected. The exact noise

played could be changed via the app interface on the computer. If this choice is made, some though will need to go into whether these noises need their own volume control or they are connected to the computer's volume control. Either way, it may be useful to also have a physical mute switch on the device.

Schedule

The following are estimates and should be adjusted as the project progresses to ensure that each component gets enough time to be developed fully.

9/10 - 9/23 Planning
This is the period between group formation and project proposals. During this time, extensive research should be done into the method of projecting the keyboard onto the desktop. This will likely be the most expensive part of the project, so it would be ideal to get this done first, perhaps in parallel with the choosing of the microcontroller as it would need to be something that could effectively manage the projection. Once those are chosen, the other sensors could be looked into. This would also be a good time to look at what additional features (layout changing, keypress sounds) it would make sense to include.

9/24 - 10/15 Prototyping
This period, between the start of proposal meetings and the Milestone 1 due date, should be used to order parts and prototype. It is especially important to get the projection mechanism working during this time as it might require some innovation and almost certainly will require learning to use unfamiliar parts. Additional sensors and parts should be ordered during this time and their working should be verified

10/16 - 11/3 Integration and PCB Design
During this period, the parts should be assembled into subsystems and tested as groups with breadboard prototypes. Once all of the components are working together, a PCB should be designed and ordered.

11/4 - 11/10 Computer Interface Construction and Testing
During this time, while waiting for the PCB to arrive, design and implement a way for the device to communicate with the user's computer. This may involve developing a simple application to manage some of the settings. 3D print a case for the device.

11/11 - 11/30 Testing on PCB and Final Integration
Get everything working on the PCB during this time. Ensure that the computer interface still works.

12/1 - 12/9 Bug Fixing
Fix every bug that can be found and test extensively. It may make sense for the builders to use the device in place of their usual keyboard to find more bugs.

12/10 - 12/11 Final Report
Complete the final report.

Budget

This budget is very rough because there are many ways to do each part of the project and some parts may end up not being done, such as the sound effects. I will assume a 5 person group is working on this, so the budget is $1250. This budget includes some extra parts in case they are needed for any reason.

Part	Quantity	Cost	Total Cost
Microprocessor	1	$30	$30
PCB	5	$30	$150
Time-of-flight sensors	10	$5.5	$55
Projection Mechanism	1	$800	$800
Misc Parts	1	$100	$100
Shipping	1	$75	$75
Total			$1210

Issues

The single biggest implementation issue is getting the projection mechanism right. There are a number of ways to do this and prototyping would likely be needed to determine which is most doable. However, each method is fairly expensive and even obtaining the parts required for prototyping could be costly. This could be combatted with both extensive research and by looking for cheaper methods.

Another issue is getting the time-of-flight sensor to talk to each other and ensure the microcontroller is able to tell which keys are being pressed, even with changing layouts. This will require extensive testing to ensure it is working properly, but should be doable. I/O expanders may be required since there may be many such sensors.

A third issue is getting the device to properly interface with computers. However, keyboards are very common and there is a wealth of knowledge on how to make this work as intended.

Holochess

Holochess, or Dejarik, is a two player game in the *Star Wars* universe. This project would be creating a hologram projector (there are known, relatively simple ways to do this) to project the board and pieces. Sensors would be used in order to detect and interpret player movements, which the microcontroller would convert into moves. There is no official ruleset for Dejarik, so it may be preferable to implement regular chess instead. The primary value of this would be entertainment, with a bit of a novelty factor. It may particularly appeal to fans of *Star Wars*.

Schedule

9/10 - 9/23 Planning
This period should be used for picking out pieces and learning how to project holograms. Existing methods of producing holograms are very limited in some ways, including the angles from which they can be viewed and the degree of detail that can be produced. These should be evaluated thoroughly and a decision should be made whether to aim to produce Dejarik or Chess.

9/24 - 10/15 Prototyping
Each individual subsystem should be prototyped and evaluated.

10/16 - 11/3 Integration and PCB Design
Integrate the subsystems and see that they are working. Then, design and order the PCB.

11/4 - 11/10 Build Enclosure
Build the enclosure for all the relevant parts while waiting for the PCB to arrive.

11/11 - 11/30 Assemble and Test
Place all required components on the PCB and ensure that it fits within the enclosure.

12/1 - 12/9 Final Preparations for Expo
Fix as many bugs as possible and ensure that the device is ready for the Design Expo. It would be useful to play full rounds of the game to find bugs.

12/10 - 12/11 Final Report
Complete the final report.

Issues

The most important issues arise from the user experience provided by existing hologram technology. It will be important to consider not just the feasibility of building the device during prototyping, but also whether the end product would actually be enjoyable to play.

Another issue would be ensuring that only movements intended as moves are registered as such by the device. This could be done with particularly exact sensing or by adding an undo button, so accidentally registered moves could easily be undone.

Off-Grid Real-Time Bus Arrival Signs

This project is a bus stop sign that shows live arrival times and needs no grid power and no wired data connection. It can be strapped to any existing pole and will show the next few arrivals for the routes serving that stop, updated once a minute. Real-time bus data already exists in Ann Arbor. AAATA publishes a developer data feed, operates a Track My Bus tool, and returns arrival estimates by text message.[1] The problem is that a rider standing at a stop with no phone cannot reach it.

Agencies do install real-time displays, but each one needs power and a data connection run out to the pole, which costs thousands of dollars per stop. So most stops have nothing, and the ones that do get a display are usually downtown transit centers.

Details

An e-ink display draws current only when the image changes and holds the last image at zero power, so a screen updating once a minute is essentially free to run. An alternative LCD backlight would drain a battery in a day. A small solar panel and battery sized for a Michigan winter will supply all the energy the node needs. A LoRa radio gives multi-mile range at very low power in exchange for a very low data rate, which works because an arrival prediction is typically a very small amount of data.

The signs themselves will not connect to the internet at all. Solar e-paper transit displays already exist commercially. For example: Papercast is the largest vendor, with deployments in more than 45 countries, and LA Metro selected them in 2023 after a 16-month pilot. But every Papercast display connects to their cloud system over 3G or Wi-Fi,[2] so every sign carries its own modem and its own monthly data subscription. Instead, we will use one gateway node placed somewhere that already has power and internet, such as a lab window or a rooftop. The gateway will then pull the agency feed, compute predictions for every stop it serves, and then broadcast small data packets over LoRa. Each sign in range would then wake up, receive the packet addressed to it, redraw, and then go back to sleep.

Each sign node will consist of a low-power microcontroller, a LoRa transceiver, an e-ink panel and its driver, a solar charge controller, and a battery. We will need to build this same board three times. The gateway node will use off-the-shelf hardware.

The power budget is the main thing we need to work out and it is the first task on the schedule. The e-ink panel only pulls current during a refresh lasting a couple of seconds and pulls nothing while holding an image, so a sign updating once a minute spends nearly all of its time asleep. The LoRa receive window is short and the microcontroller sleeps between them. We will build the real numbers from datasheets in the first two weeks and confirm them on the bench before picking a battery and panel size. We will use LiFePO4 rather than lithium polymer because it can be charged at low temperatures, which is where conventional lithium chemistry fails outdoors in winter.

Schedule

The dates below are approximate but should be close.

9/10 – 9/23	**Planning** Pick an e-ink panel, a LoRa module, and a microcontroller. Put together a rough power budget from datasheets. Draft proposal by 9/18.
9/24 – 10/6	**Order parts and get the pieces working** Get a display showing text and get two LoRa radios talking to each other. Final proposal

[1] https://www.theride.org/maps-schedules

[2] https://www.papercast.com/e-paper-display-applications/bus-stops/

	by 10/1, parts ordered by 10/4.
10/7 – 10/25	**Breadboard prototype and PCB** Get the whole path working on a breadboard. The gateway pulls the feed and the sign wakes up and shows it. Measure real current draw. Design the board and order PCBs by 10/26 to cover the 10-day lead time.
10/26 – 11/10	**Software and enclosure while boards are in fab** Finish the gateway and sign software. Build the enclosure and solar mount. Start the no-sun battery test.
11/11 – 11/25	**Get the PCB working** Bring up the board, move the software over, and build three signs. Start testing outside. Second PCB spin if we need one.
11/26 – 12/9	**Final testing, poster, expo** Aim to be done by 12/3. Collect results for the poster and finish the report.

Budget

This is a general estimate. The major line items come from current vendor listings. The 4.2 inch e-paper module is listed at $29.87 on LCSC[3] and between $18.99 and $28.99 direct from Waveshare. Bare RFM95W LoRa modules run a few dollars each, with the Adafruit breakout version at around $20.[4] JLCPCB currently advertises 4-layer 100x100mm boards starting at $7 for a small prototype run.[5] We budgeted PCB fabrication above that since the quoted price is for a small promotional board and does not include shipping. A team of five gives us $1,250. The estimate below builds three sign nodes and one gateway, with spares.

What	Quantity	Cost per	Cost total
E-ink display panels	4	$30.00	$120.00
LoRa transceiver modules	6	$10.00	$60.00
Low-power microcontroller	6	$5.00	$30.00
Solar panels	4	$15.00	$60.00
LiFePO4 cells and charge management	4	$12.00	$48.00
PCB fabrication (two spins)	1	$180.00	$180.00
Stencil and assembly supplies	1	$60.00	$60.00
Gateway hardware	1	$40.00	$40.00
Weatherproof enclosures and pole mounts	1	$120.00	$120.00
Antennas, connectors, misc	1	$80.00	$80.00
Shipping	1	$75.00	$75.00

[3] https://lcsc.com/product-detail/E-ink-Display_Waveshare-4-2inch-e-Paper-Module_C359944.html
[4] https://www.adafruit.com/product/3072
[5] https://jlcpcb.com/

Total			**$873.00**

Issues

The largest concerns at this time are

1. The winter power budget
2. Snow covering the solar panel
3. Charging a battery below freezing

Issues 1 and 2 are basically the same problem and they are our biggest risk. We think the power budget closes, but we have not measured anything yet, so we will confirm it on the bench in week 5 before committing to a battery and panel size on the board. Snow is the harder case, since a covered panel produces nothing no matter how it is rated. We will mount it steeply enough to shed snow and size the battery with enough reserve to ride out a stretch of bad weather. Issue 3 is the reason we picked LiFePO4 over lithium polymer, and we will also stop it from charging when it gets too cold. Beyond ordinary battery handling we do not see any safety issues, and we do not need access to any special facilities.

Evaluation and testing

In order to verify the device works, we will test how long a change in the feed takes to show up on the sign, how often a sign is showing stale data, how many days a node runs with no sun, and how far a sign can be from the gateway before packets drop. We will also sit at a stop and compare what the sign says against when the bus actually shows up.

Additional stretch goals

If the core system works early there are two more things we could add. The first is a button and a small speaker that reads the arrival times aloud, which would make the sign usable by blind riders. The second is having the signs report their own battery voltage and last-packet time back to the gateway, so an agency could see which signs are failing without visiting them.

Throwable Reconnaissance Sphere

This project is a throwable sensor sphere that lets first responders see inside a room before they enter it. When a fire crew arrives at a structure fire, or a tactical team stages outside a barricaded room, the only way they currently learn what is on the other side of the door is by opening it. Most injuries during interior operations happen in the first few minutes of entry, when the crew knows the least about the space.

The device is a ball about the size of a softball, built to survive being thrown through a doorway or window into a space that cannot be safely entered. On landing it survives the impact, figures out which way is up, lights the room, captures images in every direction, and sends them back through the building's walls to a receiver outside. The operator has images within a few seconds of the throw.

This product category already exists and gets bought. Bounce Imaging sells the Explorer to police departments for around $2,500 a unit, and ReconRobotics sells the Throwbot XT for around $10,000, both usually purchased with grant money. But the existing products have a documented weakness in the radio link. Bounce Imaging's own FAQ puts the range of their cameras at 40 to 60 feet in most structures over Wi-Fi, with anything beyond that requiring a 4G subscription, and they sell a separate range extender accessory to improve on it.[6] The Air Force went further and funded an SBIR specifically to add an alternative wireless technology to the Explorer, on the stated grounds that the system was designed for civilian users, communicates over 802.11 Wi-Fi and 4G cellular, and needs something more robust and longer range for environments where those are not available.[7] That link is what we want to work on.

Details

The device is somewhere a person cannot go, which means concrete, rebar, and structural steel sit between the sphere and the operator. Streaming video over 2.4 GHz Wi-Fi fails in that environment, since a poured concrete wall costs 10 to 20 dB and a reinforced floor can cost 20 to 30 dB. Moving to the 915 MHz band gets back about 8.5 dB and loses less signal through building materials, but it is slower. So we will give up live video to get through walls and design around still images instead. A single compressed image is small enough to get out over a slower link fast enough to be useful.

Four camera modules spaced around the sphere give full horizontal coverage no matter how it lands, and an IMU tells us which way is down so images can be rotated level. We will use sensors that do JPEG encoding on-chip, so the microcontroller only has to move the data and never has to compress an image itself.

Schedule and budget

Weeks 1 and 2 are for picking the radio and the cameras. Weeks 3 and 4 are for getting one camera and one radio link working on devkits. Weeks 5 through 7 are for putting them together, testing the link through real walls in the EECS building, and getting the PCB ordered by the 10-day deadline. Weeks 8 and 9 are for finishing the software and drop-testing the shell with a dummy mass while the boards are in fab. Weeks 10 through 12 are for getting the board working, putting it in the shell, and drop testing.

Budget is roughly $1,100 for two spheres and a base station. These are rough numbers: cameras ($96), microcontrollers ($72), sub-GHz and 2.4 GHz radios ($156), IMUs ($48), LEDs and drivers ($40), PCB fabrication and assembly ($260), batteries ($60), shell materials ($120), base station ($80), and shipping and miscellaneous ($175). The shell and potting materials are the least certain line, since we have not yet decided between printed TPU and cast silicone.

---

[6] https://bounceimaging.com/faq/

[7] https://www.sbir.gov/awards/180879

Issues

Getting the signal through walls is the central risk, since the device is useless if images cannot get out. We will measure this in real buildings before committing to a board layout, and we can fall back to a smaller image if we need to. Impact testing is expensive because every drop risks the hardware we need for the demo, so we will validate the shell with a dummy mass while the boards are in fab and build spares. Battery safety under impact is a real hazard, so we will use a hard-case cell mounted in the middle, away from the outside of the ball. Finally, throwing a hard object around at the Expo is not acceptable, so we will demo using a closed mock room that the sphere gets tossed into while images come up live on a monitor.

Evaluation and testing

We will measure how many packets get dropped and how long an image takes to come through, against the number of walls and floors between the sphere and the receiver. We want a usable image through two interior walls or one floor. Beyond that we will run repeated drops onto concrete from a range of heights and report the survival rate at each, score what fraction of throws produce at least one level and in-focus image, and measure the time from throw to first image on screen.

Adaptive At Home Physical Therapy System

High Level Description

Physical therapy after an ankle or knee injury changes significantly throughout the recovery process. A patient may start with simple movements to regain range of motion, then move to resistance exercises, partial weight bearing, balance exercises, and eventually strengthening. A physical therapist can closely monitor these exercises during an appointment, but patients typically only attend in person physical therapy a few times per week, meaning much of their rehabilitation and prescribed exercise actually happens at home without direct supervision. During these at home sessions, patients do not have the same equipment, measurements, or supervision that they would have with their physical therapist.

This project proposes an adaptive at home physical therapy system for ankle and knee rehabilitation. Rather than creating a machine that only performs one exercise, the goal is to build a device that remains useful throughout multiple stages of recovery. A physical therapist would prescribe exercises and their requirements through an application, including the exercise, number of repetitions, allowed range of motion, resistance, weight bearing target, and rest periods. The device would sync with these instructions and guide the patient through the appropriate exercises.

The main benefit of this system is that it would bring some aspects of the measurement and supervision available during an in person physical therapy appointment into the patient's home to supplement between in person sessions. The device could track progress in areas such as range of motion, strength, endurance, stability, weight bearing, and heart rate while also monitoring for potentially unsafe situations. As the patient recovers, the same device could adapt from basic non weight bearing exercises to resistance, standing, and balance exercises. The goal is not to replace a physical therapist, but to give patients a safer and more measurable way to complete their prescribed therapy at home while giving therapists useful information about what happens between appointments.

Implementation Details

The physical system would consist of a foot platform, wearable sensors, a controllable resistance mechanism, and a custom PCB containing the main processor and interfaces for the sensors, motor, safety hardware, and wireless communication.IMUs placed on the foot, shin, or thigh would measure joint angle, range of motion, movement speed, and shakiness. Load cells in the platform would measure weight bearing and balance, while a load cell and encoder in the resistance mechanism would measure force and movement. A heart rate sensor could also track exertion and recovery. Together, these sensors would allow the system to measure range of motion, strength, endurance, stability, weight bearing, and heart rate throughout a therapy session.

The resistance mechanism would likely use a motor driven cable or similar system to provide light, controlled resistance. The mechanical design will be one of the larger challenges because the system needs to safely interact with an injured leg. Maximum force and range of motion would therefore be

limited in firmware, and loss of communication or a system fault would automatically disable the motor. A physical HELP button would directly disable powered resistance, while the sensors could also detect possible falls, loss of balance, or movement outside therapist defined limits.

The companion application would allow a physical therapist to configure exercises with parameters such as repetitions, range of motion, resistance, weight bearing, and rest periods. Once these instructions are transferred, the embedded system would perform the monitoring and safety functions locally. The application is intentionally a smaller part of the project so that the main technical focus remains on the embedded hardware, sensing, and control.

Schedule

Time	Goals
9/10-9/23	Finalize the ankle and knee exercises the system will support and determine the measurements needed for each stage of rehabilitation. Research and select candidate IMUs, load cells, motors, encoders, and other major components. Have the overall system design and initial component choices completed by September 21.
9/24-10/6	Test sensors using development boards and build an initial prototype of the foot platform and resistance mechanism. Determine whether the selected sensors can accurately measure range of motion, force, weight bearing, and shakiness. Begin developing basic sensor and motor control firmware. Have the major sensors and mechanical approach validated by October 5.
10/7-10/25	Refine the mechanical design based on prototype testing and finalize component choices. Develop the custom PCB schematic and layout around the selected processor and sensors. Continue firmware development in parallel. Send the PCB for manufacturing by October 16.
10/26-11/10	While the PCB is being manufactured, continue building the mechanical system and develop exercise tracking, motor control, safety limits, HELP button behavior, and basic app communication using development hardware. Have the mechanical system and major firmware functions working by October 30.
11/11-11/25	Assemble and bring up the custom PCB and integrate it with the sensors, resistance mechanism, and safety hardware. Add endurance, breaks, shakiness, balance, weight bearing, and fall detection. Have the complete hardware system integrated and running the main exercises by November 20.
11/26-12/11	Test and calibrate range of motion, force, weight bearing, repetition detection, and safety features. Fix remaining hardware and software issues, complete the physical packaging, and test a full multi stage rehabilitation session. Have the final system ready for the Design Expo by December 10. Finish Final Report by December 11.

Budget

We would likely use an STM32WB55CGU6 as the main processor because it provides Bluetooth Low Energy and the interfaces needed for the sensors and motor control at around $10 per processor. The motor and mechanical system are the largest cost uncertainty, and the extra PCBs and processors are included as spares. If other costs come up, the number of sensors or complexity of the resistance

mechanism could be reduced. With a team of five and a total budget of about $1,250, this estimate of $940 leaves roughly $300 for unexpected costs or a PCB revision.

Part	Quantity	Cost per	Cost total
STM32WB55CGU6	5	$10	$50
Custom PCB	5	$20	$100
IMUs	4	$15	$60
Load cells and amplifiers	5	$20	$100
Motor, driver, and encoder	1	$175	$175
Heart rate sensor	1	$30	$30
Platform and mechanical parts	1	$175	$175
Safety and power components	1	$75	$75
Misc. parts	1	$100	$100
Shipping	1	$75	$75
Total			$940

Issues

The largest concern is safety because the device will apply resistance to a recovering leg. The motor must have strict force and range of motion limits, and the HELP button must be able to immediately disable resistance. The mechanical design should also release the patient if power or communication is lost.

Another concern is keeping the project manageable within the semester. The resistance system, motion and force measurements, and basic safety features will be prioritized first. Heart rate monitoring, wireless wearable sensors, and more advanced fall detection could be simplified or treated as stretch goals if development takes longer than expected.

Wireless Continuous Pulse Oximeter

High Level Description

Hospital patients are often connected to bedside monitors that track vital signs such as heart rate and blood oxygen. When a patient gets out of bed to walk, use the bathroom, or leave the room, the pulse oximeter must be disconnected and then reconnected when they return. This creates gaps in monitoring and adds another step for patients and hospital staff.

This project proposes a wireless pulse oximeter that stays on the patient and continuously measures heart rate and blood oxygen. While the patient is in their room, data streams over Bluetooth to the bedside system. When they leave, the device stores measurements locally and automatically uploads them when they return. An IMU also tracks whether the patient is resting, standing, walking, or experiencing a possible fall.

The system combines heart rate and IMU data into a joint activity metric by comparing changes in heart rate with the amount of movement detected by the IMU. For example, a high heart rate during very little movement is flagged differently than a high heart rate during walking, giving hospital staff more context than either measurement alone. Bluetooth receivers throughout the hospital also provide approximate room level location to help identify patients who wander or may need assistance.

Implementation Details

The wearable would use an optical pulse oximeter sensor, an IMU, and a low power Bluetooth microcontroller on a custom PCB. The PCB would also include local memory and battery management. When Bluetooth is available, measurements would stream directly to a receiver. When the connection is lost, heart rate, blood oxygen, and IMU data would be timestamped and stored until the device automatically reconnects.

Additional Bluetooth receivers placed in different rooms or hallways could estimate the patient's location based on which receiver detects the strongest signal from the wearable . The firmware would also combine IMU activity measurements with heart rate data to create a simple exertion or activity score and detect unusual combinations of movement and heart rate.

Issues

The largest challenge would be maintaining accurate pulse oximeter measurements while a patient is walking because movement can interfere with optical measurements. Bluetooth signal strength can also be affected by walls, equipment, and people, so the location system would only be expected to provide approximate room or hallway level positioning. Battery life would also need to be considered since the device is intended to remain on the patient for long periods, including how and when it could be recharged without interrupting monitoring.

Live-Cell Imaging System

Project Overview

Live-cell imaging is the ability to watch cells grow and respond to treatments over hours or days rather than looking at a single fixed snapshot and is a great tool for developing drugs against diseases. But commercial live-cell imaging systems are often incredibly expensive, let alone the software licensing, it usually costs around $200-$500k for current live cell imaging systems (Sartorius SX3, Leica Mica; The Sartorius SX3 standard configuration starts at about $200k). The real cost comes with the associated digital license that allows you to use these machines. It is essentially the “ink” that they are selling so you can use their “printers”. This quickly becomes out of reach for many research labs and institutions that don’t have access to such resources.

This project addresses that gap with a compact, low-cost live-cell imaging module designed to sit inside a standard tissue culture incubator and continuously image live cells over time, while being affordable, compact, and most importantly open source.

Details

This project is a compact, low-cost imaging system that mounts a camera and motorized stage inside a conventional CO2 incubator to capture time-lapse images (probably basic fluorescence) of live cell cultures. It automates the manual routine a researcher would otherwise perform every few hours, which is: opening the incubator, repositioning under a microscope, and imaging different wells or fields of view.

By handling this automatically, the system keeps cells undisturbed in their normal growth environment rather than exposing them to repeated temperature, humidity, and CO2 fluctuations from opening the incubator door. The finished device is a small enclosure sized to fit inside a standard incubator alongside normal culture vessels, built around a low-cost camera module, a compact motorized stage, and simple programmable LED illumination, all controlled by a small onboard computer that handles capture timing, stage movement, and image storage or streaming which avoids the bulk and expense of a full motorized microscope frame or scientific camera. (There could be other methods of taking a photo, but it is necessary to have a camera, lens, and lighting).

The target users are labs that need long-term imaging but can't afford or justify a commercial system's cost, academic labs running exploratory work, teaching labs that want students observing live cells rather than fixed slides, or other resource-limited

groups. Because the design is meant to be cheap and compact enough to build and fit multiple units, it also lets a lab run several experiments in parallel.

This project's contribution is combining genuinely accessible, low-cost commercial components with real multi-position imaging making continuous live-cell imaging viable for labs that currently have no good option between an expensive system and no continuous imaging at all.

Implementation Plan

1. **Component selection and bench setup.** Choose the camera module, single-board computer, stage actuation approach (small stepper motors vs. simpler linear actuators), and LED illumination parts.
2. **Breadboard imaging and motion control.** Get the camera capturing usable images of a test sample on the bench, and get stepper motors moving a simple stage prototype using a breadboarded driver circuit. Design and order the PCB. Begin 3D-printing the stage and enclosure parts in parallel, since that doesn't depend on the PCB arriving.
3. **PCB arrival, assembly, and bring-up.** Populate and solder the board, port over the already-debugged motor/LED control firmware, and resolve board-specific issues (power regulation, connector fit, motor driver behavior).
4. **Mechanical integration and focus/positioning testing.** Assemble the camera, stage, and illumination into the enclosure, and test repeatable multi-position imaging using a fixed target.
5. **Incubator environmental testing and live imaging trial.** Test the assembled unit inside an actual incubator for thermal/humidity tolerance of the electronics, then run a real time-lapse imaging session on a live cell culture.

Budget

We only expect to build only one live-cell imaging module. Expected budget is seen below. This could change depending on how we intend to position the image captures.

What	Quantity	Cost per	Cost total
Basler dart (camera)	1	$300	$300
AmScope A10X-V300 (lens)	1	$30	$30
28BYJ-48 stepper + ULN2003 driver	3	$5	$15
Custom PCB	1	$20	$20

LED	1	$15	$15
3D Printed Case	1	$20	$20
Misc Parts	1	$50	$50
Shipping	1	$30	$30
Total			$480

Implementation issues

- **The main mechanical risk is repeatable, drift-free positioning at low cost.** Cheap stepper motors and printed parts can introduce backlash or flex that throws off focus or field-of-view alignment between imaging cycles. This is addressable because it's testable early and cheaply on the bench well before live cells are ever involved, so we can catch and correct mechanical issues. Adding buffers by reducing how much area we accept in a single capture could also help reduce this risk.
- **We may need an incubator.** To truly test this system and its design, we would want to test within an incubator, this can be sorted out with some staff in the biomedical department. However, testing acquisition can be done without actual cells such as with table salt or microbeads. We could also test with cells that can survive room temperature. (No cells that we would use would ever be dangerous)
- **Thermal/humidity tolerance of electronics inside the incubator is a concern**, since standard incubator conditions (37°C, high humidity, CO2) are outside the comfort zone of some off-the-shelf electronics. This is addressable with basic coating or a sealed enclosure for the electronics, and should be tested.
- **LED lighting** needs more work. Standard fluorescent captures are a bit more complicated than just taking an LED and putting it near the cell cultures. However, I believe that we could sort this out before it becomes an issue.

We don't see a fundamentally unaddressable issue at this stage; the repeatable-positioning and incubator-environment questions are the two areas with real uncertainty, but both could be resolved after discussion.

Evaluation / Testing

We have two levels of success in this project.

1. Can we acquire images? Does the system reliably capture multiple focused images and repeat this on schedule over an extended period without crashing, losing position, or drifting out of focus. Using a fixed target (such as salt) instead of live cells could demonstrate success in this project.

2. Does it actually deliver useful live-cell imaging data? This is the real test of what was described earlier, and our measure of success is a direct comparison against a manual imaging baseline: image the same live cell culture both with our system and with periodic manual imaging on a standard lab microscope, then compare. Specifically:

- **Positioning consistency test**: confirm the stage returns to the same field of view accurately across many cycles, using a fixed target (such as a scratch)
- **Live-culture time-lapse test**: run an actual multi-hour or multi-day imaging session on a real cell culture inside the incubator, and check whether the resulting time-lapse actually shows expected cell behavior (growth, division, movement) clearly enough to be useful/meaningful

We'll consider the project successful if it can produce a clean, in-focus time-lapse of live cell growth over an extended period without manual intervention, at a fraction of the cost and size of a commercial system. It would be even better, if we could get a biologist to use our device and decide if the scans produced are well defined and meaningful.

Air Quality UAV Module

Problem

Concern over air pollution has been steadily increasing throughout the years. The method of tracking and monitoring air pollution has been steadily improving however they tend to neglect rural or impoverished areas. Ground-based air quality stations are expensive, fixed in place, and require stable power and network connectivity to report data, which makes them a poor fit for rural regions, disaster zones, or areas with unreliable grid infrastructure.

This project addresses that gap with a lightweight, UAV-mounted air quality module designed to bring real-time pollution monitoring to places that fixed infrastructure can't reach.

Details

This project is a compact, lightweight sensor payload that mounts on a fixed-wing UAV to measure key air pollutants, such as particulate matter (PM2.5/PM10) and target gases like CO, NO2, or VOCs, in flight.

A small onboard microcontroller reads the sensors, timestamps and geotags each reading using the UAV's GPS, logs it locally, and can optionally stream live data over the aircraft's existing telemetry link. The finished module is a small, self-contained pod, as small as possible, designed to run off the UAV's own power system and add minimal weight or drag.

The target users are organizations already doing environmental and public health work in resource-constrained settings. Because it's low-cost and low-weight, it's usable by local technicians and small survey teams rather than requiring specialized aircraft. A lightweight module that works at speed opens up air quality mapping to smaller, cheaper, longer-range fixed-wing UAVs, giving places that have no good options a great option to measure air quality.

Implementation Plan

1. **Component selection and bench setup.** Choose specific sensor parts (PM sensor, gas sensor(s), microcontroller, GPS module),
2. **Breadboard sensor validation and firmware.** Get each sensor talking to the microcontroller individually on the breadboard version, write and debug the core firmware (sampling loop, GPS timestamping, local logging to SD/flash), and start

3D-printing/prototyping the passive inlet geometry. Order the PCB once we have stuff working.

3. **PCB arrival, assembly, and bring-up.** Solder and populate the board, port the already-debugged firmware over, and resolve any board-specific issues (power regulation, connector fit, etc.).
4. **Ground-based airflow testing.** Test the passive sampling inlet and sensor response using a bench fan or a car-mounted test rig to simulate airspeed before ever attaching it to a UAV.
5. **Flight integration and test flights.** Mount the completed module on a UAV and run initial test flights.

Budget

- Microcontroller + GPS + basic passives: ~$25-40 (a low power microcontroller with built in GPS could keep this cheap.).
- PM sensor (Plantower/Sensirion SPS30-class): ~$25-40.
- Gas sensor (electrochemical or MOx sensor for CO or VOCs): ~$15-30.
- PCB fabrication: ~$20-40
- 3D-printed enclosure and battery: ~$20-30.
- Contingency: ~$30-50.

Total: $135-$240

Other implementation issues

Mechanical/aerodynamic risk is the main open question, specifically whether the passive airfoil-pressure sampling inlet actually produces a usable, steady flow. We could show in theory if its doable fairly early on, but it may take a few weeks before we could definitely say that we are able to collect particulates in a meaningful way.

Evaluation and Real-World Testing

We need to be able to test if we measure air quality accurately at flight speed. I believe this could be difficult to prove how “accurate” our system is. I think our best bet is to compare it to another known system and see how it matches up. The researchers at CLASP have many of these devices so it might be a good idea to contact them and ask if we could test our system side by side or perhaps ask how they test their air quality systems. A rudimentary method is to compare our systems with national monitoring systems, but we would want our system to be more accurate. If our system can accurately measure air quality to standards that other larger modules do, then we have succeeded.

Robotic ACL Reconstruction Drill-Guide Demonstrator

1 in 3500 people tear their ACL every year[1] and this number is becoming more common by the day among athletes. Reconstructive surgery is often required to restore knee stability and enable them to return to their sport. One of the most critical steps in ACL reconstruction is creating bone tunnels in the femur and tibia through which a replacement graft is positioned. Errors in the tunnel placement or drilling depth can alter graft alignment and knee mechanics and are an important technical cause of ACL reconstruction failure[2].

My proposed project is a desktop robotic ACL reconstruction demonstrator designed to improve the precision and safety of this drilling process. The system would combine embedded motion tracking, motorized drill-guide positioning, and real-time force and depth sensing to automatically maintain the desired angular alignment of a predefined bone-tunnel trajectory as the model knee is flexed. During drilling, the system would continuously monitor resistance and depth, automatically disabling the drill if an unsafe condition is detected.

At the Design Expo, people could manually flex or reposition the model knee through a constrained range of motion. The system would measure the change in joint geometry and automatically reposition the drill guide. The user could then initiate drilling into replaceable foam or synthetic bone blocks while the system displays orientation, drilling force, and depth. Intentionally creating an unsafe drilling condition would demonstrate the automatic safety cutoff.

Details

The prototype would consist of a 3D-printed or synthetic knee model containing movable femur and tibia sections, a two-axis motorized drill-guide mechanism, and an embedded control system. Motion sensors attached to the femur and tibia would estimate the relative orientation of the two bones as the knee is repositioned. A microcontroller would use this information to determine the required drill-guide orientation and command two stepper motors to rotate the guide toward a predefined tunnel trajectory. To keep the prototype manually feasible, the knee would be mounted with fixed constraints in translational movement and drill entry point established manually and the robotic mechanism controlling angular orientation mainly.

The drill guide would be actuated using two stepper motors and motor drivers, allowing the system to control its orientation along two rotational axes. Before operation, limit switches would be used to home the mechanism and establish a known reference position. The knee-mounted motion sensors would also be calibrated with the model placed in a known starting position. During normal operation, the controller would continuously compare the measured knee orientation with the desired drilling trajectory and update the position of the drill guide accordingly.

A small electric drill would then be positioned through the robotic guide and used to drill into styrofoam or synthetic bone material. During drilling, a force sensor would measure drilling resistance while a separate position or distance sensor would estimate drilling depth. A sudden drop in resistance could indicate that the drill has broken through the material, while excessive measured depth could indicate that the drill has traveled farther than intended. Either condition would cause the controller to disable the power to the drill. A physical emergency-stop button and software fault detection would provide additional safety in the event of a sensor failure or unexpected system behavior and could be implemented via interrupts.

---

[1] https://www.panoramaortho.com/resource/acl-injuries-by-the-numbers/
[2] https://pubmed.ncbi.nlm.nih.gov/32691095/

The embedded electronics would ultimately be integrated onto a custom PCB containing the microcontroller, sensor interfaces, motor-control connections, power circuitry, and drill-control circuitry. Initial development would be performed using development boards and sensor modules before moving the completed design onto the custom PCB. This approach would allow the sensing, motion control, and safety systems to be tested independently before final integration.

One of the main implementation challenges will be achieving repeatable mechanical alignment using relatively cheap motors and 3D-printed components. IMU drift and sensor noise may also introduce error into the measured knee orientation. These problems would be addressed through calibration, filtering, mechanical constraints, and repeated testing. The project is intended only as a desktop demonstrator and would operate exclusively on synthetic materials rather than humans or animals.

Schedule

The dates are an approximate timeline which is highly likely to change a lot.

Sept 10-23	Clinical Research and Planning In the first two weeks after groups are formed, we will study the basic anatomy of the ACL and the major steps involved in ACL reconstruction, focusing especially on how femoral and tibial bone tunnels are created and aligned. From this, we will identify the main functions our prototype needs to perform and define rough requirements for drill-guide movement, sensing, accuracy, and safety. We will then meet as a group to decide on the overall system architecture.
Sept 24-Oct 13	Design and Part Selection We will research and select a suitable microcontroller, IMUs, force and depth sensors, stepper motors, motor drivers, drill motor, and power electronics. At the same time, we will begin designing the mechanical structure for the model knee and two-axis drill guide. By the end of this phase, the major components should be selected and ordered, and we should have a clear understanding of the processor I/O and power requirements.
Oct 14-27	Actuator and Sensor Prototyping Using development boards and breadboard prototypes, we will independently test the major subsystems. This will include reading and calibrating the IMUs, controlling the stepper motors, homing the mechanism with limit switches, measuring force and drill depth, and safely switching the drill motor. We will also begin building the mechanical prototype and designing the custom PCB. The PCB should be ordered near the end of this period.
Oct 28-Nov 10	Software development and testing on integrated prototype While waiting for the PCB to arrive, we will continue developing and testing the software on the development-board prototype. The main focus will be automated guide positioning, force/depth monitoring, sensor timeout handling, and safe drill shutdown.
Nov 11-25	PCB Integration and Testing

	Once the PCB arrives, it will be assembled and tested before migrating the firmware and sensors onto the final hardware. The complete electrical and mechanical system will then be tested together.
Nov 26-Dec 9	Final Integration and Design Proposal Submission The completed system will be tested using synthetic bone or foam material. We will evaluate drill-guide alignment accuracy, depth measurement, breakthrough detection, emergency-stop response, and behavior during sensor faults. Any remaining software, electrical, or mechanical issues will be fixed, and the final prototype and Design Expo demonstration will be prepared.

Budget

At this stage, I feel that the exact parts are still subject to change so this is a very rough estimate. I think that with 5 members the project should be comfortably within the course budget. THe main costs will likely come from the mechanical set up, 3D printed parts, and the motors.

Item	Estimated Cost
Microcontroller and development hardware	$30
Motion and depth/force sensors	$60
Motors and drill hardware	$100
Power electronics and safety components	$40
Custom PCB and components	$60
3D printing and mechanical parts	$60
Synthetic bone / testing material	$30
Spare parts in case of damage	$30
Estimated Total	**$410**

With a group of five members, this would come to about $80 a person which is well within the per-student budget with enough room for change if more/different parts are needed later on.

Implementation issues

The main implementation issues that come to mind are:

1. Mechanical accuracy and rigidity with the given degrees of freedom
2. Reliable knee-position tracking
3. Detecting unsafe drilling conditions (potentially incorporating CV with a pi-cam)
4. Rapid drill shutdown while maintaining overall safety
5. Integrating and testing all the mechanical, electrical, and software components

Issue 1 may be difficult because backlash or flex in the mechanical structure could reduce the accuracy of the drill guide. Issue 2 will depend heavily on the IMUs ability to be calibrated and how much drift is observed during operation. Issue 3 is likely to require the most experimentation, since we might need to incorporate computer vision and an additional raspberry pi microcontroller if the sensor approach is not enough. Issue 4 should be manageable by designing the drill to default to an off state and including both software and hardware shutdown mechanisms through interrupts. Issue 5 will likely require a lot of testing, since each subsystem needs to work correctly independently before we can combine them all together and test again. Overall, each issue seems manageable as long as we distribute tasks well and stay ahead of timeline.

Additional Stretch Goals

If we manage to complete the project ahead of time I would love to include some computer vision aspect to the drill because I am also taking EECS 442 and would love to use the theoretical skills I learn from that class into practice in this project.

Autonomous Smart Cones to Measure Athletic Performance

For my EECS 373 final project, my group built a pair of autonomous “smart cones” intended to make athletic training drills more interactive and faster to set up. The idea was to take the normal cones used for agility drills and give them the ability to move and detect when an athlete reached them to provide visual feedback and record basic performance metrics. We were able to get a working prototype together for the final Expo, but the system still had a number of limitations, especially with localization, movement accuracy, and the fact that we only had two cones. The april tag detection wasn’t great for the lighting of the EECS atrium and I think it would be interesting to revisit the idea with a larger group of cones and redesign some of the parts that gave us trouble the first time.

The new version would consist of roughly six motorized smart cones that could automatically arrange themselves into different training layouts. Each cone would have an embedded controller, motors and wheel encoders for movement, sensors for detecting athletes or physical contact, and LEDs or a buzzer for immediate feedback. The cones would communicate wirelessly with a central controller or user interface, where an athlete could choose a drill and view their results.

Having six cones instead of two would make the system much more useful because it could support more realistic agility drills without requiring someone to manually rearrange the field between every exercise. For example, the system could support suicide/sprint drills, figure-8 drills, reaction drills, shuttle runs, and eventually custom layouts selected by the user.

Details

One of the main improvements I would want to make over our original project is the positioning system. In the first version, getting the cones to consistently move to an exact location was one of the more difficult parts of the project. Wheel encoders could still be used for short-term motion control, but accumulated wheel slip makes them less reliable over longer movements. A redesigned system could combine encoder measurements with some form of external position reference, such as an overhead camera tracking AprilTags mounted on each cone. The camera could determine each cone's actual position while the embedded controller on the cone handles the low-level motor control.

Each cone would use two motorized wheels to move around the training area and would include simple sensors for detecting when an athlete reaches or hits it. An accelerometer could detect taps or knock-overs, while ultrasonic or other proximity sensors could determine when the athlete passes close to the cone. LEDs and a buzzer would provide immediate feedback during the drill.

The cones would communicate wirelessly so that they can coordinate movement and report sensor events to a central controller. Once a drill is selected, the system would assign a target position to each cone, move them into formation, and begin timing once the athlete starts. During the drill, sensor events from the cones could be timestamped to calculate split times, overall completion time, missed checkpoints, or collisions.

Some possible drill modes would include:

- **Suicide/Beep Test Drill:** cones automatically move outward to different distances while recording the athlete's time between checkpoints.
- **Figure-8 Drill:** two or more cones form a course and record completion time and collisions while the athlete moves around them.

- **Reaction Drill:** several cones light up in a random sequence and the athlete has to reach or tap the correct cone as quickly as possible. Red cones get pressed and turn green monitoring the time taken to touch the cone.
- **Dynamic Course Drill:** the cones rearrange themselves between rounds so that the athlete does not know the next course beforehand.

The reaction and dynamic-course modes would be particularly interesting with six cones because they were not really practical with our original two-cone prototype.

Schedule

The dates are an approximate timeline which is highly likely to change a lot.

Sept 10-23	Clinical Research and Planning Review the previous EECS 373 design and identify the main causes of the performance issues we experienced at the Expo. Fix the issue of detecting the april tag in dark spaces. Decide on the localization approach, wireless architecture, sensors, and mechanical changes needed to support a larger number of cones.
Sept 24-Oct 13	Individual Cone Prototype Build one improved cone and get the motor control, encoder feedback, sensors, LEDs, and wireless communication working reliably. Test the new localization approach before attempting to scale the system.
Oct 14-27	PCB and multi-cone integration Design the custom controller PCB and begin building additional cones. Implement communication between multiple cones and develop software for assigning target positions and preventing cones from interfering with each other while moving.
Oct 28-Nov 10	Software development and testing Implement several drill modes and test automatic cone positioning, timing, athlete detection, and feedback. Expand testing from two cones to the full six-cone system.
Nov 11-Dec 9	Final Testing and Expo Preparation Once the PCB arrives, it will be assembled and tested before migrating the firmware and sensors onto the final hardware. The complete electrical and mechanical system will then be tested together.
Dec 10-11	Final Design Submission

Budget

Since many of the parts are relatively inexpensive and repeated across each cone, the main cost would come from building six complete mechanical units.

Item	Estimated Cost

Microcontroller and PCBs	$120
Motors, Drivers and Encoders	$180
Sensors	$100
Wireless Communication Hardware	$60
Batteries and Power Electronics	$100
3D printed cone bodies and chassis	$150
Camera/localization hardware	$60
Spare parts in case of damage	$75
Estimated Total	**$845**

With a five-person group, this would be around $170 per person and should still fall within the course budget. The number of cones could also be reduced if the mechanical costs end up being higher than expected.

Implementation Issues

The main issues I would expect are:

1. Getting six cones to position themselves accurately and repeatedly without colliding
2. Preventing wheel slip and accumulated localization error
3. Coordinating several moving cones without collisions
4. Maintaining reliable wireless communication between all of the units
5. Building six mechanical systems without spending too much time reproducing the same hardware
6. Reliably detecting athlete interactions during fast-moving drills

The positioning problem was already difficult with the original two-cone system, so I expect that to remain the biggest challenge having to redevelop the RTR logic at a larger scale. Using an external camera as an additional source of position information may make this much easier than relying entirely on wheel encoders. Scaling from two to six cones will also make communication and movement coordination more complicated. If building all six cones takes too much time, the system could initially be demonstrated with three or four while keeping the software capable of supporting additional units.

Additional Stretch Goal

If the six-cone system is working reliably, a stretch goal would be allowing users to create their own drill layouts rather than selecting only from predefined drills. Another possibility would be automatically changing the difficulty of a drill based on the athlete's previous performance using some custom machine learning models.

App-Independent Legacy Hearing Aid Bridge

Modern hearing aids increasingly rely on smartphones, proprietary applications, and continued manufacturer software support. The consequences of this dependence are particularly visible in 2026. LXE Hearing, the company behind Lexie and Eargo hearing aids, is winding down its U.S. operations. The Lexie app will be removed from app stores and lose cloud connectivity on October 8, 2026, followed by the Eargo app on October 22. Customer and technical support will also end on October 22. Although the hearing aids are expected to retain basic amplification and locally stored settings, the apps will no longer receive updates and future compatibility with phone operating systems cannot be guaranteed.[1]

The proposed device would not attempt to recreate proprietary fitting or manufacturer-specific software functions; instead, it would address the broader longevity problem by providing a modern audio interface whose essential operation does not itself depend on continued application or cloud support. Software support is not the only compatibility problem. Bluetooth connectivity in hearing aids has historically relied on proprietary implementations, limiting interoperability between devices and manufacturers. Newer Bluetooth LE Audio and Auracast standards are intended to improve this interoperability, but older hearing aids may not contain the hardware necessary to support them.[2] Many older hearing aids do, however, contain telecoils capable of receiving audio through an inductive neckloop.[3]

The proposed system would provide a small, standalone bridge between modern audio sources and telecoil-equipped hearing aids without introducing another dependence on a manufacturer application or cloud service. The device would receive audio through Bluetooth LE/Auracast, a microphone, or a wired audio input, perform local audio processing, and transmit the resulting signal through an inductive neckloop. Physical controls would provide volume, source selection, and audio-profile adjustment. The finished device would be a rechargeable wearable unit designed around the principle that the continued usefulness of functioning hearing hardware should not depend on continued support for a particular smartphone application or online service.

Details

The device would contain a microcontroller capable of wireless communication and basic digital audio processing, an audio input/output path, rechargeable battery, and physical controls. Audio could enter the device from a Bluetooth/Auracast source, 3.5 mm input, or onboard microphone. After optional equalization or filtering, the processed audio would drive an inductive neckloop that could be received by a hearing aid operating in telecoil mode. Neckloops are already used to transmit audio to telecoil-equipped hearing aids, making this output method technically feasible.[3]

1. https://lxehearing.com/
2. https://www.bluetooth.com/learn-about-bluetooth/feature-enhancements/le-audio/hearing/
3. https://www.hearingloss.org/find-help/hearing-assistive-technology/assistive-listening-systems/

All essential operations would occur locally. User settings such as volume, source, and several audio profiles could be stored in nonvolatile memory so that the bridge itself remains functional without an application or external server. The initial prototype would focus on reliable audio reception and telecoil transmission. More advanced noise reduction, directional microphones, or additional modern audio protocols could be added if time permits.

Schedule

9/10-9/23	**Planning** Research telecoil/neckloop operation, Bluetooth LE Audio/Auracast hardware, audio-processing requirements, and appropriate microcontroller families. Determine desired inputs, physical controls, and basic audio-processing functions. Establish initial requirements for audio quality, latency, battery life, and device size.
9/24-10/6	**Design and part selection** Select the wireless platform, processor, audio codec, battery system, microphone, and inductive output approach. Order development boards and components needed for the first prototype. Begin testing basic wired audio and inductive transmission before integrating wireless audio.
10/7-10/25	**Breadboard prototype, software development, and PCB design** Build a prototype capable of receiving an audio signal, performing basic digital processing, and transmitting it through a neckloop. Begin software for source selection, volume control, saved profiles, and wireless audio. Design and order the custom PCB.
10/26-11/10	**Software development and testing on prototype** While waiting for PCBs, continue development using the prototype. Complete basic Bluetooth/Auracast reception, audio routing, equalization, and physical controls. Measure audio latency and transmission quality. Begin enclosure design.
11/11-11/25	**Software development and testing on PCB-based system** Assemble and bring up the custom PCB. Port and test the existing software on the PCB-based device. Integrate battery operation, charging, controls, wireless audio, and inductive output. Fix hardware/software issues and determine whether a PCB revision is needed.
11/26-12/9	**Final testing, additional features, poster and expo** Test all supported input modes, battery life, latency, audio quality, and telecoil compatibility. Complete the wearable enclosure. If time permits, implement improved noise reduction, microphone-array processing, or additional audio profiles. Complete poster and Expo preparation.
12/10	**Finish final report**

Budget

With a team of five, the total available budget would be $1,250. The core system is expected to cost approximately $865, leaving roughly $385 for replacement components, PCB revisions, or unexpected costs. Multiple PCBs and development components would be purchased so that spares are available during testing. If costs increase, the number of completed wearable units could be reduced without affecting the ability to demonstrate the core system. Hardware for

advanced noise reduction or directional audio would only be purchased if sufficient budget remains after the core system is operational.

Item	Quantity	Cost per unit	Total cost
Bluetooth LE Audio/Auracast development hardware	2	$60.00	$120.00
Microcontroller/DSP development boards	3	$30.00	$90.00
Audio codec/interface components	1	$80.00	$80.00
PCB - main device	10	$12.00	$120.00
Batteries/charging and power components	5	$15.00	$75.00
Microphones, buttons, encoders, jacks, etc.	1	$80.00	$80.00
Enclosures/mounting hardware	5	$15.00	$75.00
Misc. electronic parts	1	$70.00	$70.00
Shipping	1	$50.00	$50.00
Directional microphone/noise reduction hardware (stretch goal)	1	$100.00	$100.00
Inductive neckloop/coil and driver components	5	$15	$75
Telecoil receiver for testing	1	$30	$30
Total			**$965.00**

Issues

1) Implementing Bluetooth LE Audio/Auracast reception
    a) Bluetooth LE Audio and Auracast require more complicated hardware and software support than conventional Bluetooth communication, but existing development platforms should make prototyping feasible.
2) Producing clear and sufficiently strong audio through the inductive neckloop
    a) Requires testing the neckloop driver to ensure that the transmitted signal is strong enough for a telecoil-equipped hearing aid without excessive distortion or power consumption.
3) Maintaining low audio latency while performing digital audio processing
    a) Manageable by beginning with simple filtering and equalization and only adding more computationally intensive noise reduction if time permits.
4) Achieving acceptable battery life and size for a wearable device
    a) Influences processor, radio, amplifier, and battery selection, but the relatively small number of components should make a compact rechargeable device feasible. More advanced audio processing can be treated as a stretch goal if it creates unacceptable latency or power requirements.

Stretch Goals

Stretch goal 1: add directional microphone processing to preferentially amplify speech coming from a selected direction.

Stretch goal 2: implement more advanced noise reduction and speech-enhancement algorithms.

Stretch goal 3: add a standalone hearing-profile setup process that allows personalized audio settings to be configured and stored locally without a smartphone application or cloud service.

Stretch goal 4: automatically detect and switch between available audio sources such as Bluetooth/Auracast, wired audio, and the onboard microphone.

Stretch goal 5: expand support to additional wireless audio standards or input devices.

Evaluation and Testing

The system will be considered successful if it can reliably receive audio from each supported input, process the audio locally, and transmit intelligible audio to a telecoil receiver without requiring a smartphone application or internet connection. Testing will measure end-to-end latency, audio quality, wireless reliability, battery life, and successful operation of the physical controls and locally stored settings.

Deployable Electronic Breadcrumb System for Emergency Egress

Indoor navigation remains a significant problem for firefighters and other emergency responders. GPS, while effective outdoors, is largely ineffective inside buildings, and public-safety organizations have identified reliable indoor localization as an important technology gap.[1] In 2026, NIST noted that first responders inside large buildings remain difficult to locate reliably, particularly when a responder becomes lost or fails to check in.[2] This creates a problem not only for the responder attempting to exit, but also for rescue personnel who may need to retrace that responder's path.

Electronic breadcrumb systems have been explored previously. For example, NIST demonstrated deployable wireless breadcrumb relays that could be placed by first responders as they moved through a structure.[3] However, that work primarily focused on maintaining radio communication between responders and an incident command post rather than using the deployed nodes as a navigation path.[3] A system specifically designed to preserve the route traveled through a building could provide a simpler alternative to determining an exact indoor position. Rather than answering "Where exactly is the responder?", the system would answer "Which direction along the traveled path leads toward the responder or back toward the exit?"

Details

The proposed system would consist of several small, battery-powered wireless "breadcrumb" nodes and a wearable or handheld responder unit. Breadcrumbs would be placed periodically as a responder moves through a building. Each node would communicate with nearby nodes and store information about its position in the sequence of the traveled route. The responder unit would communicate with the breadcrumb network and provide direction and distance information through a simple interface such as LEDs, a display, audio, or haptic feedback.

During egress, the device would guide the responder from the current breadcrumb toward the previous node and continue along the recorded path until reaching the entrance. The same network could also be followed in the opposite direction, allowing another responder to retrace the original path toward a missing or incapacitated responder. Because the route is established as the nodes are deployed, the core system would not require GPS, cellular service, existing Wi-Fi, or a preloaded map of the building. If time permits, additional sensing could be used to detect blocked sections of the original route and investigate rerouting through previously explored paths.

1. https://www.nist.gov/news-events/news/2026/05/z-axis-test-facility-improving-indoor-localization-public-safety
2. https://www.nist.gov/blogs/taking-measure/how-thousands-tiny-dots-can-save-first-responders-lives
3. https://www.nist.gov/news-events/news/2008/08/nist-debuts-new-approach-ad-hoc-networks-first-responders

Schedule

9/10-9/23	**Planning** Research possible wireless/ranging technologies such as UWB, BLE, or other radio platforms and compare range, accuracy, power consumption, cost, and ability to operate through walls. Determine the basic breadcrumb communication and routing scheme and define requirements for the responder unit. Identify likely microcontroller and sensor families.
9/24-10/6	**Design and part selection** Select the wireless/ranging technology, microcontroller, batteries, and user-interface components. Order development boards and enough radio modules to prototype a small breadcrumb network. Begin testing basic communication and ranging between two nodes. Finalize the expected architecture of the breadcrumb and responder units.
10/7-10/25	**Breadboard prototype, software development, and PCB design** Develop a multi-node prototype using development boards. Begin implementing the communication protocol, node association, ranging, route recording, and basic navigation software. Test whether several breadcrumb nodes can establish a route and guide a responder from one node to the next. In parallel, design the custom PCB based on lessons from the prototype and order PCBs by the end of this period.
10/26-11/10	**Software development and testing on prototype** While waiting for the PCBs, continue development of the full navigation system. Complete multi-node route recording and retracing, improve ranging and communication reliability, and implement the responder interface. Test scenarios such as node failure, branching paths, and a second responder following a previously deployed route. Have the core software largely functional and tested before transitioning to the PCB-based system.
11/11-11/25	**Software development and testing on PCB-based system** Assemble and test the PCB-based breadcrumb nodes and responder unit. Integrate ranging, routing, and user guidance. Conduct tests through hallways, rooms, walls, and multiple turns. Resolve PCB and software issues and evaluate whether a PCB revision is necessary. Begin stretch goals if time allows.
11/26-12/9	**Final testing, additional features, poster and expo** Conduct full-system testing in an unfamiliar indoor route and evaluate whether both the original responder and a second responder can successfully retrace the deployed path. Measure ranging accuracy, communication reliability, navigation success, and deployment time. Continue blocked-path detection, rerouting, or LiDAR/SLAM integration as stretch goals. Complete poster and Expo preparation.
12/10	**Finish final report**

Budget

With a team of five, the total available budget would be $1,250. The core system is expected to cost approximately $985, leaving roughly $265 for replacement parts, PCB revisions, or unexpected costs. If costs increase, the number of breadcrumb nodes could be reduced while still maintaining enough nodes to demonstrate route recording and retracing. Extra wireless modules and PCBs would primarily serve as spares. LiDAR hardware for SLAM-based

mapping would be treated as a stretch goal and would only be purchased if sufficient budget remains after the core system is completed.

Item	Quantity	Cost per unit	Total cost
UWB ranging/radio modules	10	$22	$220.00
Microcontrollers/dev boards	10	$10	$100.00
PCB breadcrumb	10	$12	$120.00
PCB responder unit	5	$15	$75.00
Batteries/charging components	10	$8	$80.00
Sensors/UI components	1	$90	$90.00
Enclosures/mounting hardware	1	$100	$100.00
Misc. electronic parts	1	$100	$100.00
Shipping	1	$100	$100.00
LiDAR (stretch goal)	1	$100	$100.00
Total			**$1,085.00**

Issues

1) Maintaining reliable wireless communication through walls and between floors
    a) Walls, floors, and building materials can reduce wireless range, but testing different radio technologies and maintaining sufficiently close spacing between breadcrumbs should make this manageable.
2) Achieving accurate ranging between breadcrumb nodes indoors
    a) Indoor multipath can reduce ranging accuracy; however, the system does not require precise absolute positioning, only enough accuracy to guide a responder toward the next node
3) Determining appropriate breadcrumb spacing and deployment
    a) Requires testing to determine how frequently breadcrumbs must be deployed while still keeping deployment practical.
4) Designing two different systems: the breadcrumb nodes and responder unit
    a) Increases the hardware and software workload, although much of the communication and processing hardware could potentially be shared between the two designs.
5) Designing a simple and reliable interface for low-visibility environments
    a) Manageable by keeping the interface simple and relying primarily on haptic, audio, or easily distinguishable visual feedback.
6) Integrating LiDAR/SLAM for blocked-path detection and rerouting

a) Adds significant hardware and software complexity and would therefore be treated as a stretch goal. Each of these issues appears manageable without preventing completion of the core breadcrumb navigation system.

Stretch Goals

Stretch goal 1: detect when a connection between breadcrumbs has become obstructed.
Stretch goal 2: use a breadcrumb graph to reroute through another previously traversed path.
Stretch goal 3: integrate LiDAR-based SLAM to construct a map of the explored environment and improve obstacle/route awareness.

Evaluation and Testing

The system will be tested along unfamiliar indoor routes containing multiple rooms, turns, walls, and stairs. Success will primarily be measured by whether both the original responder and a second responder can successfully retrace the deployed path without GPS, Wi-Fi, or a preloaded map. Additional measurements will include ranging accuracy, communication reliability, maximum breadcrumb spacing, deployment time, and battery life.

inverter/DAQ proposal

Custom Three-Phase Inverter and Data Acquisition System for UM Supermileage

University of Michigan Supermileage is a student-led project team which designs and races energy-efficient cars at the Shell Eco-Marathon. In the past, the team supported inverter projects with the eventual goal of running an urban concept vehicle in the battery electric division, which the team had never done before. The most recent attempt (an EECS 473 project) was ambitious and implemented many different features, but it was unable to control a BLDC motor at the speeds required to make the car move. The team had to fall back on an internal combustion engine for 2026. There were attempts made to debug and fix the problems we saw with the most recent iteration of the inverter project, but due to the complexity of the system and all of the previous project members having graduated, the team decided to start from scratch and redesign the entire system.

Therefore, the goal of this project is to design a simple AND functional inverter and data acquisition (DAQ) system that will meet our team's system requirements and achieve our goal of running a battery electric urban concept vehicle at the Shell Eco-Marathon in 2027.

Schedule

9/10 - 9/23	**planning** Research and understand power electronics and the inverter system we are designing Read through and learn from documentation for Magnolia and previous 473 inverter projects to avoid making the same mistakes Outline system architecture and requirements with list of non-negotiable and nice-to-have features from design meeting with UMSM electrical systems leads + chief engineer Begin researching and selecting parts (inverter MOSFETs, inverter MCU vs DAQ MCU, LoRa) based on system requirements
9/24 - 10/6	**design and part selection** Discuss feasibility of 3 boards (inverter/power supply, DAQ, comms) with our main goal of making the inverter drive a car in mind and if LoRa/comms is worth the time and resources spent. Also discuss if prototyping makes sense for the inverter baseboard, potentially skip in favor of additional PCB dev time Post-proposal meeting, order components for inverter (and comms?) and begin prototyping DAQ system with selected MCU.

10/7 - 10/25	**prototype development, PCB design** Upon arrival of components, begin prototyping inverter and PS (and comms?) Run prototypes off a power supply first before using our PS circuit prototype Make motor move/spin with prototype inverter Be able to talk to all DAQ components (LCD, SD card, CAN bus with GPS/IMU module, encoders) and complete integration Order PCBs by 10/26
10/16 - 11/10	**software development, prototype testing** During PCB lead time, individually test prototypes with PS prototype before running everything together Develop motor control software that uses current sensing and DAQ, ideally seeing functional trapezoidal commutation Verify connection between inverter sensors and DAQ MCU Speedrun PCB assembly upon arrival
11/11 - 11/25	**software development, PCB testing** Continue refining motor control software, if going well, can consider more precise methods of control Test assembled PCBs, verify CAN bus communication and LCD/windshield wiper operate correctly for DAQ Test inverter/PS PCB and debug/revise as needed, more testing or additional features will be considered at this time
11/26 - 12/9	**testing, nice to haves, final deliverables** Tentative completion date by 12/7, may move as scope and complexity of project is more clear. Expo poster should be completed and sent out for printing at least a day or two before expo, and begin to complete final report
12/10 - 12/11	**complete expo presentation, final report**

Budget

With approximately $1,500 spent on Magnolia's inverter system and $2,000 on the last 473 inverter project, we expect to budget all or most of the $1,250 provided for a 5 person team. Some expensive components and materials are already owned by the team (ex. ~$500 BLDC motor, majority of necessary DAQ sensors), so those expenses will not come out of the funding

provided by 473. Below is an estimated breakdown of costs, extrapolated and overestimated from old budget sheets. We are not sure if the comms board will be included in this project, but it is included in the breakdown below to provide a more comprehensive budget. The total contribution from 473 is likely aggressively underestimated as the cost of most things have gone up in the past few years, so this budget will be revised at a later date.

item	quantity	cost
inverter PCBs	5	$80
DAQ PCBs	5	$30
Comms PCBs	5	$30
inverter and power supply components	-	$200
Comms/LoRa components	-	$25
MCUs	2-3	$20-30
misc components (caps, diodes, connectors, etc.)	-	$30-40
shipping/taxes (2 rounds of PCBs)		~$200
	total:	$635

Challenges

The last two UMSM inverter projects absolutely indicate the scale and complexity of this project, which will likely be similar for this project. The main challenges include:

- Preventing scope creep and setting reasonable expectations in project design
- Designing two separate PCBs for inverter and DAQ
- Learning and applying power electronics knowledge
    - Designing a stable power supply for the inverter
- Full system integration
    - likely DAQ is powered by power supply on inverter board, and receives input from temperature sensors also located on inverter board

Most of these concerns are manageable with careful planning and review of previous inverter project documentation and the problems they encountered.

Tab 2

Plant Propagation/Grow Station

Many houseplants can be propagated into more plants by taking leaf or root cuttings from the original plant, or alternatively, by growing more plants from seed. These cuttings or seeds are often propagated either in water or soil, and depending on the plant, maintaining ideal environmental conditions such as humidity can be crucial for the successful propagation and growth of the new plant. This project involves creating an enclosure with several independent climate-controllable chambers. The humidity in each chamber can be controlled by the user, directly on the enclosure and wirelessly through an app and WiFi. Each chamber also has grow lights and fans which can also be controlled by the user. The enclosure display and app can help the user what conditions are best for the plant placed in the chamber. The problem this is attempting to solve is about making home gardening more accessible to people who may not have the environment or resources to do so outside year-round.

Schedule

1. design and part selection
    a. select humidity sensors, humidifiers, photosensor, grow lights, and fans/fan speed sensor
    b. research enclosure design, how to climate seal each enclosure
    c. research wifi connectivity and MCUs appropriate for the application
2. prototyping and PCB design
    a. design prototype enclosure (3d printed? acrylic sheets?)
    b. order components, and begin talking to sensors individually
    c. begin designing PCB and drafting app design/control ideas
3. software development and prototype testing
    a. continue developing app
    b. test humidity control in enclosure
    c. see sensor data over wifi communication to app
    d. assemble PCBs
4. software development and PCB testing
    a. see humidifier/light/fan control from app working
    b. fix any problems with operating enclosure climate control with PCB
5. complete final testing and deliverables
    a. complete debugging of final product, complete expo poster/final report

Budget

Key items: acrylic sheets ($30-50/pack of 10 sheets), enclosure seals (~$20-40?), PCBs (~$60-80), MCU ($20-40), sensors+misc components (~$120-150)

Challenges

Wifi connectivity and app development

Anti-Hardware Stalking Device

Background

In a 2024 study, it was found 44% of stalking victims were tracked via bluetooth trackers[1]. These trackers, such as Apple Airtags, Samsung Galaxy SmartTabs, Life360 Tiles, and multiple devices from Chipolo, are intended to help people locate anything that can be easily lost, like keys and wallets. They are compact, small, and hardly noticeable, which makes them easy to attach to anything you might lose. However, their size makes them perfect to slip into an unexpecting persons' car, bag, or coat as well. Once the device is hidden, that person can be tracked online, giving a potential stalker their location 24/7.

Apple and Google have partnered to deliver a unified method for tracking these unwanted devices through iOS and Android notifications and have partnered with bluetooth tracker manufacturers[2]. However with standard smartphone detection, the user is limited to how often the smartphone determines to scan for these devices. For example, if your iPhone has a low battery, scanning becomes more infrequent. Additionally, this "unified" method is not even comprehensive. Life360, the company that owns Tile, has not joined this agreement. This is not some small niche company not joining either. By 2021, 45 million Tiles had already been sold[3]. Instead, users must download Life360's app and manually initiate a 10-minute scan, where they have to walk around, to detect unwanted Tiles, and remember to periodically re-initiate scans[4].

Even if smartphones can provide a solution for malicious bluetooth trackers, smartphones either take scanning frequency out of the user's control, or force the user to constantly remember to do 10-minute long scans. This also ignores that those susceptible to stalking, like people running away from domestic violence, may not have their phone, may not be able to charge it, or may not have the funds to buy a new phone. Therefore, this project proposes a cheap, easily-distributable device that can continuously detect hardware stalking devices. For example, a shelter should be able to freely hand out these devices without much of a cost burden to them.

Details

The proposed device is small, capable of RF scanning to detect bluetooth trackers, battery-powered, focused on being robust and power efficient, and has a small OLED screen to show the user nearby trackers. When a scan is scheduled, the device will wake up and conduct a scan of nearby Bluetooth Low Energy Advertisements emitted by the trackers. It will process

---

[1] https://petsymposium.org/popets/2024/popets-2024-0082.pdf
[2] https://www.apple.com/newsroom/2024/05/apple-and-google-deliver-support-for-unwanted-tracking-alerts-in-ios-and-android/
[3] https://www.life360.com/blog/tile-joins-life360
[4] https://www.wired.com/story/tile-tracking-tags-can-be-exploited-by-tech-savvy-stalkers-researchers-say/

these advertisements into the MAC addresses found, the manufacturer data, the Universally Unique Identifier (UUID), flags, and any other device specific information, additionally, it will detect the signal strength (RSSI) between the bluetooth device and itself. It will then compare this data to previous advertisements and device information stored in flash memory. If it detects a device has been continuously nearby for multiple scans through an internal algorithm, it will covertly alert the user through the OLED screen.

Designed to be easily rechargeable, the device is powered through AAA batteries. To save battery, it has an automated duty-cycle, only waking every ~3 minutes to conduct a ~10 second scan, process the advertisements, and go back to sleep. Additionally, the device can have a button for the user to manually wake up the device and start a scan.

While the device must be able to exist by itself, without access to a computer or phone, it still must share information with the user in order to alarm them if a tracker is nearby. This however must be covert, so an audio cue cannot be used. Therefore, a small screen can exist on the device, displaying the tracker type (if known) and its signal strength. For example, the screen might show "Tile | -90dPm." If this is found to consume too much power, this can be replaced with an LED that glows red if a device is found.

Development Schedule

9/10 – 9/23	**Project Planning and Early Investigation** ● Select which Microprocessor/Bluetooth Low Energy SoC will be used ● Capture BLE Advertisements from major bluetooth tracker chips (Apple Airtag, Samsung SmartTag, Life360 Tile) with a dev kit and begin rough processing ● Finish Draft Proposal
9/24 – 10/20	**Hardware Design and Early Software** ● Implement raw BLE Advertisement parsing on the chosen chip/ evaluation board ● Finalize how the user will receive information from the device (LED vs OLED screen) ● Begin designing PCB with the chosen MCU, RF antenna, push button, LED/OLED screen, and necessary components for AAA battery housing. ● Submit Final Proposal ● Order all parts we expect to need
10/21 – 11/5	**Breadboard Proof of Concept and PCB Order** ● Implement algorithm to detect if a bluetooth tracker has been repeatedly detected ● Implement Human-Computer interaction with a button create an interrupt to the device and a method of alarming the user when a

	tracker device is nearby ● Verify full scenario with mock stalker scenarios on a breadboard prototype ● Finalize PCB layout, verify design rules, and order by October 25th
11/6 – 11/18	**Finalizing Software while PCB is fabricated** ● While waiting for the PCB, finalize the sleep/wake duty-cycle and test battery consumption ● Implement the button interrupt to force a wake-up and scan ● Verify flash-memory writing works between power cycles, determine amount of memory realistically needed, and determine when data can be deleted ● Once PCB arrives, assemble and verify boards ● Design the 3D-printed enclosure
11/19 – 11/25	**PCB and Full Scenario Testing** ● Write firmware onto the MCU on the PCB ● Verify full scenario test works, as well as testing for false-positives ● Print the 3D-printed prototype enclosure, allowing for multiple iterations to address any errors
11/26 – 12/9	**Final Testing and Expo Preparation** ● Test battery-life duration under standard operation and constant operation ● Print the final 3D-printed enclosure ● Complete project poster, report, and prepare for the live demo at the expo

Budget

What	**Quantity**	**Cost per**	**Total Cost**
Bare MCUs (ESP32-C3 / nRF5 / others) Multiples for investigation, research, and last minute pivoting	**10**	**~$8**	**~$80**
Custom PCB (4-layer)	**10**	**$8**	**$80**
Test Bluetooth Trackers (Airtag, Smarttag, Tile)	**3**	**$30**	**$90**

AAA Battery Holders	**6**	**$2**	**$12**
32 count AAA Battery Pack	**1**	**$20**	**$20**
1’ Monochrome OLED Screen	**4**	**$7**	**$28**
Enclosures	**5**	**$10**	**$50**
Misc Parts	**1**	**$100**	**$100**
Shipping	**1**	**$75**	**$75**
Total			**$535**

Issues

- **RF Antenna on a custom PCB:** This is our greatest challenge, without the ability to scan, we do not have a device. BLE operates on 2.4 GHz, which can be slightly difficult to work with. Because our device is based around safety and detection, having the highest possible scanning range is crucial. Therefore, creating or using the best antenna possible is not negotiable, and this will require special focus as PCBs must be ordered quickly.
- **Device Size:** We want this device to be easily portable, making it as compact as possible is ideal.
- **Battery Life:** The device should be able to operate for a long period of time as we cannot guarantee that the user will be able to constantly charge their device. AAA batteries can be relatively easy to come by, but we want to minimize how often the batteries have to be switched out. Therefore, there will have to be a major focus on a robust sleep mode and battery life.
- **Cost:** The device, ideally, is accessible to absolutely anyone, especially those in need. Rather than focusing on a very fancy device, we want something cheap and easily manufactured.

Stretch Goal

- On iOS and Android, in the pop up informing you there is a bluetooth tracker device following you, there is occasionally a button to force it to ring. A stretch goal may be to incorporate this onto our device, however, this requires a BLE handshake rather than just passive listening to advertisements.

Semi-Smart Bird Feeder

Background

Smart Bird Feeders have flooded the market. Ranging from $150-$200 dollars, they have 4K HDR[5] cameras which continuously monitor for birds and run AI models to detect the specific species, alerting the user on their phone when a bird is eating from their feeder. These same birdfeeders come with $10 monthly subscriptions for you to be able to continuously use their software, unless you choose to buy the Lifetime AI models that cost an extra $100 upfront[6].

This is far too expensive and intense for a typical bird-enjoyer who simply wants to see funny pictures and videos of birds. Additionally, most of these feeders do not even include anti-squirrel technology, a rampant problem in the bird-feeding community[7]. Therefore, this proposal focuses on a *Semi*-Smart bird feeder that can take fun pictures of birds and prevent squirrels from eating all of your food while not introducing yet another AI subscription model into the world.

Details

On the bird feeder, there is a perch directly in front of where the bird seed lays, the bird seed is not accessible elsewhere (i.e. it is not a 360 degree bird feeder). Under this perch there is a weight sensor. When a bird lands to eat, the sensor will detect it is under ~200 grams. This will trigger **Bird Mode**, waking up the camera and telling it to start taking pictures and/or recording. However, if a greedy squirrel comes up to the bird feeder to steal food, the sensor will detect it is over ~400 grams, especially if it is an Ann Arbor squirrel. This will activate **Squirrel Mode,** this will still wake up the camera as squirrel pictures are also funny (but not videos as this will waste storage), but will also turn on the motorized perch, spinning the squirrel and flinging it off the bird feeder. For even better photos and videos, the camera will have a fish-eye lens to allow the user to get a greater range of view.

The motorized perch is attached to the bottom. It consists of a brushless motor at the bottom acting as a pivot, with the perch “attached” to it, meaning it spins with the motor. After each use, it resets to be right in front of the bird seed opening. However, when **Squirrel Mode** is activated, it begins rapidly spinning to fling the squirrel off.

Rather than uploading our photos and videos to the cloud, we will simply store them locally to an SD card to minimize connections and power usage. This can then be taken by the user and downloaded onto their local devices.

---

[5] https://kiwibit.com/products/bird-feeder-2-mini-with-solar-panel
[6] https://kiwibit.com/products/bird-feeder-2-pro-with-solar-panel-lifetime-ai-early-bird
[7] https://www.youtube.com/watch?v=PCyGVMKgbYQ

The device is battery powered through a rechargeable LiPo battery. However, as the device is already stationed outside, a solar panel will be installed on the roof of the bird feeder to charge the battery passively during the day.

Development Schedule

9/10 – 9/23	**Project Planning** ● Select which MCU will be used ● Finalize hardware decision (i.e. type of motor, battery, solar panels, weight sensor) ● Order Camera ● Finish Draft Proposal
9/24 – 10/20	**Hardware Design and Physical Housing Design** ● Begin designing the PCB ● Begin designing the exterior housing and shape of the Bird Feeder ● Get camera software working ● Submit Final Proposal ● Order all parts we expect to need
10/21 – 11/5	**Breadboard Proof of Concept and PCB Order** ● Verify full breadboard prototype, with the camera taking pictures and recording when woken up by the weight sensor ● Finalize housing ● Finalize the perch layout, with the motor able to spin the perch ● Finalize PCB layout, verify design rules, and order by October 25th
11/6 – 11/18	**Finalizing Software while PCB is fabricated** ● While waiting for the PCB, finalize the sleep-mode and recording states ● Finalize all camera software, including writing to an external SD card. ● Once PCB arrives, assemble and verify boards ● Test solar power panel charging the LiPo battery ● Design the 3D-printed enclosure
11/19 – 11/25	**PCB and Full Scenario Testing** ● Write firmware onto the MCU on the PCB ● Verify full scenario test works ● Construct the final housing with all electrical parts contained
11/26 – 12/9	**Final Testing and Expo Preparation** ● Test battery-life duration under standard operation ● Complete project poster, report, and prepare for the live demo at the expo

Budget

What	Quantity	Cost per	Total Cost
MCUs	**5**	**$8**	**$40**
Custom PCB (4-layer)	**5**	**$10**	**$50**
Motors	**5**	**$7**	**$35**
Micro-load Cells	**5**	**$5**	**$25**
LiPo Battery	**2**	**$20**	**$40**
Solar Panels	**2**	**$9**	**$18**
Cameras	**3**	**$10**	**$30**
MicroSD Cards	**5**	**$15**	**$75**
Enclosure	**1**	**$45**	**$45**
Misc Parts	**1**	**$100**	**$100**
Shipping	**1**	**$75**	**$75**
Total			**$533**

Issues:

- **The housing has to be durable and weatherproof**
- **All electrical components must safely be hidden and secure:** We do not want any animals accidentally biting wires, getting electrocuted, etc. Safety from electrical components has to be top-of-mind.
- **The Perch Rotator:** This is the most mechanical component of the entire bird feeder. Being able to securely rotate a perch when a 800 gram squirrel is sitting on it may be difficult. Other methods may be better.

Not for Public/Classroom Distribution + Sponsorship is available

Project 1: All-in-1 Drone baseboard for AI GPS denied Inspection.



Control and Efficiency

Flight time is one of the main limitations of small unmanned aerial vehicles. Every additional circuit board, connector, cable, voltage regulator, and enclosure adds weight and consumes space. Separate modules also introduce electrical losses, complicate assembly, and create additional failure points. Together, these inefficiencies reduce useful payload capacity, shorten flight time, and make the aircraft harder to manufacture and maintain.

Oprones Inc. currently requires several electronic functions to operate its drone platform. These may include power distribution, voltage regulation, motor-controller interfaces, flight-control connections, sensor inputs, communication links, payload control, and system monitoring. Implementing these functions with separate commercial modules can produce a system that works, but it is not optimized for the aircraft with commercial parts.

We propose designing a custom all-in-one drone baseboard that consolidates the required electronics onto a single printed circuit board. The board would be designed around Oprones Inc.’s electrical, mechanical, communication, and safety requirements. By reducing redundant circuitry, wiring, connectors, and mounting hardware, the design can lower aircraft mass and improve electrical efficiency. These improvements would increase available flight time or allow the drone to carry a larger payload without changing its battery.

The integrated design would also improve reliability. Fewer inter-board connections mean fewer opportunities for loose cables, intermittent contacts, assembly errors, and vibration-related failures. Centralized voltage, current, and temperature monitoring would give the flight controller or operator a clearer view of the drone's electrical health and allow the system to respond safely to abnormal conditions.

Details

The proposed baseboard would act as the central electrical platform for the drone:

The board would accept power from the propulsion battery and distribute it to the drone's major subsystems. High-efficiency regulators would generate the voltage rails needed by the flight controller, sensors, communication devices, and payload electronics. Current and voltage sensors would monitor power use and help identify inefficient or unsafe operating conditions.

A microcontroller on the board would supervise power sequencing, collect health information, control payload interfaces, and communicate with the flight controller through CAN, UART, SPI, or another interface required by Oprones Inc. The design could also support configurable payload connectors so that cameras, lights, environmental sensors, or other mission equipment can be installed without adding separate control boards.

The minimum system would include:

- Battery input protection and reverse-polarity protection
- High-efficiency regulated power rails
- Centralized power distribution
- Voltage, current, and temperature monitoring
- Flight-controller and motor-controller interfaces
- UART, SPI, or I2C communication
- Payload power and control connections
- Fault detection and emergency power shutdown
- Debugging, programming, and status interfaces
- Mechanical mounting compatible with the Oprones drone platform

Primary Objective

Design and demonstrate a custom all-in-one baseboard that satisfies Oprones Inc.'s requirements while reducing the mass, power loss, wiring complexity, and number of separate electronic modules used by the drone.

Success Criteria

- Operate every required drone subsystem from the integrated board.
- Interface successfully with NVIDIA Jetson platforms from Oprones' requirements.
- Integrate with Pixhawk6X flight controller
- Can be manually controlled via Remote Control.
- Detect and report overcurrent, undervoltage, and overtemperature faults.
- Maintain acceptable component temperatures during sustained operation.
- Drone Sensors and communication integrate optimally
- Onboard computing and sensor integration communicate successfully

The main engineering challenge is integrating high-current propulsion power with sensitive digital and sensor circuitry on a compact PCB. Intentional grounding, filtering, thermal management, connector selection, and board layout will be necessary.

Schedule:

9/10-9/23	**Requirements & Planning** Confirm Oprones Inc.'s interfaces, onboard computer, flight controller, voltage and battery requirements, sensors, and communication protocols.
9/24-10/6	**Design and Final Proposal** Complete the power budget, schematic, component placement, thermal analysis, and PCB routing. Review the design for power integrity, grounding, connector accuracy, testability, and mechanical fit before ordering the boards.
10/7-10/25	**Breadboard prototype + PCB design** Prototype the power, sensing, and communication subsystems. Verify each required interface using development hardware. Complete the schematic and PCB layout, conduct design reviews, and order the PCB
10/26-11/10	**Systems integration testing** Inspect and assemble the PCB, test each voltage rail with a current-limited supply, program the processor, and verify sensing and

	communication. Integrate the board with the flight controller and representative Oprones payloads.
11/11-11/25	**Verification + Design Expo preparation** Perform load, thermal, communication, fault-response, and restrained-flight tests. Compare mass and energy consumption against the original system. Resolve defects, document results, and prepare the Expo demonstration, poster, and final report.
11/26-12/11	**Testing + poster + Final report**

The board would have:

- Power Regulation to various sensors, flight controller and onboard computer from a DC input.
- Ports to connect to several cameras via ribbon cables
- Ports for GPS (for testing), RC onboard modules, Lidar, output to Electronic Speed Controller (ESC)

For this project, testing connected to a PID tuning rig may be sufficient to test onboard connections for flight and other controls. Connecting the board to a standard f450 frame with motors would be viable.





Ref: Amazon link
Ref: Amvolab

To test the visuals showing visuals by connecting to the onboard computer via HDMI may be shown the visual mapping for SLAM.

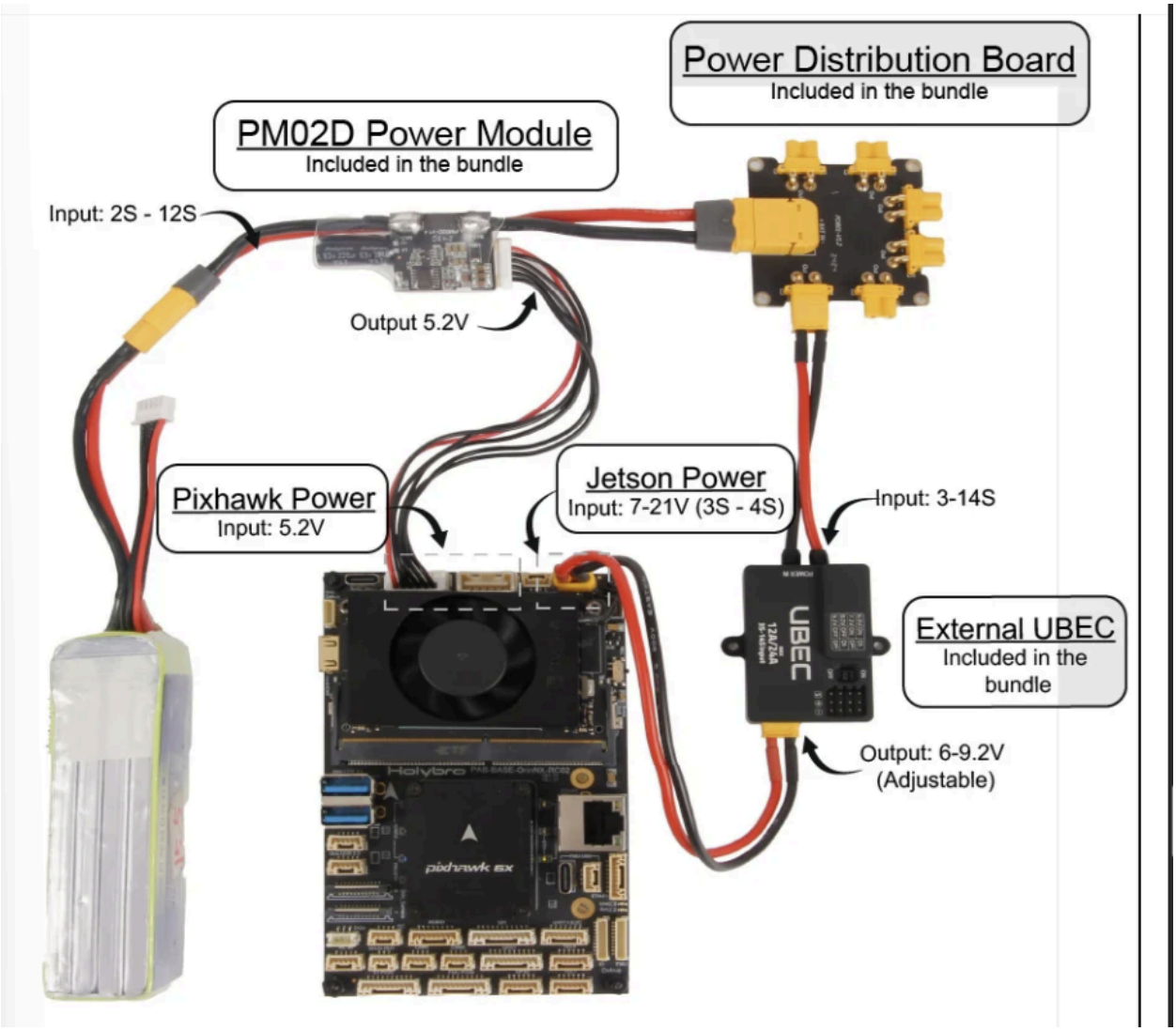
The code would be run on the onboard computer to compile and run imaging from the sensors for full SLAM. The code may be provided by **Oprones Inc.** for SLAM Visuals.

Budget:

Item	**Cost**
Jetson Nano (1)	~$399
Pixhawk 6X (1)	~$167
Pixy Cameras (3)	~$80
F450 Frame (1)	~$25
ESC 4in1 (1)	~$80
Motors (4)	~$60
RC Battery (1)	~$40
Propellors (for flight) (4)	~$20
Stereo Camera (1)	~$334
PCB Cost	~$550

Total: ~$1755 without sponsor, $550 with sponsor payment. Additional payments from sponsors may be provided upon request.

Issues:
Testing and Safety will be problematic due to running the drone. Testing in individual devices may be possible. To test Voltage and Current, ESC + Motor setup rig may be used or full frame for “flight” if safety constraints are set up. There are a lot of components to the current requirements. Suggestion: Reduce the requirements to fit the needs of Oprones Inc. while reducing the weight of the drone.



Ref: [Holybro Pixhawk Jetson Baseboard]
This is similar to the current design of Oprones Inc.

Project 2: Drone Battery Monitoring System + Telemetry Module

Goal

Build a standalone, inline battery monitoring PCB that reads high-current flight telemetry, transmits it via Bluetooth, and logs it to a custom remote application for performance analysis.

Problem

Standard drone setups require extracting SD cards from flight controllers to analyze power draw, or rely on low-resolution On-Screen Displays (OSD). Operators lack an isolated, easily attachable tool to stream high-fidelity data (voltage sag, current spikes, power consumption) to an external device for real-time monitoring and post-flight graphing.

Design outline:

Flight Battery → Custom Inline Sensor & BLE PCB → Drone ESC/PDB *(Wireless BLE Link)* → External Device (PC/Other module)

- High-current inline shunt resistor and voltage scaling circuitry.
- Microcontroller integration (e.g., nRF52 or ESP32) for high-speed Analog-to-Digital (ADC) sampling and Bluetooth Low Energy (BLE) transmission.
- Embedded firmware for data packaging, filtering, and wireless streaming.
- Custom external Graphical User Interface (GUI) to receive, plot, and log time-stamped CSV telemetry data.

Scope:

- Passive monitoring only; the board will not feature cut-off switches or interlocks that could accidentally sever power mid-flight.
- Monitors the main discharge leads (total pack voltage/current) rather than individual balance leads.
- BLE range is limited to near-field hover testing, bench testing, or pre-flight checks (standard BLE constraints).

Success Criteria

- Accurately measure inline voltage (up to 6S/25.2V) and current (up to 100A peak).
- Transmit telemetry packets wirelessly at a minimum of 10Hz without data corruption.
- External application successfully connects, displays a live scrolling graph of power draw, and saves a retrievable log file.
- The PCB operates continuously under sustained drone propulsion loads without thermal failure.

Milestones

Milestone 1

Read accurate voltage and current values from a dummy load using a development board. Establish a stable BLE connection and stream raw text data to a serial terminal. Develop the basic UI layout for the external app.

Milestone 2

Fabricate the custom inline PCB and connect it to a physical drone test stand. Stream live flight data to the finished GUI, verify graphical plotting. Perform sustained high-current thermal testing.

Cost

Item	Cost
BLE Microcontroller Dev Kits (2)	~$30
High-Current Sensing ICs & Shunts	~$20
Dummy Load / ESC Test Rig Parts	~$80
XT60 Connectors & Silicone Wire	~$20
PCB Fabrication (Heavy 2oz Copper)	~$150

Risks

The main risks are RF interference from the drone's motors and speed controllers disrupting the Bluetooth signal, and designing PCB traces/shunts capable of handling 50A-100A+ of sustained current without overheating or causing voltage drops.Proper isolation between the high-power analog paths and the sensitive digital RF circuitry is critical.

Proposal 1: ConPass – Interactive Badge And Cosplay Device

Fan conventions and expos like Comic Con are massive social gatherings where hundreds to tens of thousands passionate attendees share their interest in niche fandoms, express their creativity through elaborate handmade cosplays, and display their community pride.

Connecting others with the same interests in tv-shows, anime, or comic books is often the primary draw of these events, pushing attendees to navigate chaotic exhibition halls in search of their tribe. However, the sad reality is that finding and connecting with people who share your specific interests in a crowded convention center remains surprisingly difficult and sometimes awkward. Additionally, while fans love to express their love for their fandom through ornate cosplays, some just do not have the craftsmanship skills, time, or money to justify creating their own cosplay.

If fans had some convenient, widely-available, method of participating in their community and locating like minded people, even without a flamboyant cosplay, the overall experience of each attendee would significantly improve. For example, an electronic badge that could display your favorite character, push interactions with others in the same community, and even build a social network of similar people.

The idea of custom badges for a convention is not a new idea. Defcon, the world's largest gathering of hackers held every year in Vegas, has a thriving community of enthusiasts who create their own custom badges with different features embedded in them. However this has not been generalized outside of the Defcon space into a larger market with other specific conventions, which is what I envision.

Details

ConPass is a wearable smart badge and peripheral engine designed for attendees, makers, and cosplayers. The pass features a touch-interactive LCD screen displaying the user's favorite characters in a customized idle state.

- **Passive Proximity Triggers:** When the pass comes within 6 feet of another user with a matching fandom profile, the badge leverages Bluetooth Low Energy (BLE) scanning to recognize them. This triggers an asynchronous event such as a specific character animation and a customized soundbite played through an onboard DAC and micro-speaker. Users can customize these interactions, enabling dynamic mini-games upon meeting, or toggle the discovery feature off entirely for privacy.
- **Physical Interaction & Networking:** When attendees want to formalize a connection, they can physically tap their badges together. Utilizing Near Field Communication (NFC), the badges instantly and securely exchange unique identification tokens to register a friendship without the risk of accidental wireless pairings.

- **Customization Ecosystem:** Instead of relying on a resource-heavy native mobile app, the ConPass ecosystem uses a Web Bluetooth API. This acts as a centralized social platform where creators can share their open-source interaction logic and custom pixel art. Users can temporarily connect their smartphones to the badge via a web browser to push new graphics and audio files to the onboard flash memory. They can also push new friendships registered on the local board to the web and follow their new friends through this interface, being able to track their friends' past/current cosplays. Once synced, the badge disconnects and operates completely autonomously on the convention floor.

Schedule

- **Sept 10 - Sept 23 (Planning & Procurement):** Double check what parts and board we should be ordering with the instructors and GSI's. Order our boards and modules for NFC, BLE, and web communication. Order LCDs and audio components. Establish the C/C++ build environment and begin mapping the hardware timers. Draft the state machine and communication protocol for BLE proximity advertising and NFC-A tag emulation. Also start planning out our web interface and battery management methods.
- **Sept 24 - Oct 6 (Breadboarding & Firmware):** Bring up the breadboard prototype. Write the low-level drivers to drive the LCD display over SPI. Configure the DAC using DMA normal mode to play audio. Double check that the CPU and input capture interrupts are not blocked from handling incoming wireless proximity triggers.
- **Oct 7 - Oct 25 (PCB Routing):** Finalize the schematic and PCB layout, paying strict attention to the NFC antenna trace routing and impedance matching. Order the PCBs. During our wait for the PCB, we will shift focus to software. Develop the Web Bluetooth API interface using HTML/JavaScript and finalize the data structures for the user profiles, friendship tracking, and video/audio open source components on our website.
- **Oct 26 - Nov 10 (Assembly And Integration):** Solder components and assemble the PCBs upon arrival. Flash the firmware from the dev kit to the custom board and begin hardware debugging. Design and 3D-print a durable enclosure to physically protect the electronics from the wear and tear of a crowded show floor.
- **Nov 11 - Dec 9 (Testing, Debugging And Stretch Goals):** Dedicate this block to real-world testing. Measure battery drain during continuous display updates and verify NFC and BLE reliability in noisy wireless environments. Refine the web UI. **Stretch Goal:** If core functionality is stable, implement the customized mini-games triggered by BLE proximity. Finalize the poster and prepare for the Design Expo.

Budget:

Although I haven't looked into this too much, I believe this will be under our limited budget and I think I have done a fair estimation of parts we will need and their average costs through googling. We will most definitely need to ensure that we are getting quality and light weight products however, so that may raise this price slightly. Additionally I am unsure of what main MCU or SOC we should be using for this so I chose a rather standard STM32 MCU price that's

probably a little high, but we will have to consult with the teachers about the best chips to look into for our video processing and communication needs.

- STM32 MCU for main processing: ~ $15.00
- ST25R3916 NFC chip set: ~ $7.00
- 5.0" 40-pin TFT Display - 800x480 with Touchscreen: ~ $30.00
- Audio Amplifier & Speaker: ~ $15.00
- Custom PCB & Passive Components: ~ $20.00
- LiPo Battery & Enclosure Materials: ~ $20.00
- Solar cells if we choose to do this: ~ $20.00

Total per unit: ~ **$127.00**

We will probably need 4 units to test multiple badges interacting, so roughly **$508.00**. Then I think we can host a website for free through special programs for a limited amount of time so I will not include that in the budget yet.

Issues

The largest concerns for me at this time are:

1. Creating efficient and secure algorithms for both BLE and NFC.
2. Creating the web interface and connection that users will interact with.
3. Finding a low power LCD screen and chip that can give enough picture quality without killing our battery life.
4. Having a cheap enough product that enthusiasts will justify buying it.
5. Race conditions with multiple badges in one area.
6. Designing a robust but elegant casing so that the product won't break with one drop.
7. Keeping the product very lightweight, since it will be attached to people for a long period of time.

Issues 1 and 2 seem reasonable enough to solve with some light googling or prompting AI in order to figure out the best method of coding these algorithms. Issue 3 and 4 will be difficult to balance in order to create the best product but that will be something we have to look more into and consult with the GSI's, James, and Mark about. Issues 5 will be something that can be reasonably figured out through software timing and testing. Issue 6 will be a difficult engineering task but Mark did state that the mechanical design attribute will have some leeway, but we can still try to find an elegant solution over time and testing. Issue 7 is going to again be a challenge, especially with the battery and screen going in, however we can possibly find a different mounting system then just a standard lanyard that will solve this problem.

Stretch Goals

I would really like to implement a solar rechargeable system to this in some way in order to be a bit more sustainable and efficient as a system, although there would still most definitely be a way to just plug it in to charge as well. I would also really love to have this actually act as the

pass to a convention, so we could charge an actual ticket like Google Wallet does or something and just authenticate ourselves with our smart badge. I would also like to see some other game and interaction ideas to implement upon meeting someone. Another idea would be to have an actual localized communication network through LoRa at each convention which would be interesting. These are just some additional ideas if we want to add onto our burden.

Testing

We will confirm our product works by making sure that we can upload custom videos and animations to the badge that work properly in idle. Additionally, when a matching fandom badge comes within the radius the correct animation will be played on both badges. Friendship data will be tracked internally on the badge and we will test whether that has transferred correctly through the website interface. I would also like the website holding the open source art to correctly provide the animations to people interested in them in a short time which we will test through downloading variable sized animations. Obviously there are more thorough tests we will have to run but these are some of the basic goals I would like to achieve.

Proposal 2: SideQuest Compass

Despite being constantly digitally connected to almost every other human on Earth, the loneliness epidemic among college students and young adults is larger than ever. Modern social media apps, like TikTok and Instagram, encourage endless doom-scrolling rather than fostering actual human connection and have even formed a new anxiety for the youth to organically introduce themselves to others.

To fight this loneliness epidemic and encourage interactions, the SideQuest Compass would serve as a slightly retro pocket device designed to encourage spontaneous, face-to-face interaction while prioritizing personal safety. It is a navigational tool featuring a mechanical kill-switch, a small display, a small keyboard, and a simulated directional radar interface created through a GPS module and a magnetometer.

When a user feels open to a spontaneous hangout, they flip the switch to activate a LoRa transceiver where they can either create a sidequest with a short description and post it, or they can search for other sidequests. When a user is searching for a sidequest, they will receive others sidequests one by one and either accept or reject them. To ensure safety, when they accept, the compass never points directly to another user. Instead, an onboard microcontroller cross-references GPS data against a hardcoded list of verified, public "Safe Zones" (like the Diag). It then updates the radar to point in the direction of the agreed location to direct both users to the exact same public landmark to meet up. They will both then make their way to that location and then tap on their devices together to confirm they have met. This can be expanded to multiple users in one sidequest with additional logic.

In order to join this program and use the device, each user would have to be verified through hard identification like a MCard reader on the device along with a predistribution verification that would be handled by some club on compass. The compass would then communicate over encrypted LoRa to a base station that can verify the details on each use and allow the device to function correctly. This base station would hold a database of information of who has which device but will not track sidequests themselves.

Schedule

- **Sept 15 - Sept 30:** Finalize component selection (LoRa module, GPS module, microcontroller, keyboard, etc.) and establish basic LoRa communication protocol.
- **Oct 1 - Oct 15:** Breadboard the hardware, design the PCB, and begin the software parsing the GPS module data, along with the card reader, and keyboard. Order PCBs.
- **Oct 16 - Oct 31:** While waiting for PCBs, finalize GPS software, and finalize the encrypted LoRa communication protocol. Set up the base station software and finish MCard reader.
- **Nov 1 - Nov 15:** Finalize all software, Assemble PCBs, integrate everything into one system, and begin designing the form fitting casing.

- **Nov 16 - Dec 9:** Debugging, extensive field testing of the LoRa range, GPS accuracy, and base station verification protocols. Finalize poster and presentation.

Budget

- Microcontroller: ~ $20
- LoRa Transceivers: ~ $30
- Battery: ~ $10
- GPS Modules: ~ $25
- LCD displays: ~ $35
- PCBs: ~ $10
- Card reader: ~ $10

Total per unit: ~$140 (Well under the $250/student budget constraint).

We will need multiple so I will estimate 4, which brings the total cost to roughly $560, although this is a very rough estimate and we would prioritize cost effectiveness.

Issues

The biggest issue will be safety with authenticating people on each session. We will have to think of more authentication methods that won't leave anybody in a sketchy situation. Some other issues are that we will need to test the LoRa range within Ann Arbor to make sure we can communicate a decent enough distance. We need to make sure that we have a good encryption scheme so that no person's identity can be fabricated or stolen, which should be doable with a little bit of software. It will also be a challenge making the design small and cost effective, but this will have to be figured out with more thorough planning.

Testing

For testing we will first have to make sure that the identification information is correctly being sent to the base station on each interaction request by checking the base station database. We will then test both the sending and receiving of requests to make sure that we can receive and send to others. We will then test the range of the device and the battery life of being on. We will also have to test our encryption of the communication to see if there is any way to read the private communication.

Guitar Functionality Enhancement

Many electronic innovations have been created surrounding the guitar in recent years, accomplishing specialized tasks, like sound manipulation, tuning, etc. My goal is to leverage as many guitar enhancing features as I can into one embedded system.
Some of these features include the following:

- Customizable digital effects/tone manipulation
- Volume/Gain/EQ
- Low latency bluetooth
- MIDI
- Playback (upload audio or stream from a device)
- Tuning
- Multi purpose (quick detachable?) whammy bar
- Slide controls along neck
- Digital display on front of guitar body that can display…
    - Creative coding visual effects that follow the sound being played
    - Videos
    - Anything the user desires!
- Smaller digital display on main device with customizable UI
- Movable low profile buttons to control effects
- Multi-functional complementary floor pedal
- Anything else we can dream up!

The main idea of this project is to consolidate a large handful of cool guitar enhancing gizmos into one working system, where everything communicates with each other. All of these features will stem from one central device, which should be small enough to fit in your hand. It will take in a signal from the quarter inch output on an electric guitar and output it via any of our multiple output methods- analog (¼ in, 3.5mm), MIDI, bluetooth, etc.

We must consider a few implementation issues. One would be getting bluetooth audio to perform at low enough latency to not disrupt the user. I have experimented with this a bit and

it can be quite tricky. Another issue/goal is to make the system low-profile and easily integratable with an electric guitar. Designing a whammy bar that is able to be attached to the guitar and provides a digital signal similar to a potentiometer may be quite difficult to design from a mechanical standpoint.

Schedule (based on provided example)

9/10-9/23	Planning Brainstorming other features, exploring implementations of previously discussed features. discuss materials/device feel. (Luxury/functional/affordable). Find a microprocessor to use (note Daisy Seed). Research bluetooth solutions.
9/24-10/6	Part Selection and Design Order microprocessor + other components and begin design for whammy bar, breadboard prototypes, begin experimenting with routing wires on the guitar to find optimal ergonomic placement.
10/7-10/25	Breadboard and Bluetooth Begin testing breadboard circuitry and test bluetooth design with basic programs. If our bluetooth part(s) do not work as well as we need them to, explore other options.
10/7-10/25	PCB Design and Optimization Once we have a working breadboard prototype, we can begin work on a PCB design with size optimization in mind.
10/26-12/9	Display Implementation, Shell Design The artistic digital display leaves a lot of room for creative design. This can take as much or as little time as needed. If we go in the direction of creative coding, this opens up many doors for this feature, and we can load it full of creative visual effects. During this time, we'll have to have it outfitted on the guitar to start working within the boundaries of its limitations. During this time, we should optimize the placement of everything and design a housing for the main device.
12/10-12/11	Finish Final Report
12/25	Christmas Merry Christmas

Budget

This project should realistically not go anywhere near the limit of our budget, but if we invest in a high quality display, or if we require expensive bluetooth technology, it may increase the cost. Here is a rough estimate of some main components we will need.

Item	Quantity	Unit Cost	Total Cost
Daisy Seed3	2	$30.00	$60.00
Display	1	$100.00	$100.00
PCB (footswitch)	1	$15.00	$15.00
PCB (main)	1	$15.00	$15.00
Misc	1	$100.00	$100.00
Total Cost			$290.00

Main Concerns

Our biggest concerns are:

1. Finding/creating a low latency bluetooth creation

There are a lot of resources online for bluetooth solutions, some research and dedication surely will provide us with what we are looking for.

2. Choosing proper RGB display, creative coding implementation

Finding a display to put on the front of the guitar that a) looks good even when powered off, b) contours to the shape of the instrument, and c) isn't egregiously expensive may be difficult.

3. Shrinking device size down to handheld size

This is manageable, but it will require a lot of time spent on PCB design and research into what we need for our signal processing.

4. Designing digital whammy bar

This should be a main feature of the device, so we should start working on it early. It will require some pretty solid mechanical engineering if we want to make something that functions just like a regular whammy bar, but also isn't drilled into the guitar.

Real World Evaluation

The measure of success for this product would be user satisfaction. The end goal is to increase the abilities of the guitarist while not adding much physical overhead to their setup. We want the user to feel like they are able to be more creative and to explore new ways to play.

iSuck

Have you ever had to change a lightbulb, or plug in a projector on the ceiling and think "Man, I wish I had suction cups as hands and could crawl on the walls and ceiling."? Well, I have the solution for you.

iSuck draws inspiration from battle tanks, and uses a tank tread with suction cups to traverse vertical and inverted surfaces. It uses motors to power the tracks and servos to change the geometry of the tracks, allowing it to go from wall to ceiling. It also has a pov camera that allows the user to 1. feel like they are driving a tank (cool), and 2. watch from up close as they use the implements to accomplish tasks.

To control its implements, iSuck will feature a multi-axis robotic arm that will operate similar to the human arm, but much more limited for simplicity. It is powered by servos and will have a physical interface at the end to work with each implement. A few of the implements include the following:

- Simple grabber
- Light bulb changer
- Cable plugger/unplugger
- Button presser
- Duster

Implementation Issues

It will be a very mechanically complex system overall, and getting suction cups to work properly on a rotating tank tread will prove to be a very challenging task. Designing it to be lightweight enough to be held by suction will be quite difficult, and it may be advisable to use a pneumatic pump to help with suction.

Budget

The most expensive parts of this project will probably be the motors, pump, and microcontroller. We'll also have to pay for all the mechanical parts, like treads, suction cups, pneumatic hoses, 3D printed parts, servos, etc. It should probably be around 400 dollars on the low end and 600 dollars on the high end.

Real World Evaluation

iSuck will prove its worth by saving the day when a ladder is in need. Instead of getting a ladder and risking injury or death from a fall, iSuck can take over these dangerous tasks. The measure of success is firstly its usability. If it is able to poke, prod, screw, unscrew, plug and unplug while it is hanging upside down from the ceiling, then it is successful. Secondly, the joy and immersion that it brings the user. When the user gets to feel like they're driving a real tank on the walls of their very own home, it will surely bring a smile to their face.

ECO friendly heads up display

Cars and other various road vehicles account for 15% of total greenhouse gas emissions, making it one of the largest single sources of global emissions[1]. Passenger vehicles are the largest subsection in transportation in terms of CO2 emissions. Aggressive driving behaviors can contribute heavily to these emissions, with studies showing decreases of up to 22 percent in CO2 emissions when using automatic adaptive speed measures[2]. By utilizing environmentally friendly driving habits, a significant reduction in CO2 emissions is possible.

Eco-friendly driving techniques are often not employed as many drivers lack the awareness needed to adopt them, and run counter to current habits such as quick acceleration, hard braking and hard turns, all of which waste fuel and momentum.[3]

Devices currently available often give a simplified breakdown of how environmentally friendly a drive was, however they don't offer actionable, real time feedback on how to improve. A low cost and simple heads up display that can alert drivers of poor driving habits can result in significant decreases in transportation emissions,[4] while helping users save on fuel costs.

[1] https://www.ipcc.ch/report/ar6/wg3/chapter/chapter-10/nsport
[2] https://news.mit.edu/2025/eco-driving-measures-could-significantly
[3] https://www.osti.gov/biblio/1399958
[4] https://www.sciencedirect.com/science/article/pii/S1361920923004224

Details

In order to assist users on their driving habits we propose a system that will be broken down into two subsystems, internal vehicle metrics and external vehicle metrics, in order to provide feedback to the user through a heads up display. In order to obtain the critical engine performance parameters our system would utilize the OBD-II port. From here we can receive metrics such as vehicle speed, engine RPM, engine load, fuel rate, etc. In addition to these, an accelerometer will be placed along the chassis of the vehicle to detect aggressive acceleration and turning.

The critical external measurements that will need to be obtained will include incline, street signs and proximity to other vehicles. The incline will be obtained through topography maps, requiring the system to utilize GPS. Street sign data may be obtained similarly to incline, by using existing infrastructure, or by utilizing computer vision trained on street sign data. To obtain vehicle proximity data, a Lidar sensor mounted on the heads up display will allow the system to determine braking distance from the vehicle directly in front.

The final system should be able to collect all this information in real time and display a Eco score on a heads up display to the driver. In order to not distract the driver a simplistic display should be utilized, showing only visuals. These visuals may include actionable items to the driver including but not limited to hard braking, rapid acceleration, and aggressive turns.

Schedule

9/10 - 9/18	Planning and Research Finalize system architecture, 2 PCB(PCB1: OBD-II, IMU, PCB2: HUD, GPS, ToF). Research components and submit draft proposal by 9/18
9/19 - 10/6	Design and part procurement Begin ordering the necessary components that the system will require, finalize the design of the HUD hardware, test acquiring internal car metrics, order peripheral sensors. Submit final proposal and order all parts immediately following the final proposal
10/7 -10/25	Schematic and PCB design Finalize the HUD and data acquisition PCB's. Test bluetooth between devboards. Order PCB's by 10/12. While awaiting PCB's design of any 3D printed enclosures. Fully assemble PCB's by 10/25.
10/26 - 11/10	Software and mounting Design Testing firmware onto the boards, implementing bluetooth between them, programming the eco score algorithm and HUD display graphics. Print 3D mounts
11/11 - 11/25	Full System Testing Mount both PCBs and enclosures into a test car to collect live driving data and validate fuel rate calculations. Construct demo car simulator for expo
11/26 -12/9	Final Testing, poster and debug System should be fully assembled in car and going through minor tweaks and debug

12/10 - 12/11	Finish Final Report

Budget

Itemized estimate budget:

Item	Quantity	Cost Per Item	Total Cost
Microcontroller (ESP32)	6	$8.00	$48.00
PCB1	5	$15	$75
PCB2	5	$15	$75
GPS receiver Module	3	$12	$36
HUD Projection Display	2	$20	$40
VL53L1CX ToF	3	$4	$12
MPU6050 IMU	3	$12	$36
Misc Parts(LDO's, OBD-II plug)	1	$150	$150
Shipping	1	$100	$100
Total			$572

Issues

Largest concerns:

1. Building this system to be safely housed in a vehicle
2. Parsing internal car metrics
3. Expo presentation (simulator)
4. Accessing the internal metrics of the vehicle
5. Requiring 2 PCB's that much communicate

Most issues lie in the fact that this system involves a car to test and validate in, so gaining access to a test vehicle will help overcome the majority of these challenges.

Smart pool filter
Chemical and debris cleanser

Residential and community pools require large amounts of upkeep to maintain swimmable conditions. This is especially true towards the beginning months of summer where large amounts of chemicals are dispersed into the water, as well as debris cleaned out of it. It is important to maintain adequate chemical balances as failing to do so leaves swimmers vulnerable to health risks such as eye and skin irritation[1], algae growth, and damage to the pool. Maintaining good water quality will leave the owners with clear waters and an increased pool lifespan.

Water quality maintenance is often neglected as users don't have up to date information on the pool's chemical balances as the current method requires them to obtain a sample and bring it to a pool store. Some chemicals such as chlorine are often dispersed by slow release methods through tablets in floaters. And other chemicals that need to be released into the pool at less predictable intervals over longer periods of time will often be released at inappropriate times, wasting time and resources.

A low cost real time feedback system will help alert users of imbalances in the pool's chemicals, and will be able to disperse the needed chemicals to bring it back to balance or give warnings so that the necessary fixes can be made.

[1] https://www.cdc.gov/healthy-swimming/prevention/index.html

Details

In order to assist users in maintaining healthy water quality we propose a system that will be broken down between 2 subsystems, internal water quality and external debris collection. To obtain important water quality information our system will utilize immersed analog probes. From these we will measure metrics such as pH level, oxidation reduction potential and water temperature. In addition to these measurements, the system will also utilize dosing pumps connected to internal chemical reservoirs to dispense pH adjustments or chlorinating agents at calculated intervals.

The external debris collection system will require a small built-in pump motor to create a mini water vortex to draw surface debris towards the device. There will be some form of netting for it to collect into. In addition to this we will need to design a propeller system for the device to travel around the pool, as well as an internal IMU to orient and navigate the robot.

Schedule

9/10 - 9/18	Planning and Research Design system requirements, draft block diagram (Sensors/ Dosing and Motors/Navigation), and research waterproof analog probes (pH, ORP, Temperature). Submit draft proposal by 9/18.
9/19 - 10/6	Design and part procurement Order development boards, peristaltic pumps, probe modules, and propulsion motors. Design the hull and waterproofing and test analog sensor readings in reference water samples. Submit final proposal and finalize component orders.
10/7 -10/25	Schematic and PCB design Design custom PCBs for sensor signal conditioning, pump control relays, and motor driving. Order PCBs by 10/25. While waiting for PCB fabrication, 3D print mounts, the debris net frame, and internal chemical chamber.
10/26 - 11/10	Software and Waterproofing Develop firmware for processing data, testing automated dosing, and controlling the propulsion thruster. Seal the enclosure to waterproof standards and integrate the collection pump and mesh collection basket.
11/11 - 11/25	Full System Testing Deploy the assembled floater in a test pool environment to evaluate sensor data, check chemical dosing accuracy, and verify automatic debris collection and navigation paths. Construct a portable water tank setup for the expo.
11/26 -12/9	Final Testing, poster and debug Debug navigation, debris collection, chemical dosing and waterproofing
12/10 - 12/11	Finish Final Report

Budget

Itemized estimate budget:

Item	Quantity	Cost Per Item	Total Cost
Microcontroller (ESP32)	6	$8.00	$48.00
PCB	5	$15.00	$75.00
Analog pH Sensor	3	$15.00	$45.00
Industrial ORP Sensor	3	$20.00	$60.00
DS18B20 Temperature Sensor	3	$9.00	$27.00
MPU6050 IMU Module	3	$12.00	$36.00
Peristaltic Dosing Pumps	4	$25.00	$100.00
Propulsion Motors	2	$15.00	$30.00
Skimmer Pump	2	$12.00	$24.00
Misc Parts (Seals, 3D enclosure)	1	$150.00	$150.00
Shipping	1	$100.00	$100.00
Total Estimated Cost			$695.00

Issues

Largest concerns:

1. Waterproofing the enclosure to protect the electronics
2. Obtaining accurate chemical readings
3. Being able to dose the chemicals as required

Securing access to a pool like environment early will help mitigate many of these issues and address feasibility

Bolt

In the 2024 Paris Olympics the first, second and third place times for the men's 1500m final were 3:27.65, 3:27.79, and 3:27.80. It's no exaggeration to say that for professional track athletes, every second counts. This motto is reflected in elite athletes' training regimens, which are tracked to subsecond precision typically using advanced watches or cameras.

That said, there is a lack of robotic devices that aid track athletes in their workout. Having a physical training aid, such as an autonomous pacing robot like Bolt, would not only enable athletes to engage in high precision track workouts, but the physical/companion nature of such a vehicle would also motivate athletes to run in a way that cannot be mimicked by smart watches alone.

This idea is not entirely novel, and Puma actually created a similar vehicle in 2016. That said, their device was mainly created to inspire an ad campaign in the leadup to the summer olympics and was never marketed as a consumer product. Thus, there is an opportunity to provide elite athletes with a pacing device that they cannot find elsewhere.

Details

Bolt would be designed as a line-following pacing robot, inspired by Puma's Beatbot, that could be programmed to execute a track workout and stream results to a smartphone for analysis.

A user (either the athlete or their coach) would use a simple app on their device to program the desired workout by listing distances and their associated paces. This data would be wirelessly streamed to Bolt to initialize the workout. Bolt would then be placed on a line on the track, possibly undergoing basic calibration, and then with the press of a button (either on-vehicle or from the smartphone), would start a short countdown to begin the workout. Bolt would contain sensors to track speed and distance, ideally with as high of accuracy as possible. This may involve GPS, or more likely, optical speed sensors and wheel rotation encoders to enable higher precision measurements that could reliably work indoor and outdoor. Bolt would ideally operate in the 15-25 mph range using an onboard, rechargeable battery, however for initial safety and testing purposes this speed would likely be dramatically reduced.

During a workout, Bolt would transmit basic information to the phone such as current speed, distance, and possibly battery life. Data would then be saved to the user's smartphone to avoid the need for on-vehicle storage to persist a completed workout.

Schedule

Date	Milestone
9/10-9/24	Brainstorm design and hardware/software features for Bolt, compiling results for proposal meeting. Primarily decide what motors to use, wireless communication, IR camera for line tracking, and speed/distance sensors. Identify a processor that can handle the computation load in real time and find a battery option that can power the device for the given speed/distance requirements.
9/25-10/6	Order parts and start writing interfaces for devices we are intending to use as a group. Possibly start prototyping simple wireless communication and also work on a barebones smartphone app.
10/7-10/25	Prototype a vehicle with simple IR line detection that works with tape or on a physical track. At this point it is fine to use weaker motors and batteries to operate at slow speeds. Get basic communication between smartphone and vehicle working with either RF or cellular - test how workouts will be stored on device and converted into physical motion.
Unsure about exact dates	● Design and order PCBs based off prototyping ● Finalize smartphone interface and wireless communication (likely while the PCB is waiting to arrive) ● Thoroughly test IR line detection indoors and outdoors ● Integration testing before design expo and final report

Budget

The budget specified below is extremely rough and mainly meant to estimate the upper end of what we may expect to pay for parts. As a team of 4-5,

ATmega328	$3
On-vehicle PCB	$10-$20
4x motor	$30-$100
BLE module	$10
Magnetic encoder	$10-$20
IR sensor	$10-$20
Misc parts (wheels, chassis, etc…)	$80-$100
Shipping	$80-$100
Estimated Total	$233-$373

Issues

A list of anticipated challenges includes:

1. Monitoring speed and distance accurately with optical sensors
2. Wireless communication between smartphone and vehicle
3. Accurately following track lines with IR sensor
4. Meeting speed and weight requirements and containing all the components as a vehicle

Issue 3 is at the core of how our device works but it is far from a novel technique so it seems feasible. Issue 4 may pose a significant challenge depending on our design, however it may be possible to use the chassis of an existing RC car to mount our components. Issue 2 seems like it will vary depending on our method of wireless communication, but we are working on BLE communication in the lab so hopefully this will be translatable even if we choose a different form of transmission. I'm least certain about issue 1 but from basic searching it seems like there are a variety of options that we can weigh for cost and effectiveness (slotted optical encoders, reflective optical sensors, non-contact ground sensors).

Additional Stretch Goal

A few ideas include:

1. Finding a way to incorporate solar recharging charging mounted on the vehicle.
2. Using speech or sound detection (like a clap) to either stop or speed up the device.
3. Incorporating a small laser display aimed at the ground to communicate information to the runner like speed, time, and distance.

Fridge Buddy

Imagine a device that could be mounted inside your fridge and passively monitor that food that you are putting in to minimize food waste. This device would interface with an external display, either attached magnetically to the outside of the fridge or in the form of a smartphone app, which lists the items sitting inside the fridge along with how long they have been there for. Items would be ordered by the date they were added to the fridge to prioritize leftover items or ingredients that are closer to expiring. A simpler alternative to a camera would be a barcode scanner that allows users to scan items as a method of adding them to the fridge catalog, however this would be less flexible than a camera with object detection capabilities.

To buy a smart fridge advertised with these capabilities could run you thousands of dollars which is unreasonably expensive for most consumers. Additionally, existing products like the FridgeCam are only available in the UK and have poor reviews relating to ease of use and accuracy.

This project would pose some significant challenges, mainly relating to detecting food from a simple photo or video. In a fridge, condensation, unpredictable lighting and bulky items can obscure a camera making it difficult to detect items from a simple photo. The barcode scanner is a reasonable solution to this problem, but it increases the overhead for a user and it is less flexible since produce items rarely have packaging with a barcode. Additionally, it is unclear whether there is an ideal place to mount such a device across all fridges. It is possible that it would clip on to a door and have a wide angle lens to capture all parts of the fridge, or possibly be mounted on the exterior and require a user to log an item before putting it into the fridge.

The budget of this device would almost certainly be below $300-$400 and would leave ample room to spare for a team of four.

This project would utilize various hardware components, such as a camera for barcode scanning or object recognition, an automatic light for consistent photo lighting, on device storage of recently added objects, and wireless communication to send data to a smartphone or LCD display outside the fridge.

The Fridge Buddy also has interesting opportunities for extensibility, mainly related to improving the accuracy of object recognition in software, and also possibly making more advanced recommendations - such as meals a user could make with their current ingredients - based on data collected by the device.

WorldView: A Low-Cost Wearable Visual Narrator for Blind and Low-Vision Users

High-Level Description

WorldView is a wearable device intended to help blind and visually impaired users experience the visual side of the world around them. Humans rely heavily on vision not only for navigation but also for understanding and enjoying an environment. A sighted person sitting on a park bench can notice the color of the sky, children playing nearby, sunlight across the grass or leaves beginning to change color. WorldView would translate these visual details into spoken descriptions, allowing the user to experience more of the atmosphere and beauty of a scene through sound.

The device is **not intended to replace a white cane, guide dog, or other mobility aid**. Instead, it would act as a wearable visual narrator that a user could choose to use when they want to experience a scene in greater detail.

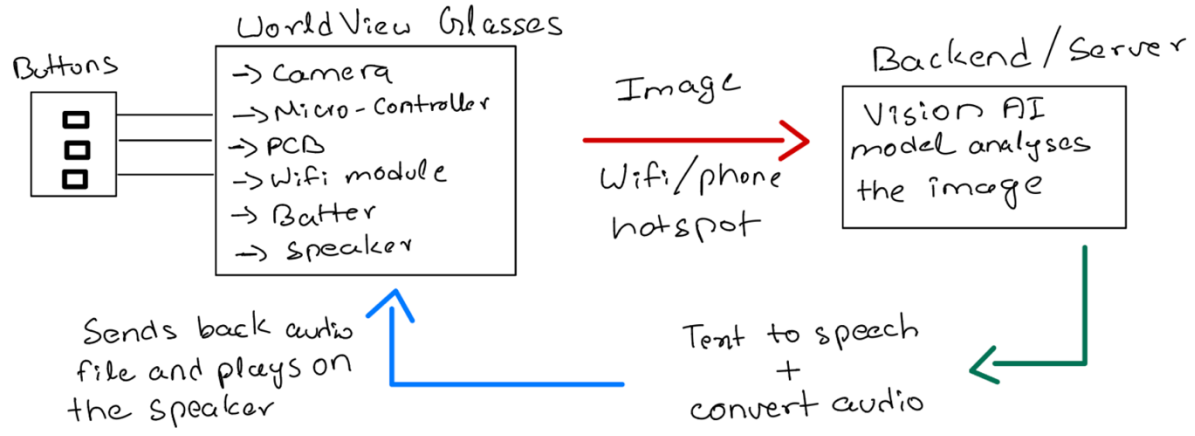
The finished product would resemble lightweight glasses or a glasses-mounted device containing a forward facing camera, functional buttons, wireless communication, a rechargeable battery and a speaker. When the user requests a description, the device would capture the scene and provide a natural spoken description of the environment, including important activities, colors, surroundings, and other visually meaningful details.

For example, instead of simply stating *"There is a tree, a bench, and three people,"*

WorldView could describe: *"You are sitting in a quiet park on a clear afternoon. Children are playing across a large grassy field, some of the surrounding trees are beginning to turn yellow, and the sky is bright blue with a few thin clouds."*

Existing visual-assistance technologies demonstrate a market for this type of functionality. WorldView would focus on providing a simpler, wearable, and lower-cost way for users to experience rich descriptions of the world around them.

Basic Functional Diagram



Possible Implementation Issues

- **Image capture and transfer:** The camera image may be too large for the microcontroller to handle or upload quickly. We may need to reduce image resolution or compress the image before sending it to the server.
- **Network delay and reliability:** Since the AI processing is done remotely, weak Wi-Fi or a slow connection could cause long response times or failed requests. The system should detect these failures and notify the user instead of waiting indefinitely.
- **AI description accuracy:** The AI may occasionally miss important details or describe something incorrectly. Testing different prompts and limiting the system to descriptive, non-safety-critical information would be important.
- **Weight and battery life:** The camera, PCB, speaker and battery may make the glasses uncomfortable if everything is mounted on the frame. If necessary, the battery or part of the electronics could be moved to a small clip-on module.

Dates	Main Work
Sept. 10 – Sept. 24	Finalize project requirements. Research similar products and select camera, microcontroller, audio, battery and wireless components. Begin testing the AI image-description and text-to-speech pipeline separately on a computer.
Sept. 25 – Oct. 8	Build an initial hardware prototype using development boards. Get the camera, Wi-Fi communication, buttons and audio working. Begin connecting the hardware to the backend so an image can be sent to the AI and a spoken response returned.
Oct. 9 – Oct. 22	Complete the first full prototype: **button → image capture → Wi-Fi → AI description → speech playback**. Begin the custom PCB schematic and layout. Because fall break is **Oct. 15–20**, keep work during that period light and mainly focused on PCB/software review.
Oct. 23 – Nov. 5	Finalize and order the custom PCB. While waiting for fabrication, continue improving firmware, AI prompts, audio playback and network reliability. Begin designing and 3D-printing the wearable enclosure
Nov. 6 – Nov. 19	Assemble and debug the PCB. Aim to have the complete system functioning on the custom hardware by the end of this period. If a PCB revision is needed, order it as early as possible.
Nov. 20 – Dec. 3	Focus on final product integration and testing. Measure response time, AI accuracy, battery life, wireless reliability and wearability. Fix major problems and finalize the enclosure. **Target Dec. 3 as the internal project-completion date.**
Dec. 4 – Dec. 9	Keep this period mainly for emergency debugging and final improvements.
Dec. 10 – Dec. 11	**Dec. 10: Design Expo.** Demonstrate the completed product. **Dec. 11: Final report due.** Make any final report edits and submit.

Component	Example	Rough Team Cost
Microcontrollers / dev boards	ESP32-S3 DevKitC-1, 2–3 boards	~$30–45
Camera modules	OV2640, 2–3 units	~$25–40
Audio hardware	Bone-conduction speaker + amplifier + spares	~$25–40
Batteries / charging / power	Batteries, chargers, regulators	~$30–50
Buttons / vibration / LEDs / connectors	Misc. interface parts	~$15–25
Custom PCB + components	First PCB + spare boards	~$60–100
Possible PCB revision	Second PCB order if needed	~$50–80
Glasses / enclosure prototypes	Frames + several 3D-print iterations	~$40–70
AI / backend / cloud testing	Vision AI, TTS, hosting	~$20–40
Miscellaneous / replacement parts	Wiring, adapters, failed components	~$30–50
Estimated Development Total		**~$325–540**

Evaluation and Testing

The project would be successful if the device can consistently capture a scene and provide a useful spoken description.

- **Response time:** Target under 5 seconds from button press to audio.
- **Description accuracy:** Test the device in different indoor and outdoor environments and compare its descriptions to the actual scene. (90% accurate)
- **Reliability:** Measure how often image requests successfully return a description.
- **Battery life:** Measure operating time under normal intermittent use. (> ~4hrs)
- **Wearability:** Confirm the final device is lightweight and remains secure during normal use.

Market Need and Product Differentiation

There is an established demand for technologies that make visual information more accessible. The WHO estimates that **at least 2.2 billion people worldwide have near or distance vision impairment**. Existing tools such as **Microsoft Seeing AI** and **Envision Glasses** already provide AI-based visual descriptions with Envision Glasses costing about **$2,499**.

WorldView would target a more focused niche: a lower-cost, hands-free wearable designed primarily to provide rich descriptions of scenery and atmosphere rather than serving mainly as a navigation or reading aid.

Sources: WHO · Microsoft Seeing AI · Envision Glasses

RearGuard: Bicycle Rear-Approach and Blind-Spot Warning System

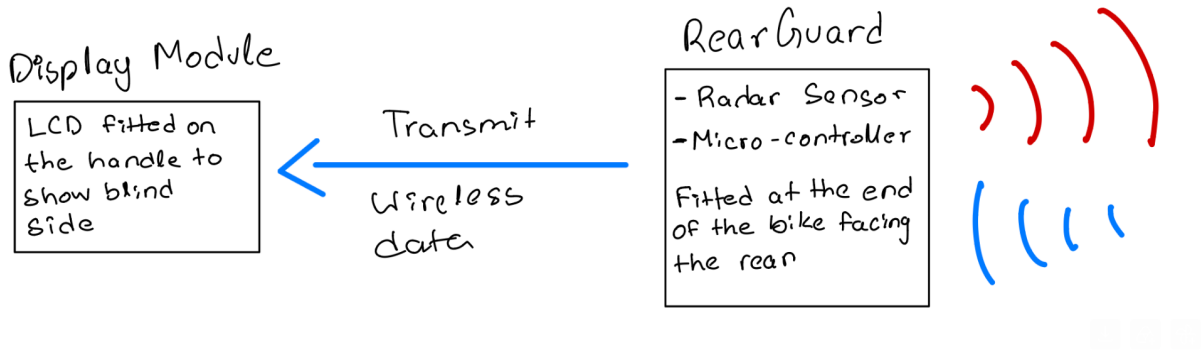
High-Level Description

RearGuard is a bicycle-mounted safety device designed to warn cyclists when a vehicle or other fast moving object is approaching from behind. Cyclists normally have limited awareness of what is happening behind them unless they turn their head which can be distracting while riding.

The system would use a **rear facing radar sensor** mounted underneath the bicycle seat. A small wireless display mounted on the handlebars would show whether an object is approaching and approximately how far behind it is. As the object gets closer, the display would provide a more noticeable warning, such as changing from green to yellow to red and flashing when the object reaches a critical distance.

A radar-based sensor would be preferred over ultrasonic sensing because radar can detect objects at longer distances and works better for approaching vehicles outdoors. *The goal would not be to replace normal road awareness but to provide the rider with an additional warning system for traffic approaching from behind.*

Basic functional diagram



Possible Implementation Issues

- **Radar detection:** The sensor must distinguish an approaching vehicle from stationary objects such as parked cars, trees, or signs. Range and relative motion would need to be tested carefully.

- **Detection angle:** The radar must cover enough of the lane behind the cyclist without frequently detecting vehicles traveling in unrelated lanes.
- **Wireless reliability:** The rear sensor and handlebar display must maintain a reliable connection while the bicycle is moving.
- **False warnings:** Too many unnecessary alerts could make the system distracting. Detection thresholds would need to be adjusted through road testing.
- **Weather and mounting:** The rear electronics should remain secure and protected from vibration, light rain, and normal outdoor use.

Rough Project Schedule

Dates	Main Work
Sept. 10 – Sept. 24	Research bicycle radar systems and select radar, processor, display, wireless communication and battery components.
Sept. 25 – Oct. 8	Test the radar using a development board. Measure detection distance and determine whether approaching and stationary objects can be distinguished.
Oct. 9 – Oct. 22	Build a complete prototype with rear radar, wireless communication, and handlebar display. Begin PCB design.
Oct. 23 – Nov. 5	Finalize and order PCB. Develop enclosure and bicycle mounting system while waiting for fabrication.
Nov. 6 – Nov. 19	Assemble the PCB and integrate the rear module and handlebar display. Begin bicycle testing.
Nov. 20 – Dec. 3	Test detection range, false warnings, wireless reliability, battery life, and mounting. Finalize the product.
Dec. 4 – Dec. 9	Emergency/debugging period and Expo preparation.
Dec. 10 – Dec. 11	Design Expo and final report.

Rough Budget

Component	Example	Rough Cost
Radar sensor	24/60 GHz radar module	~$30–70
Microcontrollers	ESP32-based boards, 2 units	~$25–35
Handlebar display	Small OLED/TFT display	~$10–20
LEDs / vibration / controls	Standard components	~$10–15
Batteries + charging circuits	Rear and front units	~$25–40
Custom PCB + components	PCB fabrication and assembly	~$60–100
Enclosures + bike mounts	3D printing / hardware	~$30–50
Spare components / PCB revision	Miscellaneous	~$40–70
Estimated Development Total		**~$230–400**

Evaluation and Testing

The project would be successful if the system can reliably detect approaching vehicles and warn the rider with enough time to react.

Testing would focus on:

- **Detection range:** Determine the maximum reliable distance for an approaching vehicle.
- **Distance accuracy:** Compare displayed distance with measured distance.
- **False alerts:** Test whether stationary roadside objects incorrectly trigger warnings.
- **Response time:** Measure how quickly the handlebar display updates after detection.
- **Wireless reliability:** Confirm communication remains stable during riding.
- **Battery life:** Measure operating time under normal use.

At the Design Expo, a moving cart or other large object could approach the rear sensor while the handlebar display shows the changing distance and warning level.

Market Need and Product Differentiation

There is already a demonstrated market for bicycle rear approach warning systems. Commercial products such as **Garmin Varia** show that cyclists are willing to use rear facing sensors to improve awareness of traffic approaching from behind. This further proves that the problem is real and that riders see value in additional rear traffic information while cycling.

RearGuard would focus on a simpler and potentially lower-cost version of this idea. Instead of depending on a smartphone or expensive cycling computer, it would aim to provide the core functionality in a standalone form: a rear mounted sensing unit and a small handlebar display that automatically warns the rider when something is approaching from behind.

Real Time Video Capture and Audio Feedback for Rock Climbing Training

A key skill in rock climbing is reading a route and finding a beta: a sequence of moves, techniques, and body positions that allows climbers to ascend and complete the route. Climbers frequently try different moves and techniques upon the same route, whether because the previous beta did not lead to successful completion, or because they want to test new approaches. Oftentimes, rock climbers use the native camera application on their phones to capture video recordings, study their movements, and learn from their mistakes. However, even with conscious effort, it may be difficult to remember the exact moves one has attempted, especially if the route is long or the climber has tried the route multiple times. This may lead to redundant effort if the climber is attempting the same approach without improvement.

Existing automatically adjusting video capture systems include tag tracking cameras[1], fixed point pan, tilt, and zoom cameras with tracking software[2], or downloadable mobile phone software which automatically center the phone camera on the user[3]. However, most market devices are currently designed for sports played on a horizontal field, such as football, hockey, or volleyball, and therefore have limited vertical panning. There are also interactive projector games such as ValoClimb[4] which project target areas upon a surface and use motion tracking to determine whether a user has reached the target, but these do not store recordings.

Among existing camera tracking systems, common limitations include their size, cost, and lack of specialization. Many systems require the camera to be placed far enough away for the entire range of action to be captured with no external mechanical device for the rotation of the camera. The software then performs automatic focusing and centering for video capture. For longer climbing routes, this may lead to lower resolution videos once the camera is zoomed in to the subject, especially if there is an angle change in the route, such as in an overhang. Additionally, although there are many systems for tracking subjects, they usually do not provide live audio feedback, especially for improving training within a rock climbing context.

Details

This document proposes a small floor mounted mechanical system to dynamically adjust the angle of a camera for higher resolution video capture of a subject, with live audio feedback over Bluetooth to alert climbers of repeated moves. By clamping a camera to a two axis mount with dedicated panning and tilting servos, and making the system battery powered, the device maintains an affordable and small footprint with ease of setup and portability. An objection detection algorithm like YOLO is run in real time on a Raspberry Pi (R-Pi), which sends processed camera data to the microprocessor to perform motor control and battery and power management via the PCB. The R-Pi also streams audio over Bluetooth to the climber's earbuds to provide unobtrusive feedback, with an option to upload recorded videos to a cloud storage site. The initial design will be implemented for bouldering only to reduce obstructive visual variables such as rope movement and belay partners.

---

[1] https://soloshot.com/collections/soloshot3
[2] https://www.veo.com/
[3] https://movtrak.app/
[4] https://www.valomotion.com/valoclimb

Schedule

The following represents an estimated schedule for the completion of a minimum viable product by the Design Expo with Milestone deliverables.

9/10-9/24	**Planning** Discuss and perform research to decide upon implementation strategies, including software databases and parts selection. Finalize intended features and stretch goals. Submit a draft proposal by 9/18 and use time between proposal submission and discussion to flesh out the timeline and note key concerns.
9/25-10/1	**Proposal Modification and Parts Ordering** Use feedback from proposal discussion to modify and finalize design, begin ordering parts and prototyping on breadboards. Submit formal proposal by 10/1.
10/2-10/15	**Prototyping and Initial Hardware Interfacing Demonstration** Complete device interface design in software. Ensure communication between all hardware, with demonstration such as passing messages wirelessly between the R-Pi and microprocessor, displaying servo position readings, and audio streaming from the R-Pi to earbuds. Begin PCB design with footprints completed. Submit Milestone 1 report by 10/15.
10/16-11/10	**Software Development and Prototype Testing** Order PCBs before 11/3. Develop tracking software, utilizing existing databases wherever possible for image processing. Test on hardware prototype before PCB order arrives. Solder and test PCBs, then assemble the system. Submit Milestone 2 report by 11/10.
11/11-11/23	**Software Development and MVP Testing** Complete software development and resolve all hardware issues with final components.
11/24-12/3	**Final Testing and Debugging** Resolve all system software issues and test the final system under multiple real applications.
12/4-12/11	**Design Expo Poster and Final Report Writing** Create and gather all expo material, including setup planning, videos, and a poster. Complete demo and final report draft by 12/10. Revise and submit final report by 12/11.

Budget

The following is a preliminary estimate for a bill of materials with the assumption of a total budget of $1250.

Item	Quantity	Individual Cost	Total Cost
Raspberry Pi	1	$45	$45
Camera	1	$50	$50
Servo	2	$32	$64
Tripod/Mount	1	$50	$50
Portable Battery	1	$60	$60
PCB	5 (minimum order quantity)	-	$30
Phone clamp	1	$30	$30
Wireless earbuds	1	$30	$30
Electronic Hardware	1	$150	$150
Miscellaneous	1	$100	$100
Total			$659

Implementation Issues

Current major concerns include:

1. Access to a climbing facility for testing
2. Safety of the test climber
3. Limited battery life
4. Concurrent image processing and audio streaming

Solutions:

1. Ideally, the system is small enough to be unobtrusive in a climbing environment, so climbing facilities such as Hadley or Planet Rock may allow testing during normal operating hours with clear communication with staff and patrons to avoid disrupting testing.

2. Test climbers will be thoroughly educated on climbing safety and will always be climbing well below their limit on the easiest rated routes.
3. This is a major concern because real time processing will make a low power design difficult. If budget allows, it may be possible to purchase a larger battery within portability constraints. Worst case, it may be necessary to use a continuous source of power such as a wall outlet.
4. It may be difficult to meet processing requirements on an R-Pi. An alternative approach is to develop an app for and utilize a mobile phone, which removes the need to purchase a camera and is individually rechargeable.

Evaluation and Testing

Testing and evaluation will involve both the video recording system and the audio feedback. For the video recording system, it will be useful to determine the percentage of time the climber remains within a predetermined area of pixels within the total image. The audio feedback can be tested by preparing and climbing several betas and determining the percentages of correct, false positive, and missed feedback as compared to the ground truth.

Robust Optical/RF Data Link Terminal for High Speed Transmission

Spacecraft, such as satellites, often acquire data faster than they can reliably transfer. This sometimes results in important data being lost due to storage limitations. One solution to increase throughput is laser communication, which enables improved data transfer speed, allowing the collection and transmission of more critical data[1]. However, laser communication beams are narrow, require precise alignment, and like other forms of communication, are susceptible to interference on the same wavelength.

Details

This document proposes a low cost IR LED and photodiode system which uses a tracking camera to control a mounted transmitter. Using information about received packet quality, the system adjusts the transmitter to point accurately at the receiver. In the case of poor optical data transmission, such as loss of communication, RF will be used to transmit compressed data that is algorithmically ranked in importance. Under normal conditions, RF communication will include telemetry reports such as packet quality. To make the system robust, there will be file checkpointing and error estimation. A Raspberry Pi on the transmitter side will capture, rank, and compress images, process the camera data, and calculate tilting. The motors will be controlled by a microcontroller, which also drives the IR LED. On the receiver side, a microcontroller is connected to the photodiode, checks CRCs, and measures signal quality, which is reported to a second Raspberry Pi that reassembles, logs, and displays received data. The Raspberry Pis and microcontrollers will sit on PCBs.

Schedule

An estimated schedule involves brainstorming and project research to determine feasible approaches and intended hardware until the draft proposal due date. Then revision and final parts selection and ordering, and submitting the final proposal. Then rotational mount prototyping and PCB design until the first milestone, at which there will be a demonstration of successful IR and RF communication, even if the automatic alignment does not yet work. Then software development and PCB and hardware assembly by the second milestone, with hopefully a minimum viable product demonstration. Finally, fine tuning software and debugging the system before completing testing and creating a poster and demonstration for the Design Expo, after which the final report will be submitted.

Budget

Item	IR LED	PCB	Photodiode	Camera	Mounts	R-Pis	RF	Micros	Misc	Total
Cost	$20	$60	$10	$60	$200	$90	$60	$30	$100	$630

[1] https://www.gov.uk/government/news/dstl-achieves-uks-first-optical-downlink-from-space

Implementation Issues

Current major concerns include:

1. Accurate alignment of IR LED
2. Data ranking algorithm
3. Different transmitter and receiver PCBs
4. Communication mode management

Solutions:

1. Requires a good control algorithm and precise hardware, but worst case can use RF transmission in place of IR communication.
2. The relative importance of data may be subjective, but the goal is consistency in ranking.
3. Developing two different PCBs may be a time consuming matter, but the receiver side should just be a simple power management board connecting the R-Pi and the microcontroller.
4. It may be difficult to determine which mode of communication to use and where to draw the line for poor communication on the IR side, but this could be solved by benchmarking the RF transmission rate and determining average, below which we choose RF over IR.

Evaluation and Testing

Testing and evaluation will involve ensuring smooth transition from IR to RF communication when optical communication is lost, packet quality for alignment accuracy, and preservation of important data in low transmission speed periods through ranking and compression.

Hotpot Guardian

A multi-diner cooking assistant that remembers what is in the broth and lifts it out on time.

High-level description

At a typical hotpot meal, several diners add thinly sliced meat, dumplings, vegetables, noodles, and seafood at different times. Once submerged, ingredients are difficult to identify, ownership becomes unclear, and diners must remember multiple cooking times while talking and eating. The result is familiar: food is forgotten, overcooked, lost at the bottom of the pot, or repeatedly disturbed to check whether it is ready. Hotpot Guardian is a removable ring that clamps around an ordinary hotpot and supports several color-coded stainless-steel cooking baskets. A diner loads a basket above the broth, selects a food preset or custom time, and presses one button. The system lowers that basket, tracks only the time spent at an adequate broth temperature, and raises it when the target is reached. The basket drains above the same pot while its matching light and tone notify the diner. Each basket therefore becomes a small, independent cooking lane without dividing the broth or replacing the restaurant's existing cooker.

Who would use or buy it?

The primary customers are hotpot restaurant operators seeking a memorable premium-table experience and more consistent cooking. A restaurant could offer the device at selected tables or as a paid upgrade. Secondary customers are frequent home hotpot users and social hosts. The customer is not buying another timer app; they are buying a physical system that prevents the timer from being ignored by removing the food from heat.

What the finished device looks like

The final device is a wipeable control hub behind the pot, with two to four detachable lift modules around the rim. Food-contact parts are commercially available stainless-steel baskets. Motors, electronics, and 12 V wiring remain outside the pot behind splash and heat shields. The Expo prototype will demonstrate two independently motorized baskets on a tabletop pot and will be usable without a phone.

Product operation and embedded architecture



The firmware will implement one safety-aware state machine per basket: IDLE -> LOADED -> LOWERING -> COOKING -> RAISING -> READY, with a FAULT state entered on a jam, missing temperature sensor, unexpected limit-switch state, or user emergency stop. A broth probe supplies real-time temperature. Rather than claiming to determine food safety, the product reports that the user-selected cooking target has been reached. The default algorithm pauses or slows the countdown when cold ingredients temporarily pull the broth below a configurable temperature threshold.

Why this is novel and worth building

Automatic-lift hotpots already validate that customers value easier food retrieval, but common consumer products lift one large basket for the entire pot. Commercial multi-basket cookers target back-of-house batch cooking. Hotpot Guardian combines four less common properties: it retrofits an existing pot, assigns an independent basket to each diner or food type, times each lane separately using measured broth temperature, and physically retrieves only the food that is ready. The novelty is therefore the multi-user interaction and closed-loop retrofit architecture, not the general idea of raising a basket.

Implementation issues

A five-person team can divide the system into mechanical lift design, custom PCB and power, sensing and controls, user interface/firmware, and enclosure/integration. Early development will use a development board and modular motor drivers. The custom PCB will then consolidate the processor, motor-driver interfaces, temperature front end, current sensing, limit-switch inputs, display and controls, buzzer/RGB outputs, emergency-stop input, and protected 12V / 5V power.

Proposed technical approach

- Processor-based PCB: an STM32- or ESP32-class MCU with sufficient timers, ADC/I2C/SPI/UART, watchdog support, and nonvolatile storage. Wireless connectivity is optional; all core functions remain local.
- Lift mechanics: low-speed geared DC motors or lead-screw steppers, upper/lower magnetic limit switches, and a self-locking or counterbalanced mechanism so a loss of power cannot drop a basket.
- Sensing: a food-safe stainless broth-temperature probe; motor-current sensing for obstruction detection; optional load sensing to prompt the diner after food is placed in a raised basket.
- Interface: one illuminated control per basket plus a shared display/encoder for presets. Color and basket position provide redundant identification; the buzzer is supplemental.
- Power: an enclosed, certified 12 V wall adapter. The project will not modify or switch the cooker's 120 V main circuit.

Six-stage schedule

Dates	Milestone	Description
Sep 10-23	Requirements and risk prototypes	Measure pot geometry and heat; validate one lift concept with cold water; freeze minimum viable features.
Sep 24-Oct 6	Bench prototype and parts	MCU reads temperature/limits, drives one motor, and executes the state machine; long-lead parts ordered.
Oct 7-25	Two-lane prototype and PCB	Two independent lifts run from development hardware; schematic/layout reviewed; PCB ordered by Oct.25.
Oct 26-Nov 8	Firmware during PCB lead time	Complete UI, presets, fault injection, logging, and automated cycle tests while the approximately 10-day PCB order is in flight.
Nov 9-25	PCB bring-up and integration	Assemble boards; verify rails/interfaces; integrate enclosure, thermal shield, drip management, and two lift modules.
Nov 26-Dec 11	Validation and Expo	Complete 100-cycle tests, thermal testing, user instructions, poster/demo, and contingency fixes; freeze by Dec 3.

Rough budget

Item	Estimate	Description
Custom PCBs and components	$180	Five boards including spares; MCU, regulation, interfaces, protection
Motors and lift mechanisms	$180	Four geared/stepper channels, couplers, rails, limit switches
Temperature/current/load sensing	$85	Multiple probes, current monitors, optional load-cell trial
Display, controls, lighting	$55	Display, encoder/buttons, RGB indicators, buzzer
Baskets and mechanical structure	$95	Food-safe baskets, clamps, fasteners, heat shielding
Power, connectors, wiring	$55	Certified low-voltage supply, fusing, splash-resistant connectors
Enclosure and spares	$100	Printed/machined parts, replacement components, contingency
Total	$750	Approximately $150 per student for a five-person team; below the $1,250 limit

Minimum viable product and stretch goals

The minimum viable product is two independently controlled baskets, one measured broth temperature, reliable automatic raise/lower motion, a complete local interface, and a custom processor PCB in a finished enclosure. If schedule pressure occurs, four lanes, automatic food detection, wireless ordering integration, and temperature-model calibration are removable without weakening the central demonstration. Stretch goals are a fourth basket, restaurant table-management integration, and a removable dishwasher-safe basket cartridge.

Reliability, safety and production issues

The largest technical risk will be reliable motion near steam, splashes, and varying pot rims. The mechanism will therefore be validated first, before the PCB is committed. A rigid prototype fixture will establish travel, load, motor torque, and clearances. The final clamp will accommodate a defined range of rim thicknesses rather than claiming universal compatibility. Drips fall vertically back into the pot, and removable food-contact parts separate from the electronics for washing.

Safety will be designed into both hardware and firmware. A physical emergency-stop button disables all motors. Upper and lower limit switches prevent overtravel; motor-current thresholds detect jams; firmware timeouts prevent continuous drive; fusing and reverse-polarity protection protect the low-voltage supply. Motion will be deliberately slow and low-force. The system will never automatically lower a basket while it is in the READY position unless the diner explicitly commands it. Electronics remain outside the steam path in a splash-resistant enclosure, and testing begins with unheated water before moving to heated-water trials under supervision.

No special laboratory is required. PCB assembly, 3D printing, laser cutting, and ordinary machine-shop access are sufficient. The team will use commercial stainless food baskets rather than attempting to certify printed parts for food contact. A real restaurant installation is not required for project success.

Evaluation and testing against real-world needs

Success is not simply that a motor moves. The project must show that multiple foods can be managed independently, that the basket is removed at the requested cooking target even when broth temperature changes, and that ordinary faults do not create uncontrolled motion.

Real-world need	Test method	Proposed success measure
Do not forget or overcook food	Run scripted hotpot sessions with two baskets and randomized target times; compare commanded and actual immersion intervals from logs.	At least 95% of 40 trials raised within +/-5 s of the temperature-qualified target; no basket missed.
Operate several baskets independently	Start overlapping recipes with different durations and temperatures.	Two simultaneous lanes complete in the correct order in 20/20 trials without state interference.
Reliable retrieval	Cycle each basket with representative 0-500 g loads using hot water and steam exposure.	At least 100 consecutive cycles per lane with >=98% successful positioning and no dropped load.
React safely to faults	Block motion, disconnect a probe, hold a limit switch, and press emergency stop.	Motor power removed within 250 ms of emergency stop; every injected fault enters FAULT and requires deliberate reset.
Temperature-aware timing	Introduce a repeatable cold-water/thermal-load disturbance and compare against a fixed timer.	Countdown pauses or compensates whenever temperature is below threshold; logged error in qualified cook time <2%.
Usable tabletop product	Give an unfamiliar evaluator a one-page instruction card and a scripted task.	Correctly start, cancel, and identify two baskets without developer intervention; setup in under 3 minutes.
Survive the environment	Operate for 60 minutes beside simmering water; inspect enclosure and log electronics temperature.	No reset, condensation ingress, or exposed surface/connector failure; electronics remain within component ratings.

Expo demonstration

Visitors will load two baskets with brightly colored reusable test pieces, select two different presets, and watch the baskets lower into a heated or visually simulated broth. A live display will show actual temperature, qualified remaining time, and each basket's state. One basket will rise first; the other will pause when the temperature is intentionally reduced and then resume.

Why you should choose this project

Hotpot Guardian combines mechanical design, embedded controls, custom PCB development, sensor integration, human-centered interaction, and rigorous testing in a product that people immediately understand. It has an interesting Expo demo and a credible paying customer, yet its minimum viable version is narrow enough to finish. Most importantly, Hotpot Guardian does not merely remind diners that food is ready; it physically prevents that food from continuing to overcook.

FoosBuddy

A clip-on robotic goalkeeper that turns an ordinary foosball table into an adaptive solo training partner.

High-level description

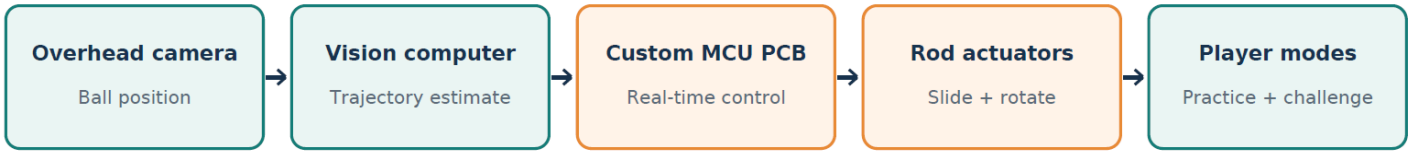
Foosball practice usually requires another person. Full autonomous tables have been built as research platforms, but they are costly and heavily modified. FoosBuddy narrows the idea into a product: a removable robot that controls only the opponent's goalkeeper rod on an ordinary table.

An overhead camera estimates the ball's position and velocity. A vision computer sends the desired interception point to a custom microcontroller PCB, which closes the rod-position loop, drives linear and rotational actuators, monitors current, and enforces safe travel limits. The player selects easy, medium, or hard mode locally.

Customers and value

Buyers include recreation centers, apartment amenity rooms, bars, youth clubs, and home owners. Venues gain an interactive attraction; players gain repeatable shooting practice. A clamp-on kit avoids purchasing or permanently modifying a table.

Embedded architecture



Novelty and scope

Autonomous foosball is not novel. The product contribution is the retrofit constraint and single-rod scope: FoosBuddy installs without drilling, calibrates rod travel automatically, stores difficulty profiles, and fails safely around human hands.

Implementation plan and feasibility

- Mechanics: toothed-belt linear stage plus a geared rotation coupling with a compliant clutch.
- Perception: Raspberry Pi-class computer and overhead camera; the custom PCB owns deterministic control and safety.
- PCB: MCU, encoders, motor-driver interfaces, current sensing, emergency stop, watchdog, and protected power.
- Minimum viable product: defend one half-field against straight and moderately angled shots. Offense and learning are stretch goals.

Schedule

Stage	Outcome
1. Requirements/mechanics	Measure table; prototype clamp, coupling, and one-axis motion.
2. Vision/motor bench tests	Track the ball and command a position on development hardware.
3. Goalkeeper and PCB	Close the loop; freeze safety; order PCB before late October.
4. PCB wait period	Build prediction, calibration, difficulty modes, and logging.
5. Bring-up/validation	Integrate PCB, enclosure, emergency stop, and compliant coupling.
6. Expo preparation	Measure block rate/latency; finish the challenge demo.

Budget, risks, and evaluation

Item	Estimate
Camera and vision computer	$140
Linear and rotational actuation	$220
Custom PCB, components, and spares	$175
Encoders, current sensors, limits, emergency stop	$85
Mechanical frame, clamps, guards, and enclosure	$140
Power and contingency	$90
Total	**$850**

The primary risks are vision latency, fast impacts, backlash, and pinch hazards. Scope control is the main mitigation: only the goalkeeper rod is automated, the camera views a limited field, and the mechanism prioritizes blocking over human-level play. A belt stage and encoder provide fast, measurable translation; a compliant rotational coupling protects the rod and hands. Guards cover the belt and couplings, motor current and velocity are limited, an emergency stop removes motor drive, and the robot automatically disables when its camera or position feedback is invalid.

Real-world success measures

Real-world need	Test method	Proposed success measure
See the ball quickly	Compare reported positions with labeled high-speed video across lighting conditions.	>=95% detection in the defensive region; median end-to-end latency <80 ms.
Reach the predicted path	Command 100 random rod positions and compare encoder result.	Position error <8 mm; no overtravel; settle within 150 ms for typical moves.
Provide useful practice	Use a repeatable launcher/ramp for straight and angled shots at several speeds.	Block >=70% of in-range straight shots and >=50% of the combined test set.
Adapt difficulty	Repeat the same shot set in three modes.	Monotonic and reproducible difference in block rate or reaction delay between modes.
Install as a product	Remove and reinstall on the target table using written instructions.	Calibration and setup in <10 minutes with no permanent table modification.
Remain safe and reliable	Run 500 defensive cycles; obstruct motion and press emergency stop.	No mechanical loosening; drive disabled within 250 ms; every obstruction produces a recoverable fault.

Expo demonstration

A visitor chooses a difficulty and takes five shots against the robotic goalkeeper. The display reports camera-to-actuator latency, predicted interception point, and score. A calibration sequence at startup demonstrates that the product can discover its own travel limits. The demo is immediately understandable, visually exciting, and technically rich even if the robot controls only one rod.

Low-Cost Bed-Turning Attachment

Bedsores (pressure ulcers) remain a severe health issue for bedridden patients in hospitals as well as homes. They are also a major financial burden, costing hospitals and healthcare facilities billions and requiring close attention and physical labor from nurses to manually rotate patients. While automated turning beds do exist, they are very costly ($5,000 - $15,000), making them inaccessible to at-home patients or underfunded hospitals.

Note: This device is only for dummy testing and is not suitable for medical tests or live patients.

This project aims to build an affordable, universal mattress attachment system that gently lifts the test dummy to their side. Instead of replacing an entire mattress, this thin device can be placed under a mattress topper with inflatable side chambers to lift a test dummy laterally. The system will also include a pressure sensor that would determine if the dummy's weight was actually shifted. The system also comes with a UI interface on the personal device to control the lifting, see real-time pressure point mapping, and view past and current patient data.

Details

The project will start with two long inflatable chambers placed on one side of the dummy. When one inflates, it tilts the dummy to one side, alleviating pressure. A small electric air pump will supply air to the system. Valves would control which chamber inflates, which chamber deflates, as well as making sure the chambers do not exceed their maximum pressure.

Pressure sensors will measure how the dummy's weight is distributed. A tilt sensor would be attached in an area underneath the torso of the dummy to estimate the turning angle.

A UI interface will allow the user to manually or automatically turn the patient, see a pressure point map, and see logs of all completed turns, pressure alleviation events, and alerts.

Schedule

The dates are rough estimates and will not portray an exact step-by-step of production and planning.

Dates	Milestones
9/10-9/23	Planning Finalize system architecture: brainstorm safety pressure thresholds, inflation/deflation techniques. Research microcontrollers best suited for the project, considering usability, cost, and power. Research valves, air pumps, and sensors. Submit Draft proposal on Sept 18.
9/24-10/6	Design Select and order test-bench parts. Select the processor platform and procure items for basic functionality (LED status light, controlling solenoid drivers). Procure a dummy, mattress, mattress topper.
10/7-10/25	Breadboard prototype and PCB Design Assemble a breadboard prototype that can inflate and deflate an air chamber and record and log live pressure values. Design a board layout for a custom PCB. Order PCB before the course cutoff.
10/26-11/10	Software and System Development Develop the rotation state machine, pressure map generation, and UI layout. Solder components to the PCB. Construct the physical hardware system.

11/11-11/25	Software and System Testing Test with the dummy and adjust any inconsistencies on the software (angle adjustments, software bugs) and hardware side (Power leaks, pressure sensor adjustments). Any adjustments, pivots, or additional features will be handled at this time. Research and add any additional features.
11/26-12/9	Final Testing, Poster, Features Target completed and working system by 12/3. Do rigorous testing. Design and print the presentation poster. Prep live dummy demo.
12/10-12/11	Expo and Final Report

Budget

The chart below contains rough estimates of potential parts that are needed for assembly. Multiple microcontrollers and PCBs are ordered in case we damage previous ones, since ordering and shipping them takes a while. The cheapest components are listed below, such as a dummy, which we can use as a training mannequin filled with water or sand. If leftover budget is available, a nicer dummy could potentially be used.

What	Quantity	Cost Per	Cost Total
Microntrollers & ICs	5	$6	$30
PCBs	5	$15	$75

Pump & Solenoid Valves	1	$120	$120
Inflatable Chambers & Tubing	2	$45	$90
Pressure Sensors	1	$75	$75
Misc Parts	1	$60	$60
Testing Dummy, mattress, mattress topper	1	$100	$100
Shipping	1	$50	$50
Total			$600

Issues

The largest concerns are:

1. Air Leakage & pressure issues under Load
2. Pressure sensors not working under heavy loads for long periods of time.
3. Sampling many sensors may cause latency.

The first issue may require brute force testing and ordering different tubes, air chambers, and pumps. Another mitigation is to use sealant or automated pumps of air if the pressure drops below the threshold. The second issue can be mitigated by software-based optimizations. The third issue can also potentially be mitigated by software implementations.

Stretch Goals

Adding heat and humidity sensors as well to detect if a potential patient is too hot or sweaty.

Testing

Testing will evaluate whether the prototype can meet all the specifications listed in the description. Below are some listed tests:

- Program a sequence of automated lifts at different angles every 10 minutes. Verify the logged data is correct. Use a 150 lb test dummy.
- Position pressure sensors in the most critical points of the body. Verify there is a significant pressure difference when lifted and unlifted.
- Complete autonomous cycles without any faults: air pressure loss, unexpected shutdowns, or wrong data.

Shrimp-No-More Backpack

Many students carry around 20lb backpacks, topped with laptops, iPads, notebooks, and energy drinks. This heavy load worsens the already arched engineering postures. This backpack will gently correct the student's posture and alleviate the weight while sprinting to an 8 am class. The backpack will also stream live posture metrics to a personal mobile device to track your progress.

Details

A chest-strap IMU detects slouching by measuring the angle of the body over a set amount of seconds. A 2-axis motorized carriage with a rigid backplate sits inside the backpack. When contents sit far horizontally from your back, your body compensates by slouching forward. So, axis 1 pulls the backpack contents inwards to sit against the backplate, reducing the horizontal distance from the contents from your back. Axis 2 lifts the contents for running or walking and lowers when sitting or slouching. This prevents any sagging and allows the contents to sit comfortably on the center of your back.

The backpack is also equipped to dampen the load when running. A high-speed actuator shifts the weight out-of-phase with foot strikes. This helps lessen the vertical bounce of a backpack when running or speed-walking. In addition, the backpack will also feel lighter.

The backpack can connect via BLE/Wi-Fi to display data, such as strap force, count steps, and render GPS commute maps, while highlighting bad posture on the route.

Issues

The largest concerns include:

1. Sudden ducking, jumping, or carrying the bag can cause a false posture to trigger.
2. GPS and wireless streaming can drain the battery very fast.

The first issue can be mitigated by a digital filter, enforcing a strict 3-second threshold before firing the motors, and detecting if the backpack is on the back of the student. The second issue can be resolved by putting the GPS into sleep during lectures and only waking up when the backpack is on the move.

SMART KNEE BRACE

Overview

ACL and meniscus reconstructions require a grueling, year-long recovery. Post-surgery, patients face three major risks: permanent loss of range of motion due to scar tissue, severe muscle atrophy, and re-injury from falling due to impaired gait. Physical therapists measure these metrics manually in-person, but at home, patients are blind to their quantitative and qualitative progress at home.

This project delivers a multi-node, wearable embedded network that provides real-time kinematic tracking, guided physical therapy, and active fall prevention.

Although most knee braces are purely mechanical, there are a couple of "smart" knee braces available on the market. One of the most popular ones is from Sensoria Health. It is an advanced rehabilitation device designed for those with post op knee surgeries. However, it comes at a very expensive price at $598.00, and it is built into a specific knee brace. The goal of this system is to be inexpensive and also to not have to rely on a specific mechanical design; the system should just be an embedded system that can attach to any brace a patient has, and it should not require the mechanical design of a specific kind of brace.

Details

Our team will build a multi-node wearable embedded system that provides real-time, quantitative feedback on leg kinematics. The system consists of four custom designed PCB nodes:

1. Knee Sensor Node: Attached to the post-op brace.
2. Wristband GUI: A receiver with an OLED interface and haptic feedback.
3. Wireless Shoe Pods (One for each Shoe): Ultra-low power PCBs placed under the insoles.

Technical aspects: The injured leg sensor node uses a 6-axis IMU on each leg sensor node to calculate exact knee flexion/extension. AN FSR in each shoe calculates weight distribution to detect if the patient is subconsciously overloading the knee. This data is transmitted via a transceiver to the knee sensor node for data logging. Pertinent data then gets sent over to the wristband GUI via wireless.

The system operates via three distinct modes:

1. Passive/Measurement Mode: The system monitors gait of the patient, and logs flexion and extension of the knee when measurement is enabled. By having a measurement mode, patients can monitor how their flexion and extension is improving over time, which is vital to ensure scar tissue does not develop, a common pitfall of ACL post-op surgery.
2. The GUI dictates a physical therapy routine (e.g. squats). The system tracks form in real-time, vibrating the wristband upon successful rep completion.

3. Danger/Correction Mode: If the knee node's IMU detects severe gait impairment (e.g. trip), the system triggers localized haptic motors on the wristband, warning the user to re-establish their gait. Furthermore, one common condition that ACL patients going through physical therapy experience is "valgus collapse," a condition where the knee calves inward; this can especially happen when muscle fatigue sets. If the knee node detects this, it should drive a haptic vibration motor on the brace, which should cause a neuromuscular reflex and realign the patient's knee.

Schedule

The dates selected are rough estimates, but they provide a good guideline/roadmap:

9/10-9/23	*Planning* Research what transceiver to use for shoe nodes, must be small and lightweight. Figure out CPU or at least a family. Write draft proposal by 9/17.
9/24-10/6	*Design and Part Selection* Look for potential dev/board kit for processor (system though has to be small to fit, this is just for prototyping), get some extremely basic functionality working like a blinking LED, Submit final proposal by 10/1. Finalize parts and order all we thing we need by 10/6.
10/7-10/25	*Breadboard Prototype and PCB design* Have a prototype of leg node working with basic functionality (IMU working at a basic level, can send/receive wireless messages, haptic motors can function when prompted). Get OLED screen to work properly on wristband GUI. Order PCB's by 10/26.
10/26-11/10	*Software Development/More Advanced Testing* While waiting for PCBs, finalize software design; Make sure FSM for the three states is working, verify that each node can communicate with each other. Get PCB's assembled by 11/10.
11/11-11/25	*PCB-Based Testing* Run all advanced tests on PCB-based system. Verify communication works. Verify data logging works and is accurate. Verify danger mode works and that it is safe and working as intended. Fix any bugs/issue.
11/26-12/9	*Poster and Expo* Get poster finished, prepare presentation/demo, fix any remaining bugs
12/10-12/11	*Final Report* Complete Final Report

Budget

Even though a realistic budget is hard to come up with right now, this budget proposal should be good enough at this stage. This budget assumes that things will break, so it includes spares for each part.

What	Quantity	Cost	Total
STM32L4 Microcontroller	6	$8.00	$48.00
XBee RF Modules	3	$25.00	$75.00
Transceiver	5	$3.00	$15.00
6-axis IMU	2	$10.00	$20.00
Force-Sensitive Resistors	4	$8.00	$32.00
SPI OLED Display	2	$12.00	$24.00
Power/Charging IC's	8	$2.00	$16.00
PCB's	6	$20.00	$120.00
Knee Brace	1	$35.00	$35.00
Haptic Motor Drivers	4	$5.00	$20.00
Shipping	1	$75.00	$75.00
Misc.	1	$100.00	$100.00
Total			$580.00

Issues

The largest issues at this time are:

1. Developing algorithms to accurately measure flexion, extension, and gait,
2. Battery Concerns
3. Wearability

Issue seems to be addressable with enough time designing/debugging. As for issue 2, running an OLED screen and RF transmissions will drain battery quickly; having the Passive state act as a sleep mode state can hopefully resolve this concern. As for the 3rd issue, extended wear can shift the IMU on the brace, leading to incorrect calculations. A highly secured, non-slip 3D strap needs to be designed.

Evaluation/Testing

To prove that the project meets real-world client needs, we will validate the system against medical baselines:

1. Kinematic Accuracy: We will use a goniometer, a physical therapy device used to measure the angle of flexion in a joint, to compare its output on the leg node to what is calculated by the system. Success is considered within +- 3 degrees margin of error.
2. Latency Testing: To prevent a fall, the alert mut be immediate. Success is achieving a total wireless system latency of 50ms or less.

3. Weight Distribution Validation: A test subject will stand on two calibrated scales. The test subject will intentionally apply more weight to one scale, and if the system accurately outputs the correct weight ratio compared to the output of the digital scales, it is a success.

AERO-SEED: Autonomous precision seeding system

Overview

Farmers face a difficult challenge when planting cover crops in the fall. Cover crops, such as cereal rye and clover, are planted to protect and improve soil during periods when the primary cash crop is not growing. However, planting cover crops after the fall harvest can be difficult because the soil may already be too cold or frozen for sufficient germination and root establishment. Planting cover crops earlier with a tractor can also be impractical because the existing cash crop is still occupying the field and may be damaged during planting.

Aero-Seed aims to eliminate this tradeoff by moving the cover-crop planter from the ground to the air. AERO-SEED is an autonomous, drone-mounted payload system designed to broadcast cover-crop seeds over an agricultural field while the primary crop is still growing. By distributing seeds before harvest, the cover crop can begin establishing while soil temperatures are still favorable, potentially providing several additional weeks or even months of growing time before winter.

Details

AERO-SEED will consist of a seed storage container, an electronically controlled dispensing mechanism, a GPS module, and an embedded controller mounted to a test drone. The seed container will use a funnel-shaped design to direct seeds toward a mechanically controlled trapdoor. The embedded system will adjust the trapdoor opening to control the rate at which seeds are released.

The system will use real-time GPS data to estimate the drone's ground speed and dynamically adjust the seed-release rate. At the desired ground speed, the system will maintain a predetermined seed density. If the drone slows down, such as when encountering a headwind, the controller will reduce the seed-release rate. Conversely, if the drone speeds up, the controller can increase the release rate. This closed-loop control approach is intended to maintain a relatively uniform distribution of seeds across the field despite changes in drone speed.

The drone will follow an autonomous lawnmower-style grid pattern to systematically cover the target area. The embedded system will also monitor system status and communicate relevant telemetry data. A safety or "danger mode" will be incorporated to disable or restrict seed dispensing when abnormal operating conditions are detected.

Schedule

The dates selected are rough estimates, but they provide a good guideline/roadmap:

9/10-9/23	*Planning* MCU selection, system planning, developing mechanical design.
9/24-10/6	*Design and Part Selection* Finalize MCU, GPS module, sensors, mechanical components, submit final proposal.
10/7-10/25	*Breadboard Prototype and PCB design* Develop breadboard prototype and begin PCB design. GPS system and PID closed-loop dispensing controller will be tested using simplified mechanical system first.
10/26-11/10	*Software Development/More Advanced Testing* PCB-software testing, advanced drone testing-dispensing tests.

11/11- 11/25	*PCB-Based Testing* Final PCB-based tests. Focus on GPS accuracy, seed distribution, data logging, and safety functionality.
11/26- 12/9	*Poster and Expo* Get poster finished, prepare presentation/demo, fix any remaining bugs
12/10- 12/11	*Final Report* Complete Final Report

Budget

What	Quantity	Price	Total
Test Drone	1	$250	$250
PCB's	2	$60	$120
Sensors & Telemetry	6	$20	$120
Mechanical System	1	$80	$80
Total			$570

StreetPass Pocket Walker to Combat the Loneliness Epidemic Among Gen Z

Details

Gen Z is the most lonely generation of them all, resulting in the phrase “loneliness epidemic” popularizing as a descriptor of social life among Gen Zers. A 2025 Cigna Group report cited 67% of Gen Zers as being lonely, as compared to 65% of millennials, who also grew up with digital technologies. Furthermore, according to a World Health Organization 2025 report, 1 in 5 teenagers aged 13 to 17 experience high rates of loneliness, with 40% of American high school students reporting “persistent feelings of sadness and hopelessness” in 2023. Psychologists report that an increased dependence on technology like social media and phone usage, compounded with the COVID-19 pandemic are what have led to more students spending time on their phones than with friends, or avoiding in-person extracurriculars, among other negative symptoms. As technology and other factors influence young minds, including many UofM students, into becoming more socially isolated and alone, solutions must arise urgently that can help target this intimacy crisis.

One potential solution is a cheap embedded systems device that can motivate students to get together in person. Since embedded devices are generally designed to be not “smart” or as costly as devices like phones and computers, if an embedded device were programmed to assist in socialization the same way social media applications on smart devices promised to, it wouldn’t be abused as easily like social media as a way to replace social interactions, and instead could be used to genuinely enhance socialization, being inherently less dopaminergic to carefully curated social media algorithms.

We propose a small, embedded device that can be put in one’s pockets or bags, that, whenever it passes by someone else holding a same replica device on the street, will notify both users that each person passed by using wireless communication. Each device will then send the other user their consented data, which includes gender, age, hobbies, goals, answers to personal questions, etc. (similar to a dating app), at which point both people can decide whether they would like to meet one another depending on how interested they are in the other user’s data. If both agree, the devices will direct both users to a nearby location using a GPS system in order for both users to finally meet and interact with one another. This is inspired by the StreetPass system of the handheld console the Nintendo 3DS, in which 2 3DS holders could pass by each other on the street and exchange game/user data with one another wirelessly. This device will hopefully utilize the serendipity and intimacy of the StreetPass system in order to drive social interaction. This device will be targeted for Gen Z, and more locally, UofM students for now. Naturally, more complex and varied features for the device are also in consideration, such as customization, planning to meet someone at a future date, etc.

Schedule

These dates are WIP and very loosely defined.

9/10-9/23	Planning Use around a couple of weeks to discuss what are the features we would like the device to finally have. Based on the complexity, power consumption, etc. of the features, decide on what kind of communication protocols to use, what kind of MCUs we would like, what kind of display the device needs
9/24 - 10/6	Design and part selection At this time, we will finalize the processor and other parts, then build 2 small minimal prototypes (whenever the two devices pass by each other, LED blinks. Also have screen display something). Final proposal submitted by 10/1, order parts by the 10/4
10/7 - 10/30	Breadboard prototype, PCB design Prototype our design (Sends user data to other device, sends users to a nearby, defined location, also have 3 or 4 devices in proximity to test that connection is stable), design the PCB and order by the 10/28 (before halloween at least).
10/31-11/13	Software development, test on prototype While waiting for PCBs, get all the software features we decided on to work completely. Assemble PCBs by 8th-10th or whenever they arrive.
11/14 - 11/25	Software development, test on PCB-based device Test PCB, including proximity of device communication, making sure multiple devices don't interfere with one another. Revise PCB if necessary. If time allows, add more features, more customizability for user, etc.
11/26-12/9	Final testing, poster, expo Aim to finish project by 12/3, 12/6 at the latest if any bugs remain. Get poster done and report mostly done.
12/10-End of semester	Finish final report, expo

Budget

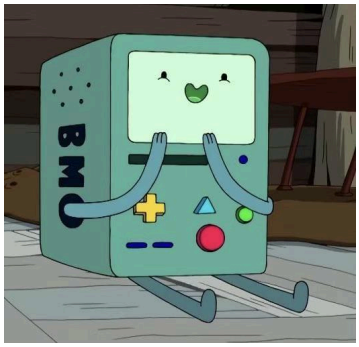
If the total budget amongst 5 people is 1250$, and we want to make at least 4-5 of the device for testing and demonstration, then we make sure each device is at most 250-300$. Assuming each MCU is around 20$, the PCBs are 10x15=150$ (accounting for the 1st iteration having errors and needing to redesign), other parts are around 200$, and shipping is roughly 100$, then in total we only use 550$. Even accounting for the fact that this is an extremely loose estimation, we have a 700$ safety net, so I believe we are safe on the budget side of things.

Issues

1. Buying material to create multiple devices
2. Wireless communication/ limiting interference between multiple devices
3. Utilizing GPS for location decision
4. Making sure device is big enough for features but small enough for bag
5. Making the device appealing enough for people to theoretically use/buy

Not sure if issue 1 is a concern or not without having full scope of what the device will do and how much each part will cost, but we do have to make sure we do not break many parts. Regardless, it is something that can be worked around. Issue 2 is something that needs extensive design, iteration, and testing, but can hopefully be programmed around. Issue 3 is hard because of how it may drive up the cost and difficulty interfacing, and we may end up cutting it altogether if it comes to it. Issue 4 is something that will be addressed at the beginning, but hopefully if the device is roughly the size of a phone or a Nintendo Gameboy, it should suffice for most users. Issue 5 is the most abstract, but we are assuming that people will naturally use it if they already have the motivation to “buy” it, and hopefully we create enough features as well as personal customization for the device in order to make it seem appealing enough for people. Gadgets and consumerism are generally appealing regardless, so as long as we make the design of the device attractive enough, it should definitely be appealing in a theoretical real-world market.

BMO from Adventure Time



Few cartoon robot characters both meet the criteria of extreme recognizability and fame as well as being a form factor and having features that are ideal for an embedded systems class such as BMO from the show Adventure Time. He is a small, gameboy-like robot that has numerous features ranging from being a small game console, to being a video player, to generally talking to people and having his own personality. The general idea is simply a desk robot character with multiple features like an 8-bit game, interactivity, etc. for amusement and companionship.

There are multiple features you could consider for an implementation of BMO. One would include designing a separate, Atari-like controller that players can use to play pong on his screen with. You could also include a camera and have BMO print out from his slit the photo he took. He could also use a sensor to turn on whenever someone is in the vicinity. With multiple people and a large budget, you can consider having him respond to voice commands, and integrating AI into his design as an AI chatbot. Overall, there are multiple possibilities and features that can be added.

A timeline could very roughly be 2 weeks of general planning, a month of building/PCB design, about a month of PCB testing and software iteration, and then a few weeks of touch ups.

There are definitely multiple issues to consider as well. You definitely need to be aware of how much time and money certain features would take to do, and plan accordingly in consideration of those constraints. The necessity of the project is also somewhat in question. The features you propose should be attractive enough to warrant someone to want this desk robot in a real world scenario. The “real world impact” argument would stem from a consumerist perspective, and embedded system’s application to toys and gadgets.

Margin: Open Source E-Reader with Writing

On May 20, 2026 , Amazon ended support for every kindle released in 2012 or earlier, despite the devices still functioning as intended. Users can no longer buy, borrow, or download a book and if you try to factory reset the device , you will end up bricking it.

Margin is a 6” electronic-paper reader with a capacitive touch screen, an on screen keyboard for typing notes directly onto a page, and wifi sync to your computer. The notes export as plain Markdown files over USB. Data remains private to the user and companies can’t decide when you are done with the device.

Details

Key differentiator - typing on E-ink with low latency. Typing on an e-reader is usually laggy since the display cannot keep up with the keyboard. My proposed target is under 100 ms from key press to feedback. We can do this by refreshing only the small rectangle around the pressed key and text cursor. A stretch goal can be applying this same logic to allow the user to pinch and zoom in on the screen.

There are cheap 6” E-ink panels (~$25) sold as replacement screens which are bare active-matrix films that connect over a 33-pin ribbon cable. They do not have a controller, memory, or a way to automatically draw the bitmap onto the screen. The panel needs someone to drive the row/column lines.

There are 2 difficult firmware aspects to this project:

1. DMA driven timing controller - Pixel data is sent out over a parallel bus on a strict schedule, fed by DMA so the CPU can lay out the text while the panel is being driven
2. A waveform scheduler - E ink pixels only move when you push them through a sequence of “voltage frames” and it is dependent on the target shade and current pixel state. Our scheduler will track which screen regions are “dirty” and based on that, perform a fast update in a specific region, a slow 16 level grayscale pass or an entire page update.

The wave form scheduler is what will make the product UX feel good when the reader is zooming in, typing, or turning a page.

System Components

Text engine on device - Epub parsing, layout, and pagination will run on the microcontroller. We can use existing libraries to help with this and restrict the scope to 1 font family. If this proves to be too difficult to be done on device we can move this over to software and transmit the “finished” text product so the device can focus on just displaying the text and managing page updates.

Touch - We’ll use a 6” capacitive touch panel with a GT911 controller, bonded ontop of the E-ink film. This is a documented part that uses I2C.

Processor - ESP32-S3-WROOM-1-N16R8 module has a an antenna for transmitting data between devices, 8MB of PSRAM which can hold a page comfortably.

Wi-Fi sync - The device can connect to Wifi and accept EPUBs over HTTP upload from a laptop and notes are exported the same way.

Power - Another goal is for the device to have strong battery life for extended use. We can buy a switching power converter and integrate it to supply the 22V necessary to drive the display.

Schedule

Dates	Milestones
9/10-9/22	Meet with team, discuss the design and the parts that we want to use. Ensure everything is technically possible and then order parts. Draft proposal 9/18
9/23 - 10/6	Final proposal 10/1. Drive the bare panel from a devkit and the purchased supply and get a recognizable image on screen by 10/6. In parallel figure out EPUB unzip and text parsing on the devkit

10/7 - 10/26	Breadboard system and PCB design. Get waveform scheduler with dirty rectangle updates working. GT911 over I2C; on screen keyboard. Schematic should be done by 10/15 and layout should be done by 10/22 so we can order by 10/26
10/27 - 11/9	Work on software while waiting on the PCB. Keystroke latency, reflow and pagination, WiFi upload, notes storage. We can also work on the enclosure.
11/10 - 11/24	Move everything to the PCB and verify all the currents are working as intended and order a new PCB by 11/14 if need be. Rail power should be working.
11/25 - 12/10	Everything should be working as intended and if not cut features. Work on poster around 12/5 and finalize the product.

Budget

The budget is pretty rough. I am not entirely sure how many of each part we will be needing to build the device or if these will end up being the parts that we actually end up using. Additionally, since we are working with high voltages there is a good chance parts will break and will need to be replaced. However, this shouldnt be an issue since there is around more than enought money left over given a group of 5 people.

Item	Quantity	Price	Total
6" E ink panel (ED060XC3/SC4)	3	$28.00	$84.00
6" capacitive Touch panel (GT911)	2	$18.00	$36.00
High Voltage Power Supply (epdiy / TPS65185)	2	$30.00	$60.00
ESP32-S3-WROOM-1-N16R8 modules	2	$7.00	$14.00
PCB	2	$15.00	$30.00
Passives, LDO, charger, microSD, switches	1	$50.00	$50.00
2000 mAh Battery	3	$10.00	$30.00
Enclosure	1	$30.00	$30.00
Shipping	1	$40.00	$40.00
			$374.00

Implementation Issues

1. Panel bring-up - Driving the updates to the screen will be difficult since we will be interfacing with raw pins and pixels. Depending on the component that we decide to use there may not be much documentation that will help us so it will be a key part of the project we need to get working early.
2. Text - Handling a large amount of text modification directly on the processor may prove to be very difficult. There are libraries that can help with this, though.
3. Working with the high voltage power supply and driving different voltages to the various components and panels will require a lot of time to ensure we wont fry any of the components. Parts are generally well documented so this shouldn't bee too big of an issue.
4. There should be limited work needed to be done on the mechanical side- just the enclosure for all of the parts.

Proposed work split

1. Timing controller and waveform scheduler
2. Text engine - EPUB, fonts, layout, reflow, pagination
3. Touch, gestures, on-screen keyboard, latency
4. Wi-Fi sync, notes storage and export, board integration
5. Power, battery life, enclosure

Evaluation and Testing

The over arching theme for evaluation is that an outsider should be able to pick up the device and think it is a Kindle.

1. Typing that "feels" good; not laggy, responsive to touch and ideally sub 100ms latency.
2. Device should be able to fit in a pocket
3. Device should be able to handle zoom in and out and the text resizing that comes with it
4. Device should be able to receive a variety of books and be able to export notes

Eternal* Coin Flipping Robot

A robot hand sits in a glass enclosure on the side of the EECS atrium. It flips a coin, reads the result, records it, and flips it again. It does this continuously until it records 10 straight heads. Then it stop permanently and displays the final count. There is no reset button.

The probability of flipping 10 heads in a row with a fair coin is 1 in 1,024. People generally cant comprehend very large or very small numbers- what they will understand is a robot chugging away day and night flipping a coin. They can walk in monday morning, see the robot and think nothing of it. They leave for the day and still see the robot flipping the coin. They walk in the next day and see the robot still flipping the same coin. Then they walk in later in the week and the robot is off. They see 1842 and appreciate the (un)likelihood of 10 heads in a row.

Details

The robot will have to have a way to flip the coin and record the result. It doesnt have to be a hand. it could instead just be a platform that springs the coin up and a computer vision camera looks down at the coin and records the result. The outside of the enclosure could a have a hex display that increments the count each flip. The flips can occur once every 15 seconds.

A key factor in building this is ensuring mechanical reliability and consistency. Each component that goes into the flipping mechanism must be able to handle thousands of flips and ensure that each flip is somewhat identical in terms of where the coin lands and how it is flipped.

Evaluation

Testing to see if the project works is fairly simple- 2 heads in a row and the machine is “bricked”. Another metric we can use is actually putting the robot out on display and see it it garners any interest from people walking by since the concept of this project is for it to be an engineering art exhibit. We can also create a live feed and website for people to tune into at any given point to watch the robot.

Bio-Adaptive Cybernetic Wrist Orthosis for Carpal Tunnel Prevention

Carpal tunnel syndrome (CTS) is the set of symptoms and signs characteristic of compressive median neuropathy at the carpal tunnel of the wrist.[a]. To help better understand, symptoms of CTS can feel like severe pain in the wrist, hand, and fingers. These symptoms can impact everyday activities such as writing and typing, something that young engineers must engage in daily. In fact, constant use of a keyboard could lead to carpal tunnel syndrome itself.[b] To help individuals diagnosed with wrist problems, wearing simple monitoring and haptic intervention schemes can prevent unintentional hand and wrist overexertion. For example, if a student is writing a paper late at night and unconsciously lets their wrists adjust into extreme extension, they can create severe, continuous strain on their nerves. If an active, bio-adaptive glove alerted the user via subtle physical feedback when strain thresholds were exceeded, the user would naturally correct their posture and avoid long-term nerve degradation.

[a] https://en.wikipedia.org/wiki/Carpal_tunnel_syndrome
[b] https://pmc.ncbi.nlm.nih.gov/articles/PMC2649727/

As a different application, if a wearable device tracks real-time muscle fatigue metrics using bio-signals, it can predict posture breakdown *before* those flashes of pain actually begin. That would allow users to adjust their typing habits or take short breaks, reducing cumulative wrist strain (as well as lost productivity time).

Ideas like this have been proposed many times. Commercial wrist braces are widely available, but rigid splints immobilize the joint, making active typing nearly impossible[1]. Medical-grade feedback systems likely exist in physical therapy clinics, but their high cost and bulky desktop consoles prevent everyday use. So, a low-cost, unrestrictive, and bio-adaptive wearable system is needed.

Details

As a user types throughout the day, the glove would ideally monitor joint motion and posture. Rather than relying on mechanical locking, the wearable could explore using motion tracking sensors (such as low-cost IMUs) to estimate wrist angles, combined with basic bio-signal monitoring to calculate muscle strain. A low-power microcontroller would process sensor inputs to detect poor typing form. When strain thresholds are exceeded, the device could deliver simple physical feedback or gentle resistance to suggest the hand back toward an ideal position.

This proposal includes building a pair of wearable glove prototypes linked to a controller interface. The team will investigate dual IMUs, basic analog bio-signal conditioning, a micro-haptic feedback motor, and a small servo-assisted tensioner to go along with it. Power management would be handled via a rechargeable LiPo power circuit integrated into a wearable wrist strap.

Important implementation questions to remember involve evaluating how clean a signal one can get from skin-surface sensors during active typing, as well as managing motion drift between hand and arm sensors. Sensor noise will need to be handled through appropriate

[1] https://www.orthoinfo.org/en/diseases--conditions/carpal-tunnel-syndrome/

filtering and tuning, while tracking drift can hopefully be handled through software updates on the microcontroller.

Schedule

The dates selected are mostly "lines in the sand" and will almost certainly not be exact though they should be close.

9/10-9/23	*Planning* In the two weeks after project groups form, brainstorm potential sensor approaches and compare motion-tracking platforms vs. simple flex sensors. Research two or three 6-DOF IMUs and evaluate them based on cost, ease of integration, and current draw. Look into basic options for haptic feedback actuators and identify a suitable microcontroller family. Write the draft proposal by 9/18.
9/24-10/6	*Design and part selection* Order sample dev kits, two candidate IMUs, and basic haptic modules to experiment with in the lab. Try bringing up a bare-bones prototype on a breakout board (blinking status LEDs, reading raw angle data, and triggering a basic motor driver). Submit final proposal by 10/1. Ideally, we aim to order all initial prototype parts by the 4^{th}.
10/7-10/25	*Breadboard prototype and PCB design* Test whether we can reliably calculate relative wrist movement using two sensors on a breadboard. Explore simple filtering techniques to handle noise during fast typing or other realistic scenarios. Begin drafting initial PCB schematics for the main controller and a lightweight sensor connection strategy. Submit PCB files for manufacturing by the 26^{th}.
10/26-11/10	*Software development and testing on prototype* While waiting for PCBs to arrive, work on software logic on the dev kits to reliably recognize bad wrist angles. Test basic threshold logic for triggering warnings or gentle servo feedback. Assemble received PCBs (hopefully without any major setbacks) by 11/8. Experiment with 3D-printing small enclosures and attaching components to a flexible glove.
11/11-11/25	*Software development and testing on PCB-based system* Begin testing the assembled PCB. Once the basic microcontroller's functionality is verified, test angle calculations directly on the physical glove harness. Debug unexpected sensor spikes or feedback glitches. Resolve any hardware issues and consider a second PCB revision if necessary. Any additional features/stretch goals will be researched and decided upon by this time.
11/26-12/9	*Final testing, additional features, poster and expo* Target project completion by 12/3. Prepare the demonstration poster and start working on the final report. Conduct simple typing trials to evaluate if the feedback effectively prompts posture adjustments, and fix any remaining minor firmware bugs.
12/10-12/11	*Finish final report*

Budget

While a realistic budget is difficult at this time, the numbers below are a reasonable first pass. If other costs came up (and they likely will), having only 1 or 2 IMU sensor modules is probably enough and would be a way to cut costs. With a team of 4, we would be under budget. Only 2 glove prototypes and 1 forearm controller unit would be built. Extra parts are there as spares.

What	**Quantity**	**Cost per**	**Cost total**
ESP32-S3 Microcontroller	4	$20.00	$80.00
PCB–glove controller	5	$15.00	$75.00
6-DOF IMU Sensors	6	$20.00	$120.00
sEMG Analog Front-End (Bio-signal sensor)	4	$50.00	$200.00
Micro-servo / Haptics	4	$12.00	$48.00
Misc parts (cables, fabric, LiPo)	1	$100.00	$100.00
Shipping	1	$50.00	$50.00
Total			**$673.00**

Issues

The largest concerns at this time are

1. Developing the algorithms needed for real-time motion tracking and filtering
2. Interfacing with the bio-signal/sEMG sensor
3. Soldering the surface mount microcontrollers and fine-pitching
4. Designing a flexible harness for a moving hand
5. Active mechanical assistance/tensioning

Assuming no one in the group has experience with this exact project, issue 1 is an unknown but doable. Issue 2 may be difficult, but http://www.adafruit.com/products/2699 makes it seem likely we'll find enough online examples of how other people have interfaced with these components. Issue 3 seems a bit difficult, but given enough time and dedication, soldering can be done effectively. Issue 4 is perhaps the most difficult as it in effect requires combining electronics with wearable fabrics. Issue 5 will be addressed by dropping that functionality if needed within time constraints. Overall, each issue seems reasonably manageable within this semester.

Additional stretch goal

While the mechanical assistance part is already a stretch goal, if time allows we will see if we can log bio-feedback trends over Bluetooth to a mobile phone app for long-term health tracking.

Automated Volumetric Foot Profiler and Shoe-Matching System

So many shoes ordered online end up being returned, and as someone who has done it themselves, it may be because the fit is slightly off in the length or the width, causing discomfort. Beyond the work of sending back boxes, wearing the wrong shoe size daily takes a real toll on your feet, leading to painful issues like sore toes, bunions, or rashes. Even a simple scanner mat could make it easier for all of us. For instance, if a person could just step onto a mat at home or in a store, a quick scan could map out their exact foot length, arch height, and weight distribution in mere seconds. Instead of buying two different sizes just to send one back, you would get an immediate, personalized recommendation that translates your exact foot dimensions across different shoe brands.

If retail footwear stores utilized a shared database of interior shoe dimensions, shoppers could scan their foot once and instantly locate inventory across multiple store-provided manufacturers to find models tailored to their exact interests, saving time and preventing costly returns.

Ideas like this have been explored in mobile phone camera scanning apps have also been introduced, but they frequently suffer from poor calibration and perspective distortion. So, a low-cost, floor-based, and weight-aware footprint profiler is needed.

Details

As a user steps onto the measurement platform, the system would attempt to map out key foot dimensions like length, width, and arch height. Rather than using an expensive 3D laser rig, the pad could experiment with a grid of distance sensors or a movable camera mounted on an overhead track, paired with load cells under the platform to measure weight distribution. A microcontroller would collect the optical and weight measurements and output an estimated foot profile. This data could then be matched against standard shoe size conversion charts via a simple software script.

This proposal envisions building a single platform node with integrated sensors and a basic screen interface. The team will look into using a few distance sensors, load cell force sensors, a stepper-driven slider (or basic optical sensor), and a budget MCU like an ESP32 or Arduino.

Implementation Issues

Main implementation concerns include figuring out how to accurately capture foot details with cheap distance sensors and managing shadow/lighting distortion if using an optical sensor. Signal calibration from the load cells when someone shifts their weight might also be difficult, but software averaging should help.

Schedule

9/10-9/23	*Planning* Brainstorm ways to measure a foot using low-cost hardware. Compare distance sensors vs. a simple camera on a frame. Look into basic load cells for measuring weight distribution. Pick a microcontroller and draft the project proposal by 9/18.
9/24-10/6	*Design and part selection* Order sample distance sensors, load cells, and a dev kit. Try getting raw readings on a breadboard. Submit final proposal by 10/1 and order initial prototype parts by 10/4.
10/7-10/25	*Breadboard prototype and PCB design* Test whether we can reliably estimate foot length and width on a benchtop rig. Draft a basic PCB layout for the sensor interfaces and controller. Submit PCB order by 10/26.
10/26-11/10	*Software development and testing on prototype* Write code to convert raw sensor readings into actual inches/centimeters. Build a rough wooden or 3D-printed standing pad. Assemble received PCBs by 11/8.
11/11-11/25	*Software development and testing on PCB-based system* Mount electronics to the physical platform and test with actual foot measurements. Compare system results against a manual tape measure. Tweak software logic to fix calibration errors.
11/26-12/9	*Final testing, additional features, poster and expo* Finish prototype build by 12/3. Prepare poster, write final report, and fix remaining software bugs before the expo.
12/10-12/11	*Finish final report*

Budget

Since this is a rough initial concept, these numbers are just a quick guess. If parts get expensive, we can drop down to fewer distance sensors or use a simple frame instead of a moving track. With 4 team members, we will easily stay under our $1,000 limit.

What	**Quantity**	**Cost per**	**Cost total**
ESP32 Dev Board	2	$8.00	$16.00
Distance Sensors	6	$10.00	$60.00
Load Cells + HX711 Amps	4	$5.00	$20.00
Platform / Frame Materials	1	$40.00	$40.00
Basic Screen Display	1	$15.00	$15.00
Misc Parts & Shipping	1	$50.00	$50.00
Total			**$201.00**

Issues

The main concerns right now are

1. Getting accurate foot shape readings from cheap distance sensors
2. Aligning multiple sensors to map a 3D foot
3. Writing a script to match measurements with shoe size charts

Issue 1 might be tricky with odd foot shapes, but averaging multiple sensor points should help. Issue 2 might take some trial and error, while Issue 3 should be fairly easy once we have clean measurement numbers. Unforeseen mechanical tweaks will probably be the real headache.

Additional stretch goal

If we get the basic scanner working quickly, a cool stretch goal would be sending the measurement data via Bluetooth to a simple phone app that directly pulls shoe size recommendations from online retail APIs.

Low Cost Retrofit Planter Monitor

Dillin Raffensberger

August 2026

Introduction

While many large and medium scale farms utilize amazing modern farming technologies like GPS guidance and auto steering, soil monitoring, and variable feed rates for implements, small family farms have fallen behind in adapting to these technologies. As of 2024, less than 13 percent of farms which have a gross income of less than $350,000 a year have attempted to implement these technologies[1].

The intention of the device would be to help these operators of small family farms to retrofit their older and small-scale planting equipment to be able to detect blocked seed tubes, empty rows, and abnormal seed flow before large swaths of field are planted incorrectly.

This product would allow small farms to begin embracing modern farming technologies to improve the efficiency and ease of use of their aging equipment, improving yields, decreasing unknown errors, and shortening the time to diagnose errors in equipment.

Details

This device would need to sense seeds as they are dispensed from seed tubes on the planters. Likely the best way to sense the seeds as they are dispensed would be a laser break sensor. The device would then need to detect spacing between seeds, if there are any seed blockages, abnormal flow, or other potential errors.

We would then need to alert the farmer driving the tractor if these issues are occurring. Data would be sent over wireless communication to a monitor on the dashboard of the tractor pulling the seeder.

The proposal outlines building one production ready prototype monitor and one set of seed detectors and the processor base stations. Each base station would be able to process several seed tubes and could be daisy chained to support increasingly larger implements.

Schedule

Selected dates and ranges are proposed and not hard due dates, but work to provide a general schedule.

Date	Activities
9/14–9/27	Planning, brainstorming, and internal design discussions. Collaborating to develop a work schedule. Researching sensors versus computer vision/AI. Begin to outline requirements for CPU selection.
9/28–10/13	After proposal is complete we can order initial sensor options. Select processor and see if there are any devkits or pre-designed boards that fit our uses. Prototype sensing behavior and monitor initial designs.
10/14–10/24	Begin breadboarding and initial design testing. Be able to get monitor to communicate via whatever communication protocol is selected. Ideally have the PCB designed and ready to order by 10/24
10/25–11/6	Order PCB by 11/3 if it has not been done already. Get Monitor completely functional and speaking to the base station as necessary for product to function. Develop base station and ensure that sensors are being correctly triggered by seed dispensing events.
11/7–11/20	Integrate monitor and seed detection so that algorithm for which errors are caused by what fault conditions can be established. Begin design of enclosures.
11/21–12/4	Finalize algorithm and robust testing to ensure no further work must be done to create a functional product. Finish enclosure design and create parts, integrate on final test bench.
12/5–12/10	Final adjustments and debugging before expo.

Budget

While a concrete budget is difficult to create this early in the process, I have outlined likely costs for the majority of potential components we would need.

Component	Quantity	Unit Cost	Total Cost
ATmega640	5	10.48	52.4
IR Break Sensor	20	5.81	116.2
Monitor PCB	5	10	50
Counter PCB	5	10	50
Display	60	2	120
Speaker	2	5.04	10.08
Misc.	1	100	100
Shipping	1	100	100
Total			598.68

Issues

Largest concern with the project would be with scale. Many of the implements we would plan on retrofitting have up to 40 seed tubes, which is why a processor with 80+ GPIO pins were selected, but with this many sensors we could find it difficult to create and debug the system. Another potential issue I foresee is that the monitor should be large enough to be obviously visible but I struggled to find a screen that was easily implementable and fit the need.

Stretch Goal

If possible the idea of adding GPS tracking so that we could track where errors in the field occur so that they could be tracked and gone over a second time if possible.

---

[1]https://www.ers.usda.gov/data-products/charts-of-note/110550

Automatic Parts Counter

Dillin Raffensberger

August 2026

Introduction

In shops and workspaces a huge problem is the lack of inventory and accountability for hardware. Small fasteners, washers, and electrical components get taken, used, lost or otherwise displaced.

The goal of this project would be to create a "vending machine" that can track and dispense accurate quantities of small components so that only the exact amount someone needs ever leaves the inventory.

Not only will this machine dispense accurate numbers, it will also track remaining inventory and message administrators or inventory managers whenever components which are running low need to be reordered.

Details

The device would need to count various components of various different sizes, masses, shapes, and colors. There are many potential ways to go about counting the components, but the initial thought would be some sort of computer vision. This could be paired with a scale which will mass the components for redundant counting.

Along with this the user interface would ideally be a keyboard and screen which allows users to look up part names or numbers to select the component they wanted. Another potential idea would be to have a touchscreen so that users could select the components graphically.

Finally the device must have an internet connection so that it can notify users when components are empty. This could possibly be direct communication using a local server, or perhaps leveraging Linux to use a machine specific email address to notify administrators over email.

Schedule

The general schedule of this project would follow what is typical for the class. The initial plans would be to create the computer vision algorithm and ensure that the camera is capable of detecting differentiating components. The next step would be to select what hardware will be actuating to dispense the components and then ensuring that the processor is capable of handling that behavior. Then get started on the PCB that houses these materials so it is ready for the order deadline. From there we would move to creating the physical components like the box and implementing the electronics to ensure the system works as a whole.

Budget

Budget

While a concrete budget is difficult to create this early in the process, I have outlined likely costs for the majority of potential components we would need.

Component	**Quantity**	**Unit Cost**	**Total Cost**
Raspberry Pi 5 4GB	1	130	130
Load Sensor	2	6.25	12.5
Device PCB	5	10	50
Display	1	19.99	19.99
Camera	1	48.25	48.25
Misc.	1	200	200
Shipping	1	100	100
Total			560.74

Issues

Some issues I think can be difficult to deal with regarding this project is the computer vision aspect. I foresee tens if not hundreds of potential sortable material and am worried about the scale of training a vision model to be able to detect the minute differences in all of them. The vision may turn into a sort of backup functionality if the force sensor works to accurately count the products being dispensed.

Cellular Mailbox Monitoring System

For someone living on a large property, checking the mail can mean walking or driving a long way just to find an empty mailbox. The mailbox may also be out of sight of the house and too far away for a Wi-Fi camera. Delivery times are not always consistent, so it can be difficult to know when the mail has arrived without checking manually. This is especially an issue for letters and smaller packages that do not have tracking information. Our idea is to build a small cellular device that can be installed in an existing mailbox and notify the owner when mail has actually been delivered.

We propose a self-contained smart mailbox system capable of detecting mail delivery and notifying the resident over a cellular network. The device would be installed inside or attached to a conventional mailbox and would require no wired network or power connection. When the system determines that mail has been delivered, it would transmit a small message through the cellular network to a remote service, which could then notify the resident. For the initial version, the notification could simply be an SMS message. A phone app or more detailed interface could be added as a stretch goal.

A basic version of the system could detect the opening and closing of the mailbox door. That may not be enough information to tell if mail was actually delivered, so the system would ideally use additional sensors to distinguish between a mail carrier delivering mail and a resident retrieving it. Possible approaches include a pressure sensor, object detection using an infrared sensor, or a simple light sensor that detects flickers of outside light when inserting objects into the box.

Cellular communication is necessary to the intended application. A mailbox hundreds or thousands of feet from a residence cannot be assumed to have access to the resident's Wi-Fi network. Cellular communication would allow the device to operate anywhere cellular service is available without requiring additional infrastructure between the mailbox and house.

The final device would consist of a weather-resistant enclosure containing our custom PCB, sensors, cellular hardware, and battery. The device should be installable in or on an existing mailbox without substantially modifying the mailbox.

Schedule

Weeks 1-2	**System design and component research** Determine how mail presence and mailbox opening will be detected. Investigate cellular modem options, sensors, processors, batteries, and power requirements. Develop an initial model of the system's power consumption.
Weeks 3-4	**Prototype development** Build a prototype using development boards and off-the-shelf sensors. Demonstrate reliable detection of mailbox opening and closing and establish cellular communication. Begin developing the state machine used to determine whether mail is waiting.
Weeks 5-7	**Sensor testing and PCB design** Test candidate mail-presence detection methods using different amounts and types of mail. Finalize the major electrical components, then design and order the PCB.
Weeks 8-9	**Software development during PCB manufacture** While waiting for the PCB, continue firmware development using the prototype hardware. Implement low-power operation, cellular messaging, sensor filtering, and the mailbox state machine. Begin designing and fabricating the enclosure and mounting system.
Weeks 10-12	**PCB bring-up and integration** Assemble and test the custom PCB. Integrate it with the mailbox sensors, battery, enclosure, and notification system. Measure actual power consumption and test communication reliability.
Remaining	**Field testing and refinement** Install the system in a real mailbox or representative test setup. Perform repeated simulated deliveries and retrievals and fix any reliability problems.

Budgeting

Component	Estimated Cost
Cellular modem/development hardware	$125
Microcontroller development boards	$50
Sensors and experimental components	$75
Batteries/power electronics	$50
PCB fabrication	$150
Enclosure/mounting hardware	$50
Optional solar hardware	$50
Miscellaneous/spare components	$50
Estimated total:	~$600

Implementation Issues

The largest concern is likely **power consumption**. Cellular radios can consume substantial power while transmitting or establishing a connection. Ideally, the mailbox should be able to run for months without needing to be charged. Because of this, the mailbox should only need to communicate when its state changes. To save power, we would keep the processor and cellular modem asleep most of the time and only wake them when something happens at the mailbox. If a reasonably sized battery can't power the device for a long enough period of time, then we may have to add a solar cell, but that would require some modification of the mailbox.

A second issue is **reliable mail detection**. Mail varies considerably in size, weight, and placement. A single envelope may be difficult to distinguish from an empty mailbox using some sensors. We will have to determine if sensing mail or door activity would be a more reliable approach.

The device must also **survive the outdoor environment**. Temperature, humidity, rain, snow, and condensation may affect the electronics. A weather-resistant enclosure and appropriate mounting location should protect the PCB and battery.

Cellular availability is another limitation. The system cannot function in areas with no cellular service, although this does not prevent it from being useful for rural properties where cellular service is available.

Evaluation

The most important measure of success is whether the system reliably tells the user when mail is waiting without requiring them to check the mailbox manually.

Testing could consist of repeated simulated delivery and retrieval events using different combinations of envelopes, magazines, advertisements, and small packages. We could measure:

- percentage of deliveries correctly detected
- percentage of retrievals correctly detected
- false notification rate
- time between delivery and notification
- cellular transmission success rate
- energy consumed per day and per notification

We would want the system to correctly identify nearly all normal delivery and retrieval events while producing very few false notifications. The notification should also arrive soon enough after delivery to actually be useful.

Water-Filter Refill Monitor

Anyone who shares a Brita pitcher with roommates has probably gone to get water only to discover that the last person put it back nearly empty. You either have to drink the small amount that's left or refill it and wait for the water to filter.

The idea is to build a device that notices when someone takes a significant amount of water and checks whether they refill it afterward. If they don't, the system eventually sends a notification reminding someone to refill the pitcher.

There are several possible ways to detect whether the pitcher has been refilled. One option would be a pressure or weight sensor underneath the pitcher to measure how much water is currently in it. Another option would be a water-flow sensor that detects water being poured into the reservoir. A simpler approach could use one or more break-beam sensors to detect whether the water level is above a certain point. Early testing would compare these approaches based on reliability, cost, and how much modification they require to the pitcher.

For example:

Pitcher removed → water consumed → pitcher returned → wait for refill → refill detected

else:

Pitcher removed → water consumed → pitcher returned → no refill detected → timeout → notification

The system would consist of a microcontroller, wireless interface, and whichever sensing hardware proves most reliable during initial testing. Ideally, the sensing system would require little or no modification to the pitcher itself.

Implementation and Feasibility

The initial prototype would test several methods of detecting when the pitcher has been emptied and refilled. Once a sensing method is selected, we would determine thresholds for distinguishing normal use from a pitcher that has actually been left empty.

After demonstrating reliable detection, wireless communication and the notification system would be implemented. A custom PCB could integrate the processor, sensor interface, and wireless hardware.

The largest technical challenge would be avoiding false notifications. A user may remove the pitcher for several minutes, partially refill it, or interact with it in ways that could confuse the sensor. The software would therefore need to determine meaningful changes in water quantity rather than simply reacting to every change in measured weight.

Maestro Sketch: A Musical Canvas & Audio Visualizer

Introduction:

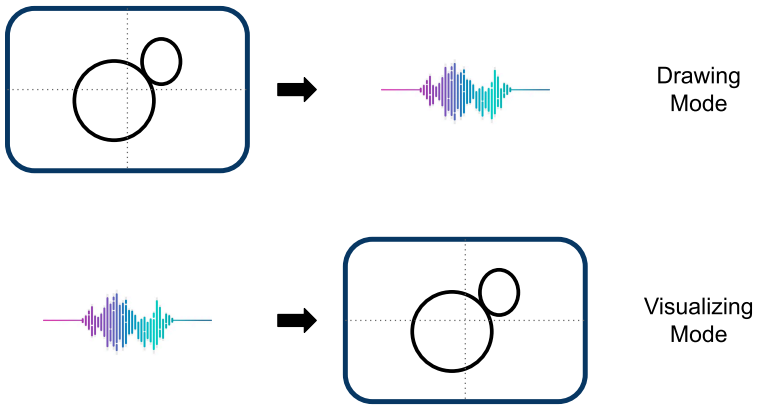
Audio visualization, where sound or music is animated using computer-generated imagery, has been around since the 70's[1]. Audio visualization adds an extra dimension or layer to the sound experienced. Musicians and listeners alike use many audio visualizers today, available for free or for purchase.

Converting drawing to sound means taking pixel data, coordinate data, or some other aspect of the drawing and converting it directly or interpreting it to produce sounds*. Both artists and researchers have made efforts to do this. This type of synthesized sound is versatile and entertaining to create in both free and paid applications.

Combining these two concepts into a single device would create a product musicians, artists, and music listeners could use to add another dimension to the experience. The Maestro Sketch would do that by serving as both an audio visualizer and a musical canvas for the user.

Details:

The Maestro Sketch will have two modes: a drawing mode and a visualizing mode. In drawing mode, the coordinate data from lines drawn on its screen will be converted into sounds at different frequencies. The music will be based on the lines drawn, with various instrument options. In visualizing mode, the device takes in sound from the left and right sides and converts it into coordinates to display on the screen. The image being shown is based on the sound received. The user can switch between different modes on the Maestro Sketch using the touchscreen's UI.



To achieve each mode, the touchscreen must act as the main interface and receive drawing input. Speakers and microphones are required for audio input and output. A MIDI synthesizer will be incorporated for sound control and instrument selection. A processor will connect these devices to create a cohesive product and handle inputs and computation with minimal lag.

Issues:

The main issues to overcome for the Maestro Sketch to be successful are

---

[1]https://en.wikipedia.org/wiki/Music_visualization

1. Developing the algorithms to quickly convert coordinate data to audio data and vice-versa
2. Interfacing with the touchscreen and establishing the UI
3. Interfacing with the MIDI synthesizer
4. PCB design and soldering
5. The price per item not being too high

Issue one can be resolved by being efficient with mathematical computations and strategic with data structures. Issue two can be solved using open-source touchscreen libraries and datasheets. Issue three will be addressed using open-source libraries and/or datasheets. Issue four is resolved, as the topic will be covered in class and workshops are available. Issue five can be addressed by considering different vendors for price and shipping, adjusting the size of the Maestro Sketch, cleaning up the physical design, or finding a partnership for a part or the whole device.

Schedule:

To complete this project within the semester, we will need to accomplish the following stages. These steps are designed to keep the design process moving and allow time for testing and debugging throughout.

Stage 1	Planning & Research With the groups now formed, we'll come together to brainstorm and discuss which features to implement, which to keep as stretch goals, and which parts to use. Additionally, we'll aim to agree upon what processor or processor family to use in time for a draft proposal by 9/18 (Friday)
Stage 2	Design & Part Selection Based on what we learn from the proposal meeting, finalize the design so we can order parts and submit the final proposal by 10/1 (Thursday). Also, test parts for functionality as they arrive.
Stage 3	Prototyping & PCB Design Begin prototyping using breadboards and Arduinos to achieve basic functionality of parts and features. Alongside that, design the PCB to meet our needs, ideally a little more than a week before the 11/3 (Thursday) deadline, to allow time for corrections. Additionally, we should start the enclosure design by this stage.
Stage 4	Software Development, Hardware Interfacing, and Testing While the PCB is being processed, we'll focus on the software, UI, and hardware interfacing design and testing. Complete all non-PCB testing by the time the PCB arrives, or by 11/10 (Tuesday) at the latest.
Stage 5	PCB Testing and Project Assembly PCB assembly and testing will begin, and the enclosure should be complete. By this point, the group should be assembling everything, ensuring functionality, meeting design constraints, and identifying any problems.

	Depending on how efficiently this stage goes, we might address our stretch goals. The ideal timeframe for this stage is at least one week before the Expo on 12/10 (Thursday).
Stage 6	Final touches, testing, and Design Expo Materials The project completion date is 12/3 (Thursday), but be ready for anything that may happen in the following week. During this time, we'll make sure everything is ready for the Expo, like the poster and the report.

Budget:

With four people, the allotted budget would cap at $1,000. This project will require a processor, PCB, touchscreen, MIDI synthesizer, microphones, speakers, and other miscellaneous parts for the design and enclosure. The table below shows how the project should stay within budget with at least 2 people on the team, with some prices based on averages of the prices found.

Item	Quantity	Cost	Total
Processor (ATmega128)	1	11	11
PCB	1	15	15
12.1" Touchscreen	1	67.8	67.8
MIDI Synthesizer	1	22	22
Speakers	2	6	12
Microphone	2	0.9	1.8
Misc parts	1	100	100
shipping	1	100	100
Total			329.6

Testing:

Different features can be tested using open-source or free software for audio visualizers and drawing conversion, like VSDC, MusicVid, draw.audio, mondile.com, etc. Other usability metrics, like a simple interface, can be tested with multiple beta testers before the Expo.

Stretch goal:

One nice-to-have feature would be a distortion mode that distorts incoming sound based on a drawn image. An external audio jack to connect to larger speakers would also be a nice feature to include.

Mosaic: Image-stitching camera attachment

Introduction:

Image stitching combines multiple images with overlapping edges to create a single panoramic or high-resolution image. It is used for image stabilization, security, medical imaging, and photographing wide or large objects[1]. However, one usually needs a specific camera with multiple lenses or external software to process multiple images. With that in mind, the Mosaic aims to end the need to upload images later and instead process them right after being taken.

Details:

After activating the Mosaic, one would take photos on their digital photo-taking device. The Mosaic then starts stitching, or combining, the photos based on the selected mode: panoramic or spherical. Panoramic mode combines images only along one direction, horizontal or vertical. Spherical mode combines images within 360 degrees. The Mosaic would have a touchscreen interface, a buzzer to indicate completion or other settings, and a processor for image stitching.

Issues:

The main issues to overcome for the Mosaic to be successful are

1. Time taken to process images
2. Potential use of a Linux processor that would require more pins and soldering
3. Portability to work with different digital cameras

We can address the first issue by tuning the algorithm and exploring different image-stitching methods. We can address the second issue by taking more time and caution with the soldering process and asking for help when needed. We can address the last issue by testing and debugging the communication protocol between different devices.

Schedule:

To complete this project within the semester, we will need to accomplish the following stages. These steps are designed to keep the design process moving and allow time for testing and debugging throughout.

[1] https://en.wikipedia.org/wiki/Image_stitching

Stage 1	Planning & Research With the groups now formed, we'll come together to brainstorm and discuss which features to implement, which to keep as stretch goals, and which parts to use. Additionally, we'll aim to agree upon what processor or processor family to use in time for a draft proposal by 9/18 (Friday)
Stage 2	Design & Part Selection Based on what we learn from the proposal meeting, finalize the design so we can order parts and submit the final proposal by 10/1 (Thursday). Also, test parts for functionality as they arrive.
Stage 3	Prototyping & PCB Design Begin prototyping using breadboards and Arduinos to achieve basic functionality of parts and features. Alongside that, design the PCB to meet our needs, ideally a little more than a week before the 11/3 (Thursday) deadline, to allow time for corrections. Additionally, we should start the enclosure design by this stage.
Stage 4	Software Development, Hardware Interfacing, and Testing While the PCB is being processed, we'll focus on the software, UI, and hardware interfacing design and testing. Complete all non-PCB testing by the time the PCB arrives, or by 11/10 (Tuesday) at the latest.
Stage 5	PCB Testing and Project Assembly PCB assembly and testing will begin, and the enclosure should be complete. By this point, the group should be assembling everything, ensuring functionality, meeting design constraints, and identifying any problems. Depending on how efficiently this stage goes, we might address our stretch goals. The ideal timeframe for this stage is at least one week before the Expo on 12/10 (Thursday).
Stage 6	Final touches, testing, and Design Expo Materials The project completion date is 12/3 (Thursday), but be ready for anything that may happen in the following week. During this time, we'll make sure everything is ready for the Expo, like the poster and the report.

Budget:

With four people, the allotted budget would cap at $1,000. This project will require a processor, PCB, touchscreen, buzzer, and a port/cable to connect to a camera/phone. The table below shows how the project should stay within budget with at least 2 people on the team, with some prices based on averages of the prices found.

Item	Quantity	Cost	Total
Processor	1	11	11
PCB	1	15	15
7" Touchscreen	1	19	19
Piezo Buzzer	1	5	5
Misc parts	1	100	100
shipping	1	100	100
Total			250

Testing:

Different features can be tested using open-source or free image-stitching software, like Hugin, Affinity, etc. Other usability metrics, like portability, can be tested with multiple cameras and phones.

Stretch goal:

If time allows, make additional units to test manufacturing multiple items of the same quality.

DropSafe: Secure Package Delivery

High-level description

Packages ordered online can travel through a secure and well-tracked delivery system, only to be left sitting in plain sight on a front porch. Doorbell cameras can help identify a thief after a package is stolen, but they do not actually protect the package. DropSafe would be a locked package box for homes that gives delivery drivers temporary access when they arrive and then secures the package until the owner gets home.

The important difference between DropSafe and a normal lockbox would be how the driver gets access. Instead of putting a keypad or permanent delivery code on the box, the courier would carry a small wireless device. For our prototype, we would build this device ourselves, although a commercial version could potentially use the handheld scanner or phone that a courier already carries. When the courier gets close to the house, the courier device and DropSafe would authenticate each other. The courier's credential would only be valid for the correct box and for a limited amount of time. This also means the outside of the box does not need to show that a package is expected.

After authentication, the box unlocks and the courier places the package inside. A load cell would detect the increase in weight and a distance sensor could provide a second check that something was actually placed in the box. Once the lid is closed, the system verifies that the package is present and that the lock has engaged. The owner could then receive a notification that the package was successfully secured.

The finished device would look like a normal outdoor storage or delivery box with an electronic lock and a small status light or display. The main controller would be a custom microprocessor-based PCB connected to the lock, package sensors, door sensors, wireless hardware, and power system. A second small embedded device would act as the courier credential during our demonstration.

The basic product would work without a subscription. An optional subscription could provide phone notifications, delivery history, remote box status, and tamper alerts. This gives the project a possible consumer-product model while keeping the main goal simple: make home delivery more secure without making the delivery process much harder for the driver.

Implementation issues

The main technical issues would be reliable authentication, package detection, and the locking mechanism. We would need to make sure that an old delivery credential cannot simply be reused later and that a credential for one DropSafe cannot open another. The wireless range should also be short enough that the box only responds when the courier is actually near the house. These features could be tested using two microcontroller-based devices and a standard short-range wireless protocol.

Package detection should be fairly manageable because we can combine two simple sensors instead of relying on computer vision. A load cell can measure whether the weight in the box increased, while a distance sensor can check whether the compartment is occupied. Door and lock-position switches would tell the controller when the lid is closed and whether the latch actually engaged. The mechanical part of the project is mainly the lock and enclosure, so we could modify an existing outdoor storage box instead of building the entire enclosure from scratch.

The custom PCB would likely contain the main microcontroller, wireless interface, load-cell interface, sensor connections, lock driver, and power regulation. Exact parts would be selected later, but none of these functions require unusual hardware or special facilities. For the prototype, the lid could be manually opened after the electronic latch releases, which keeps the mechanism simple.

Schedule

1. Planning and part selection	Choose the wireless/authentication approach, sensors, lock mechanism, processor, and overall system design.
2. Prototype	Use development boards to get courier authentication, package sensing, and lock control working.
3. PCB design	Design and order the custom DropSafe controller PCB once the basic interfaces have been tested.
4. PCB wait time	Finish the courier device, authentication software, enclosure, and package-detection code.
5. Integration	Integrate the wireless system, sensors, electronic lock, and power system.
6. Testing and debug	Test repeated deliveries, invalid and reused credentials, different package sizes, lock reliability, and the final Design Expo demonstration.

Rough budget

Custom PCB and components	$100-$175
Wireless hardware / courier device	$40-$80
Load cell, amplifier, and distance sensors	$40-$70
Electronic lock and door sensors	$40-$75
Power supply / battery hardware	$40-$70
Storage box and mechanical hardware	$60-$120
Display, LEDs, wiring, and spare parts	$50-$100
Estimated total	$370-$690

This is comfortably below the course limit of $250 per student and leaves room for replacement parts or a PCB revision. The cost could also be lower if we reuse development boards or an existing storage box during prototyping.

Evaluation and testing

The main test would be whether DropSafe can reliably complete a delivery from start to finish. A person acting as the courier would approach with the correct device, authenticate, open the box, place a package inside, and close it. The system should detect the package, relock the box, and record the delivery without any help from the homeowner.

We would also test failure cases. A device with no valid credential, a credential for the wrong DropSafe, and a previously used credential should all be rejected. We would try packages with different sizes and weights to see how reliably the sensors detect a real delivery. We would also repeatedly cycle the lock to check that it does not jam or fail after many uses.

A reasonable goal would be for valid deliveries to succeed at least 95% of the time, for all invalid credentials in our test set to be rejected, and for package detection to be correct at least 95% of the time. At the Design Expo, the project could be demonstrated by showing a normal successful delivery followed by an unauthorized device attempting and failing to open the same box.

Autonomous Cleanup Rover

High-level description

For our EECS 373 project, we built a small rover that could use a camera to recognize trash, drive toward a selected object, and collect it with a servo-controlled scoop. The project showed that a rover could detect and pick up visible litter, but it did not really understand or search the area around it. The proposed 473 project would continue this idea by turning the rover into a more complete autonomous cleanup system.

Instead of waiting for trash to appear directly in front of its camera, the new rover would be given a bounded area such as a classroom, small park, stadium, or section of sidewalk and would systematically search that area for litter. The rover would use wheel encoders, an IMU, and distance sensors or lidar to estimate its movement and avoid obstacles. While following a coverage path, the camera system would look for litter. If an object is found, the rover would temporarily leave its search path, approach and collect the object, and then continue searching the remaining area.

The finished device would be a small four-wheel rover with a camera, ranging sensors, collection mechanism, onboard computer, and a custom microcontroller PCB. The higher-level computer would handle vision and navigation, while the custom PCB would handle real-time tasks such as motor control, encoder inputs, range sensors, scoop servos, battery monitoring, and communication with the navigation computer.

The main difference from the 373 project is therefore autonomy at the area level. The old rover answered whether a robot could recognize and collect a piece of trash that it could already see. This project would try to determine whether a rover can independently search an area, find litter, collect it, and know when most of the area has been covered. A system like this could eventually be useful for cleaning relatively controlled spaces such as office buildings, schools, airports, or sidewalks.

Implementation issues

The biggest new challenge would be navigation and localization. The rover needs a reasonable estimate of where it has traveled so that it can search an area without repeatedly covering the same space. Wheel encoders and an IMU could provide basic odometry, while lidar or shorter-range distance sensors could detect walls and obstacles. The baseline project would be limited to flat, bounded environments. If full mapping or SLAM proves too difficult, a simpler coverage strategy could still allow the rover to systematically move through the test area.

Some parts of the project are already known to be feasible from the previous rover. We were able to send motor commands from a Jetson to an STM32-based controller, track a detected object with a camera, steer toward it, and run a scoop sequence. For 473, those pieces would be redesigned around the new custom PCB rather than simply reusing the previous development board. This lets more of the motor, sensor, power, and communication hardware become part of one complete embedded system.

The collection mechanism is another possible source of difficulty because real litter comes in many shapes and sizes. For the baseline project, testing could focus on common lightweight objects such as cans, plastic bottles, cups, and small boxes. Picking up very irregular objects, automatically sorting recycling, operating on rough outdoor terrain, or returning to a charging station could be stretch goals.

The main costs would be the rover chassis and motors, custom PCB, processor and motor-control hardware, camera, encoders, IMU, ranging sensors, battery system, and the scoop mechanism. The motors and scoop would operate at low voltage, and the rover would move at relatively low speed.

Schedule

1. Planning and navigation prototype	Select the processor, ranging sensors, encoders, IMU, and motor-control approach. Rebuild the useful parts of the previous rover on development hardware.

2. Autonomous navigation	Implement obstacle detection and a basic area-coverage algorithm. Begin integrating litter detection with navigation.
3. PCB design	Design and order the custom rover controller PCB with the processor, motor interfaces, encoder inputs, sensor connections, servo control, communication, and power monitoring.
4. PCB wait time	Continue navigation and vision development on development boards and improve the collection mechanism.
5. Integration	Integrate the motors, sensors, navigation computer, camera, and scoop into the complete rover.
6. Testing and debug	Run repeated autonomous cleanup trials, measure coverage and collection performance, fix reliability problems.

Rough budget

Chassis, motors, and encoders	$100-$175
Custom PCB and components	$125-$200
Camera / onboard computing	$100-$200
Lidar or other ranging sensors	$75-$150
IMU and additional sensors	$25-$50
Battery and power hardware	$75-$125
Scoop, servos, and mechanical parts	$75-$125
Wiring, connectors, spares, and enclosure parts	$75-$125
Estimated total	$650-$1,150

For a group of five students, this fits within the $250-per-student budget. The total could also be lower if parts from the earlier rover or available development hardware can be reused during prototyping.

Evaluation and testing

The main evaluation would use a bounded test area with several obstacles and a set of representative pieces of litter placed at different locations. The rover would start without being told where the litter is and would be given a fixed amount of time to search and clean the area. We could measure the percentage of the reachable area searched, number of objects detected, number successfully collected, collisions, human interventions, and total cleanup time.

A reasonable goal would be to search at least 90% of the reachable test area, avoid major obstacles, detect at least 80% of the litter it encounters, and successfully collect at least 75% of the detected objects without human assistance. We would repeat the test with different litter locations and obstacle layouts to make sure the rover is not only working for one memorized setup.

Proposal 1: Wearable Posture and Activity Monitor

High-Level Description

Poor posture and sitting for long periods of time can contribute to back pain and other physical problems, especially for students and desk workers. Most fitness trackers can currently track things like steps and general activity levels, but they rarely can track posture or tell you how long you have been sitting or staying in the same position. The proposed project is a small, wearable device that can monitor upper-body orientation, sitting/standing periods, active movement, and also prolonged periods of inactivity. This device would be designed to clip onto clothing and could be worn on the upper back, chest, or collar.

When the device detects that the user has been slouching for an extended period of time, it would provide immediate feedback, like a small vibration, as a reminder to adjust their posture. It could also remind the user to move around or stand up after sitting or staying inactive for too long. This would serve primarily students, office workers, and individuals in physical therapy, and it would be lightweight and small enough to wear for several hours without discomfort or distraction. A custom 3D-printed enclosure would protect the electronics while also allowing the device to be easily attached to different clothing types.

By providing actionable feedback directly to the user instantaneously, the device addresses a major cause of physical strain in the workplace through continuous monitoring and habit formation. Additionally, an optional wireless connection could be added that sends this data to a phone or other device for a user's own tracking and monitoring. This project would be useful because current products may be too restricting or don't provide combined posture and movement tracking, while this project would combine this in a small, portable device.

Project Schedule

The following schedule outlines the main development stages of this project leading up to the Design Expo. These dates may change slightly going forward but overall, it should follow this timeline.

9/10 - 9/18	**Planning** After forming the team, brainstorm possible approaches and decide components we plan to have. Research low-power microcontrollers, (e.g. nRF52/ESP32 series) and IMUs. Determine how the device will detect activity and posture and define power requirements. Draft the proposal by 9/18.
9/19 - 10/1	**Algorithm Testing and Final Proposal** Meet with course staff for the proposal meetings. Then, order the initial

	development boards and IMU sensors. Build a breadboard prototype and begin testing the posture detection algorithms. Submit the final proposal by 10/1.
10/2 - 10/22	**Schematic and PCB Design** Finalize the hardware schematic and PCB layout while keeping the design small and wearable. Determine how the PCB and other components will fit inside the case. Submit the Milestone 1 report by the 15th.
10/23 - 11/3	**PCB and Components** Order the surface-mount components, vibration motors, batteries, and other required parts. Complete and submit the final PCB files for manufacturing before the 11/3 deadline.
11/4 - 11/12	**Firmware Development and PCB Delivery** While waiting for the PCBs to arrive, develop and test firmware using the development boards. Focus on posture tracking and detecting whether the user is sitting, standing, or moving. Submit the Milestone 2 Report by 11/10.
11/13 - 11/25	**Hardware Assembly and Board Integration** Solder the components onto the PCBs and begin testing the completed boards. Calibrate the IMU and adjust the posture detection thresholds. Integrate the vibration feedback and fit the hardware inside the case.
11/26 - 12/10	**Final Testing and Design Expo** Test the complete device in real-world situations and verify the battery life meets the target expectation. Adjust the posture detection to reduce false alerts and fix any leftover bugs. If there is enough time, implement Bluetooth syncing. Also finish the poster and write the final report.

Budget Estimate

While setting a budget that is fully realistic is hard currently, the table below sets a reasonable first expectation. With the current budget, we would have enough buffer for any unexpected costs with a team of 4 or 5.

Item	**Quantity**	**Unit Cost**	**Total Cost**
MCU/Wireless Module	5	$8.00	$40.00
IMU Sensors	5	$6.00	$30.00
Custom Wearable PCB	5	$15.00	$75.00
Batteries and Charging Hardware	5	$12.00	$60.00

Miscellaneous Parts and Enclosure Materials	1	$90.00	$90.00
Shipping	1	$75.00	$75.00
Total Estimated Cost			**$370.00**

The main costs will come from the PCBs, batteries, and sensors. The parts listed above are estimates and may change once specific components are selected. If the project starts going over budget, we could reduce the number of extra prototypes or simplify the enclosure and other mechanical parts.

Potential Implementation Challenges and Mitigation Strategies

The main concerns are as follows:

1. Since the device is meant to be worn for the majority of the day, battery life is important. Continuously collecting data could drain a small battery quickly, so the device could use the IMU's low-power features and put the microcontroller into a lower-power state when there is little to no movement.
2. The device needs to be small and light enough for the user to wear it for several hours. A small 3D-printed case with some kind of clip would allow the user to attach it to different clothing and use it comfortably throughout their daily activities.
3. Some of the components on the PCB may be very small and hard to solder. Components need to be selected accordingly so that we do not waste money on replacement parts.
4. The device has to be able to differentiate between normal movements and actual bad posture. For example, if a user briefly leans forward, this should not trigger the posture feedback. The sensor needs to be filtered and the device should wait a certain amount of time before sending feedback.
5. The device also needs to work when worn in different locations. Since a user would wear this throughout different activities, the device may move or change positions depending on the user's clothing or activity, so the system needs to be calibrated so that it can still detect posture when attached in different locations.

Evaluation and Testing with Regard to Real-World Issues

The success of the device will be based on how accurately it detects posture and activity and how practical it is to use throughout a normal day. The device will need to be tested first in controlled situations like the user is sitting, standing, walking, or intentionally changing their posture. The results from the device can then be compared with what the user is actually doing to evaluate the effectiveness of the sensors.

The posture detection also needs to be tested for situations where there may be a false alert. Like if a user is purposely leaning forward for a short period of time, then the posture feedback should not be triggered. Testing will also help determine the appropriate time a user needs to be slouching for the feedback to be triggered. Battery life is also an important part of success in this project. This device needs to be able to last for at least 8 hours without charging.

Overall, the final prototype will be a success if it can reliably detect sitting, standing, and active movement. It also needs to be able to identify prolonged bad posture, provide feedback at a reasonable time, and stay operational for a regular work day. The device also needs to function regardless of the user's movement.

Proposal 2: Automatic Waste Sorter

High-Level Description

Recycling can often be overlooked, which can lead to recyclable items being thrown away with regular trash. This project would be a small trash bin that automatically sorts items into different compartments. The main goal would be to separate aluminum cans, recyclable items like plastic and paper, and general trash.

When an item is thrown in the bin, sensors would be used to determine the type of item. An inductive proximity sensor could be used to detect whether the item contains metal, while an infrared sensor or optical sensor could provide additional information about the item. A custom PCB would collect sensor data and control a small servo motor that directs the item into the appropriate compartment.

The bin would also have sensors inside each compartment that tracks how full it is. When one compartment is close to full, an LED would light up to indicate that it needs to be emptied. The project would be useful because it creates an easier way to recycle for users especially in an office type environment.

Project Schedule

Below is an overview of the possible schedule of the project.

9/10 - 10/1	**Planning and Prototyping** Research inductive and optical sensors and determine how they can be used to identify different types of waste. Build a basic prototype using a development board to test the metal and non-metal detection and begin testing the sorting logic. Submit the final proposal by 10/1.
10/2 - 11/3	**Hardware & PCB Design** Design the PCB schematic and layout, including the sensor connections and motor control circuitry. Determine how the PCB, sensors, and other components will fit into the physical design. Submit the final PCB files for manufacturing before the 11/3 deadline.
11/4 - 11/25	**Firmware & Assembly** While waiting for the PCB to arrive, develop the code for the sorting mechanism and test the servo control using the development boards. Once the PCB arrives, assemble the hardware and begin integrating it into the physical bin and chute.
11/26 - 12/10	**Testing & Expo Preparation** Test the sorting accuracy and adjust the sensor thresholds and servo

	movement as needed. Debug the system and prepare the poster and final report.

Budget Estimate

Below is a current estimate for the budget. This is subject to change as this is only a rough early estimate, but it leaves a lot of buffer for any unexpected costs as we go forward.

Item	**Quantity**	**Unit Cost**	**Total Cost**
Microcontroller Platform (e.g. ATmega)	3	$6.00	$18.00
Inductive and Optical Sensors	3	$10.00	$30.00
Servo Motors and Drivers	3	$8.00	$24.00
Custom PCBs	3	$15.00	$45.00
3D Printing Materials	1	$35.00	$35.00
Shipping and Buffer	1	$75.00	$75.00
Total Estimated Cost			**$227.00**

Potential Implementation Challenges and Mitigation Strategies

The main concerns for this project are as follows:

1. The servo needs to be able to move the chute reliably without getting stuck or being too difficult for the motor to move. To minimize this issue, we may 3D print the chute and the servo will be selected according to this constraint.
2. Using only basic sensors to differentiate between different non-metal items will be difficult. The sorting system will have to focus on some specified categories that can be detected more reliably.
3. Items can differ in size, which may affect how they move through the system. The opening and chute will need to be designed to accommodate different common desk trash items to limit the chances of items getting stuck.

Evaluation and Testing with Regard to Real-World Issues

The success of this project will primarily be determined by how accurately and consistently it sorts different types of trash. To test this, a controlled set of office waste items will be used, such as aluminum cans, plastic bottles, paper, and snack remainders. Each item will be tested

repeatedly and the results will be used to determine the effectiveness. The goal will be to get at least 85 to 90 percent correct. Additionally, there may need to be a time restriction set such that the system does not take too long to complete. It also needs to be able to move items correctly without getting stuck or misaligned. Overall, the project will be a success if it can consistently detect and sort different items, move the chute reliably, and indicate compartment fullness.

Touchless LED Interactive Playground for Pediatric Patient Therapy and Play

Purpose

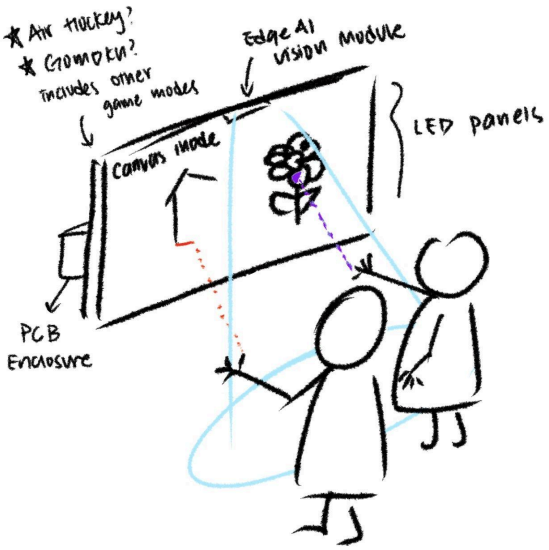
In pediatric healthcare environments, shared toys among children are often contaminated with potentially harmful bacteria and allows for the spread of infections among the patients[1]. Children are then often prohibited from using toys together like physical board games or controllers due to high risk of cross-contamination, bacterial transmission, and tedious sanitation cycles. Games are limited to single-player options like single-use craft kits or sanitized tablets for immunocompromised children. However, this implies a lack of true social interaction for those specifically in oncology units or sterile rooms: they are often faced with long hospital stays with a very limited selection of games for play and expression. As a result, the psychological problem of social deprivation has been a challenge that children's hospitals are trying to solve.

With a contactless display game sandbox, immunocompromised children will be able to socialize with other children by playing together without sharing a physical surface, replacing dirty communal screens or controllers with zero-touch interaction, letting children get creative and play competitive arcade games together through mid-air gestures with bare hands only.

High-Level Description

The system will be a responsive, touch-free interactive LED pixel display canvas mounted on hospital walls or rolling stands. When users approach the screen, sensors map their hand location in 3D space and translate gestures into real-time visual output on the screen. Some possible interactive modes include:

- Freehand Art Canvas: Users point and trace in the air to paint, switch brushes/colors and fill shapes with specific gestures. Option to download artwork by scanning on-screen QR code.
- Arcade Games: Patients, family, and healthcare staff can play quick games like air-hockey by using hands as paddles or gomoku by dragging discs across the board.



[1] Toys in a pediatric hospital: are they a bacterial source? - PubMed

A finished product will be an LED matrix panel behind an acrylic enclosure acting as the digital canvas/sandbox. At the top will be an Edge-AI vision sensor to sense hand motion location and gestures for game interaction within a cone of a certain radius. The back of the panel will house the custom PCB, microcontroller, and power supply.

Schedule

9/10-9/18	After project groups are formed, finalize the set of functions and goals we hope to achieve. Research microcontroller, Edge-AI vision module, other components, and finalize approach to mapping hand-tracking to LED pixels. Select potential components and write a draft proposal by 9/18.
9/18-10/1	After project meetings, finalize components, have a good idea of how to implement, create project progress timeline, order components and submit final proposals.
10/1-10/7	Finalize circuit diagrams, breadboard hardware components with test LED panel, benchmark power usages and current draw. Complete baseline display drivers to draw shapes on test panel, basic serial communication between gesture sensor and screen.
10/7-10/15	Create and order custom PCB, CAD chassis, scale display drivers to full LED grid, work on motion tracking and mapping gestures. Consider power distribution topology. Complete milestone 1.
10/15-10/31	While waiting for PCB, prep for hardware assembly, write and test logic for freehand drawing and games, test hand gesture tracking and mapping.
10/31-11/10	Assemble hardware and test PCB, continue testing software/hardware integration. Milestone 2.
11/10-11/24	Buffer time for debugging and testing of features and component integration.
12/1-12/11	Work on the final report, poster, and demo.

Stretch Goals

If time allows, implement the art canvas saving function by generating a QR code for people to download from. It’s not too core to the overall functionality of the project, but will be a cool

addition with computer networking. The MCU could host a local wifi access point and server, then display a QR code on the LED matrix so users can scan and download their art.

Budget

Below is the estimated cost for each component:

- LED Panels for total size of roughly 60x30cm: $120
- ESP32-S3 MCU: ~$10
- Edge AI Vision Module (Himax WiseEye2): $35
- PCB: $25
- Power supply: $35
- Miscellaneous regulators & Passives: $20
- Enclosure (Acrylic, mounts, stands): $50

All together, this puts us at under $300 minimum, not including shipping and extra components in case we break any. Since we are well under the budget for four people ($1000), we have about $700 of wiggle room for any MCU/AImodule or size upgrades, spare components, underestimated costs, and shipping.

Implementation Issues

1. Powering Multiple LED Panels
    a. Driving four panels can create current spikes that might cause voltage sags and brownouts on the MCU. To address this, we need a power distribution network with a star topology (rather than a daisy chain) for our hardware, and in firmware we can cap the brightness so we keep the peak current at a safe value.
2. Gesture Recognition with Background Noise Filtering
    a. The edge-AI camera might struggle to see hand gestures clearly if the LED panel itself blinds the lens, or other reflections/lighting affect the camera. To account for this we may design an add-on to the AI vision module to block irrelevant display light from reaching the lens.
3. LED Data Timing
    a. Continuously writing to LED screens requires high-frequency signals that could choke the processor. To solve this, we will use DMA on the MCU to automatically send screen data without needing the main processor, allowing it to read hand gestures and run games smoothly without lagging.

Evaluation

A success for this project means…

- Users will be able to play arcade games and draw with gestures in the air smoothly with minimal latency.
- LED panels can be sustained at optimal brightness without voltage sag.
- Enclosure can withstand standard hospital sanitation routines.

Fridge Monitor: Grocery Tracker for Food Waste Reduction

Purpose

It's not uncommon to forget about perishable groceries in the fridge and have to throw them out before getting to eat them. In fact, an average American throws away 103 pounds[2] of spoiled food from their fridge every year. To reduce food waste, this project automates inventory tracking and uses real-time spoilage detection to inform users what foods need to be prioritized for consumption, eliminating the need for manual inventory and expiration date management.

High-Level Description

When users put in their new groceries, the fridge registers these new items by first comparing pre and post door-open frames, and classifying them with CV. It initializes the expiration timers for each object and adds them on a list shown on the screen outside the fridge, in order of closest to expiration to furthest. If any produce starts spoiling sooner than expected, the fridge can detect so with built-in gas sensors (VOCs), and move these items on top of the expiration list.

Schedule

9/10-9/18	After project groups are formed, finalize the set of functions and goals we hope to achieve. Research microcontroller, Edge-AI vision module, other components. Select potential components and write a draft proposal by 9/18.
9/18-10/1	After project meetings, finalize components, have a good idea of how to implement, create project progress timeline, order components and submit final proposals.
10/1-10/15	Finalize circuit diagrams, breadboard hardware components, benchmark power usages and current draw. Complete LCD driver and test AI grocery recognition. Create and order a custom PCB, make a sample fridge. Complete gas sensor spoilage detection, door close sensing logic, and inventory logging logic. Complete milestone 1.
10/15-11/10	While waiting for PCB, integrate and test the entire workflow. Test PCB then assemble all hardware with PCB. Milestone 2.
11/10-11/24	Buffer time for debugging & testing of features, component integration.
12/1-12/11	Work on the final report, poster, and demo.

[2] Study Shows How Much Food Americans Waste Every Year

Stretch Goals

If time allows, we can keep track of what foods were removed/spoiled from the fridge to create a grocery shopping list for users to download to their phones wirelessly.

Budget

Below is the estimated cost for each component:

- NVIDIA Jetson Orin Nano - $250
- 13.3” Touchscreen - $150
- Fisheye camera - $60
- Magnetic reed switch - $10
- Sensirion Gas Sensor x4 - $200
- LED strips - $30
- PCB: $25
- Power supply: $35
- Miscellaneous regulators & Passives: $20
- Enclosure: $30

All together, this puts us at about $800, which is under the budget for 4-5 group members.

Implementation Issues

1. Object Recognition
    a. Some grocery items might be harder to recognize (e.g. zucchini versus cucumber) or be more hidden by other objects, resulting in an inaccurate expiration date. In this case, we can either scale down the project by only using a selection of items, or take a photo and have users fill it in instead.
2. Detecting spoiled items
    a. Gas sensors don’t work as well in big open fridges, and it’s nearly impossible to tell which specific spoiled item is emitting the gas. We could keep the product in a smaller chamber, and when a spoilage is detected we move the shortest-lived item up the list.

Evaluation

A success for this project means…

- Correctly classifies a pre-defined set of groceries, recognizes which items are new additions, and assigns a conservative expiration date.
- Reliably tracks inventory and dynamically updates front display of ordered items.
- Brings the earliest-expiring item to the top after detecting a rise in VOCs for produce.

Privacy Protecting Monitoring for Patients with Dementia

More than 55 million people worldwide live with dementia, with this number expected to increase to more than 78 million by 2030[1]. For people suffering from this condition, falls and nighttime wandering are severe safety risks, ones that can possibly be life threatening. Every year 3 million elderly patients are treated in emergency departments for fall injuries in the United States $alone$[2]. While monitoring equipment is available on the market, there are some severe shortcomings that prevent them from being well suited to this demographic. For one, any solution that uses video to monitor patients infringes on the privacy and dignity of the people being monitored. Not only this, but it is unacceptable for this type of technology to be used in more private areas like bedrooms or bathrooms, which are some of the most common areas that patients spend their time. Devices that can be worn ex. Watches or pendants rely entirely on patient compliance, and patients with dementia often remove them due to sensory $discomfort$[3]. There exist products that circumvent this, such as pressure mats or infrared sensors, however they are at risk of producing false alarms. This introduces the need for a privacy preserving, low cost, and passive monitoring system.

Details

As the patient moves around different rooms, the system needs a way to monitor them without any sort of optical lens. I propose using a radar sensor mounted to the wall. This sensor emits frequency modulated continuous radio waves and uses the reflections to generate a 3D point cloud of the room. We use this to track the human body and determine everything using this.

In order to parse and use this information we need some sort of central computing station. This would include some sort of microprocessor that could run a custom trained neural network. This means no raw data will ever leave the room, as everything is contained locally. If any sort of anomaly is detected, then the microcontroller will send an alert to the caregivers phone via a mobile app using some sort of esp32 board, separate from the one running the neural net model.

1. World Health Organization (WHO). "Dementia." Fact Sheets, 2023.
2. Centers for Disease Control and Prevention (CDC).
"Keep on Your Feet—Preventing Older Adult Falls." *National Center for Injury Prevention and Control*, 2023.
3. Alzheimer's Association. "Wandering and Dementia." *Caregiving Resources*, 2023.

Schedule

Weeks 1-3	Planning/Basic communication Finalize a block diagram demonstrating the system pipeline, calculate the power budget. Order all the hardware ASAP. Verify how the radar is working and see how to work with the data it is outputting. Write a script to read the data from the sensor on the microcontroller running the AI. Breadboard basic communication between microcontrollers.
Weeks 4-6	Machine Learning Model/PCB Design Write some sort of script to log and label training data. Collect data using the sensor. Start drafting PCB schematics as well as coding the machine learning model. Route the PCB and start training the model. Have the PCB design finished by the end of week 6.
Weeks 7-9	Software Development Get the AI running locally and get it running on the microcontroller. Start programming the non-AI board to communicate via wifi to some sort of app. Interface the wireless communication and basic firmware logic. Have a prototyped version of the final product (no wireless communication or PCB, just breadboarded connections and proof of concept.) Use CAD software to design a wall mount fort he radar/
Weeks 10-12	Hardware Assembly/Integration Start mounting the PCB. Build the established pipeline in real life. Verify everything is able to communicate with each other. Move all software from the previous steps onto their final boards.
Weeks 13-15	Final Debugging/Expo prep Test for false positives, communication errors, power errors using the ‘completed’ system. Start writing the report, prepare the poster and prepare some sort of live demo.

Budget

There are some obvious heavy lifters in this budget and honestly it's probably overkill, especially for the Edge ai brain, we just need some chip with a gpu that can run an ai model. The radar will be more difficult to cut down on so it will likely be around that price point. Miscellaneous expenses aren't accounted for here, such as cloud hosting if we wanted to actually deploy some sort of app, though a locally hosted version for proof of concept should be fine.

Component	Possible Hardware	Est. Cost
mmWave Radar	Texas Instruments IWR6843AOP EVM	$300
Edge AI Brain	NVIDIA Jetson Orin Nano Developer Kit	$499
Communication node	ESP32-S3 Dev Board + SIM7000G LTE-M	$60
Power/Thermal	5V/4A Power supplies, active heatsink	$40
Enclosure	Some 3D printed models+mounts	$30
Total:		929

Issues:

1. Signal processing and filtering noise
2. Training an accurate ML model
    a. Data collection, the model itself.
3. Thermal management for high power components
4. Wifi/Cellular feedback.

I believe the software side of things will be by far the most challenging part of this project. Everything rests on the fact that we can design a working AI model, all of the project relies on it. It seems hard to get a large enough dataset, but perhaps there is ways to do this such as writing a script to leave it and let it run in my living rooms then classify it after getting the data collected to train with. This would take a massive effort though. Perhaps it's naive but none of the hardware components seem particularly challenging on their own. Power and thermal will obviously be a big challenge though, especially if the components used end up being as high end as the ones suggested in the budget so it will be very difficult to make something that doesn't blow up or die after 4 minutes.

Stretch Goals:

It might be possible to actually monitor the patients vitals measuring the rising and falling of their chest to check for breathing and the vibration of their body for heart rate. Also, actually deploying an app and making this system connect to wifi is likely not going to be feasible in this time, so it's also a stretch goal.

Big Hero Six Microbots (Kind of)

Details:

While the neuro controlled millions of bots from the movie isn't feasible, something similar might be. I propose a group of 10-45 small robots, likely around the size of your palm, that use peer to peer communication to do things like map out areas underground or find people in rubble/fallen buildings.

These bots will not operate using any complex logic, instead since there is such a large quantity of them they will function using incredibly simple logic such as:

1. Obstacle detection: Go forward, if there is an obstacle, go towards an open path
2. Spatial Dispersion: Repel from other active bots to prevent cluttering and cover more ground
3. Basic mapping: Track local movement history to map areas.\

In large areas it will be difficult for the bots to communicate with the base station, as such the swarm will rely on a p2p mesh network. This means that a robot deep in a structure will bounce its data to a nearby robot, and this will continue until it reaches the base camp. All individual units need to do is share small packets with each other.

There is precedent for a similar project, the harvard kilobot swarm project. This project uses around 1000 low cost simple robots that are able to self assemble into complex shapes. This is similar to my proposal, but not the same. I propose a swarm optimized for dynamic environment exploration, in order to map unpredictable real world terrain.

Issues:

1. Manufacturing and assembling
2. Mesh Network Congestion
3. Power Management
4. Cost Scaling
5. Size

In any project that requires a large amount of smaller parts, size and manufacturing are the two most pressing issues. In order to make this project work, we must make many small robots and have a working design for that. Actually designing something that works at this scale is incredibly challenging. We need to have a processor on each one to run logic, some way to connect to nearby ones, some way to map location, then obviously a way to move. Each individual robot could likely be an entire 373 project. Getting this to function at scale is hard, as such the design will likely end up being much simpler. Also any cost issue will be exacerbated. If we make n robots, and each of them waste $k, the amount of money we waste will be k * n, much more than any normal project.
Power budgeting for each robot will also be complex since there are so many parts, so we would need to figure out how to mount battery packs of some sort or store power using them. Most of these issues are hardware and mechanical problems (Much different than my other proposal), and if we're able to solve them, the rest of the project wouldn't be nearly as difficult. However, we would almost certainly need to make many concessions, so hopefully we would have a product that still resembles this proposal by the end.

Stretch Goals:

Notably, all of these are software goals since the hardware itself is something like a 'stretch goal'

Environmental sensing, hazard prediction

- If there exists an environmental hazard, such as water, a big hole or anything else that could cause harm to the robot, it avoids it

A return to base protocol

- Using the data mapped from other bots and the current amount of power the bot has

Virtual Pheromone Navigation

- A more advanced way for the swarm to search areas, with the idea being that if there's a dead end, it makes bots go away, or if there's an area with little dead ends it attracts bots.

Greenhouse Rover

Growing plants in a greenhouse requires a lot of repetitive monitoring. A grower may need to regularly check moisture levels, inspect plants, keep track of which plants have been watered, and maintain the right temperature and humidity levels. This can be done manually in a small greenhouse, but as the number of plants increases, it becomes difficult to maintain and keep track of them.

This project would focus on creating a rover that navigates through the plants and does all these checks for the user. At each plant, it would take a picture of it, measure soil moisture, record local temperature and humidity, and store information like when it was last watered. The rover would be a wheeled robot with motors for movement, a water tank with a pump, sensors for navigation, a camera for plant pictures, and a movable probe to measure soil moisture. We would also have a separate hub station which would handle charging the rover and running the website that shows information for all the plants. When the robot reaches a plant, it would identify it, collect measurements, and associate new data with the specific plant.

For example, it would produce a record similar to:

Plant 12 - Tomato Plant

- Soil Moisture: 45%
- Temperature: 76°F
- Humidity: 64%
- Last watered: yesterday
- Last checked: Today, 2:05 PM
- Status: Soil moisture low
- [Plant Image]

The website would then display a record like this for each plant.

Implementation

The rover would use a higher performance board with enough processing power to handle image processing, route planning, Wi-Fi communication, and AprilTags detection. We would also have a processor to interface with the distance sensors, water pump, soil probe mechanism, motors, and encoders. The processor would be on a custom PCB that would interface with the rest of its components and would talk to the higher performance board through UART, while the hub would have a board that would be used for running the website, storing data, and performing image comparisons between past and current images. For navigation, the user would first manually drive

the rover through the area by using a calibration interface in the website, where they would be able to control the robot and move it throughout the greenhouse, similar to an RC car. There would also be a scan button where the robot would do a 360-degree rotation and scan for AprilTags. As the user controls the rover through the greenhouse, the rover records the path so it can follow the route autonomously later. Once the path is mapped, the rover would rely on wheel encoders and distance sensors to keep the rover moving along the correct path and detect any obstacles. AprilTags would give the rover stopping points to confirm current location.

When the rover reaches a plant, it would take a picture and a linear mechanism would then lower the moisture probe into the soil and retract it before it can start moving again. If moisture is below a threshold that would be specific for every plant, the rover would pump a controlled amount of water into the soil. After collecting the data, it would send it back to the hub via Wi-Fi where it would be compared with older data and displayed on the website. The picture would also be compared with older pictures to highlight changes to the plant, whether the change is in size, color, or overall appearance. If there is any data significantly different from previous measurements, it could get flagged for the user on the website. To recharge it, the rover would return to the hub, where it would position itself and a magnetic charging connector would make contact automatically. This would also allow the rover to disconnect automatically when leaving the hub.

Schedule

Weeks	Goals
1 and 2	Make base designs for rover and hub. Choose all parts, like boards, motors, sensors, camera, battery, pump, and other components. Start designing the rover chassis, soil probe, charging system, and overall website structure.
3 and 4	Test each component individually like soil probe, temperature and humidity sensors, water pump and WiFi communication, and try testing ones that would go together, like motors working with sensors and encoders, or camera capturing and scanning for AprilTags. Begin rover and hub prototypes
5 and 6	Combine navigation with calibration interface from website. Prototype soil probe and magnetic charging connector. Do breadboard prototyping and verify power to create a custom PCB design, order PCB by the end of this period
7 and 8	While waiting for PCB, create a website and database for all

	information to be stored, showing data history and images. We would also make a script to highlight differences in plants. Finish mechanical work on rover and hub as well as soil probe and connector
9 and 10	Assemble and test PCB. Integrate all components, test the charging and watering system. Begin route testing with multiple plants
11 and 12	Testing in a mock greenhouse with several plants and AprilTags. Fix navigation issues, docking alignment, probe positioning, watering, and image capture. Prepare everything for expo

Cost

A group of 4 would give us a budget of $1000, we would spend the following amount in total.

Component	Price
Rover computing board	$80
Hub computing board	$60
Custom PCB and components	$50
Motors, wheels, and drivers	$80
Distance sensors and wheel encoders	$20
Water tank, pump, and tubing	$20
Soil probe mechanism with sensor	$30
Rover chassis and hardware	$80
Battery, power regulation and charger	$60
Miscellaneous	$50
Camera	$30
Temperature and humidity sensors	$10
Total	$570

This is just a summarized and approximated cost for the project, but it puts us below our budget.

Concerns

Issues would probably come from mechanical hardware like the soil probe or the magnetic charge connector, but we should be able to do these with enough testing and research on linear actuators. Another thing to consider is the terrain, as we will have to think of a solid rover model to make it reliable in uneven terrain throughout the greenhouse. We also have a water tank on the rover, so our electronics must be carefully placed and the tank must be fully sealed. Unexpected obstacles

may also cause the rover to drift from the recorded route, but encoders and AprilTag locations can periodically confirm and correct its position.

Testing

The final system would be tested in a mock greenhouse by using potted plants and AprilTags. We expect the rover to be able to follow the recorded route, identify each plant, stop at the correct position, take pictures, measure soil moisture, and send all the data to the hub to then display it on the website. Navigation reliability could be tested by running the same route multiple times and seeing if it can reach every plant consistently. The charging system would also be tested through repeated docking to determine how good the magnetic charger makes contact with the rover. The water system would be tested by measuring water dispensed.

This project would be worth building because it would reduce the amount of work to maintain a greenhouse, while also keeping a record of plants over time. Instead of checking plants individually, a grower could check which plants need attention through the website.

Drone Catching Platform

Drones need a clear and relatively large area to safely land on. This becomes an issue when landing on uneven terrain, around obstacles, or low battery. This project would create a mobile platform that can track an incoming drone, position itself underneath it, and give the drone a safe area to land on.

We expect the platform to work with a variety of drones as long as we can put an AprilTag big enough for the camera to scan and the platform being large enough for it. The final device would be a wheeled platform with a large landing surface that would continuously estimate the drone's relative position above the landing spot and center itself under it. The goal would be for the platform to compensate for an imperfect landing rather than requiring the drone to make those adjustments. There would also be a website hosted by the platform where the user can enable drone tracking, view the live camera feed, and show information such as battery percentage, drone detection status, offset from center, and drone height.

Implementation

An upward facing camera would look up and detect an AprilTag or another marker on the bottom side of the drone. This allows us to get the relative position and height of the drone. We would also have a processor to do the camera processing to determine where the platform needs to move, while a microcontroller on the custom PCB would drive the motors. The processor and microcontroller would communicate using UART. We will also have distance sensors on all sides so it doesn't run into obstacles.

The platform would use swerve drive modules to be able to quickly switch directions and make small adjustments in position. Since buying pre-built modules would be out of our budget, we would design and build our own modules

Schedule

Weeks	Goals
1-3	Choose camera, computing board, swerve drive components, battery, and AprilTag size. Design the platform structure, swerve modules, and camera holders.
4-6	Get drone tracking working and create a swerve module. Design and order PCB. Start building a website and get it to display pictures from the camera and show tracking information

7-9	Finish the platform with everything mounted on it as well as making the 3 other swerve modules. When PCB arrives, combine camera tracking with platform movement.
10-12	Test drone approaches and landings, improve tracking accuracy, and prepare for the expo.

Cost

The most expensive thing is going to be the swerve drive modules, as they each take a drive motor and a steering motor plus encoders. Making these from scratch would cost around $80 each, or $320 for all 4. We also have a camera, PCB, battery, chassis, sensors, website-hosting hardware, which comes out to a total of around $700.

Concerns

A big concern would be accuracy and latency, as the platform would have to quickly react to any movements by the drone to be able to stay underneath it. We also would have the drone's AprilTag being too large when it is close to the camera, but a wide angle camera could help. The swerve drive will definitely bring some issues, especially since we are building them from scratch, but there are multiple open source designs we can get ideas from. They also require very precise motor control which we might run into some issues with, but with enough testing, we should be able to fine tune.

We would contact MAAV for testing with one of their drones, and will measure success as long as the drone is able to land on the platform.

Smart Bike Safety and Turn Signal System

Cyclists are vulnerable when on the road especially due to the difficulty of describing your intentions and need for constant vigilance for other cars and vehicles. There are hand signals to describe when a bike is stopping or turning, but they are not widely known and could be misinterpreted.

The goal of this is to combine a lot of different aspects into one compact product. There are separate systems like one for detecting a car approaching but not one that includes turn signals and braking as well. They are also more expensive for just one aspect so a lower cost one would be best for everyday cyclists.

Details

The device would attach to the back of the bike and would include a left and right indicator, a rear brake warning light, a small pcb that will house a microprocessor and sensors. There will also be a handlebar mounted controller that will allow the rider to indicate a turn as well as alert the rider when a car is getting closer. There will be an IMU sensor to detect when the bike is slowing down to start flashing the rear brake warning light. There will be a radar to determine when a car is getting too close to the bike and will alert the rider with a vibration and light.

The controller on the handlebars would wirelessly send a message to the device on the back of the device for indicating the direction of turning. The device would also have to be able to signal the controller to indicate if the light on the controller should flash to indicate a car is approaching.

This would be worth building as crashes with bikes have not gone down if anything they have gone up “A total of 1,075 bicyclists were killed in crashes with motor vehicles in 2024, the third-highest number ever recorded.”[1] Being able to make a biker easier to see and easier to understand will help these numbers go down.

[1]https://www.iihs.org/research-areas/fatality-statistics/detail/bicyclists

Schedule

9/10 - 9/24	**Planning** The first two weeks will be used to fully flesh out the idea and if any changes or alternate implementations can be done. Also if any additional add-ons can be included realistically. Most important will be which radar/lidar system would be best/cost efficient/power efficient. Also figure out if it is going to be able to wirelessly send signals from the controller to the device and vice versa or if it would be more reliable to do it wired.
9/25 - 10/2	**Design** The next week would be used to finally start ordering parts from the radar/lidar and the IMU sensor as well as the microprocessor. Start designing the PCB and the housing unit for all of that and the LED panel.
10/2 - 10/16	**Prototype** During this type a rough prototype that is able to communicate between the controller and rear device flash the lights and control the LED panel. It would also be best to send in the PCB to be printed. As well as have the radar/lidar working with basic functionality.
10/16 - 10/30	**Coding and Prototype Testing** While the PCB is being ordered, have all of the communication fleshed out between the controller and rear device. Finish the functionality of the prototype making sure the controller works, making sure the IMU and radar/lidar works as intended and start having it interact with the LED panel and controller. Also try to have the PCB assembled by the 30th.
10/31 - 11/13	**Testing and Software** Test on the PCB and make sure it is able to properly interact with the controller LED, Lidar/Radar and IMU. Test accuracy of the Lidar/Radar. Resolve any issues with the PCB and sensors.
11/14 - 11/28	**Finalize and Revise** Finalize the project and start to make revisions to make the product as sturdy, reliable and polished as possible.
11/28 - 12/10	**Poster and Report** Once satisfied we will start working on the poster and finish the final report.

Budget

Item	Quantity	Cost	Total
ATmega128RFA1	3	7.74	23.22
PCB	4	15	60
SIMINICS PAVO2	1	750	750
ATO-IMU-406	2	85.29	170.58
P6 Outdoor LED Screen Panel	1	21.94	21.94
Misc	1	75	75
			1,100.74

Issues

The biggest issues are going to be testing with the LIDAR/RADAR as proper testing will require someone riding in front of a car. This could be dangerous, but with proper safety this will not be too much of an issue. Potentially there could be power issues as using a small enough battery to power the LIDAR/RADAR could be an issue. However, a small less power hungry one could be used instead or increasing the size of the battery to sacrifice some of the size component to make up for this

Smart Backpack

A lot of times when rushing to class or simply going on a trip, items can be forgotten or left behind. The goal behind this is to stop this from happening. The backpack will check to make sure that the owner has not forgotten to bring any items. This would mainly be used for class where every week has the same schedule so what items to be brought are the same.

The backpack would contain several weight or pressure sensors, as well as small id tag sensors. The tags themselves can be small rfid tags that would be stuck onto items like notebooks or laptops. On the backpack would be a screen and buttons. The buttons would be pre-programmed for this such as class or gym. This would determine what is needed to be in the backpack. For example, if class is selected then the backpack would expect to see, laptop, calculator, notebook, charger, etc.. If something is missing, like the calculator, then the backpack would display on the screen calculator missing and would flash to alert the student. The backpack would also alert the student if there is an additional item within it that is not usually there, this would use the weight sensors to tell that there is too much weight. This would then flash on the screen and the student can take out the item or click a button to dismiss the weight concern.

The backpack would utilize a microcontroller to keep track of the weight sensors and RFID tags. As well as keeping track of the presets and what should be within the bag. This processor would also be tasked with determining what will be displayed on the screen.

Due to the hardware side of the project not being too advanced, more complexity can be put within the software side, for example keeping track of the days of the week. Doing this would allow the backpack to be more customizable to each individual and work better for each person as some days may need more than others.

A potential stretch goal would be to allow a person to pre-program a trip, as such the backpack would have to allow the user to type in what they want for the next day without overriding any preset options.

Automatic Ball Retrieval Robot

High-Level Description

Overview

Picking up tennis balls/pickle balls during training or recreational play is often a frustrating aspect of practicing racket sports - not only is it a tedious task that eats away at practice time, it can also be physically difficult for those who can't bend down easily.

There are existing offerings that make the physical toll easier to manage (such as these) but these still need human operation - the one option that is fully automatic is prohibitively expensive at over $2000. This proposal envisions a cheaper automatic retrieval robot would be more suitable for beginners and hobbyists.

Details

A low-profile collector robot would have two different modes - a "sweep" mode that would move in a pre-programmed path over a given section of the court and a "search" mode where the robot would use information about where balls have landed on the court to map an effective path to pick up as many as possible while prioritizing ones that have landed inbounds on the court. This would be done through 2D LiDAR sensors around the court to outline the bounds for "sweep" and to locate balls on the ground for "search", with communication between the sensors and the robot done through UWB to maximize data transfer speed. The balls would likely be picked up with a modified version of a roller in order to keep the physical design simple and cheaper, and collected in a crate or basket. Finally, after a certain amount of time has passed or a certain number of balls have been collected (the player will be able to decide which is preferrable), the robot will locate them based on a small Bluetooth wearable. The robot will then raise up the crate so the player can easily pick up the balls to continue playing.

Implementation

Rough Plan

- 9/10 - 9/25: Planning
    - Verify processor functionality, finalize communication protocols, determine number of LiDARs necessary, sketch out robot mechanics, etc
    - Wrote draft proposal for 9/18
- 9/26 - 10/7: Basic Communication
    - Establish basic communication between processor/one LiDAR sensor, and processor/wearable
    - Get algorithm for optimally collecting tennis balls/pickle balls out of the way
    - Write final proposal for 10/1

- 10/8 - 10/22: Flesh out LiDAR
    - Get accurate court line/tennis ball location communicated to processor from multiple LiDARs
    - Start on robot construction and establish basic movement controls
    - Design PCB
- 10/23 - 11/5: Robot Implementation
    - Order PCB by 10/30 (latest by 11/3)
    - Work on robot lift mechanism/housing and on full communication flow
- 11/6 - 11/25: Assembly and Testing
    - Test PCB
    - Assemble and establish full communication flow with all components
- 11/26 - 12/7: Finalize
    - Debug and work on final details
    - Aim to finish by 12/7 to have time to prep for the expo and write the report

Rough Budget

- Processor
    - Something from the SAMD Family, such as an ATSAMD51N19A, should have the processing power to be able to respond quickly to LiDAR inputs about new balls on the ground and to be able to handle the math necessary to make good paths between them
    - Total: $5 + shipping
- PCB - for collector
    - PCBWay requires a minimum order size of 5 units, and a 200mm x 200mm single-design board seems like a reasonable starting estimate
    - Total price: $97 (shipping included)
- 2D LiDAR
    - LDROBOT D500
    - In order to effectively cover the court and some of the surrounding area given the range of the devices, as well as to have spares as needed, 6 units should work
    - Total price: $50 * 6 + shipping
- Battery Packs
    - The LiDAR sensors don’t look like they have their own power sources, and the robot will also need a battery pack so a total of 6 should be sufficient
    - Total price: $40 * 6 + shipping
- Robot Mechanics
    - This is the total for a linear actuator that can raise up to 24in in order to lift up the crate, with an additional $150 for the rest of the housing, ball

collection mechanism, crate, motor controllers for the wheels, and any other physical design costs
    - Total: $142 + shipping, +$150
- Bluetooth wearable
    - A simple Bluetooth beacon should be sufficient for this
    - Total price: $18 + shipping
- LCD
    - To display mode, return condition, and battery on the robot/LiDARs
    - Total price: $10 + shipping
- **Final rough total: $962 + shipping**

Other Issues

Having a good path-finding algorithm, especially with new points to account for coming into the system as more tennis balls/pickle balls land, could be more challenging than initially expected. If that is the case, implementing only “sweep” mode might be necessary.

Evaluation and Testing

To make sure the project works, it should be tested in a real tennis/pickleball court for the duration of an actual practice session. To be successful, the robot should be able to pick up all balls on its own in a timely (e.g. requiring minimal to no stoppage in practice) and unobtrusive manner (e.g. without making excessive amounts of noise). The robot should also have a long enough battery life to last an entire session, and be portable enough to transport to and from the court with ease and without breaking. Finally, any initial calibration or setup before the start of the session should be doable by a non-technical player and not require extensive amounts of time.

Haptic Amusement Park Ride Enhancer

High-Level Description

Overview

4D theater experiences have been enhancing movies with elements like shaking seats synced to on- since 1984. However, adding these to a moving theme park ride for even more thrills would likely not be possible with the constraints of a ride car. Providing optional wearables with haptic/thermo-haptic feedback synched to specific points on the ride could allow for this type of enhanced experience in a more feasible manner.

Details

The wearables would be a chest mounted unit that would be used for vibration and pressure feedback, and a pair of long gloves that would provide vibrations, pressure, and the sensation of temperature change. These would be controlled by a FPGA also on the chest mount that would trigger different effects based on what markers along the ride it is closest to - this could rely on Bluetooth communication.

Implementation

Rough Plan

- 9/10 - 9/25: Planning
    - Wrote draft proposal for 9/18
- 9/26 - 10/7: Basic Communication
    - Write final proposal for 10/1
- 10/8 - 10/22: Add More Devices
    - Design FPGA
- 10/23 - 11/5: Wearables Implementation
    - Order FGPAs by 10/30 (latest by 11/3)
- 11/6 - 11/25: Assembly and Testing
    - Test FPGAs
    - Assemble and establish full communication flow with all components
- 11/26 - 12/7: Finalize
    - Debug and work on final details
    - Aim to finish by 12/7 to have time to prep for the expo and report

Rough Budget

- FGPAs
    - Since the form factor here is small, FGPAs might be a better option than full PCBs

    - This has to handle Bluetooth communication and control of the different haptic systems
    - Implementing this for 2 users is likely sufficient for a demo
    - Total price: $20 * 2 + shipping
- Wearable Gear and Haptic Components
    - This would likely be expensive, especially for thermo-haptic components
    - Would include the wearable chest mount/gloves themselves (~$30), multiple vibration motors (~$50) and controlelrs, and tubing/pumps for water needed for heating (~$120) for each user, at a minimum
    - Total price: $200 * 2 + shipping
- Markers
    - These could be fairly cheap Bluetooth beacons, implementing 6 should be enough
    - Total price: $18 * 6 + shipping
- **Final rough total: $448 + shipping**

Other Issues

The physical implementation of various different haptic features into such a small form factor could be an issue and is a place that could require reducing scope. Doing a thermo-haptic component at all could also be more complicated than expected

Evaluation and Testing

Because testing on an actual amusement ride is infeasible, a reasonable compromise could be a simple driving course with the markers along it that would trigger like it would in a real theme environment. Success would be triggering the right effects at the right markers along the path.

Proposal 1: Bicycle Blind-Spot Monitoring and Haptic Warning System

High-Level Description

Cyclists riding on roads must constantly divide their attention between traffic approaching from behind and hazards in front of them. Looking backward requires turning away from the road ahead, while audible warnings may be hard to hear in traffic. This project proposes a bicycle blind-spot monitoring system that detects approaching vehicles and warns the rider using haptic feedback.

A rear-mounted sensor would monitor traffic behind the bicycle. The system would determine whether an object is approaching, estimate its distance and relative speed, and determine whether it is approaching from the left or right. The rider would then be warned through vibration motors mounted near the corresponding handlebar grip. For example, a vehicle approaching from the left could cause the left handlebar to vibrate. Faster vibration patterns could indicate a more rapidly approaching vehicle.

The finished device would consist of a rear sensor module, a custom microcontroller-based PCB, rechargeable battery, and left/right haptic warning modules. It would function as a standalone device without requiring a phone during normal use.

The main goal is not simply to detect any object behind the bicycle, but to determine whether the object is actually approaching and may represent a hazard. Stationary vehicles or objects moving away from the bicycle should ideally not trigger warnings. This adds an embedded signal-processing problem beyond simple distance measurement.

Implementation Issues

The system would follow the architecture:

Rear sensors -> microcontroller -> threat detection -> haptic feedback

The custom PCB would likely contain a microcontroller, sensor interfaces, motor drivers, battery charging circuitry, voltage regulation, status LEDs, and programming/debug connections. One of the most important early design decisions would be the sensing technology. Possible options include ultrasonic, optical time-of-flight, or low-cost radar sensors. Ultrasonic sensors are inexpensive and easy to prototype, but may have limited range outdoors. Optical sensors may provide accurate distance measurements but could be affected by sunlight. Radar would likely perform better for moving-object detection but may be more expensive and complex. Because sensor performance is the largest technical uncertainty, multiple options would be tested before the final PCB is designed.

The microcontroller would analyze changes in distance over time. Relative closing speed could be estimated using:

$V_{rel} = (d_{prev} - d_{current}) / t$

An approximate time-to-collision value could then be calculated as:

$TtC = d / V_{rel}$

Two sensors aimed in slightly different directions could be used to estimate whether the vehicle is approaching from the left or right. The corresponding vibration motor would then activate.

Schedule

- Requirements and sensor testing: Define range, battery-life, and warning requirements. Test several sensor technologies using development boards.
- Prototype development: Build a basic prototype with sensors, a microcontroller, and vibration motors. Implement distance and relative-speed measurement.
- PCB and mechanical design: Select final components, design the custom PCB, and begin designing the bicycle enclosure and mounting hardware.
- Firmware development during PCB fabrication: Continue improving filtering and warning algorithms while waiting for the approximately 10-day PCB turnaround mentioned in the assignment.
- PCB bring-up and integration: Test power rails, programming, sensors, motor outputs, and charging circuitry. Integrate the system onto the bicycle.
- Final testing and Expo preparation: Perform controlled outdoor tests, tune thresholds, correct bugs, and finish the enclosure.

Rough Budget

Item	Estimated Cost ($)
Sensor	120
Microcontroller	40
Custom PCB	140
Battery and Charging	40
Haptics	40
Enclosure/Mounting	30
Other (Wring, Connector, Spares)	60
Estimated Total	430

Other Implementation Issues

The largest concern is outdoor sensing reliability. Different sensors may behave differently depending on distance, sunlight, road geometry, or target size. This risk will be reduced by testing candidate sensors early.

False warnings are another concern. The system should not repeatedly warn the rider about parked vehicles or stationary objects. Requiring several consecutive measurements showing positive closing velocity should help reduce false detections.

Mechanical packaging will also matter. The rear module must remain securely attached while the bicycle vibrates and should have basic protection against moisture. The device does not need to be fully waterproof, but the final enclosure should be suitable for normal outdoor use.

Evaluation and Real-World Testing

The project will be evaluated based on whether it provides useful warnings rather than simply measuring distance.

Example tests include:

Test	
Stationary Vehicle	No warning
Approaching Person	Detect Target
Approaching Vehicle (Slow)	Normal Warning
Approaching Vehicle (Fast)	Urgent Warning
Passing on left	Left Warning
Passing on right	Right Warning

Important measurements would include:

- Maximum reliable detection distance
- Correct left/right classification rate
- Warning response time
- Relative-speed estimation accuracy
- False-positive rate
- Battery life

Proposal 2: Smart Backpack Weight and Posture Monitor

High-Level Description

Backpacks are used by students, hikers, travelers, and workers, but users generally have little information about how the weight is distributed across their shoulders. A backpack may have an acceptable total weight while still placing much more load on one side.

This project proposes a smart backpack monitoring system that measures left/right shoulder loading and warns the user when the backpack becomes badly unbalanced. Force sensors would be placed near the left and right shoulder straps, while an inertial measurement unit could measure backpack orientation. A microcontroller would estimate total load, left/right load distribution, excessive imbalance, and possibly whether the backpack is hanging unusually far backward.

The user would receive feedback using small vibration motors. For example, if too much load is being carried by the left shoulder, the left strap could vibrate periodically.

The finished device would include a custom microcontroller PCB, force sensors, an IMU, rechargeable battery, haptic motors, and a small enclosure attached to the backpack.

Implementation Issues

The system would use:

Force sensors -> analog front end/ADC -> microcontroller -> haptic feedback

An IMU would provide additional information about motion and backpack orientation.

The main hardware challenge would be selecting the force sensors. Load cells may provide better accuracy but require more difficult mechanical integration. Force-sensitive resistors would be easier to install but may be less accurate. Several sensing methods would therefore be tested early.

Software filtering would also be required because the measured forces will constantly change while the user walks. The goal would be to detect persistent imbalance rather than reacting to every step.

Schedule

1. Research and test force-sensing methods using known weights.
2. Build a development-board prototype with force sensors and an IMU.
3. Develop calibration, filtering, and load-distribution algorithms.
4. Design and order the custom PCB and begin mechanical integration.

5. Assemble the PCB, integrate it into the backpack, and perform final testing.

Rough Budget

Item	Cost ($)
Force Sensor	80
MCU	30
PCB	120
IMU	20
Battery/Charging	30
Motor/Driver	30
Backpack	50
Other	40
Est. total	120

Evaluation and Testing

Testing would use known weights placed inside the backpack and intentionally shifted to different positions.

Example tests include:

- Empty backpack
- Evenly distributed 5 lb and 10 lb loads
- Weight shifted strongly left
- Weight shifted strongly right
- Backpack carried on one shoulder
- Normal walking

Performance could be evaluated using total-weight error, left/right load-distribution error, response time, repeatability, false-warning frequency, and battery life.

A successful demonstration would allow a user to wear the backpack, move a heavy object from one side to the other, and have the system correctly recognize the changing load distribution.

Possible stretch goals include a Bluetooth application, long-term load statistics, automatic calibration, and adjustable warning thresholds.

Idea 1: DJ Mixer

Live music mixing has increasingly growing demand as more DJs are able to provide unique and creative mixes in real time to viewers. Dancing to the overlap of fan favorite songs often becomes the highlight of a party, rave, or concert. However, DJ mixers are often either low cost and low feature, or high cost with a lot of bloat, which can make the field inaccessible. Another issue DJs face is the need for a laptop to select music. This adds a point of failure and is a poor user experience – constantly looking at a laptop makes it hard for a DJ to stay in the zone while mixing. In order to resolve these issues, this proposal describes a lower-cost, custom mixer that has a built-in screen to display music telemetry.

Technical Details

Hardware Design

The physical package of the DJ mixer is built around a custom PCB. The PCB would take in power and data from a USB-C port, which should be connected directly to a computer. For power, the PCB would take 5V from the USB and use an LDO to drop it down to 3v3 for the MCU and other peripherals. We would use multiple LDOs to separate digital and analog voltage lines, and separate their grounds to prevent noise. For data, the USB D+ and D- lines will connect to the MCU (in the STM32F4 family) with differential routing and impedance matching to bolster signal integrity. The MCU will communicate with a variety of peripherals.

First, the MCU will communicate with a PCM5102A[1] digital-analog-converter (DAC) to turn the digital music data into analog. This can then be connected to an external speaker to play music via an audio jack. The MCU can use its own I2S peripheral[2] to interface with the DAC, accelerated with DMA to provide high-speed communication. This is important because lags in sound are highly perceptible, especially to a large audience. A mixer also has knobs that are used to control volume, musical blend, and more. Potentiometers would be used to control volume since they have a finite range (0-100). They can also be used to control the pitch, base, and other tonal aspects of the music, and would be sampled using the MCU's internal ADC. To go through the music library, a push-button knob and quadrature encoder will be used to track scrolling and selection. Quad encoders will also be used for the "jog wheels" which are used to manipulate the music that's currently playing (making it faster, slower, and more). There are three LCD screens on this project. Each of the jog wheels (2) will have an LCD screen to display music telemetry such as BPM and volume. The third LCD screen covers music selection. Track data will be sent using the USB protocol to the MCU which will populate them on an LCD screen using SPI. To allow faster accesses, DMA can be used here as well.

[1] https://www.ti.com/lit/ds/symlink/pcm5102a.pdf?HQS=ds-null-null-alternatev1_582-inln-ds-rubin_high-en_us_con s&ts=1788808221800&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FPCM5102A

[2] https://community.st.com/stm32-mcus-embedded-software-32/stm32f4-i2s-dma-141373

Software Design

The firmware architecture uses FreeRTOS to execute tasks deterministically – this is critical to service the peripherals in a timely manner to provide a smooth listening experience. The RTOS tasks are ranked below in the order of priority, to ensure the MCU can address them correctly:

1. Audio Stream Handler: Using I2S and DMA interrupts, the MCU should communicate with the DAC so there aren't any blips or glitches in sound. Upon an interrupt, the MCU should populate the DMA buffer.
2. Sensor Reading Task: The MCU should secondly prioritize reading from the encoders and potentiometer ADC to update the readings on jog wheels, volume, bass, etc. A software low-pass filter could be applied to take care of noise.
3. USB Communication Task: Parses USB data and sends relevant messages back.
4. Display Task: Handles writing from the displays and reading from user inputs. This should be reasonably fast because music is mixed while it is being played, so DJs need the ability to quickly choose songs to combine them with the track that's playing.

Schedule

Date-Range	Task Description
9/14 – 9/27	Create an IOC to align on the STM32 pinout. Begin schematic with power delivery circuit while using SPICE simulations or STM Nucleos to verify appropriate filtering for the analog/digital levels. Finalize BOM for DAC, displays, and encoders. Order relevant breakout kits to enable verifying software and hardware in parallel. Configure all peripherals appropriately using HAL.
9/28 – 10/11	Finish PCB schematic and have it reviewed by the entire team and professors. Begin routing traces for PCB layout. Finish a prototyped hardware setup using a breadboard and breadboard components. Start writing device drivers. Begin CAD-ing the plastic fixture the PCB will sit inside of. Begin specifying what speakers to be used with the project.
10/12 – 10/25	Complete and review PCB layout by the entire group and professors. Order PCB. Finish writing device drivers and test using prototyped setup. Begin writing RTOS stack. Order speakers, finalize a design for the "chassis" of the PCB, it will be 3-D printed and the cover will be lasercut.
10/26 – 11/8	Complete RTOS stack and perform latency tests to

	ensure tasks are being completed in a timely manner. Tune priorities and timings as needed. 3-D print the PCB enclosure.
11/9 – 11/22	Receive PCB and solder it together. Intermittently test using testpoints baked into PCB layout. Prioritize fabricating in a way that enables testing parts of it in software (eg. soldering the USB header and MCU first). Ensure PCB fits in case appropriately with knobs and other physical components, re-design case if needed. Begin report outline and poster outline.
11/23 – 12/7	Print re-designed PCB enclosure if needed. Continue testing hardware and software on a system level, tune hardware and software as needed. Finalize report and poster outlines and begin writing them.
12/7 – 12/10	Finish report and poster, hold a full-group review session of both to proofread. Present at expo!

Budget

The below costs describe the budget to complete this project.

Component	Quantity	Unit Cost	Total Cost (Quantity * Unit Cost)
STM32F4 MCU	1	$10.00	$10.00
PCM5102A DAC	1	$4.00	$4.00
NAX9722 DirectDrive Headphone Amplifier IC	1	$2.50	$2.50
Misc. PCB components	1	$35.00	$35.00
PCB full package (4-layer board)	1	$35.00	$35.00
Custom Enclosure (3D-Printed frame & lasercut plate)	1	$40.00	$40.00
Misc. mixer	1	$10.00	$10.00

components (knob caps, sliders)			

Issues

One of the biggest concerns is with the PCB hardware design having too much noise between different voltage lines on the PCB. There are high-speed digital switching lines (USB data, SPI, ADC readings) with a lot of analog components which can also produce noise. This can corrupt the data that is being transmitted over these lines. The mitigation for this would be to use simulations and prototyping to verify that we are sufficiently decoupling and filtering these lines such that there is no cross-talk. This should be accomplished with a combination of low-pass filters (in hardware and/or software where applicable), and other physical components like ferrite chokes.

Another problem is having too much load on the software level. Supporting three displays over SPI at the same time as handling bi-directional USB traffic, scanning ADC channels, and more puts a lot of pressure on the MCU's memory and peripherals. The mitigation for this is using hardware DMA when applicable to remove software load and to specify RTOS task prioritization intentionally.

Additional Stretch Goals

Program an FPGA in SystemVerilog to sit on the I2S bus between the STM32 and the DAC to perform much faster signal processing. Because of the existing complexity of the project, it would be difficult to do this while keeping up a sufficient testing timeline to allow for a PCB re-order if needed. However, using an FPGA on the audio bus would remove a substantial load from the MCU while allowing for much faster audio processing. An alternative stretch goal would be to instead use the FPGA for a display controller. The STM32 can send commands to the FPGA and then the FPGA can write to the three displays simultaneously, removing significant traffic from the MCU's memory.

Idea 2: Steering Wheel

Video games use expensive steering wheels that do a poor job accurately communicating the haptics of the driving experience to the user. By mounting a steering wheel to a high-torque brushless motor, drivers are able to immerse themselves in the experience of driving at high speeds. Drivers are able to feel road texture and weight transfer by changing the resistance of the wheel and enabling vibration feedback. This setup also eliminates mechanical lag and gear backlash which gives the user an intense experience.

An application of this would be to use the steering wheel to play Mario Kart in a more immersive experience. This would require interacting with the Wii Remote to send the steering commands based on the roll of the remote. This project would replace these signals that the Wii Remote wirelessly sends to the motion detector.

Technical Details

Hardware Design

The main hardware complexity of this comes from building a three-phase brushless motor controller. It should take in 24-48V (depending on motor specs) to control the motor. The controls involve six MOSFETs, a gate driver, and an absolute encoder to get 14-bits of position precision. The MCU would connect to the gate driver to drive the FETs and receive encoder data for commutation and position control. In addition, current sensing can be used to monitor the health of the motor and prevent stalling. The board would also use a series of buck converters and LDOs to drop the input voltage to levels like 5V and 3.3V for the MCU and encoder.

Software Design

The software design would involve implementing a field-oriented control loop such that the stator of the motor constantly leads the rotor by 90 degrees for the most efficient control loop. This is achieved by reading the absolute encoder readings over SPI to know the position of the motor's output, and using that to spin the motor. It should then calculate the three phase currents to drive the motor. It should then send PWM signals to the gate driver to control the motor phases, allowing it to spin.

Schedule

Date-Range	Task Description
9/14 – 9/27	Select motor, complete PCB schematic (motor control and power delivery circuits), purchase breadboard package versions of components to prototype. Select steering wheel rim and design a 3D-printable steering wheel.

9/28 – 10/11	Complete PCB layout and review package. Review by team members and professors, submit for manufacturing. Write a simple trapezoidal control loop in software. Assemble prototyped components on a breadboard. 3D print steering wheel. Write a USB HID profile to interface with a Wii Remote.
10/12 – 10/25	Test software PWM generation and controls on breadboard setup, write FOC scheme in software.
10/26 – 11/8	Assemble and test hardware level of PCB, test FOC on breadboard setup. Assemble mechanical setup and ensure PCB/hardware mounting fits well.
11/9 – 11/22	Finalize FOC on assembled PCB.
11/23 – 12/7	Test playing using Wii Remote, outline project report and poster.
12/7 – 12/10	Finish report and poster, present at expo!

Issues

Most of the complexity comes from correctly implementing field oriented control given the time constraints of the class. It would require precise PWM generation and strong encoder signal integrity to read the position data accurately. In addition, writing the hardware interface for the Wii Remote to be able to send commands to it based on the PCB might be difficult to do effectively since the PCB package is moving as it's attached to the wheel.

Additional Stretch Goals

A stretch goal would be to integrate pedals and a seat with haptics integrated. The seat could vibrate when the user for example is hit by an item. Pedals can correspond to the reverse/accelerate and brake settings.

Low-Cost Rowing Telemetry System

Harry Whitten

Background

For competitive rowing programs to constantly keep improving, athletes rely heavily on feedback from coaches regarding their technique. There are some inefficiencies of how this feedback is delivered to the athletes, since coaches are not in the boat with them. Usually this consists of coaches yelling from a motor boat that follows the athletes boat, or the coaches delivering feedback after practice. Speaking from experience, this can limit how much can really be improved in a single practice. These inefficiencies in coaching have led to many commercial telemetry systems to be produced to try to fix this. These systems provide boat speed, split, stroke rate, and distance. More advanced systems provide a monitor for athletes to view power data and other information regarding how efficient their technique is on the water. These monitors can also connect wirelessly to a coaches laptop so they can view real-time data from their athletes. These systems are incredibly helpful, but are incredibly costly. I'm on the rowing team here at Michigan and a telemetry system ran us close to $20,000 for just one of our 8 man boats. This price tag puts this advantage out of reach for a lot of high school, club, and walk-on college programs, which is what this project aims to fix by creating a substantially less expensive board and sensor system that gives coaches and rowers boat-speed, stroke-rate, and a stretch goal of also providing power data.

High-Level Description

The system would be a small, waterproof, rechargeable battery powered unit mounted on the rigor of a single scull or pair rowing boat. It would consist of a low-power microcontroller, a GPS module, an inertial measurement unit (IMU), as well as low energy bluetooth module (BLE). As the boat moves it would compute boat speed and split from the GPS as well as stroke rate and boat run between strokes from the IMU acceleration signal. Data would be displayed on a LCD for the rowers and also be able to stream to the coaches laptop in their boat that follows the rowers via the BLE module. Data would be displayed to rowers/sent to coaches every second or so. Once the row is complete, the data would also be able to be pulled off the unit for further analysis.

The intended users are rowing coaches and their athletes at the high school, club, and college level who want real time feedback during practice without the cost of a commercial telemetry system. As a stretch goal detailed below, force and power sensors would also be added to this system.

Schedule

9/10-9/24	**Planning**

	Finalize team roles. Research and select parts including the microprocessor, GPS module, IMU, BLE module, LCD display and rechargeable battery. Also decided how much of the stretch goal can reasonably be committed to. Submit draft proposal by Friday 9/18.
9/25-10/8	**Design and Part Selection** Order the chosen GPS, IMPU, BLE, battery, LCD, and microcontroller as well as a devkit with I/O needed. Research and order any development boards that could be utilized. Everything should be ordered by Friday 10/2. Submit final proposal by Thursday 10/1.
10/9-10/27	**Breadboard Prototype and PCB Design** Get the devkit based prototype reading GPS and IMU data and be able to transmit wirelessly to a laptop as well as displayed on the LCD. Finish the central-unit (what the rowers will see) PCB layout and order boards by 10/27.
10/28-11/14	**Software Development while PCBs are in Fabrication** Finish the stroke detection and boat-run algorithms on the prototype hardware. Develop the coach's laptop application. If stretch goal is being pursued breadboard the strain-gauge and angle sensor signals. Also design the 3D printed casing for the central unit and get it printed before PCB arrival.
11/15-11/29	**Assembly and on-water Testing** Once PCB arrives, assemble and begin testing on water. Hard deadline before Thanksgiving as this is the last date we row on the water. Possibility of indoor testing with cooperation and permission from the women's rowing team.
11/30-12/9	**Final Testing, Poster, and Report** Wrap up any remaining bugs and continue testing if able to test indoors at the women's team's facility.

12/10-12/11	**Finish Final Report**

Budget

While specific parts have not been finalized, these parts would work and are a similar price to a different part we may choose to go with instead. Only one central unit would be built, but extra parts are ordered just in case. The total for this project is only $544.94, which is well under the total $1250 allotted for a five person group. This would financially allow for the stretch goal to be pursued, as that would include ordering strain gauges, angle sensors, more or a different mcu, and PCB fabrication.

Item	Quantity	Cost Per Item	Total Cost
MCU (Nordic nRF52840)	**3**	**$48.95**	**$146.85**
IMU Module (Bosch BNO055)	**3**	**$12.83**	**$38.49**
GPS Module (NEO-M8N)	**3**	**$21.25**	**$63.75**
PCB Fabrication of Central Unit	**5**	**$15**	**$75**
3D Printer Material, Mounting Hardware, Water-Proofing Material	**3**	**$20**	**$60**
Battery and Charging Circuit	**3**	**$12**	**$36**
Sharp Memory LCD	**3**	**$24.95**	**$74.85**
Shipping	**1**	**$50**	**$50**
Total			**$544.94**

Issues

1. Waterproofing: The central unit housing would need to be water proof as it will be exposed to splashing water, rain, and possibly could be fully submerged if the boat were

to flip. I think 3D printing an encasing and water-proofing it with seals would be a viable option, but perhaps a different method would be better.

2. Testing: There is a limited time frame of testing on the water, as the team only competes on the water until Thanksgiving break. If further testing would be required, the women's team has an indoor tank and setup to simulate rowing on the water which would allow for testing of some features. Access to this facility would need to be asked for if further testing is deemed necessary.

3. Bluetooth: Standard BLE could drop connection or lose packets if the coach trails too far from the rowers, a long range mode may need to be used to prevent this, and closer trailing by the coach could also be done.

4. IMU Calibration: Vibration from the boat and rowers can throw off accelerometer data which could complicate stroke detection and boat-run calculations. Implementing digital low-pass filtering could help reduce this.

Stretch Goal: Oarlock Force and Power Sensing

If everything is working well ahead of schedule, I'd like to add force and power sensing at the oarlock and display this information to the rowers and coaches as well. This additional functionality would be made possible by a strain gauge that measures the force transmitted through the oar and a rotary angle sensor.

Evaluation and Testing

This system would be a success if a coach in their boat could reliably receive live boat speed, split and stroke-rate data and if rowers would be able to see this data on their stroke coach as well. Correct data can be validated by doing a side by side comparison of that data generated by systems the team already has.

Self Steering Rowing Boat

Harry Whitten

High-Level Description

Steering a coxless boat like a single or pair requires a rower to physically twist their foot-stretcher to determine what direction the boat goes in. The foot stretcher is connected via two cables to the rudder. Manual steering can be very inefficient, causing fatigue, uneven leg drive, and a sub-optimal path during a windy head race. This project would solve that by automating the steering process. Rowers would be able to row regularly without having to worry about steering, while an onboard microcontroller could read predetermined GPS waypoints and course heading from an IMU. The MCU could then drive a waterproof servo motor and adjust the rudder to keep the boat on a straight line or a pre-programmed course. All of this would be powered by a rechargeable battery. There would most likely need to be a manual override switch as well in case the system is steering the boat to shore, into debris, or into another boat.

Rough Schedule

Weeks 1-3	Bench test the servo mechanics and develop the PID heading algorithms on a prototype dev board. Get GPS working and try to upload pre determined course to MCU.
Weeks 4-6	Design and fabricate a custom PCB. Design and print the 3D printed waterproof MCU and servo enclosure that allows access to steering cables.
Weeks 7-9	Assemble all parts and mount to a single or pair. Start with straight line tracking tests and then move onto pre-determined course tracking. Continue with on water testing until end of November.

Rough Budget

The total rough budget of $510.09 sits well under the allotted $1250 for a five person group.

Item	**Quantity**	**Cost Per Item**	**Total Cost**
MCU (Nordic nRF52840)	**3**	**$48.95**	**$146.85**
Waterproof High-Torque Servo	**2**	**$35**	**$70**

GPS Module (NEO-M8N)	**3**	**$21.25**	**$63.75**
PCB Fabrication of Central Unit	**5**	**$15**	**$75**
IMU Module (Bosch BNO055)	**3**	**$12.83**	**$38.49**
Battery and Charging Circuit	**3**	**$12**	**$36**
Enclosure and Mounting Hardware	**2**	**$40**	**$80**
Total			**$510.09**

Implementation Issues

1. **Testing:** Perhaps the biggest issue with this project that makes it "wacky" is the extensive on water testing it would require in order to fine tune the PID algorithms that will steer the boat. Testing is constrained by weather and end of on the water training. PID algorithms would have to be developed early to allow for enough time to tune and test.

2. **Yaw:** A boat will naturally wobble left and right with every stroke rowers take. A PID controller will try to correct every wobble which can overwork the servo. Implementing a low pass filter to smooth out this noise will allow the servo to only respond to true course drift.

Evaluation and Testing

This project would be deemed successful if it could steer a single or pair straight through a 500m straight buoyed course without crossing over buoys on either side. Additionally, if the system could steer a single or pair successfully on a pre-uploaded 3000m course without colliding with another boat or steering the boat ashore.

Henry Springsteen

EZPaint: A Desktop Paint Mixer

(Proposal #1)

Anyone who paints on small canvases, model cars, or miniature action figures has had the same exact experience: you mix a perfect custom color, you use it, run out halfway through the job, and then you can't recreate it. The color existed for all of about twenty minutes and then it was gone forever, never to be mixed properly again. Commercial paint-matching machines do exist, but they do so behind the counter of hardware stores, cost tens of thousands of dollars, and give quantities in gallons, not ounces. There is nothing between “eyeball it on your Bob Ross palette” and “bucket to paint my whole room.”

EZPaint is a desktop appliance that mixes small quantities of acrylic paint to a specific color, repeatably, and remembers the recipe. It is roughly the size of a toaster oven. You place a small mixing cup inside, close the door, pick your target color on an LCD touchscreen, the EZPaint machine dispenses measured amounts from eight pigment reservoirs, weighs the individual outputs, stirs the mixture, ensures satisfaction with an onboard color sensor, and corrects itself until the color is perfect. The recipe is then optionally saved and can be remade weeks later to produce the same color.

The targeted users for this product would be hobbyist painters, tabletop game players, model builders, art students, art schools, and small art studios. Additionally, EZPaint would be useful for color-blind painters who can specify a color numerically to get a trustworthy result without needing to rely on their own perception.

Details

Two design decisions do most of the work of making this project feasible. One, the system is closed-loop. A spectral sensor inside the mixing chamber views the paint under a fixed white LED illuminant. The firmware converts the reading, computes the error against the target, and then dispenses a correction dose. This converts a color-science problem into a feedback control problem, which is a better fit for our embedded systems course. The closed door provides the light-tight, controlled environment needed to get accurate measurements.

The second decision is that the dosing is done by weight rather than by pump run time. Cheap pumps have poor accuracy, especially with something as viscous as acrylic. Instead, the mixing cup would sit on a 500g load cell read through an amplifier with each pump simply running until the measured mass hits the target.

Eight reservoirs will hold fluid acrylic paint: warm and cool versions of blue, red, and yellow, and then black and white. Each feeding a peristaltic pump, and all eight tubes route into a fixed nozzle cluster above the cup. Fixing the nozzle eliminates a motor and potential alignment problems. A small motor will lower a disposable paddle to stir as vibration alone wouldn’t adequately mix acrylic.

Henry Springsteen

Schedule

This schedule is a rough timeline based around the due dates of major checkpoints and proposal deadlines, and is absolutely not a cut throat deadline expectation.

9/14 - 9/27	***Planning and paint characterization*** Settle the architecture and do the work of selecting paints. Buy candidate fluid acrylics and measure viscosity, tubing drying behavior, and spectral response. Evaluate a couple of pumps for flow rate and resistance to clogging. Confirm the sensor and load cell are readable and draft the proposal by 9/23.
9/28 - 10/11	***Subsystem work*** Order pumps, sensors, load cell, and screen. Get each subsystem working independently. This means pumping a known mass on command, reading a stable weight, and something like reading a repeatable color from a known swatch. Also build a crude open-frame rig so we can have real fluid moving by mid October. Final proposal is due by 10/1.
10/12 - 10/31	***Calibration data and PCB design*** With the rig running, dispense systematic grids of pigment ratios and record the resulting colors/data. This dataset will become our initial guess lookup table and will reveal how well the correction loop can converge on a desired color. Design and route our PCB in parallel to this. Ordering our PCB by the 11/3 deadline.
11/1 - 11/14	***Firmware and mechanical build*** Window reserved for everything that does not need the board, as we can expect it to be in transit during this time. This means finishing our control loop, building the UI, implementing the recipe storage, and designing and printing the enclosure, nozzle instruments, and paddle.
11/15 - 11/30	***Integration and testing on the PCB*** Port the firmware to the custom board, integrate into the built out enclosure, and begin our accuracy testing.
12/1 - 12/10	***Final testing, poster, and Expo*** Target our functional completion before the 12/10 expo so our remaining time can go to accuracy testing, rehearsing our Expo demo, the poster, and the final report.

Henry Springsteen

Budget

The numbers given below assume a team of four and are a rougher estimate dependent on our final selections after research regarding the best pumps, paint, etc. for our use case.

8x 12V DC peristaltic pumps and silicone tubing (from Adafruit)	$199.60
AS7341 spectral sensor + LED illuminant	$25
0.5kg load cell and HX711 amplifier	$20
3.5” SPI LCD touchscreen	$15
Stirrer motor	$5
3D printed parts including enclosure, paddle, etc.	$70
Fluid acrylic paint and refills	$15
Disposable cups, cleaning supplies	$10
PCB fabrication	$50
Microcontroller + a spare	$50
Miscellaneous components (wires, MOSFETs, etc.)	$100
Total	~$559.60

This very rough budget leaves about $440 of wiggle room against a $1000 team budget which is intentional. There are multiple avenues for overruns such as a secondary PCB design or finding out we need to use a different pump system mid semester.

Issues

The biggest areas for concern are:

1. Acrylic drying and clogging the tubbing/pump.
2. Convergence of the correction loop.
3. Wet-to-dry color shift of the acrylic
4. Repeatable optical measurement inside the chamber
5. Mechanical design of the nozzle system and paddle

Henry Springsteen

The first issue is the most likely, and unfortunately most devastating likely failure, and should be treated as the first-class requirement. If the tubing gets clogged, the user won't be able to easily access it to clean it (and it would be really annoying as a customer). This is potentially addressed by using ink-like consistency acrylics and keeping the travel tubes short and minimizing twist/turns. Insofar as safety is concerned, everything would theoretically run at 12V or below, acrylic paint is non-toxic, and the only moving parts would be low-torque pumps and a small stirrer which is safely behind a door interlock. So safety concerns are minimal.

Evaluation

We'll know our project works mainly via the accuracy of the paint color in comparison to the user input on the LCD. Even more so, repeatability is perhaps more important than exact accuracy for our core use case. Being able to run the same saved recipe five times across different sessions and seeing identical results would be the true test of the claim that a color can be recreated later.

Self-Regulating Freshwater Aquarium Controller

(Proposal #2)

Most aquarium fish die from owner neglect rather than disaster. Keeping a tank alive is a non-stop series of boring, menial, easily forgettable tasks like checking temperature, checking pH, feeding, light cycling, etc. If you skip enough of these, the tank gets exponentially worse under your nose.

This project is a single integrated controller that turns a standard 10-20 gallon freshwater tank into a self-regulating system controllable via your smartphone or on-tank display. It monitors water conditions, actuates heat, light, aeration, and feeding on schedule, while also dosing small corrective amounts of buffer and dechlorinator when conditions drift from the ideal, logging everything. In the worst case alerting the owner when something is so wrong that it can't fix itself. Users would be hobbyists, classrooms, or anyone who wants to leave town for a week without worrying.

Details

As water sits in a tank it's conditions drift continuously, so the controller needs a way to watch it accurately and respond accordingly. Rather than requiring the owner to test and correct by hand, this system would have a single control unit handle sensing, actuation, and dosing altogether with the owner only being involved when something requires human interaction. Temperature would be read by a waterproof sensor, while pH would come with an analog probe and ADC breakout, and a total dissolved solids sensor would round out the sensor probe. The

controller would poll all of these roughly once per minute, log the readings, and compare them against the user-set targets.

When reading drifts outside of the set targets the controller would act. Temperature being handled by switching the heater, light running on a schedule, and aeration being raised if dissolved conditions call for it. Chemical correction would be handled by two smaller pumps that draw from a few reagent bottles, more than likely a pH buffer and dechlorinator. Because a tank has large thermal and chemical inertia, and the risk of overdosing is great to the fish, the corrections would necessarily need to be small, slow, and limited with daily caps on total volume. Feeding would be separately handled by a stepper-driven dispenser with a set portion and a set schedule.

Schedule

The schedule of this project would be roughly as follows:

- Two weeks of sensor selection and probe testing in a bucket or otherwise controlled environment.
- Two weeks of bringing up the subsystems independently while designing the PCB and ordering it prior to the 11/3 deadline.
- Firmware and enclosure work while the PCB is in transit
- Integration and multi-day live tank run sometime in November
- Ample time the two weeks prior to expo for extended testing, final project report, poster, and practicing for Expo.

Budget

Tank, stand, heater, light, pump	$150
Probes (pH, temp, TDS, etc.)	$150
Relay modules and dosing pumps	$100
Microcontroller	$45
PCB fabrication	$75
Enclosure and mounting hardware/3D printed hardware	$60
Reagents, food, test kits	$50
Total	$630

Issues

Henry Springsteen

There are a few issues to consider:

1. Voltage and water in close proximity
2. Probe fouling and calibration drift
3. Testing a system which could theoretically, maybe, possibly kill a live animal

The first issue is by far the most important as it requires real care and precaution. We'd need to ensure all logic stays at low voltage and have robust physical separation between water and voltage.

Evaluation

The real test to know if we have succeeded here would be running the tank unattended for an extended period of time (up to two weeks) and intermittently testing manually to see how well the tank maintains targets. Along with this we would want to deliberately introduce faults like adding an ice cube, adding a little acid, etc. to confirm that the controller not only notices but responds within its limits and alerts.

Idea 1: Automatic Guitar Tuner

High Level Description

Right now, tuning (electric) guitars is a pain. You have to pull out a tuner and then fiddle with the guitar tuning knob until the string is in tune. It takes a while to go through each string and keep going back and forth until you've hit the right note, and then you've got to do it six times.

There are solutions to make this process a little faster, including motorized tuners like the ROADIE 3. But those ones are still handheld, and require you to tune a string, move to the next one, tune that one, etc. It's not much of an automated process. Additionally, there's other solutions that let you strum all the strings and show how out of tune all of them are, but then there's no motors to actually utilize that information.

The ideal solution would be to build a device into the guitar, not handheld, that would let you strum multiple strings and, using motors, quickly turn all the tuning pegs until the guitar is in tune, allowing for the time to be spent making music rather than tuning strings.

Implementation Details

One of the biggest considerations is whether a motor that can be small enough to be mounted on a guitar will be strong enough to actually turn the strings. Based on online data[1], the high E string pulls about 16 lbf, and all six strings of a standard set fall within a 13–17 lbf range. Acting on a 3 mm-radius string post, that tension resists with roughly 225 mN·m. The machine head's own worm gear does most of the work for us: at a typical 14:1 ratio and a pessimistic 40% worm efficiency, only about 40 mN·m is needed at the peg button. This is very feasibly achievable with motors like the Pololu 150:1 HP micro metal gearmotor, which produces 235 mN·m at stall in a 10×12 mm package[2]. Also, producing the required 40 mN·m draws roughly 0.36 A, or 22% of the motor's 1.6 A stall current, which sits inside Pololu's guidance of 25% for continuous operation.

To actually hit the pitch, I believe a PID loop is the best approach, with the pitch reading as the input, and the motor turning as the output. It'll let the motors quickly jump to the pitch, and it will adapt with different guitars and string types.

---

[1] https://www.jemsite.com/threads/string-tension-charts.73846/
[2] https://www.pololu.com/product/997

But that means that we need a reliable way to read pitch, which is tricky. At low E (82.41 Hz) one cent is 47.6 mHz, while a 4096-point FFT sampled at 8 kHz gives 1.95 Hz bins — about 41 cents, an order of magnitude coarser than our target. Even a two-second window is still roughly 10 cents off. So rather than a straightforward FFT, I believe we'd have to use a two-stage estimator: a coarse FFT to locate candidate partials, followed by phase-vocoder interpolation or an autocorrelation refinement (YIN-family) to resolve frequency well below the bin width. Both are techniques with published reference implementations, and CMSIS-DSP provides a validated FFT for Cortex-M4, so we can focus on integrating this functionality rather than inventing it.

Now the thing that might make this especially difficult is the ability to read the pitch of all the strings at the same time. It's tricky because the harmonics of some strings, like the third harmonic of the low E, lands at 247.23 Hz, within 2.0 cents of the open B string. To solve this problem, we can start off with establishing the pitch of the low E and its harmonics, and then subtract those, so we can get a clean reading on the B. In theory it's possible but it might prove challenging to implement.

These requirements set the processor choice. An 8192-point single-precision buffer alone is 32 KB. STM32G431 runs at 170 MHz with an FPU and the full DSP instruction set but carries only 32 KB of SRAM, so that won't be enough. Instead, the STM32F411 is slower at 100 MHz but provides 128 KB of SRAM with the same M4 FPU and DSP extensions, and at roughly $7 it is the better fit.

Another implementation detail is actually building it into the guitar. The basement of the Duderstadt has a nice little wood shop with all the tools we'd need to build this project. And the woodworking wouldn't be that much anyways. With a proper design, all we would need to do is route a hole in the wood and screw in the electronics. And we should be able to screw on the motor attachments to the tuning pegs. The hole doesn't need to be perfectly sized or beautiful, and if it's on the bottom it'll be hidden anyways. The final unit will be a headstock unit, control button, small LCD, and battery.

Implementation Schedule

Sept 10 - Sept 23	**Requirements and algorithm de-risk.** Get some real guitar strum audio and validate the idea behind being able to pick out individual string tunings from it (using an algorithm in Python or something). Select motor, pickup, MCU family. Measure actual peg torque on a real guitar with a torque screwdriver to validate the 40 mN·m estimate.

Sept 24 - Oct 6	**Devkit bring-up and part orders.** Port pitch detection to a Nucleo/Discovery board. Get audio in through the ADC, motor spinning through a driver breakout. Order all long-lead parts.
Oct 7 - Oct 25	**Breadboard prototype and schematic/layout**. Full PID tuning loop working on one string with one motor that tunes it to a pitch (in a reasonable amount of time). Once confirmed, order PCB by ~Oct 26.
Oct 26 - Nov 10	**The 10-day gap.** Finalize testing and development of polyphonic multi-string algorithm. Start routing holes and spaces in wooden guitar to fit enclosure. Ensure that when the PCB comes back from manufacturing that we can fully integrate it.
Nov 11 - Nov 25	**PCB bring-up and characterization.** Board assembly, debug, port firmware. Run the accuracy test protocol. Budget for one respin.
Nov 26 - expo	**Testing, poster, stretch goals.** Target functional completion ~10 days before the expo.

Implementation - Estimated Cost

Line	Qty	Est.
PCB fab + assembly, 2 spins	2 × 5 bds	$200
STM32F411 + passives, connectors	spares	$60
Micro metal gearmotor w/ encoder	3	$90
Motor driver ICs (DRV8833 class)	spares	$25
Piezo pickup / contact mic + op-amps		$40
LiPo cell + charge/protection IC	2	$40
Display (small color LCD)	2	$40
Mechanical: peg socket, shafts, hardware		$80
Beater guitar for destructive testing	1	$120
Strings (we will break some)	many	$60
Total		$755

Implementation - Safety

Snapping strings isn't too dangerous, but it can hurt skin/eyes if it happens. We can employ several ways to reduce this risk, including a hard current limit on the driver, a software sanity check that measured pitch is moving in the commanded direction (if you're tightening and pitch isn't rising, the string is slipping or about to break - halt), a hard ceiling of one semitone above target, and safety glasses during test. LiPo charging is the other one; use an off-the-shelf protection IC rather than rolling your own.

Evaluation and Testing

This is how we know we've built something functional and useful.

- **Accuracy.** We should be able to detune the string by ±100 cents, run the tuner, measure final pitch against a reference tuner (either phone app or buy a high precision one. Success: **±3 cents on all six strings across multiple trials.**
- **Speed.** Wall-clock from strum to all-six-in-tune. Incumbent benchmark: Roadie 4 claims under 10 seconds per string. If we can do better than 60 seconds (one minute) total, that's a huge win and a notable improvement over current methods
- **Polyphonic detection accuracy, measured separately from actuation.** Feed known synthetic six-note mixtures with programmed detunings, measure per-string pitch estimation error. This isolates our DSP from our mechanics - do it, because when the integrated system fails you need to know which half broke.
- **Does it meet the need?** We should be able to hand it to guitarists who have never seen it and see them use it to tune their guitars efficiently. If we can do this, we have shown it's usefulness.

Idea 2 (wacky): Electric Skates

High Level Description

Right now, getting from place to place requires either a bike, scooter, skates, or a motor vehicle. If there was a solution that could get you from place to place faster than walking, was powered, and could be carried in your backpack, this device could fundamentally change the way we travel from point to point in small distances.

Imagine two platforms with electric powered wheels. They would fit in your backpack, and when you needed to get somewhere, you could whip them out, stand on them, and ride away. When you get to your destination, you just stick them back into your backpack. It's the power of an electric scooter with the portability of normal roller skates.

Implementation Details

While it would feel like the battery for this would need to be huge (and not practical for the project) the math says otherwise. Modeling an 80 kg rider and gear at 12 mph, rolling resistance on small hard wheels contributes roughly 63 W and aerodynamic drag on a standing rider about 52 W, for a total of around 154 W at the wheel once drivetrain and motor losses are included. A 3 km commute at that speed is a nine-minute ride and consumes about 24 Wh. With 18650 cells, that's about 65 grams per shoe, or 130 grams total, which is less than a phone. So from an energy storage standpoint, this idea is practical

However, that's assuming only cruising on flat ground. A 3% grade adds 168 W, taking the total to 322 W, and accelerating at a modest 1 m/s^2 demands a 429 W transient at speed. So the motors have to be sized for roughly three times the cruise figure, split across the two platforms, and they must deliver it at high rotational speed: a 60 mm wheel spins at about 1700 RPM at 12 mph. So the combination of high RPM, high transient torque, and a package thin enough to stand on is the real design problem, not the battery. The answer is a larger wheel and a hub motor, which immediately fights the requirement that the thing fit in a backpack.

Braking might be an issue. If the shoes are moving and have their own brakes (maybe regenerative braking), and they are slightly off, that can easily trip up the guy driving them. Maybe the person riding can step off when it's time to brake, or maybe a better solution can be designed later. For now braking is an open ended question.

Implementation - Other

First of all, this project would pose a safety issue. Building battery packs and letting people drive around on motorized platforms can become risky. I strongly believe that with the proper safety requirements, we can still account for those and make it as safe as possible, both while building and while riding. For example, with

- All proper protective gear and precautions should followed at all times
- Speed should be limited for most of the testing
- Battery voltage/current should be limited for most of the testing

I think we can still deliver a useful and powerful product. But this would have to be discussed with the professors.

Low-cost AI Augmented Blind Spot Detector

Problem

More than 1100 people a year are killed in bicycle accidents and more than another 100,000 are seriously injured [1]. Additionally, the rear end accident where a driver hits the back of a cyclist accounts for 23% of these crashes [2]. If a driver is distracted, tired, or impaired when on the road, there is little a cyclist can do in order to mitigate the risk of these crashes because the car is completely unseen to them. Some sort of detection system that could distinguish to a higher degree than a mirror would be beneficial to keep the rider aware of their surroundings without them turning around to identify the issue. There is a large market for safety devices as there are 51 million annual bike riders in the US [3] and over 1 billion riders worldwide[4].

There are many options for cyclists who are looking to augment their wearable gear to improve their safety. One of the main competitors in this space is Wahoo Fitness who makes a radar powered blind spot detector. However, our goal would be to make this a more accessible price so that less serious riders could add this to their setup. A current Wahoo system would cost $250 per unit and with some development, an AI driven system could achieve the same results for a lower price.

Details

In order to use AI to remedy this, there would be a camera and other needed sensors mounted either to the right handlebar or underneath the saddle of the bike. This camera and sensor suite could communicate with a YOLO model running on an onboard processor that can analyze the image and sensor input and output to a wireless display mounted on the center of the handlebars. This would display helpful information like the type of vehicles incoming. Distance, and if the vehicles are on a collision course. There could also possibly be a live feed of the camera to the screen for easy checking.

Because of the form factor, the device would slow cyclists to take steps to improve their safety and stop waiting on improvements in vehicle safety and automation to keep them safe on the road.

---

[1] https://injuryfacts.nsc.org/home-and-community/safety-topics/bicycle-deaths/

[2]https://www.wahoofitness.com/blog/5-critical-moments-when-radar-detection-on-a-bike-prevents-crashes/

[3]https://runrepeat.com/cycling-statistics

[4]https://www.welovecycling.com/wide/2021/09/23/cycling-in-numbers-facts-you-probably-didnt-know

Schedule

9/10 - 9/23	**Ideation and brainstorming** - For a couple of weeks after the project group formation, brainstorm to make sure we have multiple approaches to all potential problems and formalize our problem statement to ensure that the whole team can be aligned on what the goals of the project and the timeline look like.
9/24 - 10/6	**Research and purchase** - Look into the technical needs of our individual components and decide which type of each is best for our scenario. This paper outlines some possible candidates but more work needs to be done to actually ensure that they are optimal. Purchasing the parts at this stage could also help avoid waiting time later in the project period.
10/7 - 10/25	**Prototyping** - This phase is pretty self explanatory. Just building each of the individual subsystems and designing their functionality with breadboards or protoboards, and ensuring that all of the components are usable together. This will not involve system level testing and only be the development of the subsystems.
10/7 - 10/25	**PCB Design** - Simultaneously to the prototyping, another team member could design and order the PCB's to match the function of the breadboards or protoboards in prototyping. Getting these ordered early is essential to staying on track
10/26 - 11/10	**System Integration** - During the period while we wait on the PCB's to arrive, we could integrate the many subsystems designed and tested in the prototyping phase into one cohesive system. This system will likely have many flaws as there will be breadboards or protoboards.
11/11 - 11/25	**PCB Integration and debugging** - One by one replace the associated breadboard or protoboard with a soldered PCB. Ensure that the debugging goes parallel to this as to localize any issues to a single PCB that has been added to ease debugging.
11/26 - 12/9	**Final preparation** - Finish any remaining design or development tasks, build the poster for the portfolio, prep for the expo, and start the final report
12/10-12/11	**Final Report** - Finish final report

Budget

This is a very rough budget. Lots of the technical elements would need more research to find the best components for the job.

Item	**Quantity**	**Price**
AI capable processor board for raspberry pi	1	150
High definition camera	2	50
Radar distance sensor	2	50
Wireless displays	1	30
Batteries	4	80
Custom PCB (1 for handlebars, 1 for sensor mount with failures expected)	4	80
Mounting hardware	—	50
Miscelaneus	—	80
Total	—	570

Issues

---

1. Feasibility of running an image detection YOLO model on a small processor. This may require expensive chips or large batteries and cooling equipment
2. Detection accuracy could be too low to be helpful, which would majorly compromise the effectiveness.
3. Processing Latency could make the AI output too late to be usable by a rider.
4. Weatherproofing must be sufficient so that there is no dust or water that gets in the container
5. Field of view of the cameras may be limited by the tire or the saddle

Stretch goal:
A smarter model that is able to integrate the sensor and image data in order to identify high risk targets, rather than just keeping the rider updated and aware of their surroundings. This could then prompt the rider when they need to take action and not require monitoring.

AI and Sensor Driven Water Ski Trainer

Problem

I am the President of the Michigan Waterski Team and one issue we frequently run into is having enough experienced skiers in the boat to teach newer skiers the fundamentals of skiing. There are not currently any commercially available computer tools that can give coaching advice and provide data driven feedback to the skier in the water. The goal of this project would be to try and create an embedded system that can be mounted to the inside of the boat and provide this assistance.

Details

In order to do this, the project would use two main sensor suites, one an AI driver camera, and the other being a tension and suite of motion sensors mounted onto the ski rope. These two suites would be connected to a central computing and display module which runs an AI algorithm to provide feedback, and displays this on an easy to view display.

Schedule

The schedule will be largely the same as the previous potential project. The less “flushed out” part here includes eliminating that redundancy of the scheduling.

Budget

This budget will be very similar to the first proposition. Fundamentally, they are both an AI camera system and other sensor computation displayed on a main display.

Item	**Quantity**	**Price**
AI capable processor board for raspberry pi	1	150
High definition camera	2	50
Radar distance sensor	2	50
Touch displays	1	30
Batteries	4	80
Custom PCB (1 for camera mount, 1 for sensor mount, one for main compute module, with failures expected)	6	120

Mounting hardware	—	150
Miscelaneus	—	80
Total	—	710

Issues

The issues for this one will be similar to the bike blind spot detector. This is because they are both most limited by the feasibility of the limitations of the AI image computation model.

1. Effectiveness of AI at analyzing movement mechanics (body angle, weight distribution, ski edge timing, etc)
2. Weatherproofing would be way more of an issue here than with the bike blindspot detector because of the constant wet environment in a boat.
3. Mounting security could be difficult because of the high g-forces on the boat and the extreme tension put on the rope.
4. Durability could be compromised during a skier's crash if the handle whips and contacts any of the sensor mounts.

Stretch goal:
Train an AI with footage and sensor data to artificially learn what good skiing looks and feels like. This would use an AI instead of a discrete algorithm made by the project team to provide the feedback to the skier.

Prep-Otle

Introduction

One of the biggest complaints against Chipotle is that they're out of ingredients on the line, which Business Insider reported that it was the customer's 2nd biggest complaint in 2016 (Business Insider). More recently, Chipotle has been under scrutiny for their portion sizes (likely attributed to them trying to make the little food that they have last longer) leading the Chipotle CEO to respond, encouraging customers to simply ask for more (USA Today).

Chipotle's lack of ingredients problem is especially true at Chipotle's State St. location. Workers are frequently caught juggling multiple tasks such as managing food on the grill, assembling burritos on the line, and checking out customers at the register. This is only made worse when Chipotle runs their BOGO (buy one get one free) promotions, leading to lines out of the door. Prep-Otle seeks to assist Chipotle employees by keeping track of all the ingredients ("toppings") and suggesting which topping to prepare next based on what's running low on the line, letting employees focus on preparing food and serving customers.

Basic Details

Prep-Otle tracks each topping and uses this information to recommend what topping to prepare next. It does this by having cameras monitoring the serving line, grill, and cutting board station to keep track of what's available and what is being prepared. Then, it recommends what to prepare next on a wall-mounted, touch-screen tablet screen.

An example of Prep-Otle assisting Chipotle employees could be Prep-Otle noticing that there are only 3 servings left of chicken and 5 servings of steak on the line. Since both items are meat, Prep-Otle would then check the grill to see if anything is being prepared. Prep-Otle sees that 10 servings of chicken will be ready in 4 minutes and that nothing else is being prepared. Knowing that 10 servings of chicken is enough to fill the serving tub and that there is space on the grill, Prep-Otle recommends the employee working the grill to put steak on the grill before the line runs out.

Ideally, Prep-Otle will help employees anticipate shortages before the serving line runs out of any topping. Unfortunately, there is not always enough space or employees to make this happen. In that case, Prep-Otle will prioritize preferred toppings (which will be set by default or can manually be set by Chipotle employees on the touch-screen tablet) over rarely ordered toppings.

As a stretch goal, Prep-Otle can learn to anticipate lunch and dinner rushes by keeping track of when and how much each topping gets ordered instead of solely relying on the line's current offerings.

Schedule

A tentative schedule is listed below to serve as a guide of where the team should be each week. Unforeseen issues might arise leading to some of these items being shifted around but they should be close to the schedule. Lastly, three PCB iterations are accounted for but hopefully, in practice, the number of iterations can be reduced. Important project milestones are listed as well.

Week 1 Sep 14 - 18 **Note:** Draft Proposal	**Brainstorming Week** The group brainstorms different approaches, such as communication between the cameras and the main tablet. Additionally, project requirements are clearly defined as the team completes the draft proposal.
Week 2 Sep 21 - 25 **Note:** Proposal Meetings	**Gameplan Week** Team members identify and take responsibility for each of the project's tasks. Additionally, parts are chosen.
Week 3 Sep 28 - Oct 2 **Note:** Final Proposal	**Real Work Begins** Members start working on the project and order parts after the final proposal.
Week 4 Oct 5 - 9	**Getting Things Started** Each part has its own interface and a breadboard prototype is completed
Week 5 Oct 12 - 16 **Note:** Milestone 1 Report	**PCB V1 & Camera Sensing** A PCB is made and ordered for both the camera modules and the wall tablet. Cameras can measure the depth of each serving tub to determine the amount of each topping available and can identify toppings being prepared on the grill and cutting board.
Week 6 Oct 21 - 23 **Note:** Fall Break + 473 Midterm + Milestone 1 Meetings	**PCB V1/V2 & Planning Prep Algorithm** Both PCBs are tested and fixed in V2. Details of the prep algorithm are re-evaluated based on the hardware's performance.
Week 7 Oct 26 - 30	**PCB V2 & Prep Algorithm Draft** Both PCB V2s are ordered and an early prep algorithm is completed.
Week 8 Nov 2 - 6 **Note:** Last day to order PCBs	**PCB V2/V3 & Testing Prep Algorithm** Both PCB V2s are tested and V3 is ordered if necessary.

Week 9 Nov 9 - 13 **Note:** Milestone 2 Report/Meetings	**PCB V3 & Prep Algorithm V2** Both PCB V3s are tested and changes are made to the prep algorithm based on last week's testing.
Week 10 Nov 16 - 20	**Testing Prep Algorithm V2** Prep Algorithm V2 is tested; it should be at least 90% completed by the end of the week.
Week 11 Nov 23 - 26 **Note:** Thanksgiving	**Finishing Touches** Fix any of the remaining miscellaneous bugs to the prep algorithm and start working on the poster and report.
Week 12 Nov 30 - Dec 4	**Flex Week** Finalize the poster and finish up any of the remaining tasks.
Week 13 Dec 7 - 11 **Note:** Design Expo + Final Report	**Report** Finalize the report and present at the Design Expo!

Budget

A reasonable first pass of the budget is listed below. This is a worst case analysis to ensure that the group can stay under budget even with 3 PCB iterations and $100 worth of surprise costs. Note that some items have hyperlinked examples (cameras and touch display). Depending on the group's spending, the size of the touch display can be easily adjusted.

What	Quantity	Cost Per	Cost Total
Depth Cameras	3	$150	$450
Batteries (5V, 5000 mAh)	4	$20	$80
ESP-32	4 (3 Cameras; 1 Tablet)	$10	$40
Camera PCB	9 (3 Iterations)	$20	$180
Tablet PCB	3 (3 Iterations)	$40	$120
LCD Touch Display	1	$62.31	$62.31
Shipping	1	$100	$100
Misc/Surprise Costs	1	$100	$100
Total			$1,132.31

Problems and Defining Success

The obvious counter point to Prep-Otle is why don't I hire someone to do this? As such, Prep-Otle needs to be a safe financial bet with a hopefully easy pitch being that it is cheaper than the salary of a single person working one week. With Michigan's minimum wage rising to $15 an hour in 2027, 12 hours of service per week, and 7 days a week this comes to a max selling cost of $1,260. For a rough 20% profit margin, the entire system should cost less than $1,000 to produce.

Additionally the biggest problem facing the group is the ease of set-up. The Prep-Otle cameras should last at least 14 hours on a full charge (adding an extra hour for opening and closing) with a plug-and-play battery. Simultaneously, they must mount easily to the wall or sneeze guard without needing to install anything special. An easy and accessible mounting solution would be two command strips each holding 15 lbs for a total max weight of 30 lbs. Meeting both of these requirements will likely be the hardest as it will require very careful part selection.

Lastly, the system should be fairly accurate. As the standard unit is a "serving size," Prep-Otle must be able to measure each topping within one serving size of their respective item. To actually test this, various shaped 3D printed objects resembling each topping (ex mini cubes for pico) can be placed in tubs with defined volumes.

D.A.S.H. (Driver Awareness and Safety Helper) Cam

Introduction

At any given moment during the day, more than 537,000 American drivers are visibly manipulating a cell phone while driving (GHSA). This has led to an estimated 36,640 people dying in 2025 from motor vehicle traffic crashes (NHTSA). Many of these deaths could've been avoided by the driver simply paying attention to the road instead of texting.

Base Details

The D.A.S.H. (Driver Awareness and Safety Helper) Cam helps address this problem by tracking the driver's eye movements to ensure the driver is paying attention to the road. If the driver doesn't check one of their mirrors or part of their windshield, the avoided area will start to slightly light up red, encouraging the driver to check that section. If the driver still doesn't look after a set period of time, local law enforcement will be notified of the driver's GPS position and their speed. Note that not paying attention is not necessarily breaking the law but instead treated as a warning of a potential incoming accident. The D.A.S.H. Cam is not meant to punish speeders but instead speeders that are not paying attention to the road.

Schedule

A tentative schedule is listed below to serve as a basic guide of where the team should be each week.

Weeks 1 - 3 Sep 14 - Oct 2	**The Proposal** The group brainstorms different ideas, defines requirements, takes on responsibilities, and orders parts.
Weeks 4 - 5 Oct 5 - 16	**Milestone 1** Each part has its own interface and a PCB is ordered.
Weeks 6 - 9 Oct 21 - Nov 13	**Milestone 2** The PCB is finalized and the project works with only a few minor bugs.
Weeks 10 - 13 Nov 16 - Dec 11	**Finishing Touches and Documentation** All of the remaining bugs are ironed out. The design poster and final report are completed.

Budget

A rough budget is listed below with the biggest cost being the eye-tracker camera and a windshield for testing purposes. For the eye-tracker specifically, I found an example used for gaming which should be overkill for our application.

What	Quantity	Cost Per	Cost Total
Windshield	1	$300	$300
Camera	1	$360	$360
Battery	1	$25	$25
Lights	5	$20	$100
Raspberry Pi	1	$50	$50
GPS/Speedometer	1	$100	$100
Shipping	1	$100	$100
Misc/Surprise Costs	1	$100	$100
Total			$1,135

Problems and Defining Success

One obvious problem is that the light has to get the driver's attention while also not being distracting. Therefore, the color and tone of the light will have to be carefully tested.

Additionally, there must be some consequence for not paying attention. Forcefully stopping the driver such as by shutting off the engine is off limits and since the project is not law enforcement, it shouldn't have any authority to punish the driver. Alerting law enforcement as a precautionary measure seems like a good idea in theory but could quickly become a problem with bad intentions. Additionally, many people will likely be uncomfortable with a device that could be used to report their attentiveness and/or speed to law enforcement. Thus, trying to make an ethical solution will be difficult and likely controversial.

Lastly, it will be tough to define the driver's attentiveness. While this project suggests defining attentiveness as periodically checking each mirror and the windshield, this could be gamed by looking around quickly in circles. Other measures will likely need to be introduced, such as ensuring that the driver is holding on to the steering wheel, to ensure attentiveness is accurately measured and defined.

Low-Cost Single-Accelerometer CPR Compression Feedback Device

Cardiac arrest is one of the leading causes of sudden death in the United States, with more than 350,000 out-of-hospital cardiac arrests occurring every year[1]. Survival depends heavily on bystanders performing immediate, high-quality CPR before EMS arrives, yet studies show that even trained rescuers struggle to consistently deliver chest compressions at the correct rate and depth without some form of real-time feedback[2]. Without feedback, most rescuers don't compress at the right rate or depth, which directly lowers the patient's chance of survival.

Devices like the CPRmeter 2, ZOLL's Real CPR Help, and TrueCPR already solve this problem, and CPR feedback is even bundled into some higher-end AEDs. That feedback, though, typically only comes on models priced at $1,800 and up. Budget-conscious schools, small gyms, and community organizations that stretch to afford any AED at all often go without compression feedback entirely, and there's no widely-adopted, low-cost standalone trainer for CPR practice and certification classes. Part of why feedback stays a costly add-on is technical: compression depth is conventionally estimated by double-integrating the raw accelerometer signal, but that process drifts badly unless something resets it every cycle. Commercial devices compensate with extra hardware, such as pressure sensors or entirely different sensing modalities, which is directly reflected in their cost and limited deployment.

To address this, we propose building a low-cost, single-accelerometer CPR feedback device that estimates both compression rate and depth without any of the extra drift-compensation hardware the expensive, commercial products use. We can achieve this through spectral-analysis and filtering techniques that extract both metrics directly from the raw acceleration signal. Published research has already shown this level of accuracy is achievable with a single accelerometer and the right math, meaning a low-cost project can achieve the same functions as those existing in hospitals[3]. From there, we engineer it into an affordable, usable device that could finally reach gyms, schools, offices, and public AED cabinets instead of hospital-only equipment.

The finished device is a small, battery-powered gadget held under the rescuer's palms during compressions, the same placement used by commercial devices today. An accelerometer feeds a microcontroller running our real-time detection algorithm, which

1. American Heart Association, "CPR Facts and Stats," https://cpr.heart.org/en/resources/cpr-facts-and-stats
2. PLOS ONE, "Feedback on the Rate and Depth of Chest Compressions...," https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0150139
3. González-Otero et al., "A New Method for Feedback on the Quality of Chest Compressions...," https://pmc.ncbi.nlm.nih.gov/articles/PMC4163344/

drives a paced buzzer (100-120 compressions/min) and an LED showing whether compressions are too shallow, adequate, or too deep. No training is required to interpret the feedback; just place it under your hands and go.

Implementation Issues

The main implementation considerations for this project are cost, reliability, and safety. Cost is central to our whole pitch, so keeping the bill of materials low is a design constraint. Reliability matters too, since a feedback device that gives inconsistent readings could actively coach a rescuer to compress incorrectly, so our algorithm needs careful validation before we'd call it functional. Safety risk is low overall, since the device only touches the outside of a rescuer's hands or a manikin's chest and runs on a small battery.

Schedule

A rough 6-item plan for reaching a working device by Design Expo, keeping the ~10-day PCB turnaround in mind:

1. **Sensor characterization:** Get an accelerometer communicating on a breadboard and confirm it captures clean, high-sample-rate compression impulses by hand-testing on a manikin.
2. **Algorithm implementation:** Implement the published spectral-analysis method for rate/depth estimation onto the breadboard, running in real time.
3. **Breadboard feedback loop:** Add the buzzer and LED feedback driven by the algorithm's live output. This is our core milestone, since it proves the system works before any custom PCB is ordered
4. **PCB design and order:** Confirm component choices from the breadboard prototype, lay out a board integrating the MCU, accelerometer, power management, and feedback drivers, and order it with a buffer for the 10-day turnaround.
5. **Validation during PCB turnaround:** Use the waiting period to test the breadboard prototype on a CPR training manikin. We'll have a separate sensor measure the true compression depth as a reference, and compare it against our device's readings while testing at different compression speeds and depths.
6. **PCB integration and final testing:** Assemble the board, transfer the validated algorithm, confirm accuracy matches the prototype, and finalize the system.

Budget

With a team of four or five (~$1,000-$1,250 total), our rough estimate is:

Component	Qty	Price per	Total

MCU Boards for Prototyping (Arduino Nano 33 BLE Sense Rev2, nRF52840)	2	$40	$80
Accelerometer (Adafruit LIS3DH)	4	$5	$20
Custom PCB Fabrication	5	$5	$25
Buzzer (Adafruit Piezo Buzzer)	4	$1.50	$6
LED (Adafruit NeoPixel single LED breakout)	4	$1.50	$6
Battery (Adafruit Lithium Ion Polymer Battery - 3.7V 500mAh)	4	$15	$60
Charging Circuit (Adafruit PowerBoost 500 Charger)	4	$8	$32
Ground-truth Distance Sensor for Testing (Adafruit VL53L0X)	2	$15	$30
Misc	–	–	$60
Shipping	–	–	$75
Total			**$394**

This leaves substantial headroom under budget for a second PCB revision, additional accelerometer parts if our first choice underperforms, or a CPR manikin rental if one isn't available through a campus organization.

Other Implementation Issues

Beyond schedule and budget, the main concern is manikin access for validation testing. We plan to reach out to organizations like the School of Nursing, Athletic Training programs, or student EMS and Red Cross clubs to borrow time on a training manikin rather than purchasing one outright. We don't see any significant mechanical challenges, as the enclosure is a simple hand-held puck that can be 3D printed on campus. Power isn't a concern either, since the device only needs to run for the duration of a CPR attempt or training session between charges. The one real technical risk is the depth-estimation algorithm itself. We're implementing it from a research paper rather than a commercial reference design, so our first attempt may underperform and need iteration. Because our schedule allocates time to work through this on the breadboard before finalizing the PCB, this risk stays manageable instead of turning into a late-stage blocker.

Evaluation and Testing

Our measure of success is quantifiable by how accurately our device estimates compression rate and depth against ground truth. Using a CPR training manikin, teammates will perform compressions across a range of target rates and depths while an independently mounted distance sensor records true displacement for comparison. We'll report our device's average estimation error in both metrics, aiming to match the accuracy shown in the literature (within about 1.5 compressions/min and 2mm of true depth), and also test challenging conditions (inconsistent pressure, brief pauses, extreme rates/depths) to see where accuracy breaks down.

Low-Cost Lecture Attention Detector

Lecture attendance does not equal lecture engagement. Students frequently disengage during class through phone use, prolonged head-down posture, or drifting attention, and current classroom tools have no way to detect or address this. This project proposes a small desk-mounted device that detects sustained disengagement and provides real-time feedback to the user.

The device sits on a desk facing the student and uses a simple camera or IR proximity sensor to track head position and orientation. Prolonged head-down posture, slow head droop, or extended gaze away from the front of the room triggers an alert, delivered through an LED, a buzzer, or a notification sent to the user's phone. A microcontroller processes the sensor data locally and drives the alert hardware, keeping the system compact and self-contained.

This project does not address a safety-critical need, but its value is that it is low-cost, quick to build, and still requires legitimate embedded systems work such as sensor integration, basic signal processing to distinguish head movement from disengagement, and a microcontroller driving real-time feedback. It demonstrates the same core skill set as a more complex project on a much smaller scale.

Implementation Issues

The main technical challenge is distinguishing normal note-taking behavior from actual disengagement, since both involve looking down. We would address this with a time-based threshold: brief head-down movements are ignored, while sustained head-down posture past a set duration triggers the alert. This avoids the more difficult problem of modeling attention directly, since the system only needs to detect prolonged lack of movement rather than judge whether someone is paying attention.

A benefit to this system is that the cost is also low, likely under $100 for a working prototype, using off-the-shelf parts such as an IR distance sensor (Sharp GP2Y0A21YK0F, ~$13), an Arduino Nano (~$20), and simple alert hardware like a piezo buzzer and LED (a few dollars each). Reliability and safety are not significant concerns, since a false alarm has no meaningful consequence.

A rough four-item schedule would be:

1. Get a camera or IR sensor communicating with a microcontroller and confirm it can track basic head position or presence.

2. Implement a time-based threshold algorithm to flag sustained head-down or head-away posture.
3. Add feedback hardware such as an LED, buzzer, or optional Bluetooth notification, triggered by the algorithm.
4. Package the system into a small desk-mountable enclosure and test it under realistic lecture conditions.

Evaluation and Testing

Success can be measured directly by having a teammate simulate both engaged behavior and disengaged behavior during a mock lecture session. We will then confirm the device correctly triggers on disengaged behavior while remaining inactive during normal note-taking. This requires no specialized equipment or outside access, only a controlled test scenario sufficient to validate the detection logic.

Smart Bike Module with Energy Harvesting

Cyclists face significant risks on shared roads, lacking the visibility and rear situational awareness inherent to motor vehicles. This project proposes a cohesive, wired embedded system featuring a rear facing camera, automatic brake lights, manual turn signals, and a custom power management system that harvests energy from solar panels and a hub dynamo to sustain the module indefinitely.

Details

The system utilizes STM32 microcontrollers to implement all features. It includes two modules. The rear module integrates a camera, LED signals, power management circuitry, and a Hall effect sensor to track wheel revolutions. By calculating rapid deceleration from the Hall sensor's speed data, the system automatically triggers the brake lights. The front handlebar module features an LCD display streaming the rear camera feed alongside a comprehensive bike computer interface showing current speed, average speed, trip distance, and overall ride statistics. To ensure reliability, the front and rear PCBs communicate exclusively via a hardwired connection, such as CAN bus.

Design Phases

Phase	Description
Planning and Parts	Select STM32 variants, camera, screen, and power components. Finalize draft proposal.
Breadboard Prototype	Configure STM32 video data transfer. Prototype solar/dynamo charging and Hall sensor speed calculations.
PCB Design	Finalize schematics and order PCBs, factoring in the standard manufacturing delay.
Software Development	Refine DMA for video buffers. Finalize speedometer UI and braking algorithms. Assemble PCBs.
Integration and Testing	Verify energy harvesting efficiency, calibrate speed/trip statistics, resolve bugs, and finalize enclosures.

Budget

With a team of 4, the $250/student allowance provides a total budget of up to $1,000. The numbers below serve as a rough estimate of parts cost.

Item	Quantity	Cost Per Unit	Total Cost
STM32 Microcontrollers	8	$8.00	$64.00
STM32 NUCLEO dev boards	2	$30.00	$60.00
Camera and LCD Modules	4	$25.00	$100.00
PCBs (Front and Rear)	10	$15.00	$150.00
Solar Panels and PMIC / BMS	4	$20.00	$80.00
Hall Effect Sensors and Magnets	4	$10.00	$40.00
Batteries	4	$10.00	$40.00
Misc Parts and Enclosures	1	$100.00	$100.00
Total			**$634.00**

Issues

- Energy: Safely regulating variable AC voltage from the hub dynamo alongside DC solar input to charge a battery without thermal issues.
- Deceleration Algorithms: Accurately calculating sudden speed drops from the Hall effect sensor data to trigger brake lights reliably without false positives from standard coasting.
- Video Latency: Optimizing STM32 memory and DMA buffers to maintain a continuous

video stream while concurrently processing other inputs.
- Hall effect sensor calibration: Ensure a convenient setup process for the user to allow for accurate speed calculation. Configure either polling or interrupt for continuous measurement.
- Mechanical design: Designing weatherproof enclosures and routing hardwired connections without interfering with steering.

Evaluation and Testing

Success will be measured by maintaining a negligible video feed latency and proving the power management system achieves a net positive battery charge during standard daytime riding. Real world testing must also verify that the Hall effect sensor accurately logs trip statistics against a commercial GPS device and triggers the brake light seamlessly during sudden stops.

Additional Stretch Goal

Integrating a rear facing ultrasonic or radar sensor to flash a blind spot warning on the handlebar screen when a vehicle approaches rapidly.

Smart Chiller Flask

Adding ice dilutes a drink, and traditional insulated flasks can only maintain the starting temperature. This project proposes a device that integrates a thermoelectric cooler (Peltier module) directly into the base of a custom insulated bottle. Contrary to similar solutions such as portable fridge or ice bath container, this design requires only carrying the sole container you can drink from that integrates cooling. The system features a simple user interface: a small external screen displaying the realtime liquid temperature and a physical button to activate the cooling cycle. An embedded microcontroller reads an internal thermistor and triggers a high current MOSFET to drive the Peltier module, actively pulling heat out of the liquid and exhausting it outward.

Design Phases

Phase	Description
Planning and Parts	Finalize core components (Peltier, battery, microcontroller, thermistor)
Breadboard Prototype	Breadboard the high current circuit and test the thermal dissipation rate of the Peltier.
PCB Design	Finalize schematics and order PCBs, factoring in the standard manufacturing delay.
Software Development	Integrate the populated PCB into the 3D printed flask base, seal against moisture, and finalize screen UI.
Integration and Testing	Final thermal and battery testing before the expo.

Budget

Item	Quantity	Cost Per Unit	Total Cost
STM32 Microcontrollers	2	$8.00	$16.00
Peltier Thermoelectric	2	$10.00	$20.00

Module			
Battery supporting high current draw	2	$25.00	$50.00
Display (0.96" OLED / LCD)	2	$8.00	$16.00
Power Electronics (MOSFETs, Drivers)	1	$15.00	$15.00
Heatsink, Thermal Paste, and Fan	1	$20.00	$20.00
PCB Fabrication and Enclosure	1	$150.00	$150.00
Total			**$287.00**

Issues

- Energy: Peltier modules are inefficient and draw significant current. Sizing a battery pack capable of sustaining the required amperage while keeping the flask portable.
- Thermal Management: Thermoelectric coolers simply move heat from one side to the other. Safely dissipating the heat from the hot side using a passive heatsink without burning the user's hand will require careful mechanical design.
- Waterproofing: Ensuring the custom PCB and battery are entirely isolated from condensation and accidental spills.

Evaluation and Testing

Success will be measured by the system's ability to actively lower the temperature of 500ml of room temperature water by a measurable margin within 15 minutes, without overheating the exterior components or immediately exhausting the battery.

Additional Stretch Goal

By replacing the single MOSFET with an H-bridge driver circuit, the system could reverse the current polarity to the Peltier module. This would allow the flask to alternate between cooling iced drinks and actively heating coffee or tea on demand

Form and Reps Tracker for Strength Training

Form and Reps Tracker for Strength Training

Strength training is heavily reliant on progressive overload and strict biomechanical form; however, most lifters track their progress using mostly notes apps or other commercial apps that require users to manually enter the number of reps, which are often forgotten by users midway through intensive training. While commercial gym equipment provides built-in rep counters, free weights (barbells, dumbbells, and kettlebells) remain completely unmonitored.

Similar ideas like this have been proposed and even implemented on different smartwatches, such as the Garmin vivoactive 5 and Amazfit bip 6, but many online user experiences reflect that these smartwatches are typically inaccurate, and they do not account for whether the users are deviating from the safety guidelines. Therefore, a more accurate and specialized weightlifting tracker/coaching system is required, as weight training is becoming more and more popular, with about one in three Americans now engaging in regular strength training compared to just one in five in the early 2000s.

Details

Imagine an edge-computing wearable that processes high-speed biomechanical data locally rather than streaming raw data to the phone. The hardware uses a 9-axis IMU (accelerometer, gyroscope, and magnetometer) to determine the absolute 3D orientation. Sensor data are streamed over the microcontroller via DMA, and once enough data are collected, a lightweight Edge AI model would automatically identify the specific exercise being performed, such as benching, squatting, or deadlifting, without any manual input from the user.

Once the type of exercise is identified, the firmware switches to coaching mode to monitor the form. It would compare the user's form to the pre-loaded perfect biomechanics, so that when the user deviates beyond the safe threshold, such as failing to reach full extension, the microcontroller would send an interrupt to end the user's music to indicate a warning. After the set, this device would package data and send it back to the phone via BLE payload to contain the exercise ID, total reps, average velocity, form score, and average heart rate, ensuring the device could be run for weeks on a tiny battery.

https://www.reddit.com/r/amazfit/comments/1o2uge4/the_bip_6_and_zepp_app_is_almost_useless_for/
https://www.nytimes.com/2026/02/02/well/move/weight-lifting-america-history.html

Schedule

9/10-9/23	**Planning** In the two weeks after the project groups are formed, brainstorm other approaches and start discussing the requirements of the processor and other hardware devices. Research two or three 9-axis IMUs and optical heart rate sensors and select the best fit(based on noise levels, power consumption, and footprint). Research the ideal biomechanics for a selected list of exercises(bench, squat, and deadlift) and establish methods of logging raw CSV data during gym sessions. Hopefully identify a microprocessor (or at least a family) with built-in BLE and sufficient RAM for Edge AI. Write the draft proposal by 9/18.
9/24-10/6	**Design and Part Selection** Once the proposal meetings are done, order at least two BLE dev kits, the selected 9-axis IMU, and the heart rate sensor. Bring up a prototype on a breadboard in minimal fashion(blinking lights, successfully reading raw sensor data over I2C/SPI, and establishing a basic BLE connection to a phone). Start taping the devkit to a wrist to log raw movement data at the gym to build the training dataset. Submit the final proposal by 10/1. All parts we think we need for the custom PCB prototype will have been ordered by the 4th.
10/7-10/25	**Breadboard Prototype and PCB Design** Get the prototype working with basic functionality: implement DMA streaming for the sensors, verify the heart rate monitor reads accurately during movement, and test the BLE media control commands (to ensure we can successfully interrupt/pause music on the phone). Begin training the lightweight Edge AI model using the collected CSV data. Design the custom wrist-sized PCB. Order PCBs by the 26th (a bit more than a week before the PCB deadline).
10/26-11/10	**Software Development and Testing on Prototype** While waiting for the PCBs to come back, complete the firmware algorithms on the devkits. Finalize the Edge AI exercise classification and form comparison logic. Ensure the system reliably triggers the music interrupt when a simulated “bad form” threshold is crossed. Have all functionality written and most tested by the 10th on the prototypes. Get PCBs assembled (hopefully correctly) by 11/8. Identify solutions to all non-electronic issues (designing and 3D printing the wristband enclosure) and get all things ordered on that front.
11/11-11/25	**Software Development and Testing on PCB-Based System** Testing PCB. Once the basics are working on the PCB, migrate the firmware and test to ensure that the Edge AI model and BLE payload transmission (exercise ID, velocity, form score, heart rate) run smoothly on the standalone PCB board. The accuracy of the IMU and Heart Rate Sensor will be tested to ensure the custom PCB layout did not introduce noise. Real-world gym testing

	will be conducted to find and fix bugs in the coaching mode. PCB issues resolved. If needed, PCB revisions will be made. Any additional features/stretch goals will be researched and decided upon this time.
11/26-12/9	**Final Testing, Additional Features, Poster and Expo** Targeting project completion 12/3. At this point, we'll get our poster done and start on the final report. Continue to fix any remaining minor bugs.
12/10-12/11	**Finish Final Report**

Budget

The numbers below are a rough estimate of the total cost, and it could act as a buffer in case other costs come up, such as broken components or extra functionality needed. PCB manufacturing costs might be underestimated. With a team of 4, we could be under budget.

What	Quantity	Cost Per	Cost Total
Nordic nRF52840	2	~$50	$100
Adafruit 9-DOF IMU BNO055	2	~$30	$60
M5StickC Heart Rate Hat (MAX30102)	2	~$40	$80
PCB Manufacturing and Assembly (JLCPCB)	1	$150	$300
Misc Parts	1	$100	$100
Shipping	1	$75	$75
Total:	$715		

Issues

The largest concerns at this time are

1. **Classification Accuracy** - The system must correctly identify the exercise being performed and accurately record the number of reps performed with > 90% accuracy across 5 different users. This issue is currently unknown, but it seems reasonable.
2. **Latency Feedback** - When a user's form deviates from the acceptable range, the music interrupt must be fired within 200 milliseconds. I believe this issue can be resolved through the custom DMA.
3. **Activation of Edge AI** - How does the Edge AI know when a set actually starts?
4. **Wrist Flexion** - Wrists are highly mobile, so it might track the user's arm movements when everyone lifts weights at a slightly different angle. We could solve this by applying the wristband lower on the arm, so it would not be affected by the flexion of the wrist.
5. **Sweat and Salt** - Weightlifting generates lots of sweat, which might impact the readings and accuracy of the IMU and heart rate monitor. However, adding coatings onto PCBs might solve this problem.

Additional Stretch Goals

If time allows, two wristband devices can be developed so that when users are performing independent dumbbell movements, each wristband can be used to count separately for each arm.

Active Snack Tracker

Active Snack Tracker

The obesity rate within the United States has become a problem. According to a 2024 study, the prevalence of obesity in adults was 40.3%, with the prevalence of severe obesity increasing from 7.7% to 9.7% between 2013 and 2023. In addition, mindless “doom-snacking” while studying is a universal college experience. Software-based diet trackers fail because they rely entirely on user discipline and honesty to manually log food.

This Active Snack Tracker physically enforces portion control through a purely embedded approach. The custom PCB acts as the base of a 3D-printed bowl. It uses a high-precision load cell to track exactly how much weight, or “snack,” is removed. In addition, it integrates a microphone and a ring of capacitive touch sensors around the rim to track human activities. The onboard processor runs a TinyML acoustic classification model that listens for the specific sound of “crunching” chips.

If the bowl detects you eating too fast or if it hears you crunching a snack you didn’t legally grab from the bowl, it warns the user by firing a high-pitched frequency from the speaker and sends a signal back to the user’s computer through Wi-Fi to black out the computer screen until the user steps away from the bowl for 10 seconds.

Schedule

Sept/Oct	Research and test the different components for the load cell, microphone, and microcontroller. Prototype the load cell and microphone on a microcontroller. Collect hours of audio data to train the acoustic TinyML model to distinguish crunching and ambient dorm noise.
Late Oct	Route the custom PCB. The board must be circular to act as the bowl’s base, with capacitive touch traces routed along the perimeter. Order the PCB.
Nov	While waiting for the PCB, test the audio feedback and punishment logic. Assemble the PCB. Calibrate the capacitive touch baseline algorithm to prevent false triggers from room humidity. Integrate everything. If needed, redesign and order a new PCB.

https://www.cdc.gov/nchs/products/databriefs/db508.htm

Dec	Finalize real-world testing with different scenarios of “cheating” in a noisy environment.

Budget

What	Quantity	Cost Per	Cost Total
ESP32-S3 Wi-Fi SoC	2	~$8	$16
1kg Load Cell	2	~$10	$20
HX711 (24-Bit ADC)	2	~$10	$20
INMP441 I2S Microphone	2	$10	$20
Audio Amp & Speaker (pre-built ones)	2	$50	$100
Custom PCB	1	~$100	$100
Misc Parts	1	$100	$100
Shipping	1	$75	$75
Total:	$451		

Issues

1. **False positives from the microphone** - It might be very difficult to distinguish a chip crunch from a cough, talking, or other factors within a typical study room/dorm. A filter must be applied to filter out all the useless frequencies before passing the data to the TinyML model.
2. **False positives from capacitive touch sensors -** Capacitive touch sensors are highly sensitive to changes in room humidity and temperature. A regular baseline calibration might be needed to prevent false positives, like when the bowl thinks a hand is hovering when no one is there.

3. **Complex PCB layout** - A load cell paired with a Wi-Fi SoC on a single PCB might cause conflicts and fluctuate the weight readings wildly. Strict routing must be implemented on the PCB.

Smart Bag Departure Monitor

Have you ever arrived at a new location only to realize that you left an important item behind? When people are in a rush, they may not have time to check every item in their bag before leaving. By the time they realize something is missing, they may have already traveled through several locations, making it difficult to determine where the item was lost.

The Smart Bag Departure Monitor is designed to prevent this problem by notifying users when they're about to leave with an item behind. The system consists of a central device attached to the user's bag and small BLE tags attached to items the user wants to monitor. When the bag is closed and moving, the system performs an inventory check. If a required item is missing, the central device provides an immediate vibration alert so the user can address the problem before traveling farther from the item. The tag attached to the missing item can also activate an LED and audio alert to help the user locate it.

This system is particularly useful in situations where departure is time-sensitive, such as emergency response or field work. For example, a firefighter may perform a standard equipment check before responding to an emergency, but an automated final check could provide an additional layer of verification immediately before departure. Rather than replacing an established checklist, the system acts as a final automated confirmation that the required equipment is present.

Existing products such as Airtag are designed primarily to help users locate individual items and provide separation alerts [1]. The Smart Bag Departure Monitor focuses on verifying that a set of required items is physically contained within a bag at the moment they start leaving. This creates an embedded-systems challenge in determining whether an item is inside the bag rather than simply whether it is nearby.

Details
The system uses an RF-shielded bag with two BLE receivers: one located inside the bag and one outside the bag. With the RF isolation from the bag, the item inside the bag should be detectable by the internal receiver while the item outside the bag should be detectable by the external receiver.

The bag only performs inventory checks only when the user begins to depart, identified with these two conditions:

1. Zipper is closed.
2. The bag starts moving.

[1]
https://www.apple.com/newsroom/2026/01/apple-introduces-new-airtag-with-expanded-range-and-improved-findability/?utm_source=chatgpt.com

A Hall-effect sensor and magnet will be used to detect the zipper state.A magnet will be attached to the zipper pull while the Hall-effect sensor will be attached near the fully closed position. When the zipper is closed, the magnet will move closer to the Hall-effect sensor and ultimately detect that the zipper has been closed.

An IMU with wake-on-motion interrupt will detect when the bag begins moving. While stationary, the system will remain in a low power state rather than constant monitoring to reduce battery consumption. When motion is detected, MCU will wake up and read the Hall-effect sensor to determine a potential departure. This helps prevent unnecessary inventory checks caused by events such as the bag being accidentally kicked or moved while the user is still at the same location.

When both conditions are satisfied, the system will perform an inventory check. If there is an item that is located outside the bag, the central device will activate a vibration motor to alert the user. The tag attached to the forgotten item will activate its LED blinking and audio playing to assist the user to find the item with more ease.

A primary design requirement is long battery life. The target is at least six months of operation without charging. Achieving this will require low-power operation of both the central bag device and the individual BLE tags, introducing sleep modes, low duty cycle BLE communication, and event driven sensing.

Schedule

9/10 - 9/23	**Planning** Explore other approaches that can work for item containment (using UWB and distance estimation). Research existing products and approaches for preventing or detecting forgotten items. Identify major technical risks and establish initial system requirements. Draft proposal due 9/18
9/24 - 10/1	**Design and Part selection** Research and select parts. Pay particular attention to power consumption for the six-month battery life requirement. Begin breadboard prototyping. Order Faraday Bag and BLE modules early to test RF containment before finalizing system design. Final Proposal due 10/1
10/2 - 10/23	**Breadboard prototyping, PCB & Casing design** Test individual components and integrate on a breadboard. Confirm all major components operate independently before full integration. Design custom PCB. Place PCB order before 10/23 (approx. 10 day delay for PCB to arrive). Design and 3D print the case for the item tags. Milestone 1 report due 10/15
10/24 - 11/6	**Software development, Integration, and PCB Testing** Complete the software required for the integrated breadboard prototype. Test the complete system and begin testing custom PCB as soon as it

	arrives. Characterize the accuracy of detecting whether an item is inside or outside the bag with the casing on. Last day to order PCB is 11/3.
11/7 - 11/20	**Full System Testing and Debugging** Evaluate the complete system under a range of operating conditions. Measure inside/outside detection accuracy, false-positive and false-negative rates, departure detection reliability, battery consumption, GPS accuracy, and cellular connectivity as applicable. Debug and address remaining hardware and software issues.
11/21 - 12/8	**Finalization** Finish up the project. Target to finish the project by 12/3. **Design Expo 12/8**
12/8 - 12/11	**Final Report** Final report due 12/11.

Budget

Part selection isn't finalized. This is a rough budget calculation after doing research for essential components based online.

Item	Quantity	Unit Cost
Faraday Backpack	1	$140.59
XIAO nRF52840 Sense	5	$15.90
Nordic nRF52840-Dev Kit	1	$48.95
IMU	1	$14.95
Hall-effect sensor + magnets	2	$3.00
Coin vibration motor	2	$2.00
Buzzer / LEDs / buttons	1 set	$10.00
Li-Po battery	6	$15.00
Battery charging / power-management components	6	$20.00
Custom PCB fabrication	5–10	$100.00
Connectors, passives, wiring, miscellaneous	1 set	$50.00
Total		$663.99

Issues

1. RF containment / inside-vs-outside detection
2. TImely departure detection & scanning
3. Power consumption and battery life

1 is the biggest technical challenge. If RF containment proves insufficient, UWB based ranging or BLE distance estimation can be explored as an alternative. It will provide a good estimate of how far away the item is from the moving bag and give alert in relatively short notice.

The system should perform an inventory check as soon as the user begins to depart to ensure that missing items are detected while they can still be retrieved. However, continuously scanning for BLE tags would waste energy. Additional research will be needed to develop a reliable departure-detection method that triggers an inventory check at the appropriate time while minimizing unnecessary wireless activity.

Additional Stretch Goal

The user could configure which tagged items are considered required for a particular trip or activity. This would allow the same system to be used for different situations without requiring every tagged item to be present for every departure.

For example, a user could configure a work inventory containing a laptop, badge, and charger, while a travel inventory could require a passport, wallet, and headphones. The system would only alert the user when an item required for the current inventory is missing.

Customizable Smart Window for Circadian Cycle

Our daily schedules do not always match the natural day-night cycle. Someone may need to wake up before sunrise, sleep during the day, or go to bed while it is still bright outside. This can make it difficult to use natural light as a cue for waking and sleeping.

The Customizable Smart Window is designed to create a programmable day-night cycle by automatically controlling both the physical curtains and artificial light from the window. The system allows the user to set a desired wake-up and sleep time. Instead of immediately opening or closing the curtains, the system simulates natural sunrise or sunset by gradually changing the amount of light entering the room. This could help users to establish a more consistent environmental cue for waking and sleeping, especially when their schedule doesn't align with the natural clock.

Details

The window will use multiple curtain layers with different light-transmission properties. One layer will be an opaque blackout curtain, while the other layers will be progressively more translucent. By moving the layers individually, the system can control how much natural light enters the room.

The user will configure a daily wake-up and sleep schedule using a low-power monochrome LCD display and four buttons. The display will show the wake-up time on one side and the sleep time on the other. Left and right buttons will move the cursor between individual time fields, while up and down buttons will increase or decrease the selected value. The interface will allow the user to configure the hour, minute, and AM/PM setting for both wake-up and sleep times.

For example, the display could show:

WAKE-UP	SLEEP
7 : 3 0 AM	8 : 3 0 PM

← ↑ ↓ →

Thirty minutes before the user's scheduled wake-up time, the system will begin a gradual sunrise sequence. Using a light sensor, the system will determine if there is enough ambient light outside the window. If there is sufficient natural light available, the system will gradually remove the curtain layers to increase the amount of sunlight entering the room. If not, the system gradually introduces light into the room by increasing the brightness of the LED light source integrated into the window.

The controller will use both the current time and measured ambient light level to determine the appropriate response. Time information will be an important cue to distinguish the dark night time environment from the early morning environment before the sunrise. This will help power consumption by using LED systems only when necessary.

The evening sequence operates in the opposite direction. Approximately 30 minutes before the user's scheduled sleep time, the system gradually decreases the amount of light entering the room. The system will read the current light level, user's sleep and wake-up schedule, and expected outdoor lighting conditions when determining how to control the curtains.

A simpler approach is to record the sunrise and sunset time for every week in the day. However, Sunrise and sunset information obtained through an online API could be used to improve this prediction. With a Wi-Fi connection, the system could obtain the expected sunrise and sunset times for the user's location and use this information to determine whether natural light is likely to be available during schedule transitions. This would reduce unnecessary curtain movement.

Schedule

9/10 - 9/23	**Planning** Research existing products and approaches for preventing or detecting forgotten items. Identify major technical risks and establish initial system requirements. Draft proposal due 9/18
9/24 - 10/1	**Design and Part selection** Research and select parts: light sensor, MCU, motors/actuators, LED lighting system, and curtain materials. Begin breadboard prototyping. Explore APIs for sunset and sunrise data from online. Final Proposal due 10/1
10/2 - 10/23	**Breadboard prototyping, PCB & Casing design** Build a prototype of the light-sensing and LED-control system. Develop curtain mechanism and test individual layers. Design custom PCB. Place PCB order before 10/23 (approx. 10 day delay for PCB to arrive). Design and 3D print the case for the item tags. Milestone 1 report due 10/15
10/24 - 11/6	**Software development, Integration, and PCB Testing** Implement the scheduled sunrise and sunset sequences. Integrate all parts on prototype breadboard and to PCB. Begin testing custom PCB as soon as it arrives. Last day to order PCB is 11/3.
11/7 - 11/20	**Full System Testing and Debugging** Evaluate sunrise and sunset transitions under different lighting conditions. Measure light-level changes, transition timing, curtain positioning accuracy, and LED brightness control. Test repeated operation at different times and identify mechanical or software failures.
11/21 - 12/8	**Finalization** Finish up the project. Target to finish the project by 12/3. **Design Expo 12/8**
12/8 - 12/11	**Final Report** Final report due 12/11.

Budget

The final component selection will be adjusted to keep the project within the available team budget while maintaining the required functionality.

Item	Quantity	Unit Cost
Microcontroller development board	1	$15
Ambient light sensors	2	$5
Servo/stepper motors and drivers	7	$10
LED lighting components (HUB75 tiled)	20	$10
Monochrome LCD display	1	$13
Buttons (4)		$2
Blackout curtain	1	$24
Translucent curtain	6	$13
Curtain rails / mechanical components	7	$27
Custom PCB fabrication	1 batch	$100.00
3D-printed/mechanical components	—	—
Power supply and wiring	1 set	$30
Miscellaneous components	—	$50
Estimated Total		$765

Issues

1. LED lighting and power consumption
2. Mechanical curtain control

The LED system must produce enough light to create a visible gradual sunrise or sunset effect while maintaining reasonable power consumption. The current plan is to use tiled LED boards to create a window-like display and simulate natural lighting. The power consumption of this approach will be analyzed during prototyping, and alternative LED configurations will be considered if the required brightness results in excessive power consumption.

The system will use a total of seven independently controlled curtain layers. Installing all of the layers in a compact window structure while allowing them to move independently is a significant mechanical challenge. The curtains must not interfere with one another during repeated operation, and their positions must be predictable enough to produce consistent changes in light transmission. Different curtain hanger, rail, and actuator configurations will be explored during prototyping. Custom brackets or other mechanical components may be 3D printed if necessary.

Additional Stretch Goal

The system could connect to Wi-Fi and obtain sunrise and sunset information from an online API for the user's location. The controller could combine this information with the user’s schedule and measured light level to predict whether natural light will be available during a

transition. This could further reduce unnecessary LED usage and curtain movement while making the system more adaptable to seasonal changes.

Project Proposal #1 - Low-Cost labor detection for livestock

High-level Description

During calving or lambing season, farmers may need to monitor many pregnant animals at the same time. Births can happen overnight or while the farmer is doing other work, and constantly checking every animal is not realistic. If labor begins and complications occur without being noticed, both the mother and offspring may be at risk.

This project would build a low-cost wearable sensor system that helps a farmer identify which animals may be entering labor and should be checked. Each animal would wear a small sensor node near the tail or rear of the body. The node would primarily use an IMU to monitor movement and orientation. It could look for behavior changes such as repeated tail raising, increased movement, or repeated posture changes.

The goal would not be to predict the exact time of birth. Instead, the system would detect unusual behavior that may be associated with labor and notify the farmer that a specific animal may need attention.

Each wearable node would include a microcontroller, IMU, wireless communication, battery, and custom PCB. The nodes would process sensor data locally and send short status updates to a central receiver. The receiver would keep track of several animals and show which ones are behaving normally and which ones have triggered an alert.

This would allow one farmer to monitor many animals without having to constantly observe each one.

Implementation Issues

The main sensor would likely be a low-power accelerometer or IMU. The device could track features such as tail angle, number of tail raises, overall movement, and how often the animal changes position. A simple threshold- or state-based algorithm would probably be used first, with machine learning as a possible stretch goal.

The wearable nodes would need enough wireless range to work across a barn or nearby pasture. Since only small status packets need to be sent, a low-power long-range wireless system such as LoRa or another sub-GHz radio would be a good option.

Battery life is also important because having to charge many devices every day would make the system less useful. The node could reduce power use by processing data locally, transmitting only periodically, and using low-power modes when possible.

The mechanical design may be one of the more difficult parts of the project. The device needs to stay attached while the animal moves, while also being protected from water, dirt, and impacts. Early prototypes could test different strap and enclosure designs.

The biggest project limitation is that we may not have access to pregnant livestock during the semester. Because of this, the project would focus on detecting repeatable motion and behavior patterns rather than claiming that the system has been fully validated to predict real births.

Schedule

1. Research and prototype sensing: Research measurable behavior changes around calving/lambing and test IMUs and wireless hardware using development boards.

2. Develop detection algorithm: Collect sample motion data and implement basic features such as orientation, activity level, and repeated movement detection.

3. Wireless system and PCB design: Get multiple prototype nodes communicating with a receiver and design the custom wearable PCB.

4. PCB fabrication period: While waiting for the boards, continue firmware development, testing, and enclosure design.

5. PCB bring-up and integration: Assemble and test the custom boards, move the firmware onto the final hardware, and integrate the battery and enclosure.

6. Final testing: Test motion detection, false alarms, wireless range, battery life, and mechanical durability.

Rough Budget

A team of five would have roughly $1,250 available based on the $250-per-student guideline in the assignment.

A rough estimate would be:

- Custom PCBs and possible revision: $150
- Microcontrollers and IMUs: $100
- Wireless hardware: $125
- Batteries and power circuitry: $100
- Central receiver: $75
- Development boards and prototype parts: $100
- Enclosures and mounting hardware: $125

- Mechanical test setup and miscellaneous parts: $100

Estimated total: about $875

This leaves some extra budget for replacement parts or a PCB revision.

Evaluation and Testing

The system could be tested using a mechanical setup that moves the wearable through known positions and motion patterns. These could include tail down, tail raised, repeated tail raising, rapid movement, random movement, and inactivity.

The detected states could then be compared to the known motion sequence to measure accuracy.

The device would also be tested with movements that should not cause an alert, such as shaking, vibration, impacts, or brief changes in orientation. This would help measure the false-alarm rate.

Wireless testing would measure communication range, packet loss, and alert latency with several nodes operating at once.

Battery current would be measured during sensing, processing, transmission, and low-power operation to estimate runtime.

The enclosure and attachment system could also be tested using repeated shaking, pulling, drops, and water exposure.

A successful prototype would be able to reliably identify predefined motion patterns, communicate alerts from multiple nodes, operate for a useful amount of time on battery power, and survive basic mechanical testing.

Stretch Goals

If the main system works reliably, possible stretch goals include learning each animal's normal activity baseline, using a lightweight machine-learning classifier, adding another sensor such as temperature, or forwarding alerts to a phone.

Project Proposal #2 - Low-Cost CPR Compression Quality Monitor

High-level Description

High-quality CPR depends heavily on the person performing the chest compressions. If compressions are too shallow, too slow, too fast, or the chest is not allowed to fully recoil, CPR becomes less effective. During CPR training, an instructor can usually watch only a few people closely at a time, making it difficult to give detailed feedback to a larger group.

This project would build a small embedded device that measures CPR compression quality and gives immediate feedback to the person performing CPR. The device could either sit on top of a training mannequin's chest or be integrated into a pad placed where the rescuer's hands normally go.

The device would measure things such as compression depth, compression rate, and whether the chest returns fully between compressions. A more advanced version could also estimate how consistently force is being applied or whether the person's hand placement is moving away from the center of the chest.

The main goal would be to create a low-cost system that could be used with basic CPR training mannequins instead of requiring an expensive mannequin with built-in electronics. This could make objective CPR feedback more available for schools, community training programs, or other organizations that need to train many people at once.

The final system would include a custom PCB with a microcontroller, motion or displacement sensing, force or pressure sensing, and some form of immediate feedback such as LEDs, a buzzer, or a small display. A wireless connection could also allow several CPR stations to report results to one instructor.

Implementation Issues

The main challenge would be measuring compression depth accurately. One possible approach would be to use an IMU to estimate vertical motion during each compression. However, IMU measurements can drift when position is estimated from acceleration alone, so another sensor such as a distance sensor, force sensor, or pressure sensor could be added to improve accuracy.

The system would also need to identify the beginning and end of each compression. Once individual compressions are detected, the microcontroller could calculate compression rate, average depth, variation in depth, and whether the chest returned to approximately its starting position.

The device would need to be strong enough to handle repeated compressions without damaging the electronics. The enclosure would also need to spread the user's force so that the sensor readings are consistent and the device is comfortable to use.

One possible issue is that commercial CPR feedback devices already exist. Because of this, the main focus of the project would be making a system that is inexpensive, portable, and able to work with basic training equipment. A system that allows one instructor to monitor several CPR stations at once could also make the project more useful and different from a single standalone feedback device.

Schedule and Budget

The project would begin by testing different ways of measuring compression depth and force using development boards and a CPR mannequin or spring-loaded test fixture. Once a sensing method is chosen, the team could develop the compression-detection firmware and begin testing depth and rate accuracy.

The next step would be designing the custom PCB and enclosure. While waiting for the PCB to arrive, the team could continue improving the firmware and build a test fixture for repeatable compression testing. After PCB bring-up, the final system would be tested for accuracy, durability, and repeatability.

The project should fit comfortably within the course budget. A CPR training mannequin could likely be one of the larger expenses, along with custom PCBs, sensors, development boards, enclosure materials, and wireless hardware if that feature is included. A rough total budget of around $500–800 should leave enough room for spare components and a possible PCB revision, which is within the $250-per-student budget described in the assignment.

Evaluation and Testing

This project would be straightforward to test because CPR compressions can be reproduced safely using a mannequin or mechanical fixture.

The device could be tested using compressions with known depths and known rates. For example, the fixture could be compressed by several different amounts while the device's depth measurement is compared against a ruler, displacement sensor, or mechanical stop.

Testing could also intentionally create poor CPR technique, such as:

- compressions that are too shallow
- compressions that are too fast or too slow
- incomplete recoil
- inconsistent compression depth

- pauses between compressions

The system’s output could then be compared against the known test condition.

The final project could be evaluated based on compression-depth error, compression-rate accuracy, ability to detect incomplete recoil, false detections, and durability after repeated use. If wireless communication is added, multiple CPR stations could also be tested at the same time to make sure alerts and results are correctly reported to the instructor.

A possible stretch goal would be to add rescuer-fatigue detection by looking for a gradual decrease in compression depth or consistency over time.

Universal Recycle Bin Helper

High Level Description

In a world where everyday we get closer to the point of no return for climate change, sustainability initiatives are becoming more important and vital for the survival of our planet and humanity. A major effort towards decreasing the amount of waste and emissions in the environment is recycling, a process and idea that isn't new but still is a factor that helps greatly in reducing carbon emissions and physical waste in our environment.

The main issue with recycling is that often the public is ill-informed on what can be recycled, and this causes the U.S. waste management system more than $3.5 billion a year[1]. Contamination is a major issue for recycling centers as well. The contamination rate in the U.S. is around 25%, meaning that whatever trash bag contains the contamination has to be thrown away which wastes actual recyclables in the bag[2]. There is also the issue of different cities having their own rules on what can be recycled and what their facilities can take. Aiding people in not having to study the rules on what can be recycled is another great step towards more efficient recycling.

To fix this issue, a self sorting attachment can be developed and placed on top of commercial, city, and personal trash bins, where you can throw away anything and the system itself will sort what can be recycled or thrown away. This takes the burden off of the individual, and ensures the recycling facilities that the garbage they're getting is actually recyclable.

Details

The vision for this product is for it to attach onto the top of a connected trash can with recycling and garbage right next to each other, and will have a system that checks what the item is and based on internal calculations it'll dump the item into one of the bins. There would be a weight sensor as well as a camera connected to the microcontroller, and through the data collected the microcontroller will decide what the item is, and following that will see if it's recyclable or not.

Multiple modes can be created for different cities and their own recycling laws, but that would be hardcoded based on the current legislation and rules. The goal is also for this product to be accessible and cheap so that companies and cities can buy in bulk as well as be available for individuals. There would be multiple cameras to create a 3D understanding of the item so that the program can be more accurate on identifying the item.

Schedule

The listed dates are based on the school year and can be changed but the timeline should be reasonably followed.

9/11 - 9/25	***Planning*** After groups are formed, rough outlines of the project need to be created, by beginning to pick out specific parts and whether or not the ones chosen for the

[1]https://routeware.com/blog/whats-the-real-cost-of-contamination/
[2]https://ecorithms.com/blog/recycling-contamination

	budget below match the needs of the project. Research recycling laws and the feasibility of Computer Vision and what methods can be used to accomplish this goal. Also find what other microprocessors could be used and what the storage needs would be. Draft proposal done by 9/16.
9/26 - 9/30	***Purchasing of Parts*** This should go along with the findings from the Planning portion of the project, with some parts already being identified. Definitely order the camera and servo, so when they arrive testing can be done as soon as possible to integrate with the final chosen microprocessor. Use a Raspberry Pi 5 to create a mini prototype that involves basic Computer Vision and also able to move servo. Final Proposal done by 9/30.
10/1 - 10/16	***PCB Planning and Breadboard Prototype*** Get the prototype to work through breadboarding by having the camera identify objects(just seeing different objects, not naming them) and moving the servo in response to it, while in parallel learning about PCB design and designing the PCB for the final product.
10/17 - 11/7	***Development of Computer Vision and Recycling Data, Testing on Breadboard*** In this time period develop a whole working program that is able to begin identifying different items shown to the camera, and moving the servo based on what is identified. A chosen set of laws on recycling should be used for the basis of the project and will be detailed during this time. Computer Vision should be able to identify 20 of the most common recyclable items. Accuracy should be 75% for 100 items being shown. Figure out the power management system for the solar panels and rechargeable battery in parallel. Order PCB by 10/25(room for design changes)
11/8 - 11/22	***Development of Software and Testing on PCB system*** Begin testing on the PCB and ensuring the power system works to power all of the Computer Vision capability and the servo itself. Debug and check for any basic issues in the PCB. Also time period for any revisions to the PCB(too big, too small, not capable for power usage, etc.), but should do the base test listed above to ensure it's ready to be implemented with the physical part of the attachment.
11/23 - 12/10	***Final Testing and Development, Design Expo, and Poster:*** Computer Vision should reach a rate of 90% correct decisions on recyclable items for 100 items. Should identify 25 of the most common recyclable items. Work on poster and design expo demonstration. Goal of done with project 12/5

12/10 - 12/11	***Finish Final Report***

Budget

The budget for this project is based on rough estimates of costs of the individual peripherals and items needed for the final result. The final budget shouldn't be too far off unless additional features are added or specific peripherals that are needed are more expensive. For the chip, the BroadCom BCM2712 from the Raspberry Pi 5 was chosen for CV capabilities.The overall project should be cheaper, with a possibility of getting a lower memory chip for the CV option, and a lower load servo.

Item	Amount	Cost Per	Total
BroadCom BCM2712 4GB	1	$80	$80
Adafruit HX711 and Load Cell(weight sensor)	1	$15	$15
20kg Servo	1	$30	$30
Camera Module	2	$30	$60
12V rechargeable Li battery	1	$30	$30
12V solar panel	2	$30	$60
PCB cost	1	$10	$10
Exterior case	1	$50	$50
Shipping	1	$100	$100
Total			$435

Issues

1) Computer vision can be difficult from scratch
2) Preloading rules for recycling
3) Differences in weight for trash thrown away
4) Universal capability for attachment

Issue 1 is a main problem due to the fact that there's most likely a lack of experience with developing a computer vision software from scratch, however there are many resources

specifically for Raspberry Pi and development of computer vision. Issue 2 is less of an important issue in making the program work with the rules given from the city laws, while looking at the laws themselves is time consuming. Issue 3 is a limitation where the product may have to have a weight limit for use simply based on the weight limit of the sensor. Issue 4 is a large issue for marketability by ensuring any current trash bin can be used with the product. The biggest issue would be the computer vision development, but this can be achieved with the amount of resources available and outside of that issue the project is very feasible and could be a gamechanger in sustainability.

Hockey Heads Up Display Trainer

High Level Description

Professional sports are constantly changing around us, with better technology on the field of play, more accurate statistics, and advancements in accessibility for the general public to join in on the fun. The sport of ice hockey is one of those that is constantly evolving and attempting to please greater audiences. However, a main accessibility issue with hockey is the cost of the game itself, with parents having to often spend more than $1000 just for gear for their child to play. This doesn't even include the cost for an ice rink, recreational leagues, and the transportation to the closest rink. With money being a major limiting factor to widen the sport to everyone, a solution must be found to open accessibility.

Details

This solution would be a single device that could be used on or off ice and can be attached to any helmet. Within this device is a heads up display that mimics a full trainer for an aspiring hockey player which would only require the individual to have a stick, some pucks, and a surface to shoot at. It would simulate in game situations, from shooting at the net to passing to your teammates, and would allow many more people to get into the sport without having access to a rink and all the expensive equipment.

The full product would have speakers, a heads up display, an outward facing camera, and a bluetooth device to connect to the users phone, saving any important information and data to improve the users experience and skills.

Budget

Items include the Raspberry Pi 5 microcontroller, as well as needed items for the HUD and speakers feedback.

Item	Amount	Cost Per	Total
BroadCom BCM2712 4GB	1	$80	$80
High Brightness Display	1	$50	$50
HUD Reflector	1	$15	$15
Camera Module	1	$30	$30
Bluetooth Device	1	$20	$20
Power Supply	1	$20	$20

Speakers	2	$15	$30
64GB card	1	$10	$10
PCB cost	1	$10	$10
Shipping	1	$100	$100
Total			$365

Issues

1) Interfacing with HUD
2) Accuracy of environment tracking
3) Processing Power

Issue 1 is due to lack of experience with a HUD from most students, and how quickly it can be learned. There also is the question of how large the HUD should be so that the user can use it effectively for training purposes. Issue 2 is the largest issue, with how accurately the microprocessor can follow objects such as the puck and how consistent it can be with putting targets on an unidentified surface and keeping them in the same place. Issue 3 is important since this does seem like a highly intensive processing project, which has capabilities that mimic VR equipment, and may not be possible in one semester. However, there is a resource available online for Raspberry Pi 5 implementation with a HUD, though it may not be as in depth as needed for assistance on this project.

Advanced Embedded Systems Project Proposals

Idea 1: swimming pool lap counter watch

High-level description

We want a simple smart-watch that can support daily/weekly swim practice needs, and a phone app that syncs data with the watch to help track and visualize long-term fitness progress.

Every lap swimmer who has ever lived, has turned to the other person in their lane and asked "what lap are we on? How many laps do we have left?". There is something pesky in the nature of humans. We love puzzles, we build complicated machines, and we coordinate so well and so precisely that we can play in orchestras, yet in the business of counting, we can often grow lazy and forgetful, even if it matters to our own physical fitness.

I want to build a swim watch that lets the swimmer focus on their swim stroke and breathing, not on their exact counting and overall workout. Less counting lap numbers, or back-calculating my pace, or writing it all down week after week to see if they are getting faster. I want a watch and phone app that makes it simpler to meet short-term practice goals and long term fitness goals.

Watch Features

1. Swim Activity Tracker

The Swim Activity Tracker feature is the live monitoring of swim exercises that tells the user their pace and distance, which controlled by the user, and provides simple feedback for their activity.

1. User Interface
    - Users can control when to Start/Stop Swim Activity
    - Users can see live pace & distance traveled since start of swim activity
        - Distance is reported in **"laps swam"**, and **"X Yards/Meters"**
    - Users can see live pace & distance traveled for all **Swim Sets**. I.e, periods of exercise separated by periods of **Rest**.
        - There should be views so users can easily see this data for the last three **Swim Sets**, and for all **Swim Sets**
2. Data Tracking
    - Data is stored in chunks called **Swim Activities**
        - Swim Activities are manually started/stopped by the user
        - Swim activities store raw speed and distance traveled by the user
        - Swim activities are processed to show user specific metrics like laps swam, and pace over different intervals.
        - The last 6 hours of swim activities shall be stored locally (on watch), with a Sync to App feature to offload this data
3. Battery Management
    - Over temperature protection

    - under voltage & over voltage protection (if lithium ion is selected).

4. User Settings and Definitions

    - [User Configurable] **Lap distance** (yards/meters)
    - [User Configurable] **exercise threshold** (minimum speed & distance to determine swimming vs resting)
        - i.e: above this threshold you are swimming, below it you are resting.
    - [User Configurable] **Rest Time**

5. Terms / Definitions

    - A Rest is a >[REST_TIME] seconds pause in activity
        - A pause in activity shall be measured as a lack of movement above a certain speed over a certain distance (exercise threshold).
    - Once a rest is detected, it requires a start of activity to consider that rest over. This is movement greater than [CONFIG: MIN EXERCISE SPEED] ft/s. We will need to find the slowest velocity while swimming we want to consider exercise.

2. Other Smart Watch Features [STRECH GOALS]

1. Clock

    - shows the current time (Face / digital)
    - syncs via bluetooth

2. Heart-Rate Monitor

    - When activated, takes active heart-rate measurements that need to be within 5% of actual heart-rate
    - only takes measurements for 30 seconds before deactivating.

4. System/Mechanical Features

- Smart Watch
    - 10-12+ battery life
        - -> (drives low power usage requirements)
    - Comfortable wrist-strap
    - Waterproof up to 15-30 meters
    - nice display

Swim Activity App

1. Sync to App

    - Allows the swim watch's data to be synced to the user's smartphone and become available for the other Swim Activity App features

2. Swim Activity Visualizer

    - Graph of a Swim Activity Event

        - A line graph of pace versus time that shows all laps and rest-periods taken.
        - Data gathered will be slow measurements, 2-3 second sample
        - manual/automatic sync of data via bluetooth to app.

3. Activity history

    - shows all swim activity events (DATE-TIME, LAPS)

4. Progress Metrics

    - [GOAL] shows synthesized results of how a swimmer is progressing
    - Graphs of different swim sets and their pace numbers over time
        - (EASY) 1-lap (50yrd) Pace time over X months
        - (EASY) 2-lap (100yrd) Pace time over X months
        - (EASY) 4-lap (200yrd) Pace time over X months
        - (HARD - STRECH-GOAL) FREESTYLE/BACKSTROKE/BREASTSTROKE/BUTTERFLY Pace time over X months

Budget / BOM

Phone App

Item	Cost	Effort	Source
Iphone &/or Android App	$0	Med/High.	Team built Flutter or React Native App
On-Device Storage	$0	Low.	Team built Solution
Cloud Data Storage	$0	High.	Team will avoid cloud based solution unless needed

Watch Device

Item	Cnt	Cost	Source
Wrist strap	1	$5-$10	Team to designed & 3D print
MCU: Rasberry Pi type	1	$20-$50	DigiKey, Mouser, etc. Custom PCBA to reach 2inches x 2inches
Bluetooth Chip	1	$5-$15	DigiKey, Mouser, etc. Custom PCBA to reach 2inches x 2inches
Acceleromter	1	$5-$20	To be researched. DigiKey, Mouser. Example: ST's LIS3DH
Li Battery	1	$5-$20	To be researched. Li-ion or Li-Po, waterproofing and safety considerations.
1.5-2 inch Display	1	$0	To be researched. Example: waveshare 2inch LCD Display Module.
Watch Case	1	$5-20	Team to designed & 3D print
Epoxy	1	$50-$110	MG Chemical or similar, need appropriate workspace

Link for example Epoxy we may use for waterproofing.

Schedule

9/10-9/24 Planning

- Decide system requirements, such as power consumption, and desired packaging/size. These will drive component and MCU selection.
- Sept 18th, project draft proposal due.
- Sept 24th/25th, staff discussion:
    - we show that we understand how to use an accelerometer to measure distance traveled, our plan to waterproof our device, and basic evidence that we can squeeze some version of our parts list into a watch sized package.

9/25-10/9 - Design and part selection

- An early mechanical design of our components to ensure they fit into the watch, because this project will have a tight packaging requirement.
- Oct 1: Formal Proposal, showing our packaging, our procedure for waterproofing, a block diagram showing our data monitoring, etc.
- Order dev-boards and create walkable prototype. Not waterproof. Data sampling from MCU and accelerometer works, simple data viewer prototype, may be PC not phone app.
- STRETCH GOAL: If heartbeat monitor circuit can be implemented, it will be.

10/10-10/25

- Effort is on protoboard prototype and PCB design
- Oct 15: Milestone 1
    - show our algorithm, our data gathering, our device's display, and a simple phone App for viewing the exercise activities
- By October 25th, we should have our circuit and our PCB designed
- Order PCB before Nov 3 deadline preferrably

10/26-11/10

- Software development on smart watch display and phone app
- 3D print smart watch case before PCB comes in.
- Assemble PCBAs as soon as parts come in
- Nov 10th: Milestone 2. Show either PCBA fits into 3D printed case, or work on fixing case. Extra large case is acceptable if mistakes occurred.

11/11-11/23

- create epoxy model for PCBA to be fitted into the watch case (Watch case needs to be done by now).
- Sample 1 epoxed and dunked in the water, live data monitoring to find waterproofing issues
- Field test in swimming pool
- Strech goal watch features: Clock Face, Heart beat monitor (if circuit was placed and is working).

11/26-12/9

Final testing, additional features, poster and expo Targeting project completion 12/3. At this point we'll get our poster done and start on the final report. Continue to fix any remaining minor bugs.

12/10-12/11

Finish final report

Risks & Issues

The following are the anticipated issues for this project:

1. PCBA / Watch case packaging

The team will need some CAD experience/skill to ensure that their PCBA can fit in the designed project size, which would be near 2in x 2in.

2. Accelerometer -> distance -> swim lap algorithm

Some exploratory development will be needed to find the best way to calculate the user-side data and avoid noise

3. Waterproof & Safe packaging

Some research is need to make sure thermal concerns related to epoxy and the battery usage are well understood and solved to ensure the device operates properly, and the battery remains in a safe, enclosed state at all times.

Idea 2: Racing Video Game w/ Real RC Cars

High Level Description

I would like to make a racing video game that could use your own house as the race-track. This would be a cool toy to buy, a combination of a PC racing game with slot car racing. The racing game will be on PCs, through a webbrowser, and your screen will show the camera on top of tiny rc cars. The RC cars are all connected via wifi, and can be selected and controlled by the players. We will also make little flags that anyone can use to mark out the course.

The race track

- Race track parts would come with multiple orange Flag Gates. These are standalone gates made of flags that are to be strung together to create the track borders.

The video game

- This is a driving game
    - it'll have a website you can go to or a download location. The vehicles will be wifi-connected and selected by the players. Once selected you can start controlling the vehicle and can see its camera and vehicle data.

The controller / HMI

- Controller will be a PC game controller setup.
- The video game will be either based on pygames or will be a webbrowser based game. If it was webhosted that would make the multiplayer more straightforward.

Accessories

- a charging station or setup. Simple USB-C if it can have on-board battery protection, or an offboard battery charger setup for a detachable battery.
- little flags that the drivers must pass between to demark the course. Drivers must pass between!

Schedule

9/10-9/24 Planning

- Decide system requirements, such as power consumption, and desired packaging/size. These will drive component and MCU selection.
- Sept 18th, project draft proposal due.
- Sept 24th/25th, staff discussion:
    - Show a basic mechanical layout for the vehicle, system specs, and ideas for how the racing game will be impelemented.

9/25-10/9 - Design and part selection

- An early mechanical design because our will have a tight packaging requirement.
- Oct 1: Formal Proposal, showing CAD, components list.
- Order dev-boards and create protoboard demo.

10/10-10/25

- Effort is on CAD model and PCB design to match
- Oct 15: Milestone 1
    - show our prototype vehicle with prototype electronics (not PCB)
- By October 25th, we should have our circuit and our PCB designed
- Order PCB before Nov 3 deadline preferrably

10/26-11/10

- Print and reprint to ensure 3D printed parts fit w/ PCBA
- Assemble PCBAs as soon as parts come in
- Nov 10th: Milestone 2.

11/11-11/23

- make sure video game works, communicates with vehicles, and add nicer features to game.
- Work on strech goal video game ideas

11/26-12/9

Final testing, additional features, poster and expo Targeting project completion 12/3. At this point we'll get our poster done and start on the final report. Continue to fix any remaining minor bugs.

12/10-12/11

Finish final report

Budget

RC Car (Qty 4)

Item	Cnt	Cost	Source
rasberry pi pico	1	$20-$30	Example: pico with wifi, bluethooth and Li-Po charger
tiny camera	1	$50-$80	Examples: CADDXFPV Walksnail Avatar HD Nano Camera / micro-drone cameras
Li-ion Battery	1	$5-$20	Standard rc car battery + charger. Needs to fit 1/48th or 1/32 model
vehicle chassis	1	$5-$20	Team to designed & 3D print. Aiming for 1/48th or 1/32 scale model RC car
Large RC car wheels	4	$2-5	Team to designed & 3D print
rc car motor	1	$5-$20	Simple DC motors

Track (Qty 1) and Controllers (Qty 1 per player)

Item	Cnt	Cost	Source
xbox controllers	1	$20-$40	xbox controllers or cheaper PC connected game controllers
orange flags	30-40	$1-$2	Team made, 3D printed and maybe orange fabric

Optional Stretch Goals

- Wifi /bluetooth based positioning

Proposal 1: Through-Wall Motion Radar

High-Level Description

A portable radar that shows you people moving on the other side of a wall. The finished device is a box you hold in one hand (may be unrealistic, backup is two hands). Ideally we'd get the box to be around the size of a thick phone, but that may be unrealistic based on the antenna arrangement we will need to adopt. A patch antenna array points to the "top" of the box, a screen on the front showing a forward sector view. You point it at a wall, press a button, and moving people behind that wall appear as blips that track as they walk around.

What originally inspired me: https://youtu.be/xyC9K1NO1TI?t=56

Commercial through-wall radars exist. The Range-R unit used by US police and the Xaver line are two examples, but they cost between six and fifty thousand dollars (The Range-R website asks for a quote, that's how you know its expensive). The pitch is that a student team can build a functional one for a few hundred dollars in parts, that's excluding our designs and RnD cost. The users who care are first responders searching the ruins of a collapsed building or tactical teams clearing a structure, both teams would want to know whether a room is occupied before they walk into it.

The device works by sending out a radio signal and listening for the echo, the same idea as a bat using echolocation. The delay before the echo comes back tells us how far away something is, and comparing the echo across a few antennas in some arrangement tells us roughly which direction it came from. The wall, the floor, and the furniture all bounce back the same echo every time, so we tune the device to ignore anything that isn't changing and only highlight signals that shift from one moment to the next, which is what a walking person looks like. Those changing echoes become the blips on the screen.

Implementation

Some definitions for the reader:

PLL: phase-locked loop, the controller chip that commands and stabilizes the signal sweep.

VCO: voltage-controlled oscillator, the part that generates the RF signal, controlled by the PLL.

Mixer: combines the received echo with the outgoing chirp to produce a much lower frequency that encodes range.

LNA: low-noise amplifier, boosts the faint echo signal

Rough Timeline

9/10–10/1	Planning and part selection. Narrow down the RF components and the processor board (likely a Raspberry Pi 5 or similar single-board computer). Finalize proposal.
10/1–10/22	Basic prototyping, have an RF emitter and antenna. Increase the number of antennas and try to get a proof of concept working.
10/15–11/5	PCB design and first order. This obviously requires knowing the antenna layout, what needs to go where physically, the number of ports we need, etc. Ideally order by 10/26.
10/22–11/19	Full DSP pipeline on the laptop. Have working FMCW range processing, moving-target recognition and indication. Verify that it works by testing with some sort of thin wall.
11/12–12/3	Integrate everything and do rigorous refining and testing.
12/3–12/10	Final testing, calibration, poster and expo prep. Have numbers for what we can advertise for the device.

Rough Budget

Category	Examples	Est. $
RF front end	antennas, electrical components	500
ADC and digital capture		150
Processor board	Raspberry Pi 5 or similar	180
PCB fabrication		300
Display, battery, enclosure	5 in LCD, LiPo, 3D-printed housing	150
Total		**1280**

Implementation Issues

For the processor board I'm leaning toward a SBC like a Raspberry Pi 5, with a cheap Jetson as a stretch, rather than an FPGA. The biggest computational load we have to worry about is whether the processer is fast enough to run several FFTs in parallel. If we end up scaling to more RX channels and need more throughput, a Jetson Orin Nano is likely the next choice. However, if someone on the team happens to want to do FPGA work, that could be a stretch goal too.

Since RF design is unforgiving, and I suspect no one on the team that will work on this project has any RF experience, the mechanical design of the arrangement of transmitters and receivers as well as the algorithms used to parse it into something readable will be by far the most difficult part of the implementation. A lot of time and effort, and possibly money, will have to go into making this work. The team should also work as quick as possible to have a proof of concept working. If the team is unable to do so, they will need to consider pivoting to an alternate project.

On the other obvious implementation questions: no. The device transmits at very low power inside the FCC Part 15 UWB mask, so there is no licensing issue and no real RF safety concern. There are no moving parts or mechanical hazards. It is a battery-powered handheld, and the display and processor dominate the power draw, so there is no energy budget problem. The only facilities we need are a vector network analyzer and a spectrum analyzer for bring-up, both of which the department lab already has.

The basic FMCW radar seems to have been implemented frequently in academia. For example, MIT's 6.S062 coffee-can radar and Henrik Forsten's 6 GHz FMCW radar are both fully documented open builds we can use as reference designs, and the RF parts they use, the ADF4159 PLL and similar, are cheap and easy to get. Doing it through a wall specifically has been demonstrated too. MIT's WiTrack localized a single moving person through a wall in about the frequency range we're

targeting[1][2], and the WiTrack2.0 paper includes collected datasets and antenna array layouts we can use as references when we lay out ours[3].

The device only sees moving people, since the subtraction effectively erases anyone holding perfectly still. A breathing-detection mode is a stretch goal. MIT's Vital-Radio project has shown breathing and heart rate can be picked out through a wall from radio reflections alone[4], so with enough time, the team could try to implement this.

Wall material changes how fast the signal travels through it, which means different types of walls are going to yield different distances. For the thin interior walls we're initially targeting (drywall, plywood), that bias is on the order of a couple of centimeters, which is to say fine and negligible. Thicker solid walls like brick add tens of centimeters, which is part of why they're not explicitly in scope. For a bonus demo, we could demo the device through something like the outer wall of EECS, but we're likely just going to run with the decreased accuracy.

Full imaging, AKA getting an actual silhouette or skeleton, is not achievable for this project. MIT's RF-Pose needed a trained neural network and camera-synchronized source to get there[5], and that's years of work.

Evaluation and Testing

Our minimum bar for success is to detect a single person walking behind one interior drywall, range error under about 50 cm, and tracks the person's motion somewhat accurately. AKA, the blip on the screen should track the person as they walk around in real time.

For more rigorous testing, we can have a person walks a marked path in the far room while a camera in that room records. We compare the estimated track against the actual one and extract detection range, range accuracy, angular accuracy, refresh rate, and how many people we can resolve at once.

[1] `https://witrack.csail.mit.edu/`
[2] `https://witrack.csail.mit.edu/witrack-paper.pdf`
[3] `https://witrack.csail.mit.edu/witrack2-paper.pdf`
[4] `https://www.mit.edu/~fadel/papers/vitalradio-paper.pdf`
[5] `https://rfpose.csail.mit.edu/`

Proposal 2 (Unrealistic): Head-Mounted Aurora Viewer

High-Level Description

A visor that lets you see aurorae the naked eye cannot. The human eye integrates light over roughly a tenth of a second, so a faint aurora appears as a dim grey smudge with no color. A camera can integrate for five seconds and pull out bright, structured green and purple curtains. This device performs that integration live and puts the result on a display in front of your eyes.

This idea originated when I first went aurora viewing here in Michigan, and was confused when everyone else began pulling out their phones and cameras to take pictures while I could not see anything in the sky. I was able to capture the faint aurora with a long exposure shot on my phone, but I thought being able to view the faint and otherwise invisible aurora in real time would be very cool.

The finished device is a visor or a pair of glasses. A low-light camera like the type used in dash cams (some fancy Sony sensor that works well with motion and low light) at the top, two micro-OLED displays, an IMU on the frame, and a compute pack on the headband or belt worn around the waist. It captures frames continuously, uses the IMU together with the camera stream to register each new frame against the running stack so that head motion cancels out, and integrates about five seconds of exposure. The stacked, denoised, color-boosted image is sent to the displays at the highest framerate we can get, which will probably be in the 1 to low 2 digits. The compute pack (probably something along the lines of a Jetson Orin) may be quite heavy, hence why we suggest it could be waist mounted.

Implementation

Rough Timeline

9/10–10/1	Part selection and laptop prototype. Select and purchase the camera, compute module, IMU, and displays. Build a prototype of the software stack running on a laptop and simulate it against night sky footage, and finalize this proposal.
10/1–10/22	Carrier PCB. Design the board, covering power, IMU, camera interface, and display drivers, and order it.
10/22–11/19	Port to the compute module. Get the pipeline running on the jetson at a few frames per second and optimize as needed.
11/12–12/3	Visor integration and field testing. Integrate the visor, run field tests on clear nights, and tune denoising and exposure.
12/3–12/10	Design the poster and prepare for the expo. Use an indoor simulated-sky demo as a fallback since aurora cannot be scheduled. Optionally take the teaching team out for demo if timing is convenient.

Rough Budget

Category	Examples	Est. $
Compute module	Jetson Orin Nano	250
Low-light camera	Sony IMX908	250
Displays	ECX334C Clone	100[6]
IMU, power, etc.		50
Carrier PCB		100
Optics, visor frame, battery, enclosure		200
Total		**950**

Implementation Issues

The compute module runs hot and draws 2 digit wattage constantly, so we will need a big battery and some active cooling.

Optics and camera selection is very finicky and iterative. There will be an extreme amount of sensor tuning and experimentation with filters to get an actually usable output. There's also the question of using the camera stream to do stabilization, which means some sort of sensor fusion algorithm, which is difficult and finicky to implement too.

CV will be a very significant part of this project, so much so that you could argue that it becomes a ML project, not an embedded project.

Evaluation and Testing

We would compare the stacked output's signal-to-noise ratio against a single frame, and against a tripod-mounted long-exposure camera shooting the same patch of sky.

Success is pulling a Kp 3 aurora into clear, colored visibility when the naked eye can barely see anything. As a convenient indoor test, we can project a dim, slowly varying synthetic aurora onto a screen in a dark room and confirm the device recovers it while a bare camera frame does not.

Eventually, real world testing should ideally be done. I captured my aurora at Pinckney Recreation Area about 30 minutes North by car of Ann Arbor. Go when there are clear skies and a high enough Kp value reported by NOAA.

[6]Sony ECX334C MicroOLED display clones could be had for $50 in China. Used in my 373 project. May not be feasible for your team if you decided to adopt. Costs $200 in the US.

1 Proposal 1

1.1 High-Level Description

Repetitive Strain Injuries (RSIs), industrial Musculoskeletal Disorders (MSDs), and grip-related peripheral neuropathies represent a widespread occupational and biomechanical challenge. According to the U.S. Bureau of Labor Statistics, work-related musculoskeletal disorders account for over 30% of all worker injury and illness cases involving days away from work, costing employers upwards of $20 billion annually in direct workers' compensation costs [1, 2]. Furthermore, in athletic and human performance settings—such as rowers, climbers, baseball players, and gymnastics—monitoring grip force dynamic distribution, dynamic strain, and kinetic chain transfers is critical for preventing acute injuries (e.g., Ulnar Collateral Ligament fatigue) and optimizing force production [3].

Current biomechanical sensing approaches are heavily constrained: optical motion-capture rigs are restricted to fixed laboratory environments, and commercial dynamometers only measure static clamping force. There is a lack of field-deployable, untethered devices that capture multi-point force vectors and joint angular rates simultaneously during natural, dynamic activity.

This project proposes a lightweight, wireless **Smart-Grip Sensing Glove** designed to perform high-frequency grip force distribution analysis and dynamic hand/wrist kinematic tracking in real time. The wearable glove features a flexible array of Force-Sensitive Resistors (FSRs) integrated across the palm and digital phalanx contact points, alongside a high-rate 9-axis Inertial Measurement Unit (IMU) positioned at the metacarpal junction.

How it works:

- **Data Acquisition:** A custom wrist-mounted PCB ingests multi-channel analog pressure signals and digital IMU motion metrics at high sampling rates (> 200 Hz).
- **On-Board Processing & Filtering:** An low-power MCU performs onboard digital filtering, dynamic impact peak detection, and complementary/Kalman orientation calculations.
- **Wireless Telemetry:** The device streams compressed data packets via Bluetooth Low Energy (BLE) or Sub-1GHz RF to an edge gateway, PC, or tablet for live ergonomic analysis, physical therapy monitoring, or athletic biomechanics research at SIX Labs.

Target User: Occupational health safety teams, ergonomics researchers, physical therapists tracking patient rehabilitation, and athletic biomechanics researchers.

—

1.2 Implementation Issues

1.2.1 Schedule

- **Weeks 1–2 (Planning & Architecture):** Finalize sensor selection (FSR matrices, low-power BLE MCU, high-rate IMU like ISM330DHCX). Draft schematics for the glove PCB and USB receiver dongle.
- **Weeks 3–5 (Prototyping & PCB Layout):** Validate analog front-end multiplexing and SPI/DMA transfers on breadboard dev-kits. Design a compact 4-layer PCB (< 40 mm $\times$ 40 mm). **Order Rev-1 PCBs by end of Week 5.**

- **Weeks 6–7 (Firmware & Wearable Design):** Develop MCU driver libraries, sensor filtering algorithms, and BLE streaming profiles during PCB fabrication delay. Stitches flexible wiring traces onto compression glove.
- **Weeks 8–10 (PCB Assembly & System Integration):** Populate PCBs using surface-mount reflow. Perform board bring-up. Mount PCB, battery, and FSR array into glove harness.
- **Weeks 11–13 (Testing, Refinement & Expo Prep):** Calibrate force channels against known weight loads. Fine-tune power management profiles. Finalize poster and live demo setup for the Design Expo.

1.2.2 Budget

The total allowable budget ceiling for a 5-person team is $1250 ($250 per student). The estimated bill of materials for building 3 functional glove prototypes and 1 gateway receiver demonstrates a cost-effective implementation:

Table 1: Estimated Bill of Materials for Proposal 1

Component Description	**Quantity**	**Cost / Unit ($)**	**Total Cost ($)**
Custom 4-Layer Wrist PCBs + Stencil	5	20.00	100.00
BLE Microcontroller Module (nRF52840 / ESP32-S3)	5	8.50	42.50
Flexible Force-Sensitive Resistor (FSR) Arrays	12	12.00	144.00
High-Precision 6/9-Axis IMU (ISM330DHCX)	5	7.50	37.50
Analog Multiplexers / AFE / Precision ADCs	5	6.00	30.00
LiPo Batteries (3.7V 350mAh) & Charger ICs	5	11.00	55.00
Wearable Glove Harnesses, Flex Cables & Enclosures	3	30.00	90.00
Miscellaneous Passives, Hardware, & Shipping	1	150.00	150.00
Total Estimated Budget			**649.00**

1.2.3 Technical & Mechanical Concerns

- **Flex-Cable Strain & Fatigue:** Repeated hand flexing can snap wiring interconnects between palm sensors and the wrist PCB. *Mitigation:* Use flat flexible ribbon cables with molded strain reliefs routed along anatomical low-strain lines.
- **High-Channel Analog Multiplexing:** Sampling numerous FSR points can cause MCU bottlenecks. *Mitigation:* Implement hardware Direct Memory Access (DMA) to auto-cycle through multiplexer channels without CPU intervention.
- **Safety & Ergonomics:** Lithium-polymer batteries positioned on the body must be shielded from puncture or impact. *Mitigation:* Enclose electronics inside a rigid, low-profile 3D-printed PETG shell padded with shock-absorbing foam on the dorsal wrist.

—

1.3 Evaluation and Testing

- **Force Accuracy & Linearity:** Validate individual FSR pressure channels against a calibrated dynamometer load cell across a 0–100 N force range with $< 6\%$ error.
- **Kinematic Tracking Accuracy:** Validate IMU angular velocity and orientation tracking against controlled rotary stage speeds (up to 1000 deg/sec) with minimal gyro drift over 10-minute continuous trials.
- **Wireless Throughput & Latency:** Ensure real-time telemetry over BLE retains $> 98\%$ packet transmission efficiency at 100 Hz update rate over a 10-meter range.

2 Proposal 2

2.1 High-Level Description

Subterranean soil and surface integrity degradation impacts civil infrastructure, municipal parks, agricultural fields, and sports venues alike. Studies indicate that non-uniform surface compaction, excess moisture retention, and unmonitored dynamic impact loads are leading causes of soil erosion, field degradation, and tripping/musculoskeletal injuries [4]. Standard ground testing relies on periodic manual drop-weight tests (e.g., Clegg Hammer) that fail to capture spatial and continuous temporal variations.

Smart Structural Turf is a low-cost, subterranean wireless sensing node designed to be driven directly into soil, turf, or athletic grounds. Formed like a heavy-duty lawn spike, each node houses a high-g shock accelerometer (measuring transient footfall or equipment impacts up to $\pm$200g), capacitive soil moisture and temperature sensors, an embedded MCU, and a Sub-1GHz RF radio module. Sitting flush beneath the foliage line, the nodes stream real-time ground health metrics to an edge base station, alerting managers when localized zones become over-compacted, waterlogged, or structural risks.

—

2.2 Implementation Issues & Rough Sketch

- **Core System Architecture:** Ultra-low-power MCU paired with a high-g accelerometer (e.g., ADXL375), capacitive moisture sensing traces embedded along the spike shaft, and a 915 MHz Sub-1GHz RF transceiver.
- **Power Strategy:** Ultra-low-power sleep modes where nodes wake on configurable shock interrupt thresholds or periodic timers. An optional flush-mounted solar trickle charging cap serves as a stretch goal.
- **Rough Budget (4 Nodes + 1 Base Station):**
  - 3D-printed polycarbonate/TPU spike housings: \$60
  - Custom Node PCBs + MCUs + High-g Accelerometers + Moisture Front-ends: \$210
  - RF Gateway Receiver & Antenna Assembly: \$90
  - Miscellaneous Hardware, Batteries, & Shipping: \$110
  - **Total Estimated Budget: ~\$470**.
- **Primary Feasibility Challenges:**
  - *RF Attenuation in Wet Soil/Grass:* Ground-level 2.4 GHz signals degrade rapidly in moist grass. *Solution:* Utilize Sub-1GHz (915 MHz) RF for superior foliage and ground penetration.
  - *Impact Survival:* The top cap must sustain direct structural loads from lawn equipment or heavy foot traffic. *Solution:* Pot internal circuitry in high-strength epoxy within a reinforced TPU/polycarbonate housing recessed 5 mm below grass height.

References

[1] U.S. Bureau of Labor Statistics, *Nonfatal Occupational Injuries and Illnesses Requiring Days Away from Work*, U.S. Department of Labor, 2022.

[2] Occupational Safety and Health Administration (OSHA), *Ergonomics: The Preventability of Musculoskeletal Disorders*, U.S. Dept. of Labor, 2021.

[3] J. R. Physical Therapy & Biomechanics, *Upper Extremity Kinematics and Grip Force Dynamics in Elite Athletes and Industrial Workers*, Journal of Biomechanical Engineering, vol. 142, no. 8, pp. 102–111, 2020.

[4] M. E. Agricultural & Sports Turf Research, *Continuous Monitoring of Soil Surface Hardness and Dynamic Impacts*, Soil Science & Turf Mechanics, vol. 45, no. 3, pp. 215–223, 2021.

Stereo Camera for 3D Point Cloud Generation

Perception capabilities are crucial in robotics for performing obstacle detection and avoidance, object detection, and to give robotic systems a form of vision.[1] Further, it is important for these systems to have some sort of depth perception to more efficiently and accurately map out different environments. The most accurate solution that currently exists is LiDAR, which is laser based. However, accurate LiDAR systems often cost thousands to tens of thousands of dollars, while cheap ones can be inaccurate.[2] Further, LiDAR is unable to capture images, so a separate camera is still needed. A stereo camera, while not as accurate as a good LiDAR system, provides the depth and image capabilities in one package, for a lower price tag.

A stereo camera can also bundle an IMU, allowing one system to provide crucial measurements most robotic systems need to function.

While many stereo camera solutions exist, the most popular use proprietary software, making it impossible to debug many classes of issues. These cameras also make significant tradeoffs in performance to remain mass-market solutions.

Design

The most important aspect of a stereo camera for many robotics applications remains to be its ability to generate accurate point clouds that extend as far out from the camera as possible. We propose a camera that optimizes point cloud generation by focusing on density, precision, and spatial accuracy. We would buy camera modules and record their intrinsic parameters. By utilizing FPGAs, the pixels captured by the cameras would be transformed (based on the intrinsic parameters), processed, and assembled into a point cloud that should have accurate depth.

At the same time, a microcontroller would handle generating frame metadata, camera synchronization (ex. exposure), and USB/Ethernet transfer to a computer/visualization software. The MCU would also need to watch for errors like dropped frames or desynchronization between the left and right lenses. A custom PCB would be used to house the hardware (FPGAs, sensors, etc) for the two camera modules.

Schedule

Below is an approximate timeline for the project.

---

[1] https://www.esa.int/Enabling_Support/Space_Engineering_Technology/Automation_and_Robotics/Robotics_Perception
[2] https://e38surveysolutions.com/pages/drone-lidar-price-guide

9/14–9/27	***Planning*** After group selection, we will start discussing and finalizing what parts we will need, with exact components chosen for the camera sensors and the FPGA. We will also start to either write out or program the math we will need for calibration, potentially practicing on two camera sensors from the lab. We will also discuss what MCU we need and plan out the software stack
9/28–10/11	***Design and Part Selection*** We will finalize all parts (MCU, FPGA, camera sensors) and order them. We will also start on PCB discussion, and hopefully design. Before PCB design, we will finalize the capabilities of the PCB. We would also ideally like to start writing some software libraries, especially for calibration and start planning VHDL layout for the FPGAs. Design 3D printed mount to hold camera sensors in place in epipolar alignment.
10/12–10/25	***Calibration and Testing*** When the camera sensors arrive, we would start calibrating the sensors to match each other. We would also wire up a breadboard to test our PCB design. Test to see closely timed sensor frames are possible to grab and operate on with the FPGAs. Also start testing and improving rectification logic. Order PCB ideally by 10/22.
10/26–11/8	***Software and Scheduling*** Set up MCU with RTOS and start writing service routines to grab rectified images/pointclouds and send via ethernet/usb to a computer/visualization software. Complete image rectification on FPGAs. Attempt to finalize image disparity algorithm. Ensure that sensor images are synced in time. Start assembling PCB when it arrives, and ensure 3D printed case can house everything, reprinting if needed.
11/9–11/22	***Final Assembly and Programming*** Test final assembled PCB and start adding everything to 3D printed housing. Ensure that camera timing works on the PCB, and ensure FPGAs are correctly buffering pixel rows. Ensure feature extraction and DMA output work on the PCB, and ensure all pixel transforms look correct, adjusting algorithms if needed. Also ensure the output image is correct.
11/23–12/10	***Final Testing and Expo Prep*** Start on the expo poster and final report. Fix any issues with 3D printed assembly and fix any minor software bugs. Aim to have the project ready for the expo by 12/5
12/10–12/11	***Finish final report***

Budget

Below is an estimated budget for parts known to be needed for the stereo camera. PCB manufacturing is estimated and is likely incorrect. A single board would be needed, so assuming the PCB layout is correct, there would be spares. The total cost would be under budget for a five to six person team

Item	**Quantity**	**Cost**	**Total**
OV9281 Camera Module	2	41.99	82.00
AMD-Xilinx Zybo Z7-20	1	314.00	314.00
MCU	1	30.00	30.00
PCB (with assembly)	1	100.00	100.00
Miscellaneous	1	200.00	200.00
Total			**726.00**

Challenges & Testing

Listed below are the main potential difficulties for this project

1. MIPI CSI-2 signal integrity on a custom board, and other general timing issues
2. Developing the actual rectification/disparity algorithms
3. Verification of the system without a computer

The calibration of the cameras needs to be very precise, and there are a lot of potential issues ensuring the left/right images/point clouds remain in sync. The pixels need to be buffered and processed in the correct order, ensuring that the limited amount of BRAM on the FPGA board does not become a bottleneck. The most common issue, it seems, is that incorrect board layout (i.e. traces are different lengths) can severely degrade the high speed CSI signals. These issues should be solvable with careful layout and testing, though. Issue 2 seems to be well documented online and should not be as much of a problem, but may prove to be a challenge once we start programming.

To test this, we would need to build testing into each part of the project. First, testing signal integrity from the camera to the FPGA and ensuring these images are transferred/start being processed at the same time via an oscilloscope. We would also calibrate the cameras via OpenCV and upload the matrix to the FPGAs. Finally, we would capture images with the cameras and test rectification/calibration in a hardware simulator and see if it looks correct.

Home Router

The FCC and other government agencies have started to crack down on 3rd party routers not made in the US for the stated reason of mitigating national security risks.[3] Many people use 3rd party routers because they can offer more features, have more range, or lower costs over the routers provided by internet service providers (ISPs).[4] Hobbyists and people looking to improve their home network also often add networking hardware like network-wide ad blockers to their networking setup. With more and more crackdown on these routers, and a general lack of routers made in the US, there will soon be a gap in the market for reliable, feature-rich, and secure routers, likely driving up costs.

Such a router might have a large market among people alienated by the new rules. There is also an opportunity to build more security into these devices from the ground up, making it a focus and protecting peoples' data.

Details

Instead of needing to buy and set up separate hardware for network tasks like ad blocking, routing, and security, a single router could bundle these tasks together into one package while also expanding the feature set of a default ISP router. We propose a router network that has an ad blocker and mesh network capabilities built in. A limited set of security features, like guest networks will also be added. A wider set of security improvements, like encryption and firewalls, are not in the scope of this class, but are possible using MCUs with hardware accelerated AES encryption, for example.

More specifically, this proposal aims for two separate modules, one for handling Linux tasks (DHCP, NAT, routing) and the other to focus on hardware telemetry and RTOS tasks. We would also create 2-3 mesh child nodes that would handle some of the device hand off capabilities. Finally, the router would be outfitted with a WiFi antenna and software in the Linux layer to help the child nodes with handoff and continuous connection tasks.

Timeline

The following is a very rough set of steps for completing this project:

1. Determine necessary MCU/hardware for each layer (linux and hardware)
2. Order hardware and start prototyping on a breadboard, while simultaneously starting to design the custom PCBs for the parent/child nodes
3. Work on software to handle handoffs, continuous connection, and ad blocking

[3] https://www.fcc.gov/sites/default/files/NSD-Routers0326.pdf
[4] https://blog.bestbuy.ca/computers-laptops-tablets/routers/replace-isp-wi-fi-router

4. Determine what linux drivers and RTOS routines need to be written and start designing those
5. Once the PCB arrives, assemble and start testing on it
6. Ensure the PCB works and finish software libraries, ensure everything works by trying to connect an actual phone

Challenges

The biggest challenges for this project will be ensuring the Linux and RTOS systems work well together and generally implementing the child nodes. Mesh networks hand off devices to the child nodes (i.e. the device can connect to different nodes at different times). This handoff needs to happen without interrupting internet connection, which will require some Linux driver development. In general, knowing Linux routing commands/rules is also needed to ensure proper setup. The other major challenge is in dealing with gigabit ethernet. At such high speeds, PCB layout needs to be very precise, and mounting components would take a while.

EECS 473 — Project Pre-Proposal (HW #0)

Proposal 1: Fall-Detection Wearable for Elderly Users

High-Level Description

Falls are the leading cause of fatal and non-fatal injury among adults aged 65 and older in the United States,[1] and worldwide falls are the second leading cause of unintentional injury deaths overall, with adults over 60 suffering the greatest number of fatal falls.[2] The fear of falling (and of not being found quickly afterward) is one of the main reasons older adults lose independence and are moved into assisted living.[3] This project proposes a low-cost, wrist- or belt-worn embedded device that continuously monitors motion, automatically detects a fall event, and sends a wireless notification with the wearer's status and rough location to a caregiver or family member (if the wearer does not cancel the alert within a short window).

The finished device is a small (roughly matchbox-sized), battery-powered enclosure containing an inertial measurement unit (IMU), a microcontroller, a wireless radio, a small buzzer, and a single cancel button. In normal operation it just samples motion data in the background. When a fall-like signature is detected, it beeps and displays/announces a countdown; if the wearer is fine, they press the button to cancel. If they don't (or can't), the device transmits an alert.

This is worth building because existing commercial fall detectors are either expensive subscription services or rely on the user pushing a button after they've already fallen, which does not help if they are unconscious or immobilized. An automatic, low-cost, no-subscription alternative has clear real-world value, and the underlying sensing problem (distinguishing a fall from sitting down quickly, bending over, or dropping the device) is a genuinely open embedded signal-processing problem.

Details

As the wearer goes about their day, the device continuously samples acceleration and angular velocity from the onboard IMU at a fixed rate (on the order of 50–100 Hz). All of the fall-detection logic runs locally on the wearable's own microcontroller: a lightweight algorithm looks for the acceleration-magnitude spike characteristic of an impact, followed by a period of near-stillness that would indicate the wearer is now lying on the ground rather than having just sat down quickly or set the device down on a table. When that pattern is detected, the device enters an "alert pending" state when it starts to beep, and starts a short countdown (on the order of 20–30 seconds) during which the wearer can press the onboard cancel button if they are fine. If the countdown expires without a cancel, the device transmits a short wireless message (a device ID, a confidence level, and, if available, a rough GPS fix) to a nearby gateway.

This proposal envisions building two or three wearable units and a single gateway station. Each wearable would contain an IMU, a low-power microcontroller with an integrated short-range radio (BLE), a small buzzer, a cancel button, and a rechargeable LiPo battery. Rather than requiring its own cellular connection (which would add significant cost and power draw to something meant to be worn all day), each wearable talks over BLE to a fixed gateway (most simply a smartphone running a companion app, or, if time allows,

[1] CDC, "Older Adult Falls Data," https://www.cdc.gov/falls/data-research/index.html.

[2] World Health Organization, "Falls," https://www.who.int/news-room/fact-sheets/detail/falls.

[3] "Preventing Falls in Senior Living Communities," Abode Care Partners, https://abodecarepartners.com/preventing-falls-in-senior-living-communities/.

a small dedicated base station of our own built around a Wi-Fi- or cellular-capable microcontroller). The gateway is responsible for the "last mile" of actually notifying a caregiver (push notification, text message, or phone call), so that the battery- and size-constrained wearable itself only ever needs to do short-range, low-power communication.

Implementation Issues

Schedule. The table below gives a rough week-by-week plan.

9/10–9/23	***Planning*** In the two weeks after the project groups are formed, research two or three candidate low-power MCU/radio SoCs (targeting a BLE-integrated part in the $8–12 range) and IMUs, and select the best fit based on cost, power draw, and available I/O. Begin sketching the fall-detection approach (threshold/feature-based vs. a learned classifier) and estimate firmware memory/CPU requirements. Write the draft proposal by 9/18.
9/24–10/6	***Design and part selection*** Once the proposal meetings are done, order the selected MCU/IMU dev board(s) and a second board to stand in as a BLE gateway. Bring up a prototype in a minimal fashion (blinking LED, can read raw IMU data and print it over a debug console, even if not meaningfully processed yet). Submit the final proposal by 10/1. All parts we think we need for the prototype will have been ordered by the 4th.
10/7–10/25	***Breadboard prototype and PCB design*** Get the dev-board prototype logging labeled IMU data during scripted falls and normal activities, and get a first-pass threshold-based fall detector running on it. Get basic BLE messages passing between the wearable prototype and gateway board. In parallel, finalize the wearable schematic and layout (IMU, MCU/radio SoC, buzzer, cancel button, battery/charging circuit) and order PCBs by the 26th (a bit more than a week before the PCB deadline).
10/26–11/10	***Software development and testing on prototype*** While waiting for the PCBs to come back, tune the fall-detection algorithm against the growing labeled dataset on the prototype, implement the alert/countdown state machine and cancel-button interrupt handling, and get low-power sleep-mode transitions working between IMU samples. Get PCBs assembled (hopefully correctly) by 11/8. Identify solutions to non-electronic issues (enclosure, wrist/belt mount) and get those ordered.
11/11–11/25	***Software development and testing on PCB-based system*** Port firmware from the dev board to the PCB and bring it up. Once basic functionality is working, run repeated staged fall trials to verify detection accuracy on the real hardware, and have each team member wear it through a full day of normal activity to measure the false-positive rate. Bugs will be found and fixed; PCB issues resolved and revisions made if needed. Any additional stretch features (e.g., a phone-app alert receiver) will be researched and decided upon by this time.
11/26–12/9	***Final testing, additional features, poster and expo*** Targeting project completion 12/3. At this point we'll get our poster done and start on the final report. Continue to fix any remaining minor bugs.
12/10–12/11	***Finish final report***

Budget (approx., team of 4–5, $250/student).

What	Qty	Cost per	Total
MCU + IMU module (e.g., nRF52-class SoC + BMI160)	3	$12	$36
Radio (BLE/Wi-Fi integrated on SoC; no separate part)	—	—	$0
Cellular backup module (SIM7000-class, stretch)	1	$25	$25
LiPo battery + charge/protection circuit	3	$8	$24
PCB manufacture (2 spins, 3 boards each)	6	$12	$72
Enclosure (3D printed / off-the-shelf case)	3	$10	$30
Misc. passives, connectors, buttons, buzzer	1	$40	$40
Shipping / contingency	1	$35	$35
Total			**$262**

Other implementation issues. No unusual mechanical fabrication is needed beyond a small 3D-printed or off-the-shelf enclosure, and no special facilities beyond the lab's standard SMD soldering/rework stations are required. There are, however, several implementation questions worth being upfront about:

- *Safety.* This is a device intended to help a vulnerable user population, so we are explicitly biasing the design toward false positives over false negatives. An occasional nuisance alert is far preferable to a missed fall. We will also make sure any local alarm are easy to operate by someone who may be dazed or in pain immediately after a fall (large button, loud/simple audio cue).
- *Algorithm risk.* Fall detection from IMU data alone is a well-studied but not fully solved problem. Published approaches range from simple acceleration-magnitude thresholds to small onboard machine-learning models. Cross-dataset evaluations show that algorithms tuned on laboratory-staged falls often degrade sharply when tested against real-world daily activity, producing an impractically high false-alarm rate outside the lab.[4] We plan to start with a threshold/feature-based approach (magnitude spike followed by a period of stillness) since it is simple to implement and debug on a microcontroller, and treat a learned classifier as a stretch goal only if the simple approach's false-positive rate turns out to be too high.
- *Power budget.* The IMU and MCU can duty-cycle aggressively between samples (e.g. sampling at 50–100 Hz but sleeping between reads), and commodity low-power BLE SoCs are documented to support multi-day battery life in comparable wearables, so we believe a small rechargeable LiPo with daily or every-other-day charging is achievable. Exact battery life will depend on final sampling rate and radio duty cycle, which we will only be able to pin down once we have real current-draw measurements on the prototype.
- *Mechanical fit.* Fitting a battery, MCU/radio board, IMU, buzzer, and button into a genuinely wearable (wrist- or belt-mounted) enclosure is a tighter mechanical constraint than a benchtop project would have. We plan to prototype the enclosure size early (even with a cardboard or 3D-printed mockup) rather than leaving it until the PCB is finished, so that board dimensions are chosen with the enclosure in mind rather than the other way around.

Evaluation and Testing

Success will be measured along three axes: (1) *detection accuracy*: staged fall trials (forward, backward, sideways, sitting-down-hard) alongside non-fall activities (walking, sitting, bending, dropping the device on a table), computing true/false positive rates; (2) *response latency*: time from impact to alert transmission; and (3) *battery life*: continuous run time on a full charge under realistic duty cycling.

[4] J. Klenk et al., "Cross-dataset evaluation of wearable fall detection systems using data from real falls and long-term monitoring of daily life," https://www.sciencedirect.com/science/article/pii/S0263224124008777.

Proposal 2: Gesture-Controlled "Air" Instrument

High-Level Description

This project is a small, self-contained musical instrument that is played entirely with hand gestures in open air with no strings, keys, or surface contact required. An array of short-range time-of-flight distance sensors mounted in a compact enclosure tracks the position and motion of the player's hand above the device, and a microcontroller maps that position/velocity data to MIDI notes or simple synthesized tones, played live through a small speaker or sent out over a standard MIDI/USB connection to a computer or synthesizer. Conceptually it's a low-cost, embedded take on a theremin, but using discrete distance sensors instead of an RF field, which makes gesture-to-note mapping (e.g., pitch from height, volume from a second axis, or note-triggering from swipe gestures) more flexible and more approachable to build as an embedded system. It's aimed at musicians, hobbyists, and honestly anyone who wants a fun, tactile-but-touchless instrument to play with at a party or the Design Expo, where a mid-air, glowing, sound-making gadget makes for a great crowd demo.

Implementation Issues

The core hardware is an array of 3–4 VL53L0X/VL53L1X time-of-flight distance sensors sharing a single I2C bus, each given a unique address at boot via its XSHUT/shutdown pin (power sensors up one at a time, assign each a distinct address, then bring the rest online), read continuously by a single microcontroller. The MCU maps hand height/position across the sensor array to pitch/volume/note-on-off events, and drives sound either through a simple onboard speaker via PWM/DAC tone generation or by sending standard MIDI messages out over USB to a laptop/synthesizer, whichever turns out to sound better and be easier to get working reliably.

Rough schedule: get a single sensor reading distance reliably over I2C (week 1), extend to the full multi-sensor array with address conflicts resolved (week 2), implement the gesture-to-note mapping and get basic sound output working on a breadboard/dev-kit setup (week 3), and design, order, and bring up a small PCB that consolidates the sensors, MCU, and audio output into one enclosure (weeks 4–5, which comfortably covers the $\sim$10-day PCB turnaround since the mapping/sound firmware is already done by this point). Any remaining time (week 6, stretch) would go toward polish: tuning the gesture mapping to feel more responsive and natural, and adding LED feedback so the player can see where the sensors are "looking" at them.

Budget is modest relative to the per-team ceiling: roughly 4 VL53L1X breakout sensors ($10 each, $40 total), a small MCU dev board or the custom PCB itself ($20–40), a speaker and small audio amplifier ($10), and miscellaneous wiring, connectors, and enclosure material ($20–30) which is well under $150 for the whole sub-team, which leaves comfortable slack in the group's shared budget for the main project. We don't see any significant safety issues here (no moving parts, no high voltages, no mechanical load-bearing structure). The main technical risk is I2C bus contention from multiple sensors that ship with the same default address; this is a well-documented problem with a standard solution (sequential power-up plus address reassignment via each sensor's shutdown pin), so we're confident it's addressable even though none of us has done it before.

Evaluation

Success here is fairly demo-driven rather than metric-driven, which fits the "wacky" nature of this proposal: can a new user, given under a minute of instruction, play a recognizable simple tune or at least clearly control pitch and volume through hand motion in a way that feels responsive (i.e., low enough end-to-end latency that the sound feels connected to the hand motion rather than noticeably laggy)? If we want a slightly

more objective check, we could also measure the sensor-to-sound latency directly and confirm it's under roughly 50–100 ms, which is in the range where audio feedback typically starts to feel immediate rather than delayed.

As a rough acceptance test at the Design Expo specifically, we'd like a passerby with no instructions beyond "wave your hand over it" to be able to tell within a few seconds that their hand motion is controlling the sound, and to be able to play at least a simple recognizable melody (e.g., a five-note nursery-rhyme snippet) within about a minute of experimentation.

EECS 473 HW #0 – Project Pre-Proposal Outline

Project Proposal 1: *Fitness Tracking Smart Shoe*

High-Level Description

Problem / Motivation

Fitness trackers like smartwatches, rings, or bands can record general activity statistics such as steps, distance traveled, and heart rate, but they have limited ability to *directly* analyze the steps and how a person's foot interacts with the ground while walking or running. Information such as foot pressure distribution, ground-contact time, cadence, and stride characteristics can provide more detailed information about a person's gait, running technique, physical activity, and fitness.

This project would create a self-contained smart shoe capable of collecting both conventional activity statistics and detailed foot-specific gait data. The device would be useful to athletes interested in analyzing their technique, athletes/recreational users recording training and activity data, or even physical therapy.

Proposed Solution

The proposed device would be a shoe containing an embedded sensing and processing system integrated directly into a custom sole. The system would measure:

- Step count
- Stride distance
- Cadence
- Ground-contact time
- Foot pressure throughout each stride
- Walking/running duration
- Potentially heart rate as a stretch goal

The shoe would communicate with a smartphone through Bluetooth Low Energy, where a basic companion app could display both real time and recorded activity information.

The electronics and sensors would be integrated into a custom 3D-printed sole. The sole would be printed with flexible filament and would contain hard-shelled protective cavities for the PCB, battery, and other electronics. These components would be sealed against water and environmental exposure. A conventional cushioning and arch support insole would sit above the electronics and pressure-sensing system to protect both the electronics and the user's foot.

Rather than manufacturing an entire shoe, the scope of this project focuses only on creating the sole which would be attached to the upper portion of an existing shoe. This would allow the project to focus primarily on the embedded sensing system and sole construction while still producing a complete and wearable device.

How It Would Work

While walking or running, an array of 11 pressure sensors distributed underneath the insole would measure how the user's weight moves from the heel towards the toes during a step. These pressure sensors will most likely be force-sensing resistors connected to ADC inputs or multiplexed ADC circuitry. Pressure sensors would be present on each shoe, so that step metrics can be recorded for each foot.

Meanwhile, an IMU would measure and record acceleration and rotation of the foot/shoe. The microprocessor would combine information from the IMU and the FSR/ADCs to distinguish individual steps and calculate stride length (estimated from integrating acceleration twice), cadence, and ground contact time.

If included, heart rate would be measured using a small UART pulse sensor with a timer. This would be located either on the top of the shoe near the tongue or on the inner ankle. Heart rate would be measured only on one foot/shoe.

Basic measurement sequence:

1. Pressure sensors detect when the foot contacts the ground and trigger an interrupt.
2. The IMU records the motion of the foot throughout the stride.
3. The MCU distinguishes/counts individual steps and calculates gait statistics.
4. Pressure measurements determine how and when force is distributed across the foot.
5. Heart rate is recorded using a pulse sensor mounted on the shoe (if included).
6. Statistics are momentarily stored locally and transmitted via BLE to a smartphone.
7. A basic smartphone app displays the collected information to the user and graphically shows the pressure at each location of the foot over the course of a stride.

Why It Is Worth Building

The fitness tracking smart shoe would provide measurements that conventional health trackers cannot obtain. Specifically, measuring pressure at various locations under the foot would allow the system to analyze how the user's foot impacts the ground during a stride. This has use-cases in organized athletic applications, recreational exercise applications, and also physical therapy.

For example, the device would be able to distinguish between heel striking, midfoot striking, or forefoot striking. It could alert users if there is excessive slapping of the foot which could cause overuse injuries if continued. It could also observe how pressure transfers from the heel to toe during a stride to provide insight on running mechanics and efficiency. These extra metrics made available by in-shoe tracking would provide a more comprehensive view of walking and running mechanics.

The system makes for an interesting embedded systems project because it requires integration of sensors, ADCs, signal processing, wireless communication, power management, battery form-factors, PCB design (possibly flex-PCB), and mechanical packaging/design. This is great experience for portable devices and wearable technology.

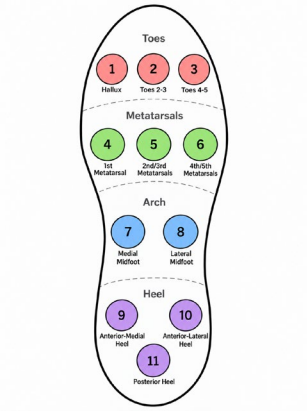
Implementation Issues

General Hardware Architecture

Processor: A low-power consumption MCU with BLE capability would control the system in each shoe. The processors would need sufficient capability for IO, ADCs, and timers for all pressure sensors, the IMU, system timing between the two shoes, stride timing, and heart rate sensing. The STM32WB family is a good starting point for MCU selection.

Motion Sensing: A small 6-axis IMU with an accelerometer and gyroscope in each shoe would measure foot motion and orientation.

Pressure Sensing: Multiple pressure sensors would be distributed across the top of the sole. Ideally, 11 pressure sensors would be used in the following locations:



Communication: Bluetooth Low Energy would transmit recorded statistics to a smartphone. Communication for IMU would likely use either I2C or SPI. Communication for the pulse sensor would likely be able to use UART as it is measuring a slow periodic rhythm so high speed is not required.

Power: A small, rechargeable 1S LiFePO4 battery would be used to power each shoe. This chemistry is chosen for its voltage range of 3.2-3.65V, its lower susceptibility to thermal runaway for safety, and its good cycle life. The Cell should be in the range of 500-1000 mAh. The PCB would contain voltage regulation and basic battery-management circuitry. A temperature sensor could monitor the battery and trigger an emergency shutdown if the cell approaches an unsafe operating temperature.

PCB: The custom PCB on each shoe would integrate:

- BLE capable MCU
- IMU
- Pressure sensors (FSRs)
- Pulse sensor (only on one shoe)
- External crystal oscillator (if needed)
- Battery charging port/protection
- Voltage regulation
- Power switch
- Programming/debug port
- Battery connection

Mechanical Construction: The sole would be custom designed and 3D printed using a flexible TPU material. The outsole that touches the ground will be printed using an abrasion-resistant TPU with roughly 95A stiffness to give grip and wear resistance. The midsole that contains the components will be more flexible TPU printed in a hollow, lattice pattern to create a supportive, soft shoe. See Adidas 4D shoes for inspiration. This material should be roughly 85A stiffness. Within the midsole there should be a hard casing printed in PLA or ABS to protect components like the PCB and battery. The insole can be purchased or printed using the softest filament possible or possibly even a foam cutout. Lastly the upper part of the shoe will be taken from an existing shoe and secured to the top of the sole assembly. This reduces excessive mechanical design and uses pre-existing designs that are known to be comfortable to users. Another benefit to this is that it allows for future iterations with other shoe manufacturers. Similar to the concept of the Ray-Ban x Meta collaboration of smart glasses and the Oakley x Meta collaboration, these soles could be made to fit different shoe uppers for different styles, sports, or activities.

Major Implementation Challenges

1. Pressure Measurement and Calibration

Problem: The pressure sensors must survive repeated loading during measurement while strides occur and retain accuracy.

Why it is manageable: This project does not necessarily require lab-grade accuracy of force from each sensor, but it important that they are accurate amongst each other so that pressure distribution can be accurately viewed. This is manageable because Adafruit FSRs have a rated life cycle of millions of actuations before failure. Also, FSRs can be calibrated during the build process to evenly measure load at all eleven points.

2. Estimating Stride Distance

Problem: Calculating stride length from accelerations can be inaccurate. IMU acceleration measurements contain noise and bias, so integrating acceleration to find distance would accumulate error. This could cause wildly inaccurate results.

Why it is manageable: Walking and running are repetitive, and the pressure sensors will provide reliable indication of when the foot is stationary on the ground. These known points in the gait/cadence can be used to correct the accumulated error. Also, knowing the user's height can make the stride easier to identify based on their theoretical max stride and nominal walking stride. Gyroscope tilt can also aid in this measurement.

3. Mechanical Durability

Problem: Electronics inside a shoe will experience repeated impact, bending force, compression, sweat, water, heat, and other harsh conditions.

Why it is manageable: The PCB and battery can be located in a low-flex area of the sole. For example, the arch area of the foot bends less than the toe area of the shoe, and it is also a lower impact zone than the heel or toe. Additionally, in this area, hard-shelled electronics cavities can protect from impact, flexing, and water. The sole above and below the cavity should be softer than the cavity to preferentially deform before the electronics casing does.

Development Schedule

Phase 1 – System Design and Component Selection *(9/10 – 9/23)*

- Finalize required measurements and system architecture.
- Select MCU, IMU, pressure sensors, battery, charging circuitry, and flexible interconnect method.
- Determine approximate PCB and battery placement within the arch.
- Begin CAD design of the sole and sensor locations.
- Order development boards, sensors, batteries, and prototype components.

Phase 2 – Sensor and Electronics Prototype *(9/24 – 10/7)*

- Interface the pressure sensors and IMU with an STM32 or Arduino development board.
- Develop initial step-detection and pressure-sampling software.
- Test BLE communication with a phone.
- Experimentally evaluate pressure-sensor placement and calibration.
- Print early sole sections or mechanical test pieces to evaluate flexibility.

Phase 3 – PCB and Sole Design *(10/8 – 10/21)*

- Complete schematic and custom PCB layout.
- Design battery protection, charging, sensor interfaces, and programming connections.
- Finalize 3D-printed midsole, outsole, wiring channels, and electronics cavities.
- Order the PCB by the end of this phase.

Phase 4 – Software and Mechanical Development During PCB Fabrication *(10/22 – 11/4)*

- Continue gait-algorithm development using development-board prototypes.
- Implement step count, cadence, ground-contact time, pressure tracking, and initial stride-length estimation.
- Develop the basic smartphone/BLE interface.
- Print and test the sole components.
- Develop synchronization between the left and right shoes.

Phase 5 – PCB Assembly and Full-Shoe Integration *(11/5 – 11/24)*

- Assemble both PCBs.
- Integrate pressure sensors, IMUs, batteries, and flexible wiring.
- Install electronics into the two printed soles.
- Attach the sole assemblies to the shoe uppers.
- Debug BLE communication and synchronization between shoes.
- Calibrate pressure sensors and gait measurements.

Phase 6 – Validation, Refinement, and Design Expo *(11/25 – 12/9)*

- Perform step-count, cadence, ground-contact, pressure-distribution, Bluetooth, and durability tests.
- Perform walking and running trials using both shoes.
- Correct remaining mechanical, electrical, and software problems.
- Seal and finish the final wearable assemblies.
- Implement stretch goals if time permits.
- Prepare the Design Expo demonstration and presentation.

Rough Budget

Component	Conservative Estimated Cost
2 STM32WB MCUs + spares	$40
22 FSRs + spares	$120
2 IMUs	$30
2 LiFePO4 Batteries	$40
Battery Charging/protection/power circuit	$40
PCB manufacture + revision	$120
Flexible wiring/ribbon cable/connectors	$50
TPU + rigid enclosure printing materials	$100
Insoles, shoe uppers, adhesives, sealants	$200
Misc. Components / spare parts	$100
Estimated Total	**$840**

With a five-person team, the available project budget would be approximately $1,250. An estimated cost of roughly $840 leaves sufficient room for failed components, PCB revisions, additional 3D printing, and

unexpected mechanical costs. The largest expected expenses are the pressure-sensor arrays, PCB fabrication, and mechanical construction of the two soles.

Evaluation and Real-World Testing

The system will be considered successful if it can:

- Correctly count steps during a known walking/running test.
- Measure cadence within acceptable error compared with manually recorded data.
- Identify the beginning and end of ground contact for each step.
- Display how pressure moves between different regions of the foot during a stride.
- Estimate traveled distance or stride length within a defined error.
- Reliably transmit collected statistics to a smartphone through Bluetooth.
- Continue functioning after repeated walking/running and reasonable exposure to moisture.

Testing Plan

Test 1 – Step Count Test: Walk exactly 100 steps while recording output from the shoe.
Success criterion: The system reports between 95 – 105 steps, error of +- 5%.

Test 2 – Cadence Test: Walk or run for a measured time while independently counting steps.
Success criterion: Calculated steps-per-minute agrees with the reference measurement within a tolerance of +-5% error.

Test 3 – Ground-Contact Time Test: Walk or run for multiple steps while recording on shoe and video on a phone. Determine the true contact time and the contact time calculated by the shoe.
Success criterion: Shoe output contact time agrees with the recorded video time within the tolerance of +-5% error.

Test 4 – Pressure Distribution Test: Walk or run with different gait styles. For instance, a heel strike, forefoot strike, weight biased toward inside, weight biased toward outside of foot.
Success criterion: The measured pressure distribution changes in the expected direction.

Test 5 – Bluetooth Test: Collect activity information while wearing the shoe.
Success criterion: Confirm that measurements are successfully displayed in the smartphone application.

Test 6 – Durability Test: Perform many tests, such as all of the above, and a ¼ mile run.
Success criterion: Verify that the electronics, sensors, sole, and wireless system continue operating accurately afterward.

Stretch Goals

If the primary system is completed ahead of schedule, possible additional features include:

- Heart rate measurement (hardware)
- Real-time visualization of foot pressure in phone app (software)
- Wireless charging (hardware)
- Battery temperature measurement (hardware)
- Running mechanics analysis (software)
- GPS tracking

Project Proposal 2: Hybrid Mechanical Smartwatch

High-Level Description

Problem / Idea

Modern smartwatches provide useful features such as vibrating for notifications, fitness tracking, and wireless connectivity, but they generally replace the appearance and mechanical operation of a traditional watch with a digital display. On the other hand, traditional automatic/mechanical watches can be visually and mechanically appealing, but they lack the connectivity and additional functionality of a modern smartwatch.

This project would combine both concepts into a single hybrid watch. The watch would contain a conventional mechanical watch movement with physical watch hands for normal timekeeping and appearance while also containing an independent embedded electronic system connected to a smartphone through Bluetooth. This would appeal to users who desire the technology of a device like an Apple Watch or Garmin but appreciate the craftsmanship and design of classic mechanical watches. Currently, these users are forced to trade one for the other, and some users even wear two watches at one time to solve this issue. The hybrid watch aims to eliminate this issue.

Proposed Device

The proposed hybrid watch would consist of:

- Conventional automatic or hand-wound mechanical watch movement
- Analog dial and hands
- Low-power BLE microcontroller
- Small rechargeable battery
- Accelerometer/IMU for activity and wrist-motion sensing
- Vibration motor for notifications
- Heart rate sensor
- GPS
- Solar charging system integrated underneath or around the dial
- Inductive charging coil as a secondary charging method
- Custom PCB designed to fit within the watch case
- Smartphone application or basic Bluetooth interface

How It Works

The mechanical portion of the watch would operate independently from the electronics. A conventional mechanical movement would drive the hour, minute, and second hands, meaning that basic timekeeping would continue even if the smart electronics lost power.

A separate low power microcontroller would communicate with the user's phone using BLE. The electronic system could receive notifications from the phone and alert the wearer through a small vibration motor. An accelerometer could also provide basic activity measurements such as steps and wrist motion. A heart rate sensor would also allow for fitness and health tracking.

The electronics would operate from a small rechargeable battery. During normal wear, photovoltaic cells underneath or surrounding the dial could slowly recharge the battery. Because indoor light or limited dial area may not provide enough energy for continuous operation, an inductive charging coil could provide a secondary charging method by placing the watch on a charging base.

Implementation

The primary electronics would use a low-power Bluetooth-enabled microcontroller, such as an MCU from the STM32WB family, along with a small accelerometer, heart rate sensor, battery-management circuitry, vibration motor driver, and wireless charging circuitry.

The largest technical challenges would probably be:

- Mechanical packaging and watch case design and manufacture.
- Power consumption and solar charging being insufficient.
- Interaction between the mechanical and electrical systems. Wiring and avoiding magnetizing the mechanical movement.

These challenges could be overcome by starting with a 3D printed prototype instead of a stainless steel case, creating a larger sized watch to allow for sufficient battery life, and a steel magnetism blocking plate separating the systems.

Rough Schedule

1. Planning/component selection (9/10 – 9/23)

2. electronics prototype: (9/24 – 10/7)

3. PCB & mechanical design: (10/8 – 10/21)

4. Software and charging development: (10/22 – 11/4)

5. Assembly/Integration: (11/5 – 11/24)

6. Testing/Expo preparation: (11/25 – 12/9)

Rough Budget

Component	Estimated Cost
Mechanical movement, dial, hands	$75
MCU, sensors, vibration motor, gps	$100
Battery and charging components	$50
Solar cells/experimental solar dial	$75
PCB manufacturing	$75
Case, crystal, strap manufacture	$300
Inductive charging	$40
Misc. & replacement parts	$100
Total	**$815**

Evaluation

The project would be considered successful if:

- Mechanical movement reliably keeps time while installed alongside electronics system.
- Watch can maintain a Bluetooth connection with a smartphone and communicate with app.
- The electronic system operates from its battery and can recharge by solar or inductive charging.
- Can be comfortably worn and is not too large or heavy.

EECS 473 HW 0

Layth Abdelkarim

Low-Cost Parking Lot Availability System

Introduction:

Finding an open parking space is often difficult in busy parking lots. Drivers may spend several minutes driving through a lot searching for an available space, creating unnecessary traffic and frustration. This is generally known as Parking Cruising, and is commonly seen even in UM parking lots during busy class hours. The Federal Highway Administration identifies Parking Cruising, where vehicles drive around in search of parking, has been identified as a contributor to congestion, air pollution, wasted time, and driver frustration. Our project would develop an embedded parking detection and guidance system capable of determining whether parking spaces are occupied and wirelessly communicating that information to a central device. The basic system would allow drivers to determine how many spaces are available without having to drive through the entire lot, and could also direct them to the available parking space through light indications.

This system is not unheard of in the US. UM parking spaces all have camera systems capable of identifying the availability of parking spaces, where the information is then posted online for drivers (such as on MGoPark). Despite this, camera systems produce certain accuracy and security risks that current systems need to work around. Additionally, searching for parking online while driving produces obvious risks and inconveniences that could be negated with a physical light indicator or display. Such lights are occasionally seen in indoor parking, but only scarcely in the US. Our proposal should provide a method for adding a physical display and find a more cost-efficient or accurate detection method over cameras.

Details:

The system needs a way to determine which spaces are occupied and communicate that with an online system or display device. There are two methods for this: one is to scan a wide range of parking spaces to identify the objects within it and whether they overlap with the parking spaces themselves. This would require a long-range sensor or camera system connected to a smart device for object/vehicle detection and a wireless interface. Current camera systems provide detailed information about an entire parking lot, and can be connected to open-source vehicle detection software to be easily configurable. These softwares do require significant processing power, and the cameras themselves may be affected by weather or objects obstructing the lens. Cameras are also security risks and can be hacked for previous images taken by them, so this implementation would require the algorithm to be done completely offline while also only transmitting the number of spaces available wirelessly. An alternative would be

the use of long-range radar sensors, as they can detect objects without relying on visible light and are largely weather-resistant, however compared to cameras may be more costly and would require custom-made algorithms to categorize vehicles and their locations.

The second method is to use a dedicated sensor for each parking space. Instead of scanning a large area, each sensor would only need to determine whether a vehicle is present within its assigned space. A low-range, low-cost radar sensor could be placed above, underneath, or around each space. The sensor would send the occupancy status to a microcontroller, reducing the amount of data that needs to be processed.

Each parking-space unit would contain a sensor, microcontroller, and wireless communication interface. An LED indicator could optionally be added to each unit, changing from green to red when the space changes from available to occupied. Alternatively, the system could use a single physical display at the entrance showing the total number of available spaces. These two options could be considered depending on the cost and complexity of the final system.

The parking-space units would wirelessly communicate their identification number and occupancy status to a central computer or online system. The system could then track all spaces and update the selected indicator system. Since only small amounts of information need to be transmitted, a high-bandwidth communication link would not be necessary.

Schedule:

Date	Task
9/7 - 10/1	After forming a team, we should decide on the type of system we will design (long-range sensors or dedicated parking-space unit) and begin looking at components that will fit our requirements. This will involve picking a sensor type and finding available sensors to purchase and test, deciding on the wireless communication method and possible parts, identify a microprocessor that supports the communication, and other components needed in the design. By the end of this period we should have a detailed high-level system and have picked a part or parts to use for each sub-section. Parts should be ordered as soon as possible. Buy backup sensors or cameras in case the original sensors don’t work down the line.

	Write the draft proposal by 9/18.
10/2 - 10/15	Parts should arrive within at most the first week of this section. Begin prototyping and getting familiar with the sensor and modules. Use computer software, raspberry pis or Arduinos first if they are easier to make test code on. Demonstrate that a sensor can detect a vehicle and communicate its state to another device. If it fails then pivot to another sensor and change requirements as needed.
10/16 - 10/31	Begin designing the custom PCB based on results from prototyping. Order the PCBs by the end of this period.
11/1 - 11/14	Continue developing the occupancy detection and wireless communication software while waiting for the PCBs. Implement the display and begin testing the complete prototype.
11/15 - 1/25	Assemble and test the PCB-based system. Test detection accuracy, wireless range, and communication reliability. Resolve hardware and software problems and begin implementing any additional features. At this point we should have a working prototype as a reference and as a backup if the PCB fails.
11/26–12/9	Perform final testing with multiple parking spaces and vehicles. Finish the enclosure and any stretch goals. Complete the poster and prepare the system for the Design Expo.
12/10 - 12/11	Finish final report.

Budget:

The budget is largely undecided as we will first need to finalize whether we will commit to long range systems or individual-space ones. The largest costs in both will likely be the vehicle-detection sensors, wireless modules, microcontrollers, and PCB manufacturing. A camera or long-range radar system could increase the cost, while an individual-space system would be cheaper to prototype but could require more sensors, so to make it cost-efficient heavy analysis on the price ranges would be needed.

Only a small number of parking spaces would need to be constructed for the demonstration, so it should not be necessary to build enough hardware for a full-sized parking lot. This also leaves room in the budget for replacement sensors, PCB revisions, and mechanical components.

Issues:

The largest concerns at this time are:

1. Determining which sensing method can reliably detect vehicles.
2. Deciding which part of the system should be the primary focus of the project.
3. Ensuring cost makes the system advantageous.
4. Developing a reliable wireless communication system.
5. Designing the mechanical mounting/enclosure for the sensors.
6. Preventing false detections from objects other than parked cars.
7. Completing and testing the PCB within the semester.

Issue 1 is the largest technical unknown because the best sensing method will depend on its range, cost, and ability to distinguish vehicles from other objects. Issue 2 will also be important because we should decide whether to prioritize developing a unique sensor system that could replace existing parking detection methods, with the LED or display system serving as an extension, or instead prioritize the parking availability display system and use a more generic camera-based detection system as a backup. This decision will determine which components and features we prioritize in the final design. The wireless system should be manageable because only small amounts of occupancy data need to be transmitted. The outdoor environment could also create problems with weather and sensor placement, although these should be manageable for a small-scale prototype.

The basic goal would be considered successful if the system can reliably determine which prototype spaces are occupied and communicate that information to a central system. Depending on which direction is prioritized, the LED indicators, physical display, or alternative sensing method could serve as additional features or stretch goals.

We will try to build our project with issue 3 in mind but we may find that our proposal is simply more expensive than current options. This is unless we decide to prioritize adding a display indicator over a new sensor, in which case cost is still important but secondary to making sure the new feature is implemented unless said feature drives up the price too much, which is unlikely.

Issue 5 is also tricky if we decide to make individual units, as most parking spaces are flat and empty, meaning we would either need to build a mount and find a place to put it while close to the parking space, or would need to put it underneath the space at the risk of being driven over by a vehicle.

Evaluation:

The system will be evaluated by testing how accurately it detects whether each parking space is occupied and whether it is applicable for current parking spaces. We will test the system by repeatedly having vehicles enter and leave parking spaces and comparing the detected status with the actual status. The wireless system will also be tested to ensure that occupancy updates are reliably transmitted to the central computer. A successful system should accurately identify occupied and available spaces, update the display or LED indicators with minimal delay, and continue operating reliably when multiple spaces change status. The project will be further considered a success if it has significant advantages over current camera-based systems, such as reduced cost, greater flexibility, etc.

Cost-Efficient Translator Accessory for Glasses

Introduction:

Language barriers can make it difficult for people to understand written information when traveling, working, or interacting with unfamiliar environments. Signs, menus, instructions, product labels, transportation information, and other written material may be difficult or impossible to understand when they are written in an unfamiliar language.

Current solutions typically require users to stop and use a smartphone. A user must take out their phone, open a translation application, point the camera at the text, and look down at the screen. This process is generally time consuming and may be disruptive to others who may see it as someone recording them. A wearable translation system could reduce this problem by allowing users to see translated information while continuing to look at their surroundings.

For example, a user traveling in another country could look at a sign written in Japanese. The camera on the glasses would capture the sign, the processing system would recognize and translate the Japanese text, and the glasses would display the English translation.

Current smart glasses that can perform this are quite costly and usually feature multiple other features in them. Moreover, the user has to buy the entire glasses for the feature. The proposed project would instead be an accessory that can be added onto the user's current glasses, potentially reducing the cost of the device and making it more user-specific. The accessory would also only provide translation abilities to improve real-time efficiency.

Details:

The system would consist of a small detachable module mounted onto a normal pair of glasses. A camera will capture an image of text in front of the user and send it to a processor, which will perform optical character recognition (OCR) to determine the words in the image. The recognized text will then be translated into the user's selected language. The translated text will be sent back to the glasses and displayed in the user's field of view.

A major design consideration is displaying the translated text without making the glasses large, heavy, or uncomfortable. Rather than placing a conventional screen directly in front of the user's eye, the system will use a small microdisplay and optical element mounted along the side or upper edge of the glasses. The microdisplay can produce the translated text, while a small angled reflector, prism, or transparent optical element directs the image toward the user's eye. This allows the display itself to remain very small while making the virtual text appear as though

it is floating in front of the user. The optical components will be positioned so that they do not physically touch the user's eye or obstruct their normal vision.

For the initial prototype, the display does not need to cover the user's entire field of view. It will instead provide a small area where short translated phrases can be viewed. This reduces the required display size and allows the electronics to be concentrated near the side of the glasses. The attachment will be designed to distribute its weight along the glasses frame rather than placing a large component on the nose or directly in front of the eye. A 3D-printed enclosure can then be designed to shape the module around the frame and made removable so that the glasses can still be used normally when the translation system is not needed.

The processing system will consist of a camera, microprocessor, wireless communication hardware, and display. Because performing OCR and translation entirely on a small wearable processor may be computationally demanding, the first prototype can use a smartphone or other nearby processing device for the more intensive image processing. The glasses module would send the captured image wirelessly to the processing device, receive the translated text, and display the result. This approach reduces the size, power consumption, and cost of the electronics mounted on the glasses while still demonstrating the complete system.

The system will also need to avoid continuously processing the same image. Instead, the camera can periodically capture frames and the software can determine when new text has entered the user's view. Only new or changed text would need to be processed and translated. This would reduce processing time, wireless communication, and battery consumption, helping the system provide a more real-time experience. To avoid security risks, images should be removed as soon as the OCR is performed. Further research into preventing images from being taken by hackers is needed.

The final device will be evaluated based on how accurately it recognizes and translates text, how quickly the translated text appears, how readable the display is, and whether the attachment is comfortable enough for extended use. The design will also consider privacy because the camera could unintentionally capture faces, personal documents, passwords, or other sensitive information. The system will therefore minimize image storage and, where possible, process images locally or transmit them only to a trusted processing device.

Schedule:

Date	Task
9/7 - 10/1	Research possible cameras, processors, displays, and wireless communication methods. Find open-source OCR apps or code to integrate. Determine the approximate processing and memory requirements for

	OCR and translation. Investigate whether processing should occur on the glasses, a smartphone, or a combination of both. Develop an initial mechanical concept and begin testing OCR and translation software. Write the draft proposal by 9/18.
10/2 - 10/15	Select the processor, camera, display, battery, and other major components. Order development boards and displays for prototyping. Begin testing the camera and display independently and establish communication between the glasses and external processing device.
10/16 - 10/31	Build a prototype containing the camera, processor, display, and wireless communication. Demonstrate that an image can be captured, transmitted, processed, translated, and displayed. Begin designing the custom PCB and order the PCB by the end of this period.
11/1 - 11/14	While waiting for the PCB, complete the OCR and translation software using the prototype hardware. Optimize image capture and processing to reduce latency and power consumption. Develop the communication protocol and begin testing the system under different lighting conditions and viewing distances. Design and manufacture the glasses attachment.
11/15 - 1/25	Assemble and test the PCB. Integrate the PCB, camera, display, battery, enclosure, etc. Identify and fix hardware and software issues. Test the complete system while attached to conventional glasses.
11/26–12/9	Perform final testing of translation accuracy, response time, battery life, weight, and usability. Complete any remaining improvements, prepare the Final Project Expo Poster and demonstration, and check stretch goals if the primary system is complete.
12/10 - 12/11	Finish final report.

Budget:

The following are the estimated costs for each potential component. Additional costs may be needed for extra or backup components. Our goal is to show that the primary function of our project can be used with as low-cost components as possible.

Component	Cost
Processor	~ $40
Mini Camera	~ $20
Mini Display	~ $20
Battery/ power electronics	~ $20
PCB	~ $20-40
3D-printed enclosure/mount	~ $20
Total	~ $140-160

Issues:

The largest concerns at this time are:

1. Achieving sufficiently fast OCR and translation for real-time use
2. Finding a display and setting it up to be readable when mounted to glasses
3. Keeping the system lightweight and comfortable
4. Maintaining reasonable battery life
5. Designing a reliable attachment for different glasses
6. Preventing camera data from creating privacy and security problems
7. Completing and debugging the custom PCB within the semester

Issue 1 is probably the largest technical unknown. OCR and translation can require significantly more processing than a typical embedded system can provide, especially if the system is expected to operate in real time. Using a smartphone as an external processor provides a relatively straightforward fallback if local processing is not fast enough. Moving the processing onto the glasses would therefore be considered a stretch goal.

Issue 2 is also potentially difficult. A normal OLED display is relatively easy to interface with but is not necessarily designed to be viewed as a heads-up display. The initial prototype can use a small display mounted near one eye to demonstrate the concept, while more advanced optical hardware can be investigated if time allows. There are also concerns with the display causing eye-strains or being inconvenient to look at, so research into custom lens design will be needed.

There are already DIY smart glasses online that may provide methods for achieving this at low cost.

Issue 3 and Issue 4 are related. A larger battery would improve operating time but would make the glasses heavier. Similarly, a more powerful processor could reduce processing time but increase power consumption. These tradeoffs will need to be evaluated during prototyping.

Issue 6 is particularly important because the camera will be continuously pointed at the user's surroundings. The system should avoid unnecessary storage or transmission of images and should provide an obvious indication when the camera is active. If images must be sent to an external device, secure communication will be used. These measures should make the privacy risks manageable, although completely eliminating the risks associated with a wearable camera is unlikely.

Evaluation:

The success of the project will be measured by how well the glasses can provide useful translations quickly, accurately, and comfortably in real-world situations. The system will be tested using signs, menus, labels, and other forms of text at different distances and under different lighting conditions. The OCR and translation accuracy will be measured by comparing the displayed translation with a known correct translation, with the goal of achieving at least 80% correct text recognition and translation for clear, commonly used text. The system will also be tested for latency, with a goal of displaying a translation within a few seconds after new text enters the camera's view. The glasses will also be tested over the user’s movement to see whether the system can adjust to slow or fast movement.

In addition to technical performance, the physical design will be evaluated by having users wear the glasses for an extended period and analyze the comfort, weight, and readability of the accessory, and whether the display obstructs their normal vision. Battery life would also be measured to determine whether the device can operate for a practical period without requiring frequent charging.

Adaptive Breathing Relaxation Recliner

Modern life moves at an increasingly fast pace, and many people rely on short periods of downtime during the day to rest or sleep. A student may have only thirty minutes between classes, an office worker may want to rest during a lunch break, and a traveler may have limited time to recover between activities. In these situations, the amount of available rest time is already limited, so spending a large portion of it trying to fall asleep significantly reduces the benefit of the break. At the same time, difficulty falling asleep is a widespread problem, with 15.4% of U.S. adults reporting trouble falling asleep most days or every day in 2024[1], which creates a need for technologies that can help users transition into sleep more quickly and naturally.

This project proposes an adaptive relaxation recliner designed to help users fall asleep by gradually slowing and stabilizing their breathing rhythm. Once the user lies down, piezoelectric sensors embedded in the recliner detect breathing-related body motion and estimate the user's current respiration rate. The chair then generates a slower target breathing rhythm through two forms of physical feedback: a speaker array that produces patterned low-frequency vibroacoustic stimulation, and a pneumatically actuated headrest that slowly inflates and deflates. Rather than requiring the user to consciously count breaths, watch a screen, or follow spoken instructions, the recliner provides subtle physical cues that encourage the user's breathing to gradually follow a slower pattern.

Details

The system can be divided into three major subsystems:

Respiration sensing: One or more piezo sensors would be mounted underneath the padding of the backrest. As the user breathes, motion of upper body creates small changes in pressure against the chair, so the system can establish a real-time monitor on user’s breathing rhythm.

Low-frequency vibration system: A 2*3 array of low-frequency speakers would be mounted inside the chair. These actuators would generate a gentle vibration to guide the user to a desired breathing rhythm and adjust based on the continuous input from the piezo sensors.

Pneumatic headrest: The second feedback mechanism would be a pneumatically actuated headrest which helps the user to relax. A small pump would allow the airbag to inflate and deflate.

[1] Short Sleep Duration and Sleep Difficulties Among Adults: United States, 2024. Amanda E. Ng, Ph.D., M.P.H., Lindsey I. Black, M.P.H., and Dzifa Adjaye-Gbewonyo, Ph.D. NCHS Data Brief No. 559, April 2026.

Schedule

9/10 – 9/20	*Planning* Discuss the feasibility of all proposed functions and investigate alternative implementation methods for each subsystem. Estimate the required processing and I/O capabilities and select a processor. Complete the project proposal draft by 9/17.
9/21 – 10/4	*Part selection* Select all the components for all subsystems and buy some backups within the budget. Select a processor based on the ports needed by the components. Have everything ordered by . Complete and submit the final proposal by 9/30.
10/5 – 10/18	*Breadboard prototype* Divide the team to develop and test each subsystem independently on breadboards. The piezo sensing subsystem should be able to acquire real-time data. The speaker/transducer subsystem should be able to activate each actuator individually. The pump should be able to inflate and deflate a balloon.
10/19 – 11/1	*PCB design & Algorithm* Two team members will rearrange the required microcontroller ports, complete the schematic, and design the PCB by 10/25. One team member will assemble the pump and the airbag together with a good air tightness. Another team member will develop the control algorithm that determines how the speaker array responds to the respiration signal.
11/2 – 11/15	*Assembly & Testing on PCB* Three team members will assemble the subsystems onto the PCB and perform system-level testing. In parallel, one team member will add the pressure sensing and safety mechanism to the pneumatic system.
11/15 - 12/10	*Assembly & Testing on the recliner* Install the complete system into the recliner. Test respiration sensing, vibrotactile feedback, pneumatic actuation, and the closed-loop control algorithm with the system in its final physical configuration. Based on the results, adjust sensor and actuator positions, vibration patterns, pneumatic timing, and control parameters as necessary. Prepare the poster and demonstration for the Design Expo.
12/10 - 12/11	Complete the final report.

Budget

Components	Quantity	Estimated Cost
Cortex-m4	3	$30
Piezo ribbon sensors	3	$100

Speakers & amplifiers	6	$100
Air pumps	2	$80
PCB fabrication and components	1	$100
Inflatable airbags	2	$30
Pressure sensors	2	$20
Power supplies	4	$120
Recliner	1	$100
Shipping	1	$75
Total		$755

Implementation Issues

1. Reliably detecting breathing using piezoelectric sensors.

   This can be addressed by testing multiple sensor locations and using filtering or multiple sensors to improve signal quality.

2. Preventing the vibration actuators themselves from interfering with the breathing sensors.

   This might be the most difficult part. We can reduce this interference through actuator-signal subtraction or by sampling during low-vibration portions of the output cycle.

3. Actuator effectiveness might be greatly reduces after installation.

   The recliner's padding may weaken the vibration or restrict the expansion of the pneumatic airbag. Therefore, both subsystems should be tested through padding with approximately the same thickness as the final chair from the beginning.

4. Make sure the pneumatic airbag won't explode in any case.

   The system will use pressure sensing and an emergency venting mechanism so the airbag can quickly deflate if abnormal pressure is detected.

Evaluation

We can conduct a small user study to evaluate the system. Each participant would first rest on a normal recliner for five minutes, remain awake for five minutes as a washout period, and then rest on our adaptive recliner for another five minutes. A chest-mounted respiration band would provide ground-truth breathing data for comparison with the piezo sensors. Participants would also self-report how quickly they felt they were able to fall asleep or become drowsy under each condition.

Intent-Aware Assistive Glove for Hand Movement

People recovering from stroke or other neurological injuries may retain some ability to initiate hand movements but lack the strength or coordination to complete them. This can make everyday tasks such as grasping objects or opening the hand difficult.

This project proposes an assistive glove that detects intended finger movement and provides mechanical assistance. Flex sensors on the back of the fingers would measure joint motion, while surface EMG sensors on the forearm could detect muscle activation associated with movement intent. These signals would be combined to determine when assistance is needed and activate tendon-driven actuators to help flex or extend the fingers. Heavier components such as motors and the battery would be placed near the wrist or forearm to keep the glove itself thin and lightweight.

Details

The glove would contain several flex sensors positioned along the back of the fingers, together with one or more sEMG sensors on the forearm. A microcontroller would continuously sample these inputs and run an intent-detection algorithm.

For example, if the sEMG signal indicates activation of the finger-flexor muscles and the flex sensors detect the beginning of finger bending, the system could classify the event as an intended grasp. The actuator system would then assist the fingers in closing. Similarly, an extension-related signal could trigger assistance in reopening the hand.

Mechanical assistance could be implemented using small tendon-driven actuators, where cables routed along the fingers are pulled by motors mounted near the wrist or forearm. Keeping the motors away from the fingers would reduce glove thickness and weight while leaving the palm relatively unobstructed.

Schedule

The project would begin with testing flex sensors and sEMG signals and determining whether they provide sufficient information for movement classification. The team would then prototype the actuation mechanism and develop the intent-detection algorithm independently. After both subsystems function reliably, they would be integrated onto a custom PCB and wearable glove. Final development would focus on reducing size, improving comfort, and testing assisted hand movements.

Implementation Issues

1. Reliable movement-intent detection
   The largest challenge is distinguishing voluntary movement attempts from noise, involuntary motion, or changes in hand position. Flex sensors alone may only detect motion after it has already begun, while sEMG signals can vary significantly between users. The system may therefore require per-user calibration.
2. Actuator safety
   The system should have a strict physical constrain on range of motion, and be able to immediately release assistance if excessive resistance is detected.
3. Actuator function
   Since heavier components such as motors and battery should be placed near the wrist or forearm, the mechanical design might require a lot of trials.
4. Water and moisture resistance
   A wearable glove may frequently be exposed to water. Sensors and wiring should be protected by water-proof fabrics.

Portable Renewable Energy Site Assessment Node

A field-ready embedded system that measures renewable resources and turns measurements into practical sizing guidance.

Communities and project developers often must choose solar, wind, or hybrid power systems with too little local data. Our portable node will collect site-specific measurements for weeks or months, survive outdoors, and provide a transparent first estimate of the renewable system that best fits the location.

Details

The Portable Renewable Energy Assessment Node is a self-powered outdoor instrument for deciding whether a location is suitable for a photovoltaic, wind, or hybrid energy system. It will measure solar irradiance, wind speed and direction, panel voltage, and battery state; timestamp the data; and store it locally. From those measurements, its embedded firmware will compute daily solar energy availability, wind-speed statistics, and an initial recommended mix of solar generation, wind generation, and battery capacity for a user-selected load profile.

The intended users are rural electrification teams, small farms, disaster-response organizations, and researchers evaluating sites that lack reliable grid power or weather-station coverage. The finished device will be a weather resistant enclosure mounted to a small mast. A cup anemometer and wind vane will sit above the enclosure, while a reference photovoltaic sensor and temperature sensor will be exposed to the environment. A microSD card will preserve raw measurements even when wireless communication is unavailable.

Unlike a generic weather station, this device is designed around an engineering decision: “What renewable system is likely to work here?” The display or companion computer report will summarize the measured resource, estimated daily energy, confidence flags for missing data, and a conservative preliminary configuration. This makes the prototype valuable even in remote locations with no cloud service.

The central deliverable is a custom microcontroller-based PCB. The board will integrate a low-power ARM microcontroller, analog front-end circuits, a microSD interface, real-time clock, battery monitoring, and connectors for the wind and solar sensors. The firmware will use FreeRTOS to keep periodic sensing, data logging, power monitoring, sizing calculations, and optional wireless communication as separate tasks. A modular architecture will let the team validate sensors before the PCB arrives, then transfer stable firmware and tested signal-conditioning circuits to the final assembly. Firmware interfaces will be small and independent of the application logic: wind_sensor, solar_sensor, storage, rtc, and power_manager modules will provide measurements and status through simple APIS. The sizing and logging tasks will call these interfaces rather than directly manipulating ADC, SPI, GPIO, or timer registers. This makes each hardware interface testable on a development board and portable to the final PCB.

The core system architecture includes:

1. Wind measurement: 3D-printed cup anemometer with a Hall-effect sensor for pulse counting, plus an eight-direction wind vane or magnetic angle sensor.
2. Solar measurement: Calibrated reference photovoltaic cell or irradiance sensor, panel-voltage measurement, and temperature compensation.
3. Embedded electronics: custom PCB with ARM microcontroller, ADC inputs, microSD logging, real-time clock, watchdog, low-power sampling, and fault detection; FreeRTOS will manage the concurrent periodic tasks.

4. Autonomous power and packaging: small solar panel, protected rechargeable battery, a simple charging circuit, and low-noise linear/LDO regulators. Reverse-polarity and short-circuit protection will be included, together with an IP55 or better enclosure.
5. Sizing algorithm: measured data are converted into daily energy statistics and a first – pass recommendation for photovoltaic area, wind contribution, and battery storage.

Schedule

The dates listed below are only estimates:

Period	Milestone and planned work
Sept. 10 - 24	Define user scenarios and success metrics; select sensors; demonstrate individual wind and solar sensor reading on a development board; freeze the PCB block diagram and order long-lead parts.
Sept. 25 – Oct. 15	Demonstrate all core interfaces on a development board: timestamped wind, solar, temperature, and battery samples are saved to microSD and processed by a preliminary sizing script. Complete schematic and PCB layout.
Oct. 16-Nov. 3	Order the PCB no later than Nov. 3. During fabrication, demonstrate the enclosure/mast mockup, power management test, documented data format, and desktop analysis workflow.
Nov. 4-17	Assemble and bring up the PCB; verify rails, ADC scaling, microSD, and sensor interfaces; integrate the weather enclosure and autonomous power system.
Nov. 18 – 25	Demonstrate the field-ready note: it logs a complete timestamped data set for at least 24 hours from its solar-and-battery supply, retains data through a controlled power interruption, and produces a solar/wind/hybrid recommendation.
Nov. 26 – Dec. 9	**Final testing, additional features, poster and expo** Targeting project completion 12/3. At this point we'll get our poster done and start on the final report. Continue to fix any remaining minor bugs.
Dec. 10 – Dec. 11	**Finish final report**

Budget

The prices listed below are only estimates:

Item	Estimated cost
Custom PCB fabrication and assembly supplies	$120
Microcontroller, Real-Time Clock, microSD, connectors, passives, and analog ICs	$115
Wind sensors, reference PV sensor/cell, temperature sensor, and small mast hardware.	$150
Solar panel, protected battery, charger, regulators, and protection components.	$110
Outdoor enclosure, cable glands, mounting, and 3D-printed mechanical parts	$95
Spare parts and contingency	$75
Estimated total	$665 for a five-person team ($133/person)

Issues

- Outdoor reliability: use sealed cable glands, conformal coating where appropriate, strain relief, and an enclosure rated at least IP55. Bench thermal and water-splash tests will precede extended outdoor deployment.
- Wind-sensor accuracy and friction: use bearings in the anemometer shaft, calibrate against a commercial anemometer or controlled fan setup, and report uncertainty rather than hiding it.
- Battery safety and uptime: use a protected battery, a charger designed for the chemistry, fusing, reverse-polarity protection, and firmware brownout logging. The device will gracefully save data before low-voltage shutdown.
- PCB integration risk: sensor and firmware validation will occur before PCB ordering; each board subsystem will have test points and serial debug access to speed bring-up.
- Scope risk: wireless telemetry and a polished dashboard are stretch goals. The system is successful without them because microSD logging and a local sizing report are part of the core design.
- Test-site risk: if outdoor access or a commercial reference instrument is unavailable, the fallback is a controlled fan, known-angle fixture, reference thermometer, and indoor solar-source test setup; development-board results remain available while PCB is fabricated.

Success will be evaluated at three levels. First, each sensor will be calibrated: wind speed will be compared with a commercial anemometer or controlled fan, wind direction will be checked at eight known angles, temperature will be compared with a reference thermometer, and the solar channel will be compared with a reference irradiance measurement or calibrated reference cell. We will target wind-speed error below 10% over the operating range, wind-direction error within one sector, temperature error within 2 degrees C, and logged data completeness above 95% during the required 24-hour autonomous run. A one-week outdoor trial is an extended validation target.

Second, the complete device will be tested as a field instrument. It must operate from its own solar-and-battery supply, retain timestamped data through a power interruption, and remain functional after controlled exposure to heat, dust, and light water spray. Third, we will validate the decision-support feature by comparing its estimated daily resource and preliminary system sizing against calculations performed independently from the same logged data. A successful final demonstration will show the physical node, an exported data set, calibration results, and a clear solar/wind/hybrid recommendation for a test site.

Additional stretch goal

- LoRa (Long Range) telemetry to a base station for sites where periodic remote status is more useful than physical SD-card retrieval.
- A simple dashboard that maps trends, flags sensor faults, and lets a user enter a load profile interactively.
- Multiple nodes to compare microclimates across a farm, campus, or rural community.

Mimicup

A Transforming Focus and Hydration Companion

Minicup is a playful embedded desktop companion that appears to be an ordinary cup, but transforms into a small robot at meaningful moments during a work or study session. A user starts a focus timer from a capacitive touch button or a Bluetooth phone app. At the end of the session, Minicup rises to reveal legs, deploys its head and arms, and delivers a short LED-and-audio celebration or a hydration/break remainder. It is intentionally not a drinking vessel: the cup-like outer shell is a visual disguise for a safe, self-contained electronic character.

The finished device will be a palm-sized, rechargeable 3D-printed object for a desk, dorm room, or makerspace. Its core experience is a synchronized transformation sequence that makes routine breaks visible and fun rather than another notification on a screen.

Details

A custom PCB will integrate an ESP32-S3, USB-C Li-ion charging, separate LDO-regulated logic/audio and servo power rails, capacitive touch, an audio amplifier, and interfaces for our micro servos and addressable LEDs. FreeRTOS will run separate tasks for motion, audio/LED effects, touch/Bluetooth events, and battery monitoring. Hardware will be abstracted through modules such as servo.h, leds.h, audio.h, touch.h, and battery.h, allowing each interface to be tested on a development board before PCB arrival. An earlier transforming-cup proof of concept establishes mechanical feasibility.

Schedule

- Weeks 1-2: define the transformation sequence, prototype the servo linkage, body shell, and balance with 3D-printed parts; select components and create the PCB schematic/layout.
- Weeks 3-4: order the custom ESP32-S3 PCB and enclosure parts. During the approximately 10-day PCB wait, develop firmware on a development board, create sound/light sequences, and test servo load and battery life.
- Weeks 5-7: assemble and bring up the PCB; integrate servos, speaker, LEDs, USB-C charging, and the printed mechanism; add fault detection for stalled servos and low battery.
- Weeks 8-10: tune motion choreography, conduct user tests, refine the enclosure, and prepare the final live demonstration.

Budget

The prices listed below are only estimates:

Item	Estimated cost
ESP32-S3, PCB fabrication/assembly, and passive components	$55
Four micro servos, LEDs, speaker/amplifier, touch sensor	$50
Battery, USB-C charging/protection, wiring and connectors	$35
3D-printed enclosure, gears/linkages, fasteners, contingency	$70
Estimated total	$210 for a five-person team ($42/person)

Issues:

Key risks are servo stalls, unstable motion, audio/servo electrical noise, and battery safety. We will use end stops, load testing, separate LDO-regulated rails with bulk capacitance, FreeRTOS watchdog/timeouts, protected Li-ion charging, and guarded moving parts. Minicup is not a drinking

vessel and will not be exposed to liquids. Success will be demonstrated by 100 consecutive transformation cycles and brief user tests with at least 8 people.

Additional Stretch Goals

- Bluetooth configuration from a phone app, proximity sensing, and selectable personality modes.
- Battery-runtime reporting and a small dashboard that summarizes focus sessions.

Inexpensive Traffic Light Detection Camera

More than 6 billion gallons of gasoline and diesel fuel are wasted every year in the United States by passenger cars, and light to heavy duty vehicles (1). Roughly 3 billion of those gallons of fuel are wasted due to these vehicles idling (1). This amount of wasted fuel is only exacerbated by drivers who are distracted at traffic lights and don't notice them changing from red to green. If these distracted drivers were to be alerted by an affordable traffic light color detection system to the presence of a green light, they could reduce their own wasted fuel and time, as well as the wasted fuel and time of the drivers behind them.

This technology has already been implemented in Tesla vehicles with Hardware 3.0, however these vehicles cost around $35,000 at their most affordable price (2). The technology also has yet to be sold as an aftermarket addition to existing vehicles, and competing technologies offer only front vehicle departure warnings, not true traffic light detection alerts. While front vehicle departure warnings are helpful to drivers behind moving cars, they don't offer a solution to a distracted driver at the front of a traffic light line creating congestion. An inexpensive green light detection system could save drivers time and money wasted on idling at lights for a small up front cost.

Details

The proposed system would include some sort of velocity or acceleration sensor (maybe an IMU) to detect if the vehicle is in motion or if it has come to a stop. If the sensor detects the vehicle has come to a stop, a windshield mounted camera will turn on and begin searching for traffic lights. If the camera detects a traffic light, it will then determine the color of the light. If the light is initially red and eventually changes to green, a short timer will begin, and if the acceleration sensor detects no movement by the end of the timer, an audible warning will be issued to the driver alerting them to the green light.

The main body of the proposed system would be housed in a compartment on the vehicle's dashboard, out of the way of any airbags. The main housing would contain a microcontroller running the necessary software, as well as the motion sensor, and would be connected via cable to the windshield mounted camera and to some on vehicle port for power (maybe OBD-II).

While several open source computer vision softwares are capable of traffic light recognition (3), the primary technical challenge for this system would be the reliability of the light detection. Poor weather or glare from the sun could likely confuse the color detection algorithm leading to false positives on green light detection, or a failure to detect a green light at all.

Schedule

Date	Stage
9/10 - 9/24	Research and Planning Once groups are formed, we will have two weeks to research and discuss the best hardware options, as well as potential other solutions to the problem. We will have to research the best options for vehicle movement sensing, speaker, camera, object detection algorithm, and microprocessor. We will also need to complete the draft

	proposal by 9/18.
9/24 - 10/8	Part Selection and Testing Once we have submitted the draft proposal, we will order a set of decided upon parts for testing. We can first order the microprocessor we decide with the computing power necessary for object detection and color detection, as well as the I/O capabilities needed for the vehicle motion sensors. We can then order a camera and IMU and test each part of the system independently. We also will submit the final proposal by 10/1.
10/8 - 11/3	Breadboard Prototype and PCB Design Once all components of the system have been proven capable of the task required of them, we can combine everything for a prototype system to confirm everything works at the same time. We will have to verify that the camera will operate under the condition that the IMU has measured that the vehicle has stopped, and then that the speaker operates under the condition that a green light has been detected but no vehicle movement has been detected. We will also have to design and order the PCB, hopefully by 10/27.
11/3 - 11/10	Continued Testing on Prototype and PCB Assembly We will shoot to have the PCB assembled by 11/8 for the milestone 2 report. While we do so we will work to iron out any issues in the breadboard prototype
11/11 - 11/25	Testing on PCB Once the issues on the breadboard prototype have been resolved and the PCB has been assembled, we will begin testing on the PCB. We will first verify that all the components work separately with the PCB, and then that the whole system operates how we expect it to. Throughout the process we will resolve any software bugs and make any necessary changes to the PCB.
11/26 - 12/10	Final Testing, Poster, and Expo We will aim to have finalized the project by 12/3, allowing for us to complete the poster and start on the report.
12/11	Final Report We will finalize and submit the final report.

Estimated Budget

Part	Cost per	Cost total
Raspberry Pi Compute Module 5	$67.50	$67.50
USB Camera	$30	$30
IMU	$20	$20
Speaker	$10	$10
Custom Vehicle PCB	$40	$40

Housing	$20	$20
Cables/connectors	$30	$30
Miscellaneous	$100	$100
Shipping	$30	$30
		$347.50

Evaluation/Testing

The primary method for testing the device will be on road tests. Group members can take the assembled prototype with them in their vehicle, and track how often it correctly detects a transition of a traffic light from red to green. If it is able to detect greater than or equal to 90% of the red to green transitions experienced, then the device will be determined to be a success.

Issues

My biggest concerns for this project are as follows:

1. Traffic light recognition reliability
2. Integrating the Raspberry Pi processor into the custom PCB

False positives for a green light recognition system caused by issue 1 could result in accidents, but reliability won't be determined until testing is done. Issue 2 will be difficult due to the need to include several peripherals for the raspberry pi processor, which may result in us needing to incorporate an existing raspberry pi board into a whole system PCB.

1. Alternative Fuels Data Center: Idle Reduction
2. Design Your Model 3 | Tesla
3. GitHub - RomeroRodriguezD/Real-Time-Traffic-Lights-classification---YOLOv3-OpenCV: Classifying traffic lights on a video streaming with OpenCV, classic computer vision using HSV channels to determine colors. · GitHub

Low-Cost Active Noise Cancellation Add-on Device for Pillows

Noise disturbances can make trying to sleep incredibly difficult, and many existing solutions are lacking. Normal methods of passive noise cancellation such as headphones and ear plugs are uncomfortable to be worn when laying down. In addition, current market active noise cancelling pillow devices have the electronics built into the pillow, making cleaning the pillow difficult. If there were a small, inexpensive, removable active noise cancelling device, users could remove the device and either wash or replace their pillow, without breaking the bank for a new electronic pillow entirely and also achieving undisturbed sleep.

Details

This technology would consist of microphones to listen to disturbing noises, a microprocessor to invert the audible disturbances, a speaker to play the inverted noises, and a rechargeable battery. The user could easily insert the device under the pillow they already own, switch it on, and sleep without disturbances. Once the pillow is dirty or needs to be replaced, the user can easily remove and recharge the device, wash or replace the pillow, and reinsert the device to continue sleeping undisturbed.

Schedule

Date	Stage
9/10 - 9/24	### Research and Planning Once groups are formed, we will have two weeks to research the best inexpensive hardware that can achieve active noise cancellation, as well as looking into methods of inverting measured sounds .We will also need to complete the draft proposal by 9/18.
9/24 - 10/8	### Part Selection and Testing Once we have submitted the proposal, we can order the agreed upon parts. Once the parts arrive we can test them individually, verifying that the speaker can produce the correct sounds, and that the microphone and microprocessor can read and measure sounds. We also will submit the final proposal by 10/1.
10/8 - 11/3	### Breadboard Prototype and PCB Design Once the individual components are verified to be working correctly we will combine everything for the Breadboard prototype. We will also have to design and order the PCB, hopefully by 10/27.
11/3 - 11/10	### Continued Testing on Prototype and PCB Assembly We will shoot to have the PCB assembled by 11/8 for the milestone 2 report. While we do so we will work to iron out any issues in the breadboard prototype
11/11 - 11/25	### Testing on PCB Once debugging is complete on the breadboard prototype, we will begin testing on the PCB prototype.

11/26 - 12/10	Final Testing, Poster, and Expo We will aim to have finalized the project by 12/3, allowing for us to complete the poster and start on the report.
12/11	Final Report We will finalize and submit the final report.

Estimated Budget

Part	Cost per	Cost total
ESP32	$5	$5
Microphone	$5	$5
Speaker	$5	$5
Rechargeable battery	$15	$15
Custom PCB	$40	$40
Housing	$20	$20
Cables/connectors	$10	$10
Miscellaneous	$50	$50
Shipping	$30	$30
		$180

Evaluation/Testing

We will test this device by measuring the sound with the device through a pillow. If the sound detected on the side of the pillow opposite the noise cancelling device is measured to be at least 10 dB less than the sound produced by the disturbance, the product will be determined to be successful.

Issues

1. Noise damping from the pillow may make it difficult for the microphone to pick up noise and for the speaker to cancel out disturbances.
2. The device inside the pillow may be uncomfortable.

If issue 1 is encountered, we can likely rectify the issue during the breadboard prototype phase and replace the speaker and microphone with more powerful options. If issue 2 is encountered we can experiment with different shapes and materials for the device's housing until we find a comfortable option.

Lucas Li

Moving Trash Bin

There have been too many times where I have been watching TV on my couch, perfectly wrapped up in a blanket, and just finishing my order of panda express. I try to throw my garbage into the trash bin, but I miss, and can never recover the same comfort again.
Now imagine a trash bin that tracks pieces of garbage in the air and moves to catch them. Living and working areas would be kept clean without anyone having to get up from their TV shows or from filling out excel spreadsheets.
Stationary trash cans rely on human accuracy, or physically moving your body, both which aren't fully optimal in a world where time is so sparse. A "smart" trash can, capable of tracking airborne trash, a rechargeable battery system, and a system that alerts users when it is almost full, ensures a clean space and a fun environment.
This smart trash can would be available in personal as well as commercial uses. For example, in kitchens, where your prep work counter is too far from the trash can, you can fling food waste over your head and expect that the smart trash can will go and catch it. Or, in art classrooms, where students can effortlessly throw drafts of sketches in a general direction without having to interrupt their thought process.

Details
There are a few key implementations to discuss. There would need to be a vision system mounted on each trash bin to survey surrounding areas for airborne garbage pieces. Cameras like the raspberry pi camera modules allow for high fps video capture, while small enough to mount on a trash can and cheap enough for the project's budget.
Next, we would need a processor with fast enough capabilities for a trajectory prediction algorithm, along with being able to identify and differentiate incoming pieces of trash, which could send data to the movement system of the trash can. The tracking algorithm, as well as the model for identifying pieces of trash will be heavily software implemented, so this may be a priority towards the beginning of the semester.
There will also be a 3 wheel movement system powered by DC motors to actually move the trash can. Here, we can use an ATmega to receive bytes from the tracking algorithm and drives PID motor loops to accurately steer the trash can to correct places.
The trash can should also be rechargeable, so an onboard battery like a LiFePO4 pack, will allow the trash can to remain operational during hours of use, and then transform into a stationary trash can during hours of minimal activity to wait for the battery to recharge. A reach goal would be to have a docking station the trash can could automatically return to so that the user does not have to manually charge and switch batteries for the trash can.

Schedule
The dates here represent a very rough timeline, as there may be many issues with debugging and troubleshooting, as well as other unforeseen conflicts.

9/10-9/21	Planning Spend about a week brainstorming and fully finalizing functionality and needs of the trash can, as well as getting to know members strengths/what parts of the project they feel most comfortable working on. Also start research on all the viable parts that we could use for each subsystem, taking into account cost, power, and functional requirements (such as fps for camera). Finish the draft proposal by 9/17, including revisions.
9/22-9/31	Part finalization Start finalizing parts used for each subsystem, taking into priority the camera and movement subsystems. Order those parts by 9/26. Also finalize a processor for the prototype, and start designing breadboards to test for basic functionality. Start very initial design and prototyping of individual subsystems, without worrying about integration, just making sure the selected parts meet requirements and limits of the project. Submit the final proposal by 10/1.
10/1-10/25	Algorithm design and PCB design Try to finish the design of identifying model as well as tracking algorithm. Ideally, we are able to identify a piece of garbage in the camera's FOV, and then output coordinates of the object's movement which we can then use for tracking algorithm. ALso want to have basic functionality of movement finished so that we can start integrating the two systems, probably using SPI protocol. From the breadboard prototypes, start designing the PCB layout and order the PCB's by the 25th. Start working on any CAD models needed for camera/motor mounts.
10/26-11/10	Integration Complete communication between tracking subsystem and movement subsystem. Start with basic functionality testing of SPI protocol between the two systems, before sending coordinate communications to DC motors. Once completed, work on PID loop testing so that the robot can move and slow down when the garbage piece starts its downwards trajectory. Should spend a lot of time troubleshooting and debugging software. Also make considerations for power once PCB arrives. Start assembling PCBs, done by 11/10 so that they can be integrated.
11/11/-12/5	PCB integration and testing Test PCBs and make sure integration works. Make sure power is accurate the that the trash can can operate for extended periods of time. Integration should be progressing, and the movement control loops should be ready for final tuning. Work on fixing PCB hardware issues as well as any software issues with the tracking model or communicating where the robot needs to move. Should be focusing on troubleshooting, debugging, and making the project presentable and functioning smoothly.
12/6-12/9	Testing, poster, and expo Finish final testing/troubleshooting. Finish poster and present at expo. Start working on the final report.
12/10-12/11	Finish final report

Budget

Here is a rough estimate of the budget. There are many other goals and functionalities that can be added to increase the complexity of the project once a meeting with 473 staff have been made, which will increase project costs. While this is a very rough estimate, there is a lot of leeway even with a group of 4.

Item	Quantity	Cost per part	total
Raspberry pi camera module 3	2	40	80
esp32	2	8	16
atmega	2	8	16
PCB	2	15	30
LiFePo4 battery	3	15	45
Trash can	2	10	20
Drive chassis	2	75	150
Misc Parts or replacements			100
shipping			80
total			537

Issues

Some concerns might be, 1. Developing a model to track objects. Might need CV background or need to do more research into a viable camera system. 2. Having a powerful enough processor that is able to update trajectory of garbage piece and send data to movement subsystem in time. There might definitely be some challenges in integrating the two subsystems and getting communication to work. 3. Mechanical CAD issues, such as figuring out where and how to house all the electronics, attaching the drive chassis, and doing test with the camera FOV. 4. Being able to communicate wirelessly (most likely through bluetooth) between multiple trash cans, so that the project operates like a system of trash cans, with the closest trash can moving to catch a piece of garbage.

Testing and Stretch Goals

To test our project, we will throw numerous different types of trash in a general vicinity of an 8ft by 8ft square. If the trash can is able to collect all types of trash with 100% accuracy and without tipping over, that is a success.

A stretch goal would be to either have multiple trash cans working together, or having them return to a docking station so they can charge automatically.

BCI Keyboard

Someone I know recently broke their hand, making it hard for them to type. There are also other individuals with medical conditions that impair their fine motor skills or their ability to type on a normal keyboard. This project is a BCI keyboard, allowing users to look at a matrix of letters on a screen and type words using signals emitted from the brain.

Details

This project would include the design of a headset with EEG electrodes that are able to measure the p300 wave your brain sends out when it wants to register a keycap hit. We would also need to design a PCB with a low noise analog design using filters, as well as gain amplifiers to capture the signal sent by the brain, which is in microvolts.

The processor would need high speed SPI capabilities to register the signals coming from the brain, as well as an ADC to convert the analog signals from the PCB to the software filtering algorithms.

The signal processing algorithm would include a bandpass filter to isolate the brain wave we are looking for on the EEG frequency spectrum, as well as a classifier to identify which letter in the matrix the user wants to type.

Schedule

9/10-9/25	Research, planning, and part selection Brainstorm ideas on how to implement the project, doing research on signal processing methods, current BCI interfaces and products, as well as think of part needed. Study EEG headwear, as well as the P300 signal. Start to order test gear and parts.
9/26-10/26	Design of subsystems Begin initial prototyping of analog circuit on PCB and submit order by 10/26, as well as begin signal processing implementation software. Spend a lot of time here working on PCB design.
10/27-11/13	PCB assembly and DSP Assemble PCB and integrate it into the project. Finalize DSP software and start working on interface for the keyboard, which will be a matrix of letters. Pass functionality testing and ensure there is a clear signal, meaning filters and amplifiers are working.
11/14-12/8	Integration and testing Mount PCB and CAD a housing, then integrate it into EEG headwear. Run live user tests and train classifier so that it accurately detects which letter the user wants to press. Troubleshoot hardware and software syncing, so the letters that are being typed are accurate with what the user wants.
12/9-12/11	Finish testing, make poster, present at expo, and write final report.

Budget
A rough budget is detailed as follows.

Item	quantity	Cost per	Cost total
EEG headset	1	80	95
ADC (measure biological signals)	2	60	120
PCB parts and order	2	60	120
Power (battery)	1	20	20
Miscellaneous		100	100
shipping		100	100
total			645

Issues
There are many implementation issues, 1. Low SNR. Because there is so much noise, it will be hard to extract p300 waves. Need analog circuits and DSP background. 2. Powerline saturation. Because the EEG signal is so weak, E&M signals from lights, PCB switching, and outlets can drown out EEG signals, adding more noise. 3. Data synchronization. The software algorithm needs to have precise synchronization between screen flashes and EEG timestamps, or else the p300 signal won't be caught and the key hit won't be registered.

Manasa Gollapalli

Offline Wearable Device for Group Hiking Coordination

Background and Overview

Outdoor groups often spread out from each other for reasons such as different walking speeds, stopping to rest, or missing a trail junction. However, the National Park Service recommends that hiking groups stay together or use a buddy system because separation can lead to participants becoming lost or injured[1]. NPS also recommends planning routes in advance, monitoring time and distance, and carrying conventional navigation resources such as a map and/or a compass[2]. The issue is that phone-based navigation resources, which people are increasingly reliant on, are often a poor solution because you would need a charged phone, cellular connection, and frequent screen use. Even when a Global Navigation Satellite System (GNSS) is available, accuracy depends on atmospheric conditions and factors such as signal blockage and receiver design.

Although devices exist that allow for communication in low-signal environments, they are often expensive and have features such as keypads and texting/calling capabilities. In addition, they are still handheld devices that require looking at a screen, which can be inconvenient and unsafe if people need to use their full attention to navigate the trail.

Details

The proposed device is a set of battery powered, portable, and wearable electronic devices that give a group awareness of their proximity to each other and their adherence to the route. Each of the participants would carry one wristband which receives GNSS position and time, uses an IMU to estimate movement, exchanges short status packets with other nodes through long range, sub-GHz radio, and delivers feedback through haptics and a small display. It is designed to complement but not replace conventional hiking preparation, giving information such as when a user is separated from the leader or assigned buddy, appears to leave a preloaded route, or has low battery/stale information.

Before the trip, each group will be able to load the route in GPX format via a physical connection. Each device will compare its location with the estimated route to check progress and track any error, with a threshold large enough to account for normal deviances. Each device logs its route, GNSS quality, radio status, battery, and events locally, then exports non-urgent data afterwards.

Additionally, there will be physical buttons for ease of use and communication rather than a complex touchscreen interface. These will be buttons such as STOPPED, NEED ASSISTANCE, or ACKNOWLEDGED, and they will transmit signals to the other nodes in real time.

Implementation Ideas

Each of the nodes will use a microcontroller running FreeRTOS, a GNSS module, a LoRa/sub-GHz radio module, IMU, display, haptic motor and driver, microSD storage, and rechargeable Li-ion battery. A u-blox MAX-M10-series GNSS module is a possible candidate since it is low-power. The

documentation for it reports continuous tracking power below 25 mW[3]. Also, SX1262-class radios are designed for long-range communication and have reported receiving current near 4.2 mA[4]. We will prototype with some development boards/modules before building one final custom PCB and enclosure.

Schedule

Dates	Milestone
9/10 - 9/23	RTOS task design, planning interfaces based on potential user needs. Order development modules. 9/18 Draft Proposal.
9/24 - 10/6	Get all parts ordered before formal proposal deadline 10/1. Get working: GNSS, radio, IMU, display, haptics, battery measurement, and multiple node communication on development boards. Begin GPX route processing.
10/7 - 10/25	Integrated prototype and PCB design: Demonstrate a prototype with three nodes that exchanges position/status data, calculates leader or buddy distance, detects a sustained deviation from a preloaded route, confirms delivery of check-in messages, and saves local trip logs. Finish up the system schematic and begin PCB layout, including component placement and preliminary RF/power routing, by the 10/15 Milestone 1 review. Finalize the PCB and enclosure design in time to place the board order by 11/3.
By 11/3 (but ideally earlier)	Order custom PCBs. During the 7–10 day PCB turnaround, finish route viewer/export, outdoor range tests, tuning haptic motors, and battery life testing on prototypes.
11/3 - 11/10	Milestone 2: 11/10 is when the PCB should be back and assembled.
11/10 - 11/23	Assemble and test PCB nodes; test cases like route deviation, separation alerts, stale behavior, node rejoin, radio outages, and GPX/CSV export, ideally

	outdoors in a real use case. Target completion of the enclosure by 11/23.
11/24 - 12/11	Refine enclosure/UI, collect final metrics, prepare poster and repeatable Expo demo. Due dates of Design Expo: 12/10 and Final report: 12/11.

Budget

This is only an estimate; however, the budget mostly constrains how many nodes or units we are able to build. We aim for a minimum of 3 to show the functionality.

Item	**Estimated cost**
PCBs, stencils, and shipping	$120
Development boards and prototype modules	$125
GNSS modules/antennas and LoRa modules/antennas	$225
IMUs, displays, haptics, storage, controls, and power hardware	$250
Batteries, enclosures, straps, assembly supplies	$150
Spares and unexpected costs	$130
Estimated total	**$1,000**

Issues

The concerns at this time include but are not limited to:

1. GNSS Location and Direction Uncertainty
2. LoRA implementation with minimal information losses
3. PCB integration and RF considerations with antenna and proximity
4. Multi-way communication (3+ devices)
5. Potential mechanical damage

Picking the right parts and thresholds can help with the first two concerns. However, the main issue will be transforming it into a wearable (issues 3 and 5). If the physics does not allow the device to be so small/watch-sized, it is possible we will have to pivot to a larger clip-on wearable. Multi-way communication (issue 4) is possible but will require very carefully planned prioritization.

Definite vs. Stretch Goal

Must-haves include set proximity tracking, major trail checkpoints, and critical message delivery. Nice-to-haves include more precise trail directions and more adaptive distance/proximity tracking

Karaoke Coach

Background and Overview

Music education has been increasingly defunded over the past few decades. This has led to a lower teacher-student ratio, meaning elementary and middle schoolers often see a decline in quality and individual attention. Especially during a whole-class song, it becomes increasingly difficult to develop fundamentals such as melody, pitch, rhythm, and musical expression. Children may also hesitate to sing alone because they are unsure whether they are matching the notes or entering at the right time. Research on elementary music education indicates that digital tools can improve motivation and support skills such as pitch and rhythm recognition, performance, and music literacy[5]. Therefore, a tool such as the live feedback/accuracy measurement karaoke device may increase motivation among these young students and improve music skills. The device would work better than an app in situations, for example, where there are multiple students holding separate mics. It also promotes active learning and social interaction unlike a typical mobile device.

Details

The proposed device is a self-contained, tabletop, singing practice station for elementary school students in music classrooms, after-school programs, and community music programs. It turns short singing exercises into a live interactive game where you can see immediate visual pitch and rhythm feedback and be able to go back to difficult sections to practice those again. In addition, a teacher can select a lesson, like echoing a four-note melody, and the device will automatically show lyrics or prompts, play accompaniment, and record results.

Implementation Ideas

The device will use a microphone input, a speaker, filtering to distinguish device playback from student voices, and a UI suitable for young children. It will run an RTOS because the system must continuously process microphone audio, simultaneously playing accompaniment, updating animation and visuals, and saving session information.

Continuous audio capture and playback require the use of buffering, so the device can process one portion of audio while the next is being played or recorded. This lets the lesson interface and progress-saving functions run without interrupting the live activity. Besides this, this device requires significant music signal processing and accurate filtering to process the audio properly.

Schedule

Dates	Milestone

9/10 - 9/23	Define classroom use cases, age-appropriate feedback, exercise format, and evaluation plan. Order prototype audio, display, and control hardware.
9/24 - 10/14	Brainstorm and develop interfaces. Demonstrate microphone input, reference-tone or accompaniment playback, animated display prompts, and simple note-matching feedback on prototype hardware. Write formal proposal by 10/1
10/15 - 11/2	Demonstrate one complete lesson with a short melody, visual pitch feedback, phrase replay, and progress summary. Complete initial PCB/enclosure design by Milestone 1 on 10/15; order PCBs by 11/3
11/3 - 11/9	While PCBs are fabricated, create additional exercises, refine feedback language and visuals, test the system with classmates, and improve audio reliability on prototypes.
11/10 - 11/23	Assemble and test the device with PCB, add lesson modes and progress tracking, measure feedback accuracy, responsiveness, and usability. Milestone 2: 11/10; target completion 11/23.
11/24 - 12/11	Finalize interface and enclosures, conduct a small usability test, prepare expo activity, poster, and report. Design Expo: 12/10; final report: 12/11.

Budget

Item	Estimated cost
Three custom PCBs, fabrication, and shipping	$120
Microcontrollers / development boards	$90
Microphones, audio amplifiers, speakers/headphones	$140
Displays, illuminated buttons, LEDs, and haptic motors	$110
Wireless modules, storage, and miscellaneous sensors/connectors	$75
Batteries/power hardware and enclosures	$85

Spares, assembly supplies, and unexpected costs	$100
Estimated total	**$720**

Issues

1. Real Time Audio Processing may have delays
2. Wireless Synchronization and Coordination
3. Room Noise

The ways we plan on solving these include giving audio processing the first priority, applying robust filtering, and having each station run its own local lesson/audio feedback loop.

Stretch Goal

A stretch goal could be having a haptic wristband to indicate rhythm and help students feel it better. Also making this work with a large group singing or choral music would be another stretch goal.

Sources

[1]: National Park Service, “Trip Planning Guide—Health & Safety.” https://www.nps.gov/subjects/healthandsafety/trip-planning-guide.htm

[2]: National Park Service, “Outdoor Emergency Plan.” https://www.nps.gov/subjects/healthandsafety/upload/NPS-Outdoor-Emergency-Plan-508c.pdf

[3]: u-blox, “MAX-M10S Integration Manual.” https://content.u-blox.com/sites/default/files/MAX-M10S_IntegrationManual_UBX-20053088.pdf

[4]: Everything RF, “SX1262—Semtech RF Transceiver.” https://www.everythingrf.com/products/rf-transceivers/semtech/531-690-sx1262

[5]: Stavropoulou, S., Georgaki, A., & Moschos, F. (2014). The effectiveness of visual feedback singing vocal technology in Greek elementary school. Proceedings of ICMC|SMC 2014, pp. 1786–1789.

Autonomous Emergency Smart Medical Payload (AESMP)

September 2026

1 Introduction

Response time for emergency medical services in rural areas can take almost 20 minutes longer compared to the national average.[1] This time can easily be the difference between life and death in emergency scenarios. Even in perfect environmental conditions, the distance alone that each fire station must cover is considerable compared to their urban or suburban counterparts. This is just one of the complicated scenarios where delivering aid becomes difficult. If there were a way to deliver medical supplies and remotely diagnose with the assistance of bystanders, we could better prioritize EMS resources and provide immediate care.

In order to provide this, we could use Unmanned autonomous systems to provide a device to the scene that would allow bystanders to communicate with first responders in sending critical vital signs back to emergency personnel to better inform them when they arrive on the scene. Additionally, we could provide a simple triage kit that would allow bystanders to stabilize an individual during this critical time before the arrival of medical services.

One of the leading companies in this space is Swedish Everdrone, which recently introduced next-generation systems capable of deploying crucial medical payloads to emergency calls in under three minutes. The current system they have developed delivers AED and anti-bleeding kits to the scene. [2]

2 Details

The scope of this project is limited to the deliverable device itself, not the platform that transports it. The drone and its dispatch, navigation, and payload delivery system are treated as an existing, third-party capability already provided by systems such as Everdrone's. The device would be designed as a compact, ruggedized package suitable for release from such a platform, containing two integrated subsystems: a vitals-monitoring and communication module, and a basic triage kit. The monitoring module would include simple, bystander-operable sensors such as a pulse-ox sensor paired with a two-way audio/video link over WiFi or cellular connectivity, allowing a remote dispatcher to guide the bystander through both vitals capture and use

of the triage supplies while the scene is unattended by trained personnel. Captured vitals and communication logs would stream to a web-based interface accessible to the incoming ambulance crew, so that by the time EMS arrives on scene, they already have continuous data and a documented timeline of the bystanders administered care. Because the device is designed only as a payload, its form factor, weight, and packaging would need to conform to whatever specifications the delivery platform requires, but the internal design, sensors, communication stack, and user interface are the sole engineering focus of this proposal.

3 Schedule

Below is a high level schedule that is subject to change as things progress throughout the semester.

9/10 - 9/26	**Planning phase:** After the teams form, I believe that the first order of business is to separate responsibilities for different parts of the stack to individuals. There are many different sub-components that need to be integrated together in order to provide the full deliverable. Therefore, we will take on different roles on the project and begin discussion on other options for each of the components. We will need to identify external sensors, communication devices and microprocessor devices in this stage. I hope to also get in contact with Everdrone during this time and ask for their advice.
9/27 - 10/10	**Design and Part selection:** At this stage we should begin selecting the individual components we need. For the microprocessor we need to select a prototype board that would allow us to communicate with all the devices we need to communicate with. The final proposal for our system must be submitted by the 1st and the majority of parts ordered by the 5th.
10/7-10/25	**Breadboard prototype and PCB design:** At this point we should have a decent idea of what our hardware and software stack will look like. At the end of this phase we should be ordering the PCB(s), therefore we should primarily spend this phase designing our PCB(s) and using our breadboard prototype in order to test the basic funcitonaility of our sensors and communication stack.
10/26-11/10	**First phase of software development and hardware setup:** We should now be focusing upon our sensor pipeline and backend system. We should also ensure during this step that we are properly encoding our data as this is a HIPAA compliance problem. Hopefully we also begin to work on the backend system that we will be using to display the data and communicate with the user.
11/11-11/25	**Second phase of software development and setup:** At this point we should have received the PCBs and we will begin work on programming them and implementing them into our final prototype design. If time allows it, testing this system on a deployment platform would be ideal (Just simply dropping it from a drone would be a good test).
11/26-12/9	**Polishing and Presentation:** This is the final phase of the design and mainly just polishing the different user interfaces. We also need to put together or final project presentation for this system.
12/10-12/11	**Final report due**

4 Budget

Pinning down a budget for this is honestly difficult simply due to whether or not we can find combined sensing equipment to integrate into our system or if each component will need to be handled individually.

Below is a first pass of the equipment we may need for the initial bread boarding prototype stage.

Device	Quantity	Cost per unit	Cost total
LILYGO T-SIM7600G-H R2 ESP32-WROVER	2	$75.00	$150.00
MAX30102 (Pulse ox sensor)	4	$2.50	$10.00
Advanced Trauma Kit	1	$120.00	$120.00
Misc. parts	1	$200.00	$200.00
Shipping	1	$100.00	$100.00
TOTAL COST	$580.00		

5 Issues

The biggest concerns with this project are the following

1. Communicating over cellular network
2. Interfacing with the medical sensors
3. Designing the housing for each component and the main package
4. Limiting scope creep

Connectivity is not the main issue with cellular connection, rather it's the rate and size of the data we would be sending through the communication hardware. My main concern is that the hardware will not be able to handle that, more research is nedeeded. Issue 2 seems doable. Most of these devices communicate via I2C or similar protocols and for the time being the only sensor we will attempt to implement is a pulse oximeter and the communication stack. Issue 3 is a non-issue if one of the team members on this project is a competent CAD designer. One of my biggest concerns in this whole project is adding additional features to the stack and not creating a solid demo. This system must be constrained to the MVP which is stream pulse ox data and communicating over cellular with a first responder. One stretch goal of this project would be to design a circuit that also provides blood pressure information to the medical personnel. Introducing new sensors will be discussed as we discover additional features.

References

[1] American College of Surgeons. *EMS Call Times in Rural Areas Take at Least 20 Minutes Longer Than National Average*. Accessed: Sep. 7, 2026. Oct. 2025. URL: https://www.facs.org/media-center/press-releases/2025/ems-call-times-in-rural-areas-take-at-least-20-minutes-longer-than-national-average/.

[2] Everdrone AB. *Everdrone – Drone as first responder*. Accessed: Sep. 7, 2026. 2026. URL: https://everdrone.com/.

Urban Search and Rescue Robot

September 2026

1 Details

In urban disaster scenarios, collapsed floors and debris piles create vertical obstacles that instantly trap standard wheeled robots. A lightweight search and rescue drone equipped with a jumping mechanism can vault over rubble, scale ruined stairs, and bypass ledges, acting as a highly mobile vehicle to locate trapped individuals. However, navigating deep into the core of a concrete and steel structure poses a massive challenge for standard high-frequency radio links, which quickly absorb and degrade. To be truly effective, the rover requires a low-frequency, high-penetration communication system capable of punching through multiple reinforced walls, ensuring operators maintain continuous remote control and receive live detection alerts from deep within the hazardous environment without risking the lives of human responders.

To achieve this, this system relies on a Raspberry Pi serving as the central compute node, paired with a sub-gigahertz telemetry system—such as 900 MHz LoRa transceivers—which penetrate building materials significantly better than standard Wi-Fi. While the Pi runs a lightweight YOLO model to optically detect human forms, it can interface with a small dedicated microcontroller that manages the motor control and the jumping mechanism. The jumping action is driven by a servo that compresses a spring, storing energy that releases instantaneously to launch it upward. By tying these different subsystems together using a publisher-subscriber framework like ROS, we can efficiently route the bounding box coordinates, motor commands, and vital telemetry back through the 900 MHz link to a remote ground station, ensuring reliable long-range operation regardless of structural interference.

Appliance Power Monitoring System

High Level Description

The Appliance Power Monitoring System is a compact, non-invasive device predictive maintenance for household appliances. The device would be a plug-in monitor placed between the appliance and the wall, monitoring AC voltage, current, power, and power factor. The device would use these electrical characteristics to detect overloads, unusual power signatures, or impending faults in the appliance. The device would transmit this data wirelessly to either an external monitor or an app via Bluetooth or Wi-Fi (TBD). The device is for those troubleshooting home appliances trying to determine if the appliances are behaving properly electrically. The device could also be used by homeowners trying to track energy consumption in their home.

The finished product would be a small, compact device that plugs into a wall outlet and has a plug for the appliance to connect, an example shown below:



Johnson Connect

The Appliance Power Monitoring System is worth building because it is a non-invasive device that could give real-time insights on the electrical performance of home appliances. Unlike other monitors on the market, this device will characterize what normal performance for the appliance is and alert the user of faults and anomalies the appliance experiences. The device doesn't just track how much power is used, it is able to use that data to alert the user when something goes wrong.

Implementation

Design Plan:

1. Hardware Prototyping (2-3 weeks): The beginning of the project will be spent on determining the hardware for AC sensing and isolation, power management and wireless communication.
2. Interface Prototyping (1-2 weeks): Concurrent with hardware prototyping, the interface for the device will be prototyped, focusing on accurately tracking AC voltage and current,

RMS voltage and current, and power factor. The interface should also track normal operating cycles and be able to detect when the appliance is operating outside of the norm.

3. PCB Design (1-2 weeks): The PCB will be designed after the hardware prototyping complete and seek into include the following:
    - STM32 or equivalent MCU IC
    - Programming/debug header
    - 3.3 V power regulation
    - AC Current/voltage isolation
    - AC Current/voltage sensing
    - ADC inputs
    - Flash memory or microSD
    - Wireless module/antennas
    - LED status indicators
    - Reset/reboot button

4. Firmware Development (2-3 weeks): After the firmware has been prototyped, the firmware for the MCU will be developed. The firmware would implement the same goals as the interface prototyping, but bringing it to the MCU level.
5. Debugging (Rest of Time): The rest of the time (4-6 weeks) will be spent debugging the firmware and hardware if needed. This includes making an enclosure for the PCB to be housed in that allows it to be plugged into the wall and for the appliance to be plugged into it. This would also include the data presentation and fault alerts.

Budget:

- PCB: ~$80 x 5
    - STM32 or equivalent MCU IC: ~$5
    - Voltage sensing ~$5
    - Current transformer ~$5
    - Analog op-amps/filters ~$3–5
    - 3.3 V regulator ~$1
    - Flash/microSD ~$2
    - Programming header ~$1
    - Protection components ~$5–10
    - Wireless module: ~$5
    - PCB fabrication ~$50
- PCB/Device Housing: ~20 x 5
- Miscellaneous Spending: ~$100
- Total: ~$600

The budget of this device is relatively low. This estimated budget could increase once deeper research is done into the PCB development and actual hardware implementation. The housing is planned to be 3-D printed so is relatively inexpensive.

Potential Implementation Issues

AC Voltage Input:
The device acts in between a home wall outlet and the appliance. Special attention needs to be placed on ensuring that the device is able to be powered by the home AC voltage, without damaging the PCB and using too much power. The device should also allow most of the AC voltage to go to the appliance without affecting energy consumption or appliance performance in a significant way, if at all. AC-DC conversion circuitry is complicated and can be difficult to properly design, so a lot of time should be spent on designing and optimizing it.

Appliance Characterization:
Fault detection depends on the device being able to characterize normal device operation and altering the user when it performs outside the norm. This characterization is done by tracking normal operation cycles and power consumption. The device must be able to first know when the device is operating normally. This means that the characterization must be some time after the appliance is first plugged into the device, to avoid characterizing the start-up cycle as normal operation. It will also have to be left alone for a non-insignificant amount of time for the device to capture “normal operation” The appliance must also be working properly in the first place to get a proper characterization, potentially conflicting with the troubleshooting goal of the device (the device would still show AC voltage and current, and power consumption, but would not be able to detect faults reliably). This can be solved by tracking normal operations of an appliance beforehand and hardcoding it into the firmware, but it would be better if it was more general and adaptable to any appliance.

Evaluation

The device will be working when it is able to accurately track AC voltage, current, power and power factor, and be able to determine when an appliance is operating outside of normal operating conditions. In order to determine its effectiveness, it will be tested with a small home appliance, such as a mini fridge. The device will be tested under a multitude of conditions, such as door closed with nothing in it, door closed with shelves filled, door closed with items blocking vents, door left slightly open, and door left wide open. When the device is able to determine when the mini fridge is using more power, such as when the door is left open, we will know that the device has characterized normal operation and can tell when the appliance is operating outside of it. For the device, success is being able to detect a fault with at least 80% accuracy with almost no false alarms, i.e. no faults detected during normal operation

Electronic Instrument Practice Mute

High Level Description

Practicing is the only way for musicians to improve and master their instrument. Whether they are playing exercises or audition pieces, being able to hear oneself when practicing is essential to know if you are playing correctly. But in situations, practicing can lead to a lot of frustrations from others and yourself. In houses with thin walls, practicing at full volume may lead to an angry text from your downstairs roommate trying to study. If you are in a rehearsal hall preparing for an upcoming audition with others already playing, you may not be able to hear yourself as clearly as you would like.

An Electronic Instrument Practice Mute would be a physical mute that can be placed in the bell of any brass instrument that would significantly reduce the sound output of the instrument. The mute would also have an incorporated microphone that would record your playing and play it back to you through headphones, allowing you to hear what you are playing despite the mute. The mute would also analyze your playing, giving your feedback on tuning, dynamics and pitch stability through an external monitor. The audio and analysis would either be sent wirelessly through bluetooth or through a wired cord.

Implementation

PCB:

- MCU
- Microphone
- Gain/filtering circuitry
- USB connection to computer or Bluetooth module

DSP:

- Sampling
- Digital filtering
- FFT
- Pitch detection
- Dynamic-level estimation
- Note detection

Mute:

- Off-the-market mute

Or

- Custom designed and produced mute

Evaluation

The mute will be tested on a trombone and compared against the app TonalEnergy. The mute will be tested to determine its accuracy in determining pitch and dynamics. It will be successful if its pitch is within 90% of what is determined by TonalEnergy and its dynamics within 90% of an external decibel measurement tool. In addition, sound will have to be reduced by at least 50%

EECS 473 Homework 0

Project Idea Proposal

Matthew Magro | mmagro[at]umich.edu

Project Proposal: Climb Tracker

In the United States alone there are over 800 climbing gyms spread across the country[1]. This represents a portion of the over 3.5 billion dollar climbing gym market which is predicted to over double in the next 10 years[2]. Climbing gym goers are made up of people trying it for the first time, the hobbyist, and those who have dedicated years to improving. However, frequently common among them all is the desire to track their climbing in a tangible way. There is a plethora of existing apps such as Mountain Project, Strava, Kaya, and more which allow climbers to record how long they are climbing for, and depending on the app which routes they have climbed. However, a common flaw with all of these is the often-tedious manual entry and cross-checking to determine which route was climbed and trying to remember how many attempts it took.

To attempt to solve this problem, I propose a system which could be sold to climbing gyms that in a privacy-conscience (i.e. without cameras) manner would track climbers on the wall to determine which routes they were climbing, how many attempts a particular route took and other data interesting to the climber. The primary method for doing this would be an Indoor Positioning System (IPS) with transmitters set up around the gym, and recievers worn by the climber.

Details

The device would consist of two primary components. The first and most important is a watch-like device which would be responsible for tracking the climber on the wall and in the gym. These would be the main way the user interacts with the system, and they would be the end "product" that patrons of a climbing gym would use. The tracking would take the form of an IPS using Ultra-Wideband (UWB) technology[3]. Therefore, the second, component would be a collection of beacons placed around the climbing gym such that every wall in the gym has line-of-sight two three or more beacons. These beacons would be responsible for broadcasting GPS-like signals such that the watch-like devices can use their pre-loaded knowledge about the climbing gym wall to place themselves on the wall.

Images

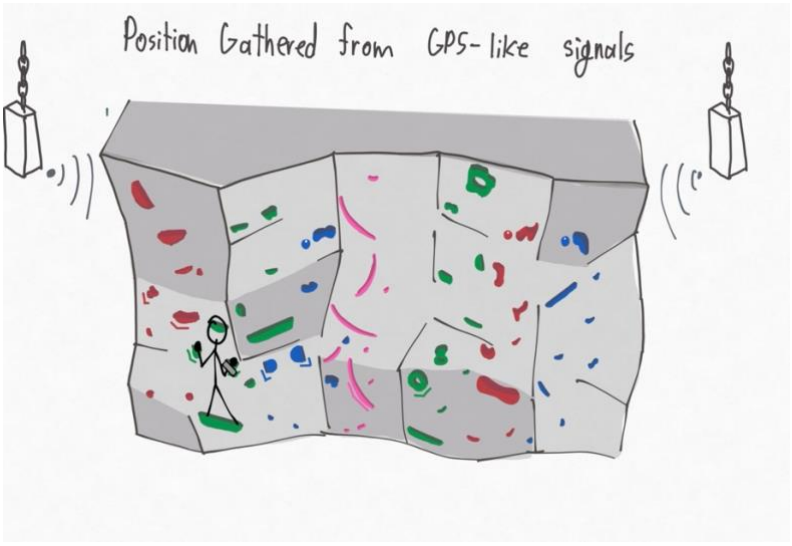


Figure 1: General Illustration



Figure 2: Details of Device

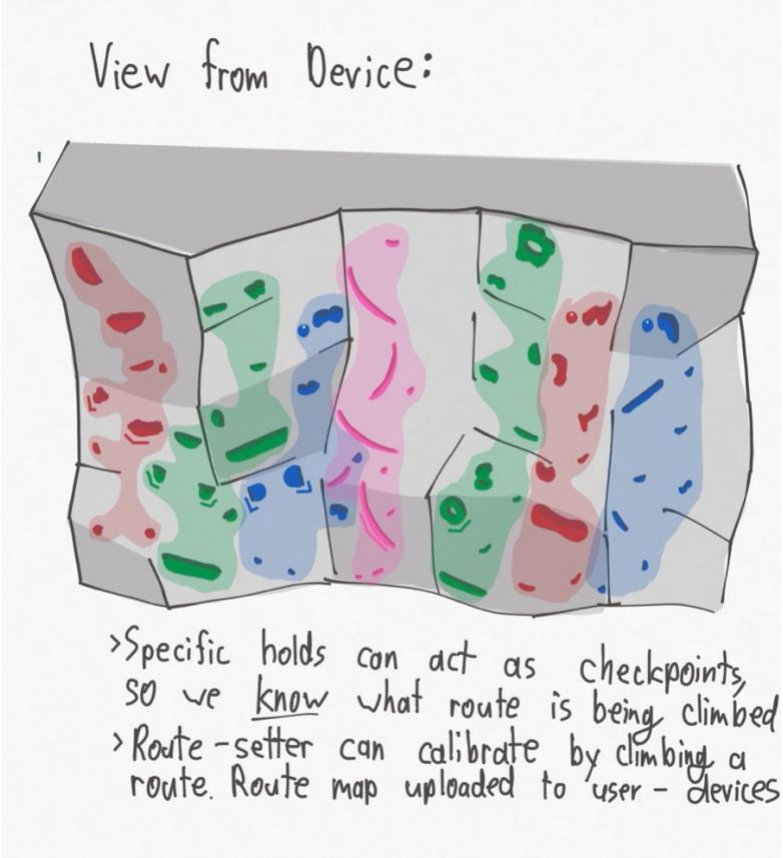


Figure 3: IPS visualization

Schedule

Dates	Activity	Description of Activities
9/17 - 10/1	Planning	Decide what technical decisions need to be made, what protocols we will be using, and the general architecture of the system.
10/1 - 10/6	Part Selection	What parts support the architecture we created above
10/6-10/17	2D loc Breadboard Prototype	Get a prototype on a breadboard capable of placing a receiver on a 2D plane
10/17-10/24	3D loc Breadboard Prototype	Get a prototype on a breadboard capable of placing a receiver in 3D space

10/24-11/3	Finalize PCB Design	Hopefully, this will not being started now and any decisions we made in the prototype stage can be implemented here
11/3-11/13	Finalize Algorithms	Prototypes likely are not incredibly accurate. Use this time before the boards come to sure-up the algorithms and adjust were necessary to improve accuracy
11/13-11/21	Assembly	Assemble PCBs, begin integration hell with the PCBs
11/21-12/7	Final Touches/Buffer	Add bonus features or use this time when another deadline is inevitably not met
12/7-12/11	Final Report	Write final report

Budget

I anticipate building 4-5 of EACH the wrist-worn receivers and 4-5 of the transmitters used for ranging. This should provide redundancy for the receivers. Additionally, since the ideal customer for this is a climbing Gym, multiple of these devices will be being used in the same space. Therefore, I want this to be something that this project accomplishes as well.

Component	Price	Explanation
UWB transceivers	~$120	For use in transmit and recieve
Misc Components	~$300	All the other random PCB components
Housing Costs	~$50	3d print filament for building boards
PCB board Costs	~$300	JUST the boards (i.e. No Assembly)
MCU costs	~$100	Processors for all boards
	Total: $870	

Issues

The largest concerns currently are

1. The amount of DSP/Comms work required for an accurate IPS system

2. The number of modules (especially beacon modules) that will be required for a functioning system. This is a concern in terms of cost and time
3. A space to test the system

Regarding Issue 1, this is perhaps the most brazen problem with this proposal. However, any project of this magnitude will have an area likely to have been unexplored by the students completing it. It is possible that given adequate time to begin learning these concepts the level of knowledge required could be well within the scope needed to complete the project.

Regarding Issue 2, I believe that with intelligent part sourcing and the manpower of a full team the price and complexity of the build would not be prohibitive enough to warrant not attempting it

Regarding Issue 3, given that the university opened the climbing center, with some wise connections I believe access to the wall could be granted. Therefore, in previous years this may have been an issue, but I do not believe it to be one now.

Sources

[1]https://usaclimbing.org/visit-your-local-climbing-gym/
[2]https://www.grandviewresearch.com/industry-analysis/climbing-gym-market-report
[3]https://en.wikipedia.org/wiki/Indoor_positioning_system

Project Proposal: Vision-Based Deer Deterrent

The state of Michigan has a large deer population that presents a persistent issue for gardeners in the state. This is primarily because deer will break into gardens and eat whatever they can get their hands on. There currently exists motion activated systems which will spray deer with harmless jets of water to spook them away from the garden. However, in high-trafficked areas these systems are not feasible due to their propensity to spray anything that moves including, cars, other animals, and humans. This is an issue that was brought up to me and suggested by a relative of mine who currently faces this issue. A vision based system which distinguishes between potential targets can reduce the possibility of a misfire on an animal or object that was not intended.

Details

The device would contain a camera (or array of cameras) responsible for running a real-time ML vision model capable of distinguishing between targets, and most importantly picking out, and placing deer on the systems map. Then the second system consisting of aimable water-equipped armatures can lock onto the identified target and deter it with a concentrated stream of water. The system will need the ability to identify and locate targets. Then aim and fire water at them.

This will likely take the form of two subsystems which communicate with each other. One subsystem responsible for the vision and the second, likely a simple robotic aiming system to deter the deer.

Schedule:

Dates	Activity	Description of Activities
9/17 - 10/1	Planning	Decide what technical decisions need to be made, and the general architecture of the system. This is also a good time to being the physical design to ensure it is feasible.
10/1 - 10/6	Part Selection	What parts support the architecture and design we created above
10/6-10/24	Prototype	Get a prototype on a breadboard capable of identifying and marking a target (perhaps with a laser at this stage)
10/24-11/3	Finalize PCB Design	Hopefully, this will not being started now and any decisions we made in the prototype stage can be implemented here

11/3-11/21	Assemble	At this point all parts should be sourced and assembly of the final project can begin
11/21-12/11	Buffer/Final Report	Bonus time for when deadlines are inevitably not met. Write final report

Budget:

Component	Price	Explanation
Mechanical Build	~$100	3D print costs, bearings, etc…
Servos/Stepper Motors	~$150	Multiple motors for multiple “outposts”
MCUs	~$50	Good processors, ML runs on personal-laptop, otherwise a jetson must be added to this total cost.
Misc Components	~$200	Resistors, Caps, etc…
PCBs	~$150	PCB ordering costs
	Total: $650	

Issues:

1) I am not well versed in Machine Learning technologies.
2) Significant mechanical work must be done to accomplish the goal of the project

Regarding issue 1, I do not believe it to be a large issue as there are likely many members of this class which have significant experience in this field. Therefore, this issue can be mitigated by recruiting the correct people.

Regarding issue 2, I believe that the mechanical work is within the scope of what could be accomplished by non-mechanical persons. I have some CAD experience and 3D printing is very accessible these days. Therefore, the barrier to entry is lower than in the past. With a little bit of pre-planning this issue can likely be mitigated.

EECS 473, HW #0: Project Pre-Proposals
Matthew Windsor

Proposal 1 (Primary): Non-Contact Post-Anesthetic Recovery Monitor for Veterinary Patients

High-level description

The greatest risk for any dog having surgery at a veterinary hospital is not while it is on the operating table but rather after the operation, in the recovery cage. During anesthetic recovery it is most likely that hypothermia, slow or irregular breathing, and dangerously low heart rates will go unnoticed since usually only one veterinary technician looks after several animals and can only check on any one of them every few minutes. At present most clinics still have a person walk down the row of cages, have a quick look in each one, and then move on. The present project suggests the use of a non-contact vital signs monitor which remains with the animal in the recovery cage and continuously records its temperature, heart rate, respiratory rate, and general movement, triggering an alarm as soon as there is any deviation outside a safe range.

The device is not attached to the animal; when a dog is coming out of anesthesia it is dazed, uncoordinated and generally tends to bite at anything it can reach, so a monitor with clip-on probes or adhesive leads would be pulled off, chewed through or cause irritation to the animal. The system on the other hand is incorporated into the bedding and the cage: a thin sensor pad is placed underneath or woven into the bedding in order to detect breathing movements and large body movements, while a small sensor pod, which is mounted on the cage bars above the animal, measures the heart rate and skin surface temperature from a short distance. A vet technician can look at a small display on the front of the cage or at a central station showing all the cages that are occupied in order to see the animal's current vital signs and is immediately given an audible or visual alert if anything goes beyond a predetermined threshold, so that one technician has the same coverage as a dedicated observer at each cage.

There is definite practical value in this approach since it addresses the main causes of morbidity associated with anesthesia in small animals (namely hypothermia and respiratory or cardiac depression), does not require any additional training for the staff in terms of attachments, cuts down on the physical effort involved in carrying out constant manual checks, and is cheap enough for a clinic to equip an entire recovery ward rather than keeping vital monitoring for just one "critical" cage.

Implementation schedule

- Sept 10 – Sept 23: Planning. Survey existing approaches to non-contact vital sensing (mmWave radar vital-sign literature, thermal-array skin temperature, pressure-mat respiration sensing as used in human sleep monitors). Choose a

radar module, a thermal array, and a pressure-sensor material. Set the alert thresholds based on the literature concerning veterinary anesthesia and recovery (for example, the normal resting respiratory rate, temperature, and heart rate ranges for dogs). Draw up the sensor-fusion and communication protocol between the pad, the sensor pod, and the display. Submit the draft proposal on September 18.

- Sept 24 – Oct 6: Part selection and dev-board bring-up. Order the radar module, the thermal array, the MCU development board, and the pressure-sensing material. Bring each of the sensors online one at a time on the development board, make sure that the radar produces a usable raw signal, that the thermal array feeds frames, and that the pressure mat generates a clean breathing waveform. The final proposal is submitted on October 1st , and all of the prototype parts are to be ordered by October 4th.
- Oct 7 – Oct 25: Carry out sensor fusion and integrate the breadboard. Combine all three sensor signals on a single MCU, implement extraction of the respiratory rate, check for a temperature threshold, and carry out an initial attempt at extracting heart rate using radar; research indicates that this is possible. Take advantage of this period to finalize and place an order for the custom PCB so that the delay involved in fabrication coincides with the development of the algorithms.
- Oct 26 – Nov 10: PCB bring-up and alerting. Assemble the PCB and transfer the fused firmware to it. Implement alert logic based on thresholds and trends that triggers both a local buzzer and LED as well as a wireless push to the cage-front or central display. Start to construct the washable pad and enclosure.
- Nov 11 – Nov 25: System validation. Test the complete system on a volunteer dog (or a heated/motion-simulating mannequin if repeatable trials are required), adjust the radar heart-rate algorithm in order to decrease the number of false alarms, and complete the construction of the pad and enclosure.
- Nov 26 – Dec 9: carry out final testing, produce the poster, and hold the Expo. Conduct head-to-head accuracy trials against reference instruments and complete the poster as well as finish off the documentation.

Rough budget:

What	Qty	Cost per	Cost total
MCU dev board / custom PCB (STM32)	2	$15.00	$30.00
mmWave radar module (e.g., Acconeer XM125)	1	$50.00	$50.00
Thermal array	1	$60.00	$60.00
Piezo/FSR pressure-sensing film for bed pad	1	$20.00	$20.00

Local display + buzzer/LED alert module	1	$15.00	$15.00
Battery + power management	1	$15.00	$15.00
PCB manufacture (2 boards, with spare)	1	$30.00	$30.00
Enclosure, washable pad materials, wiring, misc.	1	$40.00	$40.00
Total	**Total**	**Total**	**$260.00**

Because it is relatively cheap for a 5-person team, we can attempt multiple sensors and route them to a single display.

Other implementation issues:

- Reliability: because nothing is attached to the animal, the sensors have to tolerate the dog shifting position or curling up. The pressure mat and the radar have been selected, as they continue to work across a reasonable variety of postures, whereas a clip-on pulse oximeter does not.
- Safety: the device is placed in a cage containing an animal that is recovering and may be disoriented, which means that the cage must have rounded edges, use low-voltage electronics, and not include any small parts that a dog could chew and lose.
- Production/hygiene: veterinary bedding must be sanitizable between patients, which is why the pad has a washable, non-absorbent outer material with the sensing film sealed in rather than left exposed.
- Access to special facilities: ideally, full validation on an actual post-surgical patient would require the involvement of a veterinary expert, but the sensing pipeline can be developed and mostly validated using a healthy volunteer dog or a mannequin which simulates motion and heat.
- Power: Since the cages are stationary, the bed pad and sensor pod can be powered by mains or USB rather than relying solely on batteries, thus eliminating any concerns about battery life.

Evaluation and/or testing with regard to real-world issues

Success is measured against how well the system matches trusted reference measurements and how reliably it catches abnormal events, not just whether it reports numbers:

- For the respiratory rate, compare the number of breaths recorded by the system with those counted by a human observer using a stopwatch over a number of 1-minute trials; the aim is to achieve an agreement within ±2 breaths per minute.

- For the temperature, compare the reading obtained from the thermal array with that of a reference ear or rectal thermometer over a range of body temperatures and make sure that the readings agree within ±0.5°C.
- When it comes to heart rate, the rate measured by radar should be compared with that obtained from a pulse oximeter or by using a stethoscope manually since the latter method is most likely to fall below 'clinical grade'; the only objective then is for it to act as a useful show of trend (for example, within ±10 bpm) not to attain diagnostic accuracy.
- Make sure that the system is able to correctly identify gross movements (for example, the dog sitting up) and does not give false alarms when people are breathing normally over a multi-hour period during which the system is not in use.
- In order to respond to the alerts, simulate abnormal conditions, for instance, by covering the sensor pod to mimic a cooling trend or by holding one's breath to mimic apnea, and make sure that the system triggers its alarms within a clinically reasonable time frame, even though the number of false positives remains low over a long period of inactivity.

I think it’s also important I mention that my father owns a veterinary clinic and has verbally agreed to sponsor this project and be a customer.

Proposal 2 (Wacky): Live-Captioning Neckband

It is a small pendant attached to a lanyard worn around the neck which picks up the wearer's own speech and shows a live, scrolling caption of what has just been said on a small screen directed outwards, so that it functions like a wearable subtitle feature for a conversation. The device is designed for situations in which it is difficult for other people to understand the wearer's speech, such as when there is a hard-of-hearing person at the table, in a noisy environment, because of a strong accent, or due to a speech difference, as well as for more minor uses like a novelty item to aid conversation or as part of an accessibility demonstration.

The more realistic technical approach in this case is to focus on the display and the packaging, rather than starting from scratch to develop a speech-recognition system; it is not possible to run real-time automatic speech recognition completely on a microcontroller within one semester. Instead, the neckband's own hardware, consisting of a small microphone, a microcontroller such as an ESP32, and a scrolling LED matrix or a small OLED, pairs via Bluetooth with a companion phone app, which carries out the transcription by using the speech-to-text API built into the phone's operating system and then sends the resulting text back to the neckband to display it.

Rough plan: In the first two weeks have the microphone, the MCU, and the display working on their own and streaming test text; during weeks 3 to 4, overlapping with the delay caused by the PCB fabrication, get end-to-end Bluetooth text streaming working with a simple phone app; in weeks 5 to 7 order the custom PCB, set it up, and build the neckband case; in weeks 8 to 9 adjust the scroll speed, the way the text is chunked, and the latency to ensure its readable and stress-test the battery to see how it holds up over a full day of use; and in the final two or three weeks carry out the final integration, produce the poster, and the Expo.

The estimated cost is about $10 for the MCU module, $20 for the small display (this can be a scrolling LED matrix or an OLED), $5 for the mic, $15 for the battery and charging circuit, $20 for the PCB fabrication, and $20 for the enclosure, lanyard and other miscellaneous items, making the total come to around $90 to $100 per unit, which is well within budget and leaves room for a second unit or for having spares.
The major open risks include mic pickup in noisy rooms, the readability of the display and scroll speed at a glance, the end-to-end latency between speaking and the appearance of the captions and ensuring all-day battery life and comfort for a device worn around the neck.

Homework 0

Idea 1 - Wearable Glove-based Input Interface for EVA

High-level Description

Astronauts performing extravehicular activities (EVAs) have to operate with various computing interfaces while wearing bulky gloves that limit hand movements and dexterity [1]. Because of that, conventional interfaces such as keyboards, computer mice, and small touchscreens can be difficult to operate. NASA has explored various options such as a forearm-mounted touchscreen, voice recognition, gesture-tracking, and tactile interfaces [2] [3]. Previous work such as the Glove-Enabled Computer Operations (GECO) demonstrated that finger inputs can be detected through an EVA glove and that vibrations can be used for tactile feedback [4]. At the University of Michigan, the Collaborative Lab for Advancing Work in Space (CLAWS) is actively researching augmented reality (AR) and physical peripherals for EVA missions [5]. This project aims to build upon previous work done by NASA, SpaceX, and CLAWS by developing a physical, hardware-based for a glove-interface for these systems.

The goal of this project is to develop a low-motion, reliable input method that allows users to interact with an AR or computing interface without requiring a touchscreen. The proposed device will use sensors such as capacitive touch sensors or contact-based sensors to detect thumb-to-finger interactions. Similar approaches utilizing conductive fibers on the finger to mimic the natural conductivity of a bare human finger have been used by SpaceX to allow the use of a capacitive touchscreen [6]. The different thumb-to-finger interactions will be processed by a microcontroller and mapped to different commands such as selecting, scrolling, or navigating an interface. Haptic feedback would be utilized to provide confirmation of what command was utilized. Additionally, the glove will connect to a forearm-mounted LCD screen that displays the computing interface. The other side of the forearm will contain an electronics box that contains the custom PCBA, microcontroller, and power management system for the device. Wireless communication such as bluetooth would be used to send commands from the glove's fingertips to the forearm-mounted modules.

This proposal builds upon pre-existing sensing, haptic feedback, and display technologies to develop a compact and contained interface for environments with limited hand dexterity such as EVA missions while utilizing the gloves already worn by astronauts.

Implementation

Schedule

Week	Phase	Details
9/10 - 9/23	Planning	● Brainstorm ways to approach the problem, sensor types and communication protocols ● Nail down exact scope of the project
9/24 - 10/6	Part Selection, Schematic Design	● Select all parts needed for electronics modules, sensors, etc. ● Begin schematic design of the PCBA

10/7 - 10/25	Prototyping & PCBA Design	• Develop functional prototype without focus on form factor • Develop proof-of-concept for sensing aspect of the project • Finish the PCBA design for the board
10/26 - 11/10	PCBA Reviews & Software Development	• Perform a DFM and overall PCBA and schematic review • Order the PCBA • Develop the software functionality and RTOS aspects of the project
11/11 - 11/25	PCBA Testing & Software Testing	• Perform bring-up and bench-top testing of the PCBA • Combine the PCBA with the electronics box and verify overall functionality • Develop fixes to any PCBA, hardware, or software issues that arise • Narrow down the scope of any extra features
11/26 - 12/9	Last Minute Fixes, Deliverables	• Fix any remaining bugs • Work on poster, final report, and any other deliverables
12/10 - 12/11	Final Report	• Finish the final report

Budget

This budget is a first-pass estimation of overall costs of this project. While this may be an underestimate, this is still well below the total amount allocated to a typical 4-person group.

Component	Quantity	Cost per Unit	Total Cost
ATmega128	1	$15.01	$15.01
Capactive Touch Sensor	1	$9.06	$9.06
Bluetooth Module	2	$9.99	$19.98
LCD	1	$20.74	$20.74
Battery	4	$10.00	$40.00
PCBA	4	$50.00	$200.00
Other Components	N/A	$250.00	$250.00
			$554.79

Issues

The largest concerns for this project are

1. Distinguishing between intentional commands from accidental finger contacts or movements
2. Reliability of finger contact and sensing to send the respective commands
3. Compatibility and wearability of the system
4. Power consumption balancing the sensing, microcontroller, wireless communication, and the display

Issue 1 can be resolved by having specific finger interactions to enter command mode. Issue 2 can be resolved by careful selection of the sensing and placing of the sensing approach followed by sufficient

testing. Issue 3 can be mitigated by careful development of the PCBA and component selection to prioritize form factor. Issue 4 can be resolved by careful circuit and power-system design with LTSpice simulations as necessary, followed by testing and verification of the system's power consumption.

Evaluation & Testing

In order to verify that the system works, a hockey glove can be used to simulate an astronaut's suit glove to restrict hand dexterity in a similar manner. A successful version of this project will enable users to reliably control the LCD screen without requiring touchscreen inputs. This system can be tested both with and without the glove to compare command accuracy and determine whether the interface remains reliable despite the reduced hand dexterity caused by the glove.

Additional Stretch Goal

A stretch goal would be to include physiological monitoring into the forearm electronics box. The skin-facing part of the PCBA could include an interface for wearable sensors that monitor physiological signals such as heart rate and blood oxygen saturation. Another stretch goal would be to develop a flexible PCBA integrated into the glove or the electronics box to improve upon the overall form factor of the system.

Sources

[1] "EVA Systems - NASA." Accessed: Sep. 07, 2026. [Online]. Available: https://www.nasa.gov/reference/jsc-eva-systems/

[2] R. J. Adams, A. B. Olowin, B. Hannaford, and O. S. Sands, "Tactile data entry for extravehicular activity," in *2011 IEEE World Haptics Conference*, Istanbul: IEEE, Jun. 2011, pp. 305–310. doi: 10.1109/WHC.2011.5945503.

[3] "Space Suit RoboGlove (SSRG) | T2 Portal." Accessed: Sep. 07, 2026. [Online]. Available: https://technology.nasa.gov/patent/MSC-TOPS-80

[4] R. J. Adams *et al.*, "Glove-Enabled Computer Operations (GECO): Design and Testing of an Extravehicular Activity Glove Adapted for Human-Computer Interface," presented at the American Inst. of Aeronautics and Astronautics, Reston, VA, United States, Jul. 2013. Accessed: Sep. 07, 2026. [Online]. Available: https://ntrs.nasa.gov/citations/20140006198

[5] "CLAWS." Accessed: Sep. 07, 2026. [Online]. Available: https://claws.engin.umich.edu/

[6] "What is the SpaceX IVA Suit?," New Space Economy. Accessed: Sep. 07, 2026. [Online]. Available: https://newspaceeconomy.ca/2025/11/27/what-is-the-spacex-iva-suit/

Idea 2 - Seizure Detection Wearable

High-level Description

Epileptic seizures can occur when a person is alone or not in the presence of a caregiver. A seizure-alert device can provide an added layer of safety in those scenarios by detecting seizure-like activity and notifying emergency personnel and emergency contacts. The current seizure detection methods on the market mainly rely on detecting the shaking movements that occur during seizures. However, this approach only allows the detection of tonic-clonic seizures or focal-motion seizures where there is enough movement to be detected [7]. These devices typically rely on accelerometer data to detect a seizure. In the hospital, electroencephalography or EEG is typically used to detect seizure activity but is impractical for the continuous monitoring necessary for daily life [8]. This project aims to explore whether added physiological signals can allow for seizure detection and monitoring for seizures that do not produce enough motion to trigger the existing seizure detection methods on the market.

The goal of this project is to combine an accelerometer and gyroscope for movement patterns with additional physiological signals for a multi-modal wearable interface for seizure detection. The Embrace wristbands from Empatica received FDA approval for seizure detection while combining accelerometers and electrodermal sensors [9]. Additionally, photoplethysmography (PPG) via PPG sensors and electromyography (EMG) via EMG sensors have been shown to also be useful in detecting seizure activity [10] [11]. Therefore, this interface would combine the motion detection, cardiovascular, and muscle-activity measurements. The sensor data would be processed via a microcontroller while an algorithm would analyze the signals and determine whether the physiological signals are consistent with a seizure-like event. If an event is detected, the device would utilize wireless communication to communicate with a phone to send an alert.

This proposal builds upon pre-existing wearables by creating a more robust detection and alert system for seizure-like activity.

Implementation

Schedule

Week	Phase	Details
9/10 - 9/23	Planning	● Research sensing methods, brainstorm sensor types to be utilized for different physiological signals
9/24 - 10/6	Part Selection, Schematic Design	● Select all parts needed for electronics modules, sensors, etc. ● Begin schematic design of the PCBA
10/7 - 10/25	Prototyping & PCBA Design	● Develop functional prototype without focus on form factor ● Develop proof-of-concept for sensing aspect of the project ● Finish the PCBA design for the board

10/26 - 11/10	PCBA Reviews & Software Development	• Perform a DFM and overall PCBA and schematic review • Order the PCBA • Develop the software functionality and detection algorithm
11/11 - 11/25	PCBA Testing & Software Testing	• Perform bring-up and bench-top testing of the PCBA • Combine the PCBA with the electronics box and verify overall functionality • Develop fixes to any PCBA, hardware, or software issues that arise • Narrow down the scope of any extra features
11/26 - 12/9	Last Minute Fixes, Deliverables	• Fix any remaining bugs • Work on poster, final report, and any other deliverables
12/10 - 12/11	Final Report	• Finish the final report

Budget

This budget is a first-pass estimation of overall costs of this project. While this may be an underestimate, this is still well below the total amount allocated to a typical 4-person group.

Component	Quantity	Cost per Unit	Total Cost
ATmega128	1	$15.01	$15.01
Accelerometer	1	$30.00	$30.00
Bluetooth Module	2	$9.99	$19.98
PPG Sensor	1	$40.00	$40.00
Electrodermal Sensor	1	$20.00	$20.00
Other Sensors	3	$50.00	$150.00
Battery	4	$10.00	$40.00
PCBA	4	$50.00	$200.00
Other Components	N/A	$250.00	$250.00
			$764.99

Issues

The largest concerns for this project are

1. Develoipng & training the algorithm for proper detection of seizures using physiological activity
2. Integrating the various physiological signals together for detection
3. Minimizing false positives for detection
4. Developing a form factor that does not interfere with daily activities

Issue 1 will be mitigated by developing the algorithm utilizing publicly-available datasets of seizure activity physiological signals. Issue 2 will be mitigated by careful placement of the sensor for best-quality measurements as well as proper processing and filtering of the signals. Issue 3 can be mitigated by proper algorithm development and utilizing data of both seizure and non-seizure activity. Issue 4 can be mitigated by careful design of the PCBA and the overall device fixture.

Evaluation & Testing

The system would have to be evaluated in conjunction with a medical facility and a physician. The primary measures of success would be success rate in determining seizure activity, false alarm rate, and how long it takes to recognize seizure activity. A reliable system would be able to accurately determine seizures in real-time while minimizing false alarms.

Additional Stretch Goal

A stretch goal would be developing some kind of interface to display the physiological data collected in real-time along with whether or not a seizure is detected.

Sources

[7] "The Role of Seizure Alerts SUDEP | Epilepsy Foundation." Accessed: Sep. 07, 2026. [Online]. Available: https://www.epilepsy.com/complications-risks/early-death-sudep/role-seizure-alerts

[8] A. Wesal *et al.*, "Modalities and algorithms for generalized motor seizure detection and prediction: a scoping review," *Expert Rev. Med. Devices*, vol. 23, no. 4, pp. 409–424, Apr. 2026, doi: 10.1080/17434440.2026.2632757.

[9] G. Regalia, F. Onorati, M. Lai, C. Caborni, and R. W. Picard, "Multimodal wrist-worn devices for seizure detection and advancing research: Focus on the Empatica wristbands," *Epilepsy Res.*, vol. 153, pp. 79–82, Jul. 2019, doi: 10.1016/j.eplepsyres.2019.02.007.

[10] M. Glasstetter *et al.*, "Identification of Ictal Tachycardia in Focal Motor- and Non-Motor Seizures by Means of a Wearable PPG Sensor," *Sensors*, vol. 21, no. 18, p. 6017, Sep. 2021, doi: 10.3390/s21186017.

[11] S. Beniczky, I. Conradsen, O. Henning, M. Fabricius, and P. Wolf, "Automated real-time detection of tonic-clonic seizures using a wearable EMG device," *Neurology*, vol. 90, no. 5, pp. e428–e434, Jan. 2018, doi: 10.1212/WNL.0000000000004893.

High-level description

Background / Motivation

Bluetooth audio is used by hundreds of millions of people every day, but the technology is still restricted by single-device pairing limitations and noticeable transmission latency.

In everyday life, these constraints can create constant issues. Roommates watching TV late at night cannot connect two pairs of headphones together without experiencing lip-sync delay. Party hosts trying to play music across multiple rooms cannot sync audio across different speaker brands without running long aux cables across the floor.

Existing industry solutions fail to solve these everyday problems universally. High-end multi-room audio systems (like Apple AirPlay) cost hundreds of dollars per speaker, require strong Wi-Fi networks, and lock users into proprietary hardware ecosystems. Software-based group listening features (like Spotify Jam) attempt to bridge the gap, but they force every user to have a paid subscription, require a continuous internet connection, and fail to work with any audio source like a TV or offline MP3 Player. Newer hardware protocols like Bluetooth Auracast are capable of multi-device streaming, but adoption is slow, expensive, and incompatible with traditional wired speakers and headphones already in use. Because current alternatives are either expensive, locked behind specific apps, or restricted by brand ecosystems, a low-cost, universal, and low-latency hardware solution is needed.

Details

This project is a plug-and-play wireless audio system that broadcasts real-time sound from a single audio source (a phone, laptop, soundbar, or iPod) to multiple speakers or headphones simultaneously. The purpose of this project is to simplify connections through a direct plug-and-play setup, linking multiple audio devices together regardless of their manufacturer. It allows several people to listen to the same music on different pairs of headphones, achieving the group-listening functionality of services like Spotify Jam, but without being restricted to a specific app, subscription, or internet connection. Crucially, the system ensures that every receiving device plays audio in perfect synchronization.

The finished system consists of two primary hardware units:

Central Transmitter Hub**:** Plugs directly into the 3.5mm or line-out jack of a phone, laptop, soundbar, or media player. It features a visual pairing indicator, a rechargeable battery for portable use, and physical control buttons to pause or skip tracks on the source media.

Receiver Dongles**:** Multiple identical, compact receiver units that plug into any standard audio jack on a speaker or pair of headphones. Each receiver is equipped with individual volume control buttons, a visual pairing indicator, and an onboard rechargeable battery for easy placement anywhere in a room.

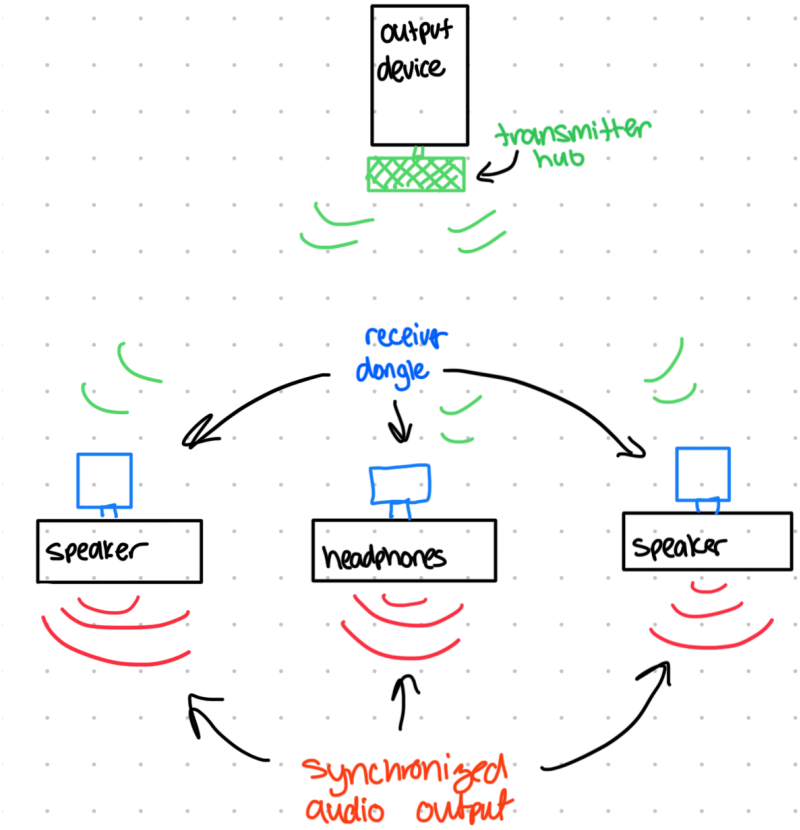


Figure 1: High-Level Block Diagram and Signal Flow

This system is built for party hosts, roommates watching TV late at night, or friends who want to listen to the same music together while studying. This project is worth building because it creates a multi-speaker sound system without audio lag, expensive proprietary ecosystems, or complicated network setups. It transforms standard wired speakers into a synchronized, wireless audio network with zero hassle.

Development Schedule

- **9/10 – 9/23 (Planning & Wireless Architecture):**
    - Map out transmitter-to-receiver wireless transmission specs and select MCU/RF chips and audio converters.
- **9/24 – 10/20 (Part Selection & Breadboard Bring-Up):**
    - Order all parts and hardware.
    - Build a working point-to-point wireless audio link on breadboards.
    - Finalize Rev 1 schematic/layout and order PCB REV 1 early (by 10/20).
- **10/21 – 11/5 (Rev 1 Assembly & Hardware Debugging):**
    - *During 10-day PCB shipping:* Continue software development on the breadboard setup.
    - Assemble PCBs after they arrive (~10/30).
    - Test PCB power, audio quality, and wireless antenna performance to identify footprint errors, noise issues, or trace miswiring.
- **11/6 – 11/18 (Rev 2 PCB Revision & Lead Time):**
    - Apply Rev 1 hardware fixes to board layout and place the **Rev 2 PCB order (by 11/7)**.
    - *During 10-day PCB shipping:* Finalize synchronization firmware using Rev 1 boards
    - Design and print enclosures for the hub and receivers.
- **11/19 – 11/25 (Rev 2 Final Assembly & System Integration):**
    - Assemble Rev 2 PCBs and fit components into final 3D-printed housings.
    - Verify clean audio streaming, battery management, and multi-unit synchronization.
- **11/26 – 12/9 (Final Stress Testing & Design Expo):**
    - Test range and sync stability in congested wireless environments.
    - Complete project poster, report, and live demo setup.

Budget:
Expected Total Project Spend ~$465
Bare MCUs & Audio Hardware ($50):
4x bare wireless microcontrollers and digital audio converters for one transmitter and three receivers.
Prototyping & Adapters ($20):
Protoboards, wires and sensors for breadboard prototyping.
Custom PCB Fabrication — 2 Runs ($160):
Manufacturing costs for two 4-layer custom circuit board runs.
Power & Battery System ($50):
Rechargeable LiPo batteries, USB-C charge management chips, and low-noise voltage regulators.
Passives, Jacks & Antennas ($45):
Audio jacks, buttons, 2.4 GHz antennas, crystals, capacitors, and resistors.
Enclosures & Mechanicals ($40):
3D-printing filament, hardware fasteners, and rubber feet for the transmitter and receiver casings.
Spare Parts & Hardware Buffer ($100):
Extra bare chips, passives, and connectors to cover soldering mistakes and damaged components.

Implementation issues

High-Frequency RF & Antenna Layout: 2.4 GHz RF signals are hard to work with. If the PCB trace connecting the chip to the antenna isn't designed correctly, or if the ground plane under the antenna is improperly laid out, wireless range will drop significantly and cause audio dropouts.
Mitigation: Avoid designing a custom antenna circuit from scratch. Instead, use a pre-tuned surface-mount ceramic chip antenna and copy the exact PCB copper layout.

Mechanical & Spatial Layout: Fitting a PCB, battery, antenna, and audio jack into a small receiver requires careful spacing. Antennas can be detuned if they're too close to metal jacks or battery casings.
Mitigation: Use 3D-printed enclosures with explicit spacing between the antenna, battery, and audio jack.

Energy Budget: Continuously receiving and processing a wireless audio stream drains batteries quickly.
Mitigation: The receivers can use a 600mAh LiPo battery paired with sleep modes when not being used.

Evaluation and testing

Latency & Lip-Sync Test
Feed an audio pulse into the transmitter and measure the time delay to the audio output at the receiver using an oscilloscope.
Success Metric: Achieve an end-to-end audio latency of under 20 ms to ensure video lip-sync is imperceptible to users when watching movies or gaming.

Multi-Receiver Synchronization Jitter Test
Connect two separate receiver pods to an oscilloscope and compare the time alignment of their audio output waveforms while playing the same audio track.
Success Metric: Phase offset between multiple receivers must remain under 1 ms. This eliminates echo effects when receivers are placed in the same room.

Range & Interference Stress Test
Stream audio through the system in a crowded environment with heavy 2.4 GHz Wi-Fi traffic while moving the receivers behind walls and 10 meters away.
Success Metric: Maintain continuous audio playback with zero audible dropouts or loss of sync at a range of 10 meters line-of-sight and through 1 standard interior wall.

Real-World Usability & Battery Benchmark
Time the auto-connect sequence from power-on and run continuous audio playback at 75% volume until the receiver battery dies.
Success Metric: Receivers auto-pair and begin playing audio within 5 seconds of powering on (without needing a phone app), and run continuously for at least 6 hours on a single charge.

High-level description

This product is a desktop coaster that heats or cools your drink to keep it at their ideal drinking temperature for hours. Instead of letting coffee go cold or energy drinks turn room-temperature during long study sessions, this product can actively hold hot beverages at 135°F or chill cold drinks down to 40°F. The station intelligently detects when a cup is placed on its surface, measures the drink's temperature without touching the liquid, and automatically safely powers off the moment the cup is lifted or emptied.

The finished device is a low-profile desk coaster featuring a smooth aluminum top plate, a glowing status ring that changes color based on temperature, an integrated display showing real-time temperature metrics, and a tactile dial for quick manual adjustments. Users can also sync their ideal temperature preferences wirelessly through a desktop widget.

Designed for students and remote workers who spend long hours at their desks, this project addresses a daily frustration: wasted, ruined drinks. It is worth building because it turns an ordinary item into a daily tool and eliminates constant trips to the microwave or fridge to keep you focused on your work.

Development Schedule

- 9/10 – 9/23 (Thermal Planning & Hardware Specs):
    - Calculate thermal power needs for heating/cooling elements and select bare controller chips, thermal sensors, and weight scale components.
- 9/24 – 10/6 (Part Selection & Breadboard Bring-Up):
    - Order parts, and wire basic power and sensor lines on breadboards to verify raw temperature and weight readings.
- 10/7 – 10/20 (Breadboard Controls & Rev 1 PCB Order):
    - Build a breadboard prototype with thermal controls, sensor reading, and a screen display.
- 10/21 – 11/5:
    - Design PCB (seperate high-current power and sensor ground planes), order by 10/25
    - Develop thermal control logic and weight safety routines on breadboards (while waiting for PCBs).
- 11/6 – 11/18
    - Correct Rev 1 electrical trace issues (with raw wires if possible)
    - Refine multitasking software and make aluminum contact top plate.
- 11/19 – 11/25
    - Calibrate safety limits and fine-tune temperature adjustments for heating vs. cooling.
- 11/26 – 12/9
    - Run continuous multi-hour temperature holding tests with real beverages.

Budget

Estimated Total Project Spend: ~$450

Bare MCU & Power Switches: $45
Sensors: $50
Prototyping & Adapters: $40
Custom PCB Fabrication: $80
Power Supply & Thermal Hardware: $110
Displays & User Controls: $25
Spare Parts & Hardware Buffer: $100

Implementation Issues:

Mechanical & Thermal Management: Peltier modules generate significant waste heat on their exhaust side during cooling modes, requiring thermal dissipation without loud noises. Also, the weight scale sensor sits underneath the heating assembly without thermal expansion skewing weight readings.
Mitigation: Mount the top aluminum contact plate on a floating mechanical support that isolates heat. Dissipate waste heat using a low-profile heatsink and a quiet PWM-controlled fan.
Special Facilities: 3D printers for the outer shell and a simple metal saw/mill

Safety Considerations: Operating a heating surface up to 135°F presents mild burn risks or fire risk if the MCU crashes while the power switch is open.
Mitigation: Install a thermal fuse attached to the heating plate that cuts power if temperatures exceed 150°F. Also add software watchdogs that disable heat when the load cell senses the cup has been lifted.

Evaluation and Testing:

Thermal Holding Precision Test
Place 10 oz of liquid in a standard mug (for heating) and an aluminum can (for cooling), monitoring liquid surface temperature over a 1-hour period
Success Metric: Hold hot beverages at 135°F ± 5°F and cold beverages at 40°F ± 5°F in a 70°F room.

Safety Test
Abruptly lift a mug off the coaster while the heating plate is operating at 100% power, and simulate software lockups to test the hardware backup thermal fuse.
Success Metric: The load-cell watchdog cuts power to the heating element in under 200 ms upon cup removal, and the plate surface never exceeds 150°F.

3 Phase Inverter

As a member of SuperMileage team competing in the Shell Eco-Marathon, the goal of our car is energy efficiency rather than pure speed. Switching to a fully battery powered electric drivetrain provides the foundation for ultimate efficiency. However, standard commercial electronic speed controllers are designed for peak power output, which wastes energy during low-power cruising. To solve this problem, I want to design and build a custom, high-efficiency 3-phase motor inverter and integrated data acquisition system.

Details

The custom inverter will operate off a 48V DC battery, and convert power into low-loss 3 phase AC using Field-Oriented Control so that a BLDC motor can turn. In order to ensure smooth, high torque startup , the controller incorporates closed-loop trapezoidal control using feedback from a motor encoder. To maximize overall system efficiency, the board will feature a series of buck converters (48V -> 12V -> 5V -> 3.3V). Real-time phase current sensing (hall or shunt) provides tight torque control, while temperature sensors will be implemented to monitor the MOSFET switching bridge and the motor.

Directly integrated into the project is a comprehensive data acquisition (DAQ) system communicating via a CAN transceiver network. The DAQ node will feature data logging, speed and position tracking using a GPS/IMU module, read wheel encoders, and manage auxiliary hardware like a windshield wiper. It will also output telemetry data to a LCD display for the driver.

Schedule

9/7 - 9/23	**Planning** ● Establish baseline requirements ● Part selection ○ MOSFETs ○ Gate drivers ○ Current sensors ○ DAQ Modules ● Draft initial diagrams/schematics ● Complete draaft proposal by 9/18
9/24-10/6	**PCB Design** ● Finalize part selection ● Proposal meeting ● Complete detailed schematic ● Design PCB and order by 10/2 ● Order components, especially ones with long lead times

10/7-10/25	**Firmware Development and Hardware Bring Up** ● During PCB turnaround time work on core software ○ SVPWM generation ○ ADC DMA drivers ○ CAN communication stack ○ LCD Drivers ● Once PCB is delivered work on soldering components ● Milestone 1 report by 10/15 ● Milestone meeting
10/26 - 11/10	**Low Voltage Validation and Debugging** ● Power the board at low voltage to verify gate switching ● Testing ○ Current sensing ○ Thermistor inputs ○ Throttle response ○ GPS logging ○ LCD output ● MIlestone 2 report by 11/10
11/11 - 11/25	**Full Power Testing and Debugging** ● Test the inverter at full 48V DC supply on dyno ● Verify handoff between trapezoidal startup and FOC ● Debug any issues with board or firmware ● Milestone 2 meeting
11/26 - 12/11	**Final System Validation and Expo Prep** ● Perform comprehensive bench testing of the combined inverter and DAQ package ● Debug any remaining last minute issues ● Prepare poster for expo, 12/10 ● Target product completion for 12/4 ● Finish final report by 12/11

Budget

This is a rough estimate of pricing for this project. Parts have not been fully selected but by estimating the price of components we most likely need, this project is well under the budget of $250 a person. There are also parts that will likely be sourced from the supermileage inventory, but are included in the estimate because again, parts have not been selected yet. If other unexpected costs arise such as elevated shipping costs, there is plenty of room in our budget to manage.

Item & Part Number	Qty	Est. Unit Cost ($)	Est. Total Cost ($)
STMicroelectronics STM32G431C8T6 MCU	3	5.78	17.34
NUCLEO-G431KB Board (Bench Testing)	2	14.5	29
Infineon BSC014N06NS 60V N-Ch MOSFETs	12	3.29	39.48
TI UCC27211A Half-Bridge Drivers	6	2.29	13.74
TI INA240A1PWR High-Side Sense Amps	6	3.33	19.98
TI LM5164DDAR 100V Step-Down Buck IC (48V->12V)	2	4.19	8.38
TPS54202 (12V->5V) & AMS1117 (5V->3.3V)	1	12.5	12.5
TI TCAN1042HVDR CAN Transceiver IC	4	2.59	10.36
Adafruit 2.8" SPI TFT LCD (ILI9341 + MicroSD)	1	29.95	29.95
NEO-6M GPS & MPU6050 6-DOF IMU Breakouts	2	18.5	37
JLCPCB 4-Layer Board Runs + SMT Stencil (2 Revisions)	2	110	220
Misc. Items	1	55	55
Total			492.73

Issues

- High Switching Noise:
  High frequency PWM switching creates rapid current transitions that can generate a lot of noise which may corrupt current sensing feedback, encoder position signals, and CAN bus data transmission
    - This issue can be fixed by using dedicated ground and power planes on a 4 layer PCB, use differential routing for high-side current sensing
- Real Time Calculation Deadlines:
  Transitioning from closed loop trapezoidal commutation to FOC at speed without current spikes or torque dips
    - Mitigation of this issue can be accomplished by using DMA to trigger the ADC current measurements and use the STM32G4's floating point unit to ensure measurements are done on time
- Overheating:
  If heat dissipation is inefficient, heavy usage of inverter can lead to overheating or even fire
    - Thermistors on the board can be used to continuously monitor for high temperatures, and software can trigger a system shutdown at particularly high temperatures

****Part of SuperMileage project team + inverter group****

Self Driving Golf Ball

This golf ball has a mind of its own and will guarantee you never miss a putt again. The ball will use a pendulum drive to change it's direction towards an IR beacon in the cup. All of the components will be encased in a waterproof 3D printed shell.

Implementation

- An IR beacon will be used so the ball can track the cup
- IR photodiodes will detect the beacon, along with filtering out the outdoor sunlight
- A high torque pendulum will allow for the ball to steer
- An STM32 microcontroller will sample an imu along with the beacon in order to calculate the steering angle for the ball
- A two piece SLA polymer 3D printed shell will encase the components along with a battery, and sealed with an O-ring to ensure it is waterproof

Schedule

9/7 - 9/23	**Planning** ● Component selection ● Block diagrams ● Initial sketches ● Work on firmware to calculate steering angle ● Complete draft proposal by 9/18
9/24 - 10/6	**PCB Layout and Enclosure Design** ● Design PCB and order by 10/2 ● Order components ● During PCB downtime design ball casing ● Also complete firmware ● Proposal meeting ● Final Proposal by 10/1
10/7-10/25	**Mechanical Integration** ● Assemble golfball ● Test on flat surfaces ● Debug as needed ● Milestone 1 report 10/15 ● Milestone 1 meeting
10/26 - 11/15	**Testing** ● Thorough testing on varying slopes/real greens ● Debugging ● Test IR beacon in outside light

	• Milestone 2 report • Milestone 2 meeting
11/16 - 12/11	**Final Validation and Expo** • Complete testing on varying slopes • Work on project poster • Finish final report

Budget

This is an estimate of the budget, parts have not been fully selected but the cost is well under the budget for the project, making it feasible.

Item	QTY	Est. Unit Cost ($)	Est. Total Cost ($)
Microcontroller & Sensors	1	35	35
Drive Motors & Drivers	2	24	48
IR Tracking Optics	1	25	25
Power & Qi Charging	1	25	25
Custom Stacked PCBs	2	30	60
3D Printed Shell & Weight	1	50	50
Target Cup Beacon	1	25	25
Misc Parts & Shipping	1	30	30
Total			298

Issues

- Solar Saturation:
  Direct sunlight may blind optical receivers
  - Use a bandpass filter to only detect beacon signal
- Turf Friction:
  Not all surfaces will have the same friction
  - Use a high density material to maximize torque to handle any level of green friction

Configurable embedded peripheral emulator

Mingxuan He

In embedded systems development, firmware typically cannot function independently of external hardware. The main controller often needs to interact with peripheral devices such as temperature sensors, IMUs, encoders, ADCs, GPS modules, and communication modules—yet these peripherals themselves frequently become a limiting factor in the development process.

First, early in product development, peripherals may not yet have arrived, may still be under design, or may only be available from other teams, which causes software development for the main controller to be blocked by hardware progress. Second, even once real peripherals are available, many test conditions are difficult to reproduce reliably and repeatably—for example, extreme temperatures, abnormal rotational speeds, transient faults, communication timeouts, corrupted data, or sensor freezes. If developers rely entirely on real hardware, it becomes very difficult to systematically verify whether firmware responds correctly to these edge cases and fault conditions.

Moreover, real peripherals can generally only provide the data actually produced by their physical environment, whereas what testing truly requires is often controllable, repeatable, programmable input. For instance, to verify a controller's protection logic as temperature slowly rises from 20°C to 120°C, rather than actually heating a sensor, it is more practical to have a simulator return corresponding register values according to a predefined time curve. Similarly, to test I^2C communication error handling, a simulator can deliberately generate NACKs, delayed responses, or anomalous data. This approach is not only more efficient for testing, but also allows the same test scenario to be run repeatedly, enabling direct comparison of results across different firmware versions.

Therefore, to tackle the problems, we propose Configurable Embedded Peripheral Emulator, a hardware testing platform that reproduces the communication behavior and dynamic outputs of real embedded peripherals, enabling firmware development and fault testing without requiring the physical devices themselves.

Details

The proposed system would act as a configurable substitute for common embedded peripherals during firmware development and testing. Rather than requiring a physical sensor, encoder, or communication module to be connected to the device under test, the emulator would reproduce the electrical interface and communication behavior expected by the controller. A user would select or load a device model based on the peripheral’s datasheet, including information such as the communication protocol, device address, register map, data format, and expected read/write behavior. The emulator would then respond to the controller as if the actual peripheral were present.

The initial system would support a limited set of commonly used embedded interfaces, most likely I^2C, UART, and GPIO/PWM. For an I^2C device, for example, the emulator could behave as a slave at a

selected address and return configurable values when the controller reads particular registers. For a UART-based device, it could transmit predefined or dynamically generated messages and respond to commands from the controller. For GPIO-based peripherals, it could generate signals such as encoder pulses, interrupts, or PWM waveforms with user-defined frequency and timing. The user would be able to modify the simulated values in real time or define simple test sequences, such as increasing a temperature value over time or changing an encoder speed.

In addition to reproducing normal device behavior, the emulator would be designed to generate conditions that are difficult to reproduce with real hardware. For example, it could return out-of-range sensor values, delay a response, stop updating a value, generate an I^2C NACK, or produce abnormal GPIO timing. This would allow developers to test whether the firmware correctly detects and handles sensor failures and communication faults. Because the same test conditions could be repeated exactly, the device could also be used to compare different firmware versions under identical inputs.

This proposal envisions building a portable emulator unit containing a microcontroller, configurable digital interfaces, voltage-level translation and protection circuitry, and a simple user interface. The microcontroller would handle timing-sensitive protocol behavior and waveform generation, while the interface would allow the user to select a peripheral model and modify simulated parameters. A small library of predefined device models could be included initially, with support for additional models created manually from device datasheets. The goal would not be to emulate every possible embedded peripheral, but to provide a reusable platform for a defined class of register-based and signal-based devices that can reduce dependence on physical hardware during embedded firmware development and testing.

Budget

Item	Quantity	Cost Per	Cost Total
STM32 microcontroller	1	$15–30	$15–30
Circuit Components (voltage-level translators, protection components, connectors, regulators)	1	$30–50	$30–50
User Interface	1	$20–40	$20–40
PCB fabrication	1	$50–100	$50–100
Shipping	1	$20–40	$20–40
Total			$135–260

Schedule

9/7-9/21	Planning In the first two weeks our team may gather to brainstorm detailed implementations for this project and discuss the system architecture of the emulator. Then come out with a first version of BOM with reasonable price

9/22-10/4	Design and part selection In the next week we may come out with a schematic design of the project, select the best and order the core components
10/5-10/18	Coding and PCB designing This week we may try to use breadboard to make a functional prototype, and send JLCPCB the first version of the designed PCB board. In the mean time we may work on coding for our embedded system
10/19-11/1	More coding and testing on PCB This week we would make sure all components function normally and then transfer the system to our PCB board for testing emulation of basic sensors. We may also update all our designs
11/2-11/22	More testing and additional functions We may finalize the tests for basic functions and start testing on additional functions. For instance, emulating SPI port, Laptop Web UI. Also we may design the exterior of our product to make it look neat and tidy
11/23-12/6	Start writing final report and prepare for the events At this time we would finalize the features of our product and try to maintain its functionality. And we will be working on the reports and other stuff
12/7-12/11	Final report and Design Expo Finish our posters, demos and final products

Issues

The largest concerns at this time are:

1. Reliable timing for I^2C/UART/GPIO emulation
2. Designing a configurable but manageable device-model format
3. Supporting different logic voltage levels safely
4. PCB bring-up and debugging
5. More complex SPI or fault-injection features

Issue 1 is the main technical risk, but using hardware peripherals, timers, interrupts, and DMA should make the required timing achievable. Issue 2 will be addressed by limiting the first version to relatively simple register-based and signal-based peripherals rather than arbitrary devices. Issue 3 should be manageable with level shifters and protection circuitry. Issue 4 may take significant time, so early prototyping on development boards and allowance for a second PCB revision will be important. Issue 5 will be treated as a stretch goal and can be dropped if necessary.

Additional Stretch goals

Integrating an autonomous datasheet reading functionality into our emulator with the Laptop Web UI to emulate the certain device.

Wearable Visual Awareness Glasses

Mingxuan He

Many people with severe visual impairment rely on canes or guide dogs to detect nearby obstacles while walking. These methods are effective for detecting objects near the ground, but provide less information about objects at upper-body height or about the surrounding environment. Existing computer-vision systems can recognize objects, but they are often implemented on phones or computers rather than as a low-latency wearable embedded system.

We propose Wearable Visual Awareness Glasses, a portable embedded device that detects nearby obstacles and selected objects and communicates this information to the user through directional vibration and audio feedback.

The goal is not to replace a cane or independently navigate the user across streets. Instead, the device would provide an additional layer of environmental awareness while remaining wearable and responsive.

Details

The glasses would contain a forward-facing camera and distance sensors. Images would be processed locally using a Linux-based embedded computer running a lightweight pretrained vision model. Rather than attempting general scene understanding, the initial system would recognize a limited set of useful object classes such as people, bicycles, vehicles, doors, and stairs.

A microcontroller running an RTOS would handle time-sensitive functions including distance-sensor acquisition, user input, haptic feedback, power monitoring, and communication with the vision processor. A custom PCB would integrate the microcontroller, sensor interfaces, haptic drivers, voltage regulation, protection circuitry, and connectors.

When an obstacle is detected, the system would estimate whether it is to the left, center, or right of the user. Vibration motors located near the two sides of the glasses would provide immediate directional feedback. Audio would be reserved for less urgent semantic information such as "person ahead" or "door detected."

The feedback system would prioritize safety-related events. For example, a nearby obstacle warning would interrupt a lower-priority object announcement. The target is to produce obstacle feedback within approximately 250 ms while keeping the head-mounted portion lightweight enough for normal use.

To reduce weight and heat, the camera and sensors may remain on the glasses while the main processor and battery are placed in a small pocket-mounted unit connected by cable.

Budget

Item	Quantity	Cost Per	Cost Total
Embedded Linux processor	1	$30-70	$30-70
Camera	1	$20–30	$20–30
MCU, ToF sensor, and IMU	1	$30–50	$30–50

Haptic/audio components	1	$15–30	$15–30
Battery and power circuitry	1	$20–30	$20–30
PCB fabrication	1	$40–70	$40–70
Mechanical parts and shipping	1	$30–50	$30–50
Total			$185-330

Schedule

9/7-9/21	### Planning and architecture Define system requirements, compare processor platforms, and prototype the vision pipeline.
9/22-10/4	### Sensor and hardware design Select camera, depth sensor, MCU, power system, and design the first PCB.
10/5-10/25	### Software and PCB development Implement vision processing, RTOS sensor tasks, communication, and haptic control.
10/26-11/15	### System integration Assemble the wearable prototype and integrate vision, distance sensing, and feedback.
11/16-12/6	### Testing and refinement Measure detection latency, battery life, thermal behavior, and feedback reliability under representative walking scenarios.
12/7-12/11	### Final report and Design Expo Finish our posters, demos and final products

Issues

The largest concerns are:

1. Achieving sufficiently low vision-processing latency
2. Combining vision and distance sensing reliably
3. Keeping the glasses lightweight and comfortable
4. Managing battery life and processor heat
5. Avoiding excessive or confusing feedback

The first version will therefore limit the number of recognized objects and prioritize reliable obstacle detection over complex scene understanding. Processor selection will be based on early latency and power benchmarks. If a fully head-mounted processor is too heavy or warm, computation and the battery will be moved to a pocket-mounted module.

Stretch Goals

Possible additional features include OCR for signs and room numbers, traffic-light recognition, voice commands, GPS integration, and configurable object classes.