Principles of Game Design and Basic Game Architecture

John Laird and Sugih Jamin

University of Michigan

Last changed 9/8/06

[drawn in part from Doug Church]

What is a Game?

- How are games different from toys?
  - Games have rules and goals to be achieved.
- Some “computer games” are really toys
  - The Sims, Simcity, Nintendogs, …
- Games can have story, but …
  - Interactive, demands participation

Computer Games

- Completely new form of entertainment
- Completely new worlds to play in
- Allow players to take on a new persona

Game Technology: Computational Worlds

- Representing physical objects – real or imaginary
  - Terrain: mountains, hills, plains, desert, river, coastal area, ocean, …
  - Buildings (exterior and interior – walls, floors, staircases, townhalls, mills, ...)
  - Game objects (furniture, fluids, weapons, vehicles, potions, bullets, ….)
  - Animate objects (units, players, opponents, animals, trees, …)
- Bringing the simulated world to life:
  - Physics: collision detection, lighting and shadow, explosion
  - Behavior and decisions: game AI
- Supporting interaction
  - Graphics
  - Audio: ambient sfx, music, and speech
  - Input devices: speech? haptic devices?
  - Networking

Engineering: Traditional vs. Game

Traditional engineering:

- High precision
- Total realism

Game engineering:

- High speed
- Low memory
- No spiking in resources
- Scalability – O(n) expected
- Believability
- Control
- Low cost development

True for graphics, physics, AI, audio, etc.

Cheat if can't get caught.

Game “Engineering”

- Design around the computer’s limitations
  - Character wears dungarees so easier to see arms move
  - Wears a hat because don’t have to have hair
  - Has mustache because couldn’t draw nose and mouth
- Laser beam takes time to travel
- It takes a spray of bullets to take down hostile
- Hostile always makes noise before firing
- Elevator doesn’t come instantly

Game Design

- Create an experience for the player
  - The player should have fun--not the designer, programmer, or computer
- Many different kinds of experiences
  - Some games are based on a story (dramatic arc, etc).
  - Put the player in his/her dreams, where he/she is the hero (Sid Meier)
  - Some are puzzles without stories or personification
- The player is in some environment
  - Abstract or representational
- The environment has rules of interaction
- The rules and some criteria for success define a game
- All managed by the computer

What Makes for Good Gameplay?

1. Pursuing and achieving goals (challenges)
2. Interactivity
3. Feedback about position relative to goals
4. Interesting choices required to achieve goals
5. Consistency and fairness
6. Avoid repetition

- Play testing to ensure that you get it right
- This is what is behind “Fun” (which ≠ Reality)

Pursuing and Achieving Goals

- Always something to achieve
- Always achieving something
- Not too easy or too hard
- Often at least three levels of goals with rewards
  - Long-term goal [complete game]
    - "I can conquer the world."
  - Medium-term goal [10-30 minutes]
    - "I can take over a city."
  - Short-term goal [seconds to minute]
    - "I can win a battle."
- What are levels of goals from different game genres?

Common Goals in Computer Games

- Eliminate other players
  - Action games
- Score points
  - Sports games
- Get somewhere first
  - Racing games
- Solve puzzles
  - Adventure games
- Gain territory
  - Strategy games
- Improve abilities
  - Role-playing games
- Develop social relationships
  - Massively multiplayer games
- Play god
  - Simulations

Game Interactivity

- Player's decisions determine success/outcome
  - At least the player thinks their actions do
  - Avoid decisions that aren't related to success of some goal
- Player is not just a passive observer
  - Can be fun, but is different kind of entertainment
  - *Always ask: What is the player going to do?*
- Minimize the player's confusion
  - What to do should be clear without consulting a manual
- User interface should not get in the way of interactivity
  - Simple, consistent

Feedback

- Feedback at all levels so players
  - Know where they are
  - Know where they are relative to other players
    - high score list
  - Know where they are relative to goal
    - Dead and don't even know it
  - Know what they need to achieve
  - Know what is important in the world
  - Can use knowledge to make choices
- Game should not be about
  - how to get information out of the interface
  - how to randomly explore the world (unless that is "fun")
  - how to recall what just happened

Decisions, decisions, decisions

- Great game-play is a stream of interesting decisions that the player must resolve
- Must develop & execute strategies/tactics to achieve goals
  - Applies across all levels of goals
- Interesting decisions to achieve goals
  - Different strategies/tactics
  - Tradeoffs between strategies/tactics: game balance
  - Avoid tedious goals/decisions: micromanagement
    - Let computer do those
    - Ask – if you take out the tedious activity, what decisions are left?
- The inverted pyramid of decision making
  - Few, simple decisions at first, and then more and more
    - Begin your game with a great first few minutes
    - Start with a simple concept: “running, climbing, jumping”
- Decisions have impact on how game turns out

Gameplay Consistency and Fairness

- There is consistency in the actions and associated outcomes for trying to achieve goals
  - Must be a *reason* for failure (or success)
  - Not arbitrary: Players know what to expect and can plan ahead
    - A pinball game uses “pinball” physics *all* the time
    - No “dead man” walking
  - Don’t solve problems by unique, unlikely actions
  - Don’t break suspension of disbelief
    - Kill self but dead body falls over wall, hits lever to open door
- Fairness
  - Player thinks they have a fair chance – game balance
  - Incorporate a smooth learning curve
  - Accommodate all skill levels
- Can still be plot twists, but must be explainable

Avoid Repetition

- Allow user to skip parts already seen
  - Skip cut scenes
  - Good save system
  - This includes audio: allow user to turn off the music
- User must do the same thing over and over again
  - Endless waiting to heal
  - Lining up at spawn points waiting for an item to respawn

Gameplay/Game Design Sins

- Poor production
  - Break the suspension of reality
  - Bad writing, bad voice acting, long load times, saving, …
- Linear plot/gameplay
  - Player's actions don't affect how the plot progresses
- Micromanagement
  - Player is forced to perform meaningless tasks
  - AI should take care of all the obvious choices
- Repetition
  - Player must do same action over and over again
  - Player must sit through same cut scenes every time they play
  - Have to replay 90% of level to fight boss
- Doesn't track user's learning curve
  - Should start easy and get harder as game progresses
- Poor game balance
  - Same strategy always works
  - Trial and error is not fun gameplay

Gameplay/Game Design Sins

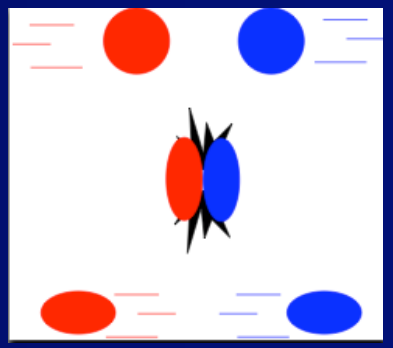
- Not enough variety
  - Same graphics, objects, monsters, level design, sounds...
- Awkward user interface
  - Must do lots of mousing to do simple task
- Limited feedback
  - Player is confused about goals
  - Player is confused about current progress to goals: no map
- Inconsistency in story
  - There are not compelling and consistent goals for the player
- Dead and you don’t even know it

Game Engine = World Simulation

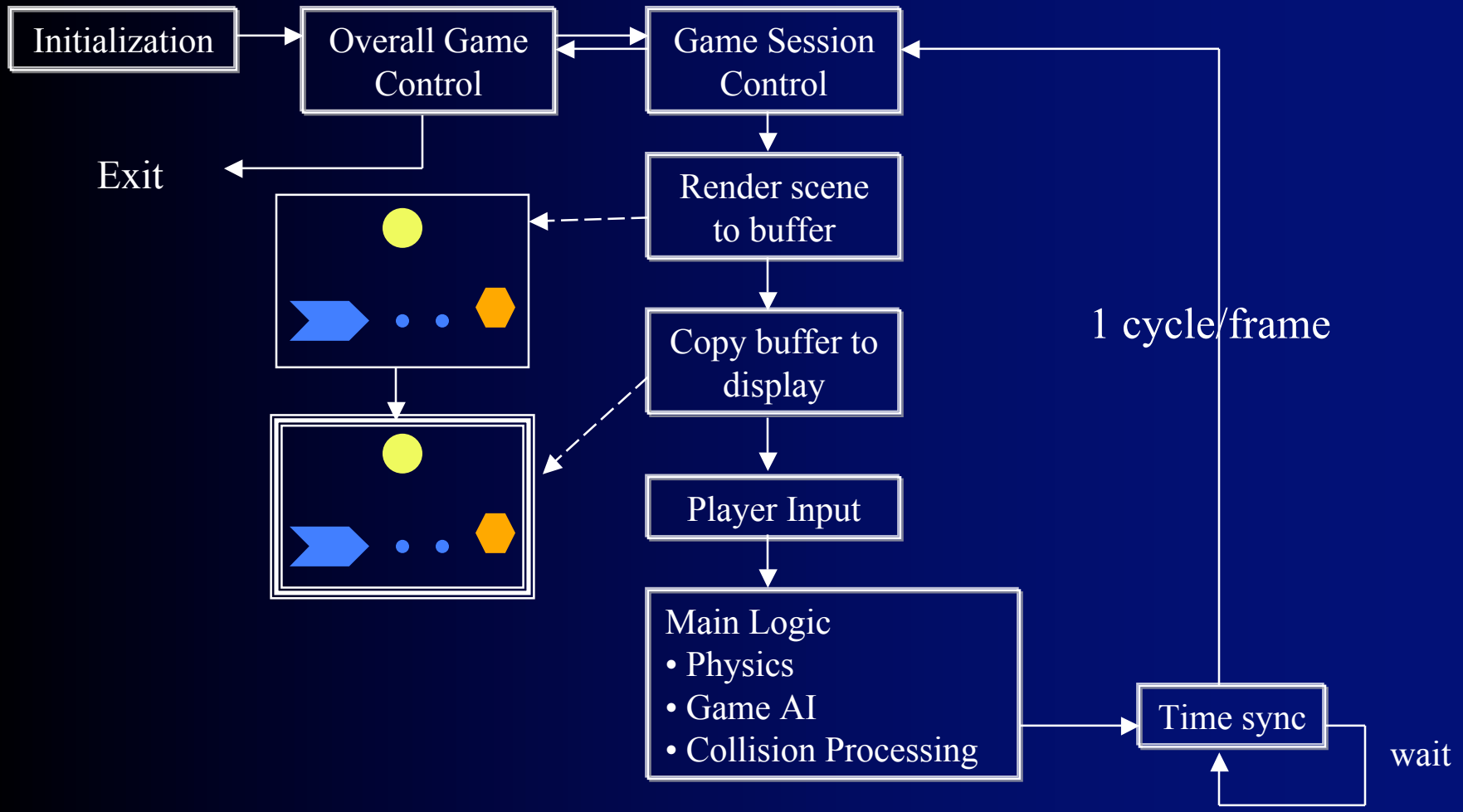
Discrete simulation: each cycle computes a ``snapshot'' of the world

Difference between a technical simulation and a game:

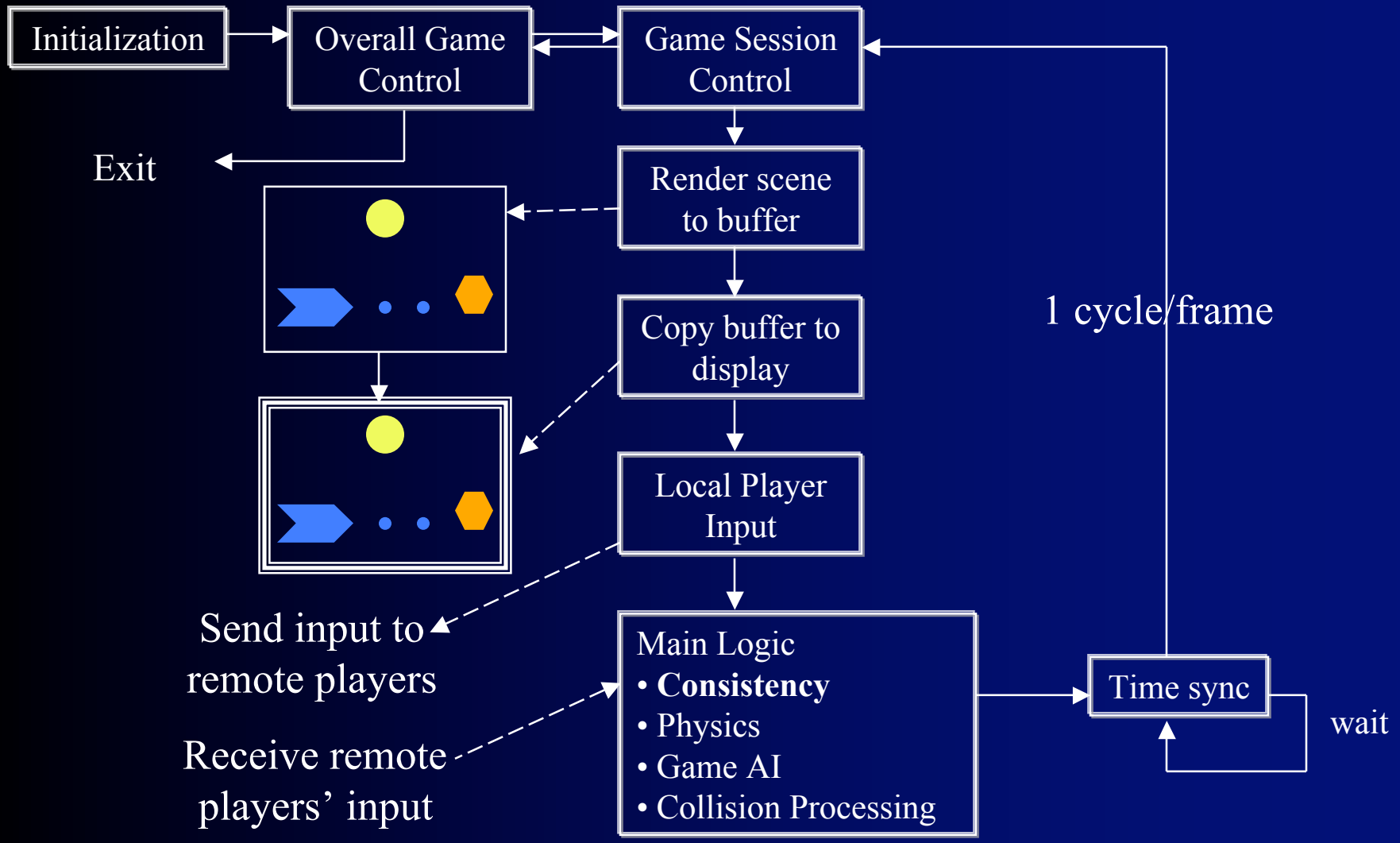
- Continual behavior
  - Not just run a program and get an answer
- Real-time and highly interactive
  - Update at around 30 times/second
  - Rendered time always lags behind real time/game time by at least 1 frame
  - Necessary to avoid clunky action or missed player input
  - Consistency is important (to both)
- Simple physics: velocity, elastic collisions
  - No mass, accelerations, momentum
  - Easier in fixed time simulation than variable time simulation



Game Loop

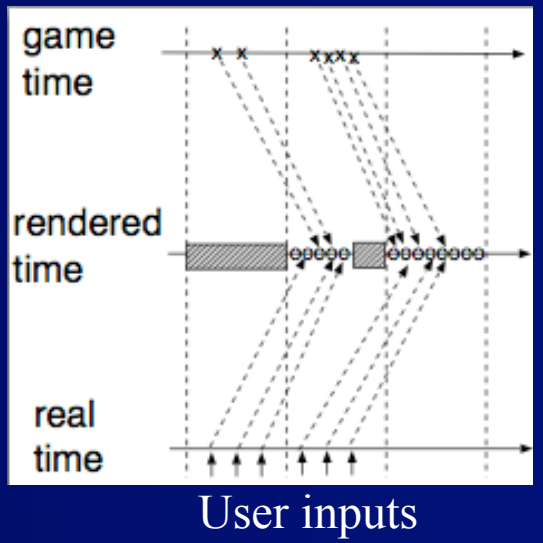

Initialization
Overall Game Control
Game Session Control
Exit
Render scene to buffer
Copy buffer to display
Player Input
Main Logic
• Physics
• Game AI
• Collision Processing
Time sync
wait
1 cycle/frame

Multi-player Game Loop



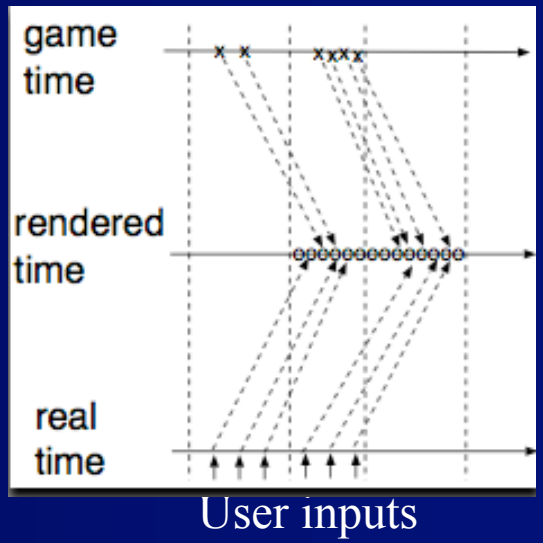
Simulation Types

- Fixed Discrete (tied to real time)
  - Update world model each time step
  - Each time step is same size
  - Detect interactions by examination
  - Wait if done too early
- Pros:
  - Simpler physics
  - Smoother graphics
- Cons:
  - Computation must fit into time step
  - Time step limits max frame rate



Simulation Types

- Variable Discrete
  - Time steps are variable - dictated by amount of computation, but fast as can be
- Pros:
  - No busy wait
  - Smoother interaction at higher frame rate
- Cons
  - Requires a bit more work on physics calculation
  - Jerky motion if frame rate too low



Simulation Types

- Event-based
  - Skip ahead to next predicted event (collision)
  - Game time stretches and contracts as necessary
  - Computed analytically
  - Not suitable for real-time interactive game
  - But can be used with some “simulation” game (fast forward to when the ship reaches port or meets pirates)