Toward an Objective Measure of Developers' Cognitive Activities

ZOHREH SHARAFI, YU HUANG, KEVIN LEACH, and WESTLEY WEIMER,
University of Michigan

Understanding how developers carry out different computer science activities with objective measures can help to improve productivity and guide the use and development of supporting tools in software engineering. In this article, we present two controlled experiments involving 112 students to explore multiple computing activities (code comprehension, code review, and data structure manipulations) using three different objective measures including neuroimaging (functional near-infrared spectroscopy (fNIRS) and functional magnetic resonance imaging (fMRI)) and eye tracking. By examining code review and prose review using fMRI, we find that the neural representations of programming languages vs. natural languages are distinct. We can classify which task a participant is undertaking based solely on brain activity, and those task distinctions are modulated by expertise. We leverage insights from the psychological notion of spatial ability to decode the neural representations of several fundamental data structures and their manipulations using fMRI, fNIRS, and eye tracking. We examine list, array, tree, and mental rotation tasks and find that data structure and spatial operations use the same focal regions of the brain but to different degrees: they are related but distinct neural tasks. We demonstrate best practices and describe the implication and tradeoffs between fMRI, fNIRS, eye tracking, and self-reporting for software engineering research.

CCS Concepts: • **Software and its engineering** → **Maintaining software**;

Additional Key Words and Phrases: Neuroimaging, eye tracking, code comprehension, prose review, data structures

ACM Reference format:
Zohreh Sharafi, Yu Huang, Kevin Leach, and Westley Weimer. 2021. Toward an Objective Measure of Developers' Cognitive Activities. *ACM Trans. Softw. Eng. Methodol.* 30, 3, Article 30 (March 2021), 40 pages.
https://doi.org/10.1145/3434643

1 INTRODUCTION

Understanding how humans carry out computer science activities improve software engineering productivity. There is significant interest in such increases, with some Fortune 500 companies retraining up to half of their workforces in programming-intensive areas [23, 31]. Despite this growing demand, research often rests on methods such as self-reporting (e.g., think-aloud protocols, questionnaires, surveys, and interviews) to study software engineering tasks [6, 19, 115]. Although these methods contribute important evidence and advance the state of the art, they suffer

Authors' addresses: Z. Sharafi, Y. Huang, K. Leach, and W. Weimer, University of Michigan, 2260 Hayward Street, Ann Arbor, Michigan, 48109-2121; emails: {zohrehsh, yhhy, kjleach, weimerw}@umich.edu.

1049-331X/2021/03-ART30 $15.00
https://doi.org/10.1145/3434643

from the Hawthorn (observer) effect [45, 49] and may not be reliable [36, 54, 71, 126]. To complement and enhance data collected using such traditional methods, we favor biological objective measures to provide insights into the cognitive processes that underlie various software engineering activities. While findings related to cognition have guided behavioral and developmental improvement in domains like mathematics [33] and education [119], we still lack a foundational understanding of the neural correlates of fundamental computer science activities. From writing programs to code review to manipulating data structures, we lack an accepted theory explaining, at a level of abstraction useful for practice or pedagogy, what goes on in the brains of people conducting the many activities collectively called "programming." Though researchers have highlighted the need and importance for such objective measures (e.g., [47]), there is no systematic solution for this challenge yet.

In this article, we give a combined presentation of two previous controlled human studies [51, 71], involving 112 participants in total and three different evidence modalities, to provide objective measures of cognitive load in computing activities. We extend our previous work by adding eye-tracking data acquisition and analyses, as well as an investigation of problem-solving strategies and cognitive load measured by eye-movement data (visual effort). Investigating visual attention trends coupled with two methods of measuring brain activity provides novel insights into developers' cognitive processes when completing a series of programming tasks. We also present comparative recommendations among eye tracking, neuroimaging, and the implications for reproducible software engineering research.

Measurements. We use two neuroimaging techniques, *functional magnetic resonance imaging* (fMRI) and *functional near-infrared spectroscopy* (fNIRS), as well as eye tracking, to measure developers' cognitive processes.

We use fMRI and fNIRS because both are non-invasive in vivo neuroimaging techniques that have enabled new and complex studies of brain function [56]. In contrast with conventional research that focuses on one specific cognitive subprocesses (e.g., attention or working memory), neuroimaging provides a comprehensive view of the activated brain regions involved. By indirectly measuring changes in oxygen consumption [81], these two modalities can both be used to isolate the brain regions recruited for specific tasks and provide a more objective evaluation of their associated cognitive load. While fMRI and fNIRS allow us to investigate the physical substrates underlying software engineering tasks, they do not provide any substantial data about participants' ways of interacting with visual information. *Eye-tracking* cameras can record significant and substantial evidence about participants' visual focus, attention, and interactions, including reading patterns and visual cues during search [124, 138]. The software engineering research community has used eye trackers to study various tasks, including the comprehension of source code and software artifact representations (e.g., UML class diagrams), source code reading, debugging and bug fixing, code review and summarization, and software traceability [106, 138]. However, the use of neuroimaging in software engineering is still exploratory; since 2014, only about a dozen publications have studied associated cognitive processes with either fMRI or fNIRS alone [24, 41, 48, 51, 74, 104, 116, 142, 143]. Moreover, only a few use eye tracking with either fNIRS [48] or fMRI [116] to study software engineering due to the high cost and effort of conducting neuroimaging studies and the lack of proper tool support for simultaneous data capturing and analysis. Our work is the first to use both fMRI and fNIRS with eye tracking to study software engineering.

Activities Studied. We focus on code comprehension, code review, and data structure manipulation as fundamental activities in software engineering. Studies have shown that developers spend more time understanding code than any other activity [34, 62, 123, 127]. A NASA survey ranked understanding as more important than functional correctness when making use of software [105]. Similarly, with companies such as Facebook [159] and Google [79] mandating code

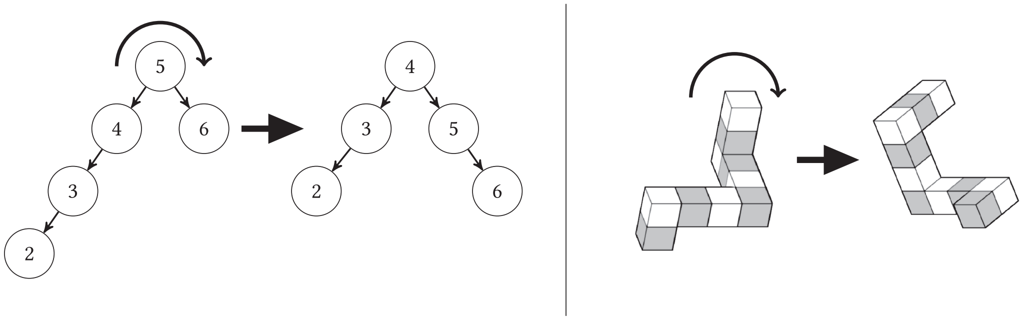


Fig. 1. A representation of the investigated relationship between data structures and spatial ability. On the left, an unbalanced binary tree is rotated about node 1 to produce the tree on the bottom left. On the right, a three-dimensional object is rotated in space as shown in the bottom right. We investigate how the brain represents these two activities using neuroimaging and eye-tracking techniques.

review for new check-ins, code review has taken on practical and research prominence [172]. In both code comprehension and review, data structures are a fundamental element that affect performance and cost [7, 50, 156, 166]. Data structure choice and usage influence many aspects of software, including but not limited to maintainability [111], fault tolerance [12], reliability [150], and scalability [129].

We compare these software engineering tasks to the everyday activities of reading prose and moving objects in space. These baselines serve as contrasts in our experiments, ground our findings, and admit the potential to inform programming training (see Section 2). *Prose* reading involves the comprehension of natural language and has long been a preferred point of comparison for code reading research [21, 142]. *Spatial ability* encompasses the determination of spatial relationships between objects and the mental manipulation of spatially presented information, and is a well-studied subject in psychology. Psychology research has shown spatial ability, often measured via *mental rotation* tasks like the one illustrated in Figure 1 [28, 38, 93, 140], to be a major factor in proficiencies such as mathematics [66, 164], natural sciences [157, 169], engineering [8], meteorology [17], and map navigation [102]. We design our experiments to be based on contrasts between programming tasks and two well-studied activities (spatial ability and prose reading) to ground our results.

Experiments. We present two sequential studies that make different use of various modalities (e.g., considering their relative costs and explanatory power [48, 51, 142]). We use highly controlled stimuli and small tasks to support the artificial and isolated environment of fMRI. All stimuli are static images and are presented for a short fixed time (30 or 60 seconds). Participants could not interact with the computer, navigate or scroll through code and other artifacts, or execute test-cases. See Krueger et al. for a discussion of the difficulties of typed participant input in fMRI studies [85].

In our first study, we use fMRI alone to understand the relationship between code review, natural language prose review and code comprehension, as well as the role of programming expertise (we refer to this study as the *Code Study*). Thirty-six students are presented with three types of stimuli, each with an associated judgment task. In a code comprehension task, participants are shown a snippet of code and asked an associated software maintenance question [144]. In a code review task, participants are shown an image of a GitHub pull request (i.e., code, a patch to that code, and a comment) and asked whether they would accept it or not. In a prose review task, participants are shown English writing with simple editing markup and asked whether to accept the changes or not. This study admits the investigation of the neural similarities between prose and code comprehension and review.

Subsequently, we present the results of a second study using both fMRI and fNIRS as well as physical eye tracking (captured simultaneously during the fMRI session) to investigate the neural representations of several classes of data structures and their manipulation (see Figure 4; we refer to this study as the *Data Structure Study*). Seventy-six students mentally manipulated lists, arrays, and trees. Participants also completed a spatial manipulation task from psychology (i.e., to determine if two perspective drawings portray the same three-dimensional shapes). This study admits the investigation of the neural similarities between reasoning about real-world objects in space and reasoning about abstract data structures.

We leverage two key insights to conduct these studies. First, we use different modalities (fMRI, fNIRS, and eye tracking) to provide objective measurements and establish a grounded understanding of mental processes associated with a series of programming tasks. By comparing and combining these modalities and discussing the implications, we inform best practices for cognitive research of software engineering. Second, we design our experiments to measure contrasts between programming tasks and activities from other domains that are already well-studied (i.e., spatial ability, prose reading) to ground our results. Through such experimental controls, we isolate one feature between tasks and observe the corresponding differences in brain activation or visual attention trend.

Summary. Empirically, we find that the neural representations of programming and natural languages are distinct. Our classifiers can distinguish between these tasks based solely on brain activity. We find that the same set of brain locations is relevant to distinguishing all three tasks. Finally, we find that expertise matters: greater skill accompanies a less-differentiated neural representation. Our results demonstrate that data structure and spatial ability operations are related: both fMRI and fNIRS evidence demonstrates that they involve activation to the same brain regions (e.g., Section 5.3). However, the similarity relationship is nuanced: spatial ability operations and tree operations admit a significant contrast and are characterized by differentiated activation magnitudes (e.g., Section 5.3). Further, some regions relevant to data structures are not accessible to fNIRS: fNIRS lacked the penetrating power to uncover the full evidence reported by fMRI (Section 6.1) and was unable to distinguish between two distinct tasks. We also found that difficulty matters for data structure tasks: more complicated stimuli result in greater neural activation and different visual attention trends: an increase in cognitive load (Section 5.4). Our eye-tracking data analysis shows that participants use a more active scanning pattern and more exploration for List and Tree stimuli than mental rotation ones. While a neural relationship between spatial ability and data structure manipulation may seem clear in retrospect, it was not obvious to our participants, 70% of whom reported no subjective experience of similarity (Section 6.2).

Both the code and prose comprehension experiment [51] and the data structure manipulation experiment [71] were previously published separately. The unified two-stage data analysis presentation (Section 4), eye-tracking data acquisition and analyses (Sections 3.2 and 4.2), the results of cognitive load (visual effort) measured by eye-movement data (Section 5.4), problem-solving strategy results (Section 5.5), comparative recommendations between eye tracking, neuroimaging (Section 6.1), and the implications for reproducible research from the three modalities' perspective (Section 6.3) are presented for the first time in this article. We also significantly restructured the formal development for a software engineering journal audience, moving certain technical details most relevant for reproduction to an appendix.

2 BACKGROUND AND MOTIVATION

We first summarize results and techniques related to psycho-physiological measures for a computer science audience in Section 2.1. Second, we review the software engineering tasks involved in this study, including code comprehension and code review (Section 2.2.1), while summarizing the study of mental rotation in psychology, supporting our experimental use of it as a neurological basis for spatial ability (Section 2.2.3).

2.1 Psycho-Physiological Measures

In this subsection, we overview the mechanism and research use of fMRI, fNIRS, and eye tracking, including their relative advantages and disadvantages for our experiments.

2.1.1 Neuroimaging. Functional neuroimaging techniques are used to study brain activity. Over the past 30 years, non-invasive in vivo functional neuroimaging techniques have emerged as important tools in understanding cognitive processes. The most popular of these techniques, fMRI, and its counterpart, fNIRS, provide several advantages.

First, as non-invasive tools, fMRI and fNIRS pose significantly less risk and can access a wider range of brain regions than previous invasive techniques (e.g., electrocorticography). Second, fMRI and fNIRS provide a wider field of view and higher spatial resolution than other functional neuroimaging techniques (e.g., EEG, MEG), allowing for the characterization of a brain region's contribution to a specific task. Third, fMRI and fNIRS avoid the use of ionizing radiation or radioactive elements that is common in many other neuroimaging modalities (e.g., CT, PET). Instead, fMRI and fNIRS rely on the *hemodynamic response*, the metabolic changes (e.g., oxygen, glucose) in neuronal blood flow to active brain regions, using oxygen consumption as an indirect measurement for brain region activity [22].

As a result, fMRI and fNIRS are popular in research. In 2010 alone, fMRI was used in more than 1,500 published articles [145]. Among other examples, fMRI has been used to study face recognition, decision making, resting, and vegetative states [94, 103, 108, 145, 146, 163]. Similarly, the use of fNIRS is also on the rise [18]. The applications of fNIRS span many fields such as behavioral development, psychiatric conditions, and brain injury [18, 42, 95, 107].

However, fMRI and fNIRS rely on the hemodynamic response, and share several limitations. One limitation arises from *hemodynamic lag*: the onset of changes in neuronal blood flow peaks several seconds after the onset of stimuli [1, 67]. Similarly, the hemodynamic response saturates over time [92], resulting in weaker signals for tasks involving sustained activity. The hemodynamic response enforces experimental restrictions such as lower and upper limits on stimuli (commonly 30 seconds), as well as demanding robust mathematical analysis [16, 134].

How fMRI Works. fMRI provides indirect measurements of brain activity through calculations of the *blood-oxygen level dependent* (BOLD) signal, defined as the ratio of oxygenated to deoxygenated hemoglobin [109]. fMRI measures BOLD signals via the application and removal of a series of magnetic fields. The energy that nuclei emit upon returning to their original positions can be used to determine their locations. As task-related brain activity is mapped onto an anatomical scan of the participant's brain in the associated mathematical analysis, participants must lie still in the narrow fMRI machine throughout the experiment with minimal head movement.

How fNIRS Works. fNIRS also measures the hemodynamic response to determine active brain regions. fNIRS relies on differences in the absorption of chromophores, groups of atoms that generate color through the absorption of light, between oxygenated and deoxygenated hemoglobin. Light is emitted and detected through devices placed at specific locations on a scalp cap worn by the participant. Unlike fMRI, fNIRS measures concentration changes in oxygenated and deoxygenated hemoglobin separately. fNIRS admits relative freedom of motion and has few environmental restrictions. For example, participants can sit in front of a standard computer and monitor and perform in a more realistic software development setting.

Comparison of fMRI and fNIRS. Both fMRI and fNIRS have been widely used in psychological and clinical research to develop a deeper understanding of brain functions such as sensory, verbal, and motor processing [3, 55, 91, 110, 118, 149]. fMRI provides excellent spatial resolution and deep penetrating power. It is a precise neuroimaging modality that captures activations across the whole brain. In contrast, fNIRS provides inferior spatial resolution and depth compared to fMRI due to inconsistent photon paths and the limited penetration of near-infrared light. As a result, fNIRS also provides a noisier signal, leading to more careful considerations in experiment and analysis design. Likewise, fNIRS places a burden on the researcher to decide, in advance, on the placement of light emitter-detector devices. Given finite placement space on the scalp, the number of regions fNIRS can measure simultaneously is limited. However, fNIRS is gaining traction as a neuroimaging technique due to its portability, ease of administration, ecological validity, and lower cost. In contrast, the high cost, restrictive environment, and high sensitivity to participant motion of fMRI limit its practicality. In this article, we present recommendations for the use of fMRI and fNIRS to study software engineering.

2.1.2 Eye Tracking. Modern eye trackers are non-invasive, versatile, easy-to-use devices that have been used to study diverse topics, such as surgery [68], driver-vehicle interfaces [173], human-computer interactions [121], gaming [9, 151], and software engineering [106, 138]. Eye trackers are designed to collect a participant's visuo-spatial attention by recording eye-movement data [124]. Visual attention triggers the mental processes required for comprehending and solving a given task, while cognitive processes guide the visual attention to specific locations. Thus, eye tracking provides useful information to study the participant's cognitive processes and effort while performing tasks [59]. Compared to the conventional self-reporting methods, eye trackers are a cost-effective way of collecting data at a fine level of details with minimal intrusion.

An eye tracker also provides information that is not available from conventional methods, including fine-grained patterns of visual attention (visual attention trends) [15, 78]. A *visual attention trend* encapsulates changes in participant's visual attention over time.

How Eye Tracking Works. Modern, non-intrusive eye trackers consist of two miniature cameras and one infra-red light source. They measure and track the human eye's focus point using the "corneal-reflection/pupil-center" method [59, 75]. The invisible infra-red light is directed into the participant's eyes. After entering the retina, a large proportion of the emitted light is reflected back and creates a strong reflection which causes the pupils to appear very bright. A corneal reflection is also generated and shown as a sharp glint over the iris. Cameras then record the center of the pupil and location of the corneal reflection while image processing identifies and tracks the eyes. Raw data recorded by an eye tracker is processed by an event detection algorithm and results in *eye gaze* data. Eye gaze data is studied with respect to certain *areas of interest* (AOIs) in a stimulus. AOIs are manually defined by the experimenter based on research questions and variables [58, 78, 121, 137].

Eye gaze data is typically divided into two categories [124] based on ocular behavior. A *fixation* is a spatially stable eye gaze that lasts for approximately 200–300 ms (on average, three eye fixations happen per second during active looking). During a fixation, visual attention is focused on a specific area of display. Researchers in psychology claim that most of the information acquisition and processing occur during fixations [78, 121] and that a small set of fixations suffices for the human brain to acquire and process a complex visual input [58, 78, 124]. Fixation data has been extensively used to measure the visual effort (cognitive load) representing the tasks and stimuli being assessed [121, 136, 137]. Longer fixation duration and higher number of fixations indicate higher visual effort [121, 136]. A *saccade* is a continuous and rapid eye-gaze movement that occurs between fixations. Saccadic eye movements are extremely rapid (within 40–50 ms). Cognitive processing during saccades is very limited [78, 124].

2.2 Software Engineering Tasks

In this subsection, we present some relevant background on the software engineering tasks considered, as well as results related to expertise and imaging.

2.2.1 Code Review. Static program analysis methods aim to find defects (or other critical information) in software and often focus on discovering those defects early in the code's lifecyle. Code review is one of the most common forms of static analysis today [148]; well-known companies such as Microsoft, Facebook, and Google employ code review regularly [79, 159]. At its core, code review is the process of developers reviewing and evaluating source code content and changes. Typically, the reviewers are someone other than the author of the code under inspection. Code review is often employed before newly written code can be committed to a larger code base. Reviewers may check for style and maintainability deficiencies as well as defects. Numerous studies have affirmed that code review is one of the most effective quality assurance techniques in software development [4, 39, 46, 72]. While it is a relatively expensive practice due to high developer input, it successfully identifies defects early in the development process. This benefit is valuable because the cost to fix a defect generally increases with the time it goes unnoticed [165, 167, 171].

2.2.2 Code Comprehension. Much research, both recent and established, has argued that reading and comprehending code play a large role in software maintenance [63]. A well-known example is Knuth, who viewed this as essential to his notion of Literate Programming [84]. He argued that a readable program is "more robust, more portable, [and] more easily maintained."

Knight and Myers argued that a source-level check for readability improves portability, maintainability, and reusability and should thus be a first-class phase of software inspection [83]. Basili et al. showed that inspections guided by reading techniques are better at revealing defects [141]. An entire development phase aimed at improving readability was proposed by Elshoff and Marcotty, who observed that many commercial programs were unnecessarily difficult to read [44]. A 2012 survey of over 50 managers at Microsoft found that 90% of responders desire "understandability of code" as a software analytic feature, placing it among the top three in their survey [20, Fig. 4].

2.2.3 Data Structure Manipulation. To date, one previous line of research has considered data structures at a cognitive level. In a qualitative study involving nine computer science majors, Aharoni investigated student thought processes when dealing with data structures [5, 6]. Aharoni found that visual representations influenced students' perceptions of the overall properties of data structures, suggesting that programmers use visual representations to reduce levels of abstraction. While we draw inspiration from Aharoni's investigation of data structure mental manipulations, rather than focusing on qualitative self-reporting, our studies use objective measurements of associated visual and neural representations.

Mental rotation is defined as the capacity to quickly and accurately rotate two- or three-dimensional figures in imagination [38]. Mental rotation tasks generally involve comparing two three-dimensional objects rotated about an axis (Figure 1, right), and are a standard paradigm for testing spatial ability [28]. Neuroimaging has provided evidence that mental rotation involves the right parietal lobe, a region believed to be responsible for spatial ability [27, 30, 65]. In our experiments, we use mental rotation as a validated test case for spatial ability. Shepard and Metzler found that the time required to solve mental rotation tasks is a linearly increasing function of the angular difference between the orientations of the two objects [140]. Gogos et al. studied the difficulty of mental rotation using fMRI to identify rises in the BOLD signal with increased angles of rotation [57]. Mental rotation is a meaningful comparison for the investigation of difficulty in our studies.

Table 1. Demographic Data of Eligible Participants

		Code Study (n = 29)	*Data Structure Study* (n = 70)	
Demographics		fMRI	fMRI (Eye Tracking)	fNIRS
Gender	Men	18	16 (16)	30
	Women	11	14 (10)	10
Degree Pursuing	Undergraduate	27	23 (21)	31
	Graduate	2	7 (7)	9

For the *Data Structure Study*, we capture eye-tracking data simultaneously during the fMRI session.

3 EXPERIMENT SETUP AND METHOD

In this section, we describe our experimental protocol for the two studies. Materials (e.g., all stimuli and de-identified data) are available at the project's website.[1] Detailed technical specifications of the fMRI and fNIRS acquisitions are available in our previous publications [51, 71].

3.1 Recruitment

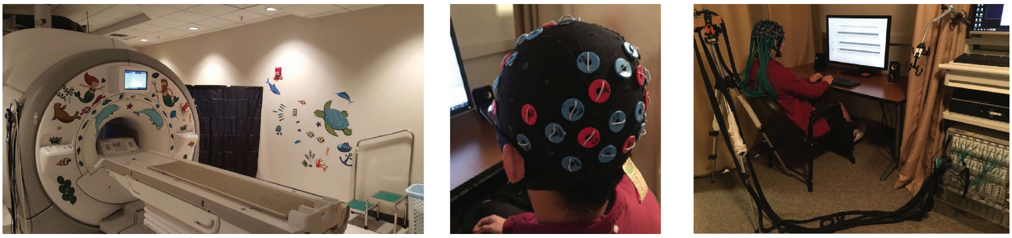
We recruited students for both studies. Solicitations were made via fliers in engineering buildings and brief presentations in two upper-level undergraduate CS classes. All participants were right-handed native English speakers and had normal or corrected-to-normal vision. They were also screened for basic experience in the programming language of interest. Prior to an experiment, each individual provided written informed consent for a protocol approved by the Institutional Review Board. Monetary compensation and course extra credit were offered. Table 1 summarizes the demographic information for all participants. We performed standard filtering including the exclusion of pregnant women, people with metal implants, head tattoos, and the left-handed (because the location of language processing in the brain strongly depends on handedness) for fMRI and participants with dark, thick hair for fNIRS. Also, data from six individuals were removed from the present analyses, either due to technical difficulties at the imaging center (yielding an incomplete dataset) or excessive head motion during the fMRI task. The final pool contained measurements from 29 participants using fMRI for the *Code Study* (35 recruited), 30 participants using fMRI and eye tracking (36 recruited), and 40 participants using fNIRS for the *Data Structure Study* (40 recruited).

3.2 Data Collection

Each full experimental protocol was completed over a single session per participant. Upon arriving, participants provided informed consent and completed a background questionnaire. After watching a training video, they were prepared for scanning and began the task activities. Following an initial anatomical scan, participants completed the tasks. They were encouraged to respond as quickly and accurately as possible within the time allotted for each trial—neural responses were considered from the start of the trial until a decision was made. Inter-stimulus intervals ranged from 2 to 8 seconds and consisted of a fixation cross displayed in the center of the screen. After completing the tasks, participants were given a chance to review some of the tasks outside of the scanner and offer verbal explanations for their responses.

For the *Code Study*, participants completed four 11-minute blocks of the code/prose tasks. In each run, stimuli were presented in alternating blocks of Code Review, Code Comprehension, and Prose Review; the blocks were ordered quasi-randomly across runs. All stimuli were presented for

[1] https://web.eecs.umich.edu/~weimerw/fmri.html.



(a) fMRI Machine (b) fNIRS cap (c) fNIRS environment

Fig. 2. (a) fMRI machine used in the *Data Structure Study*. The participant lies flat in the center of the bore. (b) The fNIRS cap on the head of a participant providing coverage of Brodmann areas 6–9, 17–19, 21, 39, 40, 41, and 44–47 is shown on the left. (c) A participant is shown completing the tasks in the fNIRS experimental environment.

a fixed time (30 s for prose, 60 s for code) and required an Accept or Reject response, made on an MR-compatible button box held in the right hand.

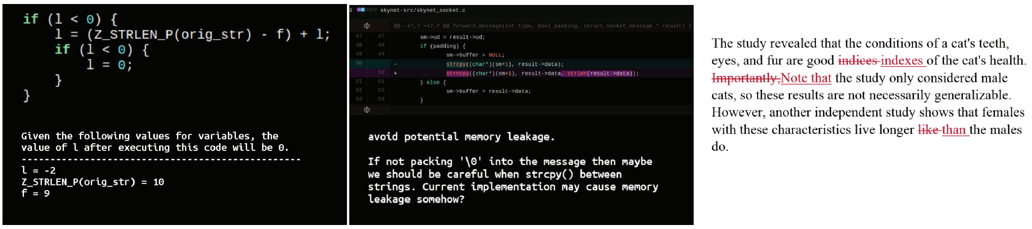
For the *Data Structure Study*, participants completed three 17-minute blocks of tasks. Stimuli used in the tasks were subdivided into three categories: (1) lists and arrays (collectively referred to as "sequences"), (2) trees, and (3) mental rotation. Each task block consisted of 10 stimuli from each category (30 stimuli in each block, 90 stimuli in total). The stimuli order was chosen randomly per participant. All stimuli were presented for up to 30 s and required an *A* or *B* response. All fMRI, fNIRS, and eye-tracking experiments used the same set of 90 stimuli.

fMRI Acquisition. All fMRI data were collected on a 3T MRI system using a 32-channel head coil following best practices from neuroimaging [56, 158]. Participants lay in an fMRI machine holding MR-compatible buttons and remained in the machine for the entire scan (see Figure 2(a)). The stimuli were presented as images on a screen in the back of the scanner. Stimuli were presented via a mirror that was placed above the head coil [25] with an approximately average distance of 4 inches between the mirror and the head of a participant.

fNIRS Acquisition. The fNIRS data were collected using a TechEn Inc. CW6 system with an above-average number of light detection channels, allowing for a broader view of the brain activities than many published fNIRS studies (cf. [74, 104]). The stimuli were presented as images on a computer monitor next to the fNIRS device (Figure 2(c)). The fNIRS participants sat in a chair wearing an fNIRS cap using a standard keyboard and monitor. The cap included 16 light emitters and 32 detectors, spaced 3 cm apart, yielding 61 data collection channels.

Eye-Tracking Acquisition. The eye-tracking data were collected using an MRI-compatible Avotec RE-5701 eye tracker by remotely monitoring and tracking participants' eye-gaze data during the fMRI session. The Avotec RE-5701 eye tracker generates 60 raw samples per second (i.e., sampling frequency of 60 Hz) and can be adjusted to view either of the participant's eyes. The viewing mirror and infrared illumination are mounted on the head coil, while the camera is located outside the bore. Using a slide projector, the galvanometer-driven mirror reflected the picture of a computer screen with a resolution of 1,920 × 1,080 and a font size of 24 pixels in height on top of the head coil. Participants viewed the stimuli via a mirror while supine. The eye tracker was installed at the head end of the scanner and received the images of the eyes via the second mirror. We calibrated the eye tracker for every participant before collecting data for each task.

As soon as fMRI acquisition begins, the eye tracker automatically starts capturing eye-gaze recordings. However, the eye tracker and fMRI software often run on two different machines, which makes synchronizing the stimulus presentation and fMRI timestamps with eye gaze recording a slight technical challenge. We created custom scripts to embed markers in the recorded data



```
if (l < 0) {
    l = (Z_STRLEN_P(orig_str) - f) + l;
    if (l < 0) {
        l = 0;
    }
}


Given the following values for variables, the
value of l after executing this code will be 0.
------------------------------------------------
l = -2
Z_STRLEN_P(orig_str) = 10
f = 9
```

(a) Code Comprehension

```
2 skynet-src/skynet_socket.c
@@ -47,7 +47,7 @@ forward_message(int type, bool padding, struct socket_message * result) {
        sm->ud = result->ud;
        if (padding) {
                sm->buffer = NULL;
-               strcpy((char*)(sm+1), result->data);
+               strncpy((char*)(sm+1), result->data, strlen(result->data));
        } else {
                sm->buffer = result->data;
        }

avoid potential memory leakage.

If not packing '\0' into the message then maybe
we should be careful when strcpy() between
strings. Current implementation may cause memory
leakage somehow?
```

(b) Code Review

The study revealed that the conditions of a cat's teeth, eyes, and fur are good ~~indices~~ indexes of the cat's health. ~~Importantly,~~Note that the study only considered male cats, so these results are not necessarily generalizable. However, another independent study shows that females with these characteristics live longer ~~like~~ than the males do.

(c) Prose Review

Fig. 3. Task stimuli of *Code Study*. Code comprehension stimuli feature true and false claims in the style of Sillito et al. [144]. These stimuli include the code difference (in color and with symbols) as well as the Git pull request message. Prose review stimuli are English paragraphs with proposed changes presented in a Microsoft Word "track changes" style.

streams. To handle our factorial experiment design consisting of three different blocks (Mental, Tree, and List) of randomized stimuli, we marked the start time of the first and each subsequent stimuli as well as information about block switching.

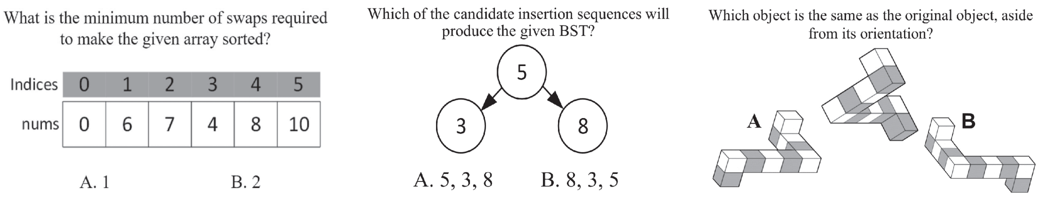
3.3 Materials and Experimental Design

3.3.1 Study 1: Code Study. In a controlled experiment involving 29 participants, we examine code comprehension, code review, and prose review using fMRI. Stimulus selection and design were informed by the experimental need to, in general, admit task completion within the time available.

Stimulus Type 1: Code Comprehension. A code comprehension stimulus consists of a snippet of code and a candidate assertion about it (Figure 3(a)). Judging whether the assertion is true or not about the code requires comprehending the code. For comparability with previous research, we used the same code snippets as Fry et al. [54]. Some samples were reduced slightly in size to fit the fMRI projection screen and colors were inverted for readability. The candidate questions were also taken from Fry et al., and thus ultimately adapted from Sillito et al.'s study of questions asked by actual programmers during software evolution tasks [144]. For each snippet an appropriate question type was selected at random. Assertions were induced from questions by including the correct answer or an incorrect answer (random coin flip). Assertions were used because the fMRI installation only allowed yes-or-no answers.

Stimulus Type 2: Code Review. A code review stimulus consists of an historical GitHub pull request, including the code difference and the developer comment (Figure 3(b)). Participants are asked to review the change and indicate whether they would accept it or not. A pool of candidate pull requests was selected by considering the top 100 C repositories on GitHub as of March 2016 and obtaining the 1,000 most recent pull requests from each. We considered the pull requests in a random order and filtered to consider only those with at most two edited files and at most 10 modified lines, as well as those with non-empty developer comments; the first 20 valid requests were used. Code was presented using the GitHub pull request web interface, simplified, and inverted for readability.

Stimulus Type 3: Prose Review. A prose review stimulus consists of a snippet of English writing marked up with candidate edits (Figure 3(c)). Participants are asked to review the changes and indicate whether they would accept them or not. We included two sources of English writing. First, we selected random chapters from an English writing textbook [40] that provides explicit correct and incorrect versions of candidate sentences and created examples based on grammar rules contained in those chapters. We created 20 stimuli of this type. Second, we selected random exercises


What is the minimum number of swaps required to make the given array sorted?
Indices
0
1
2
3
4
5
nums
0
6
7
4
8
10
A. 1
B. 2
Which of the candidate insertion sequences will produce the given BST?
5
3
8
A. 5, 3, 8
B. 8, 3, 5
Which object is the same as the original object, aside from its orientation?
A
B

(a) Sequence (List or Array) (b) Tree (c) Mental Rotation

Fig. 4. Example task stimuli for our *Data Structure Study*, reduced for presentation space. Sequence and Tree stimuli examples shown are simplified for clarity.

in "Paragraph Improvement" from the College Board's SAT study guide [155] (recall that all participants were native English speakers). Each such exercise has a paragraph and a set of questions on how to improve various parts of it; we created 10 stimuli by applying or reversing all changes.

3.3.2 Study 2: Data Structure Study. In this human study, we recruited 36 participants for fMRI and eye-tracking measurements and 40 participants for fNIRS measurements. We used 90 stimuli consisting of data structure (i.e., list, array, tree) and mental rotation tasks with varying levels of difficulty.

Stimulus Type 1: Data Structure. Each stimulus from the data structure category included a starting data structure, an operation to perform, and two answer choices (Figure 4). Answers were either numerical values to describe the outcome of an operation or candidate data structures resulting from an operation. A *sequence task* appeared as either a linked list or an array. For simplicity of modeling, we defined the *difficulty* of a sequence or tree task to be the total number of elements present—the N in Big-Oh notation. The sequence tasks include merge, insert, and swap operations. The tree tasks include binary search tree (BST) rotation, insertion, and traversal operations.

Stimulus Type 2: Mental Rotation. Each stimulus from the mental rotation category included a starting three-dimensional object and two candidate objects. Participants chose the candidate that could result from a rigid rotation of the original (Figure 4(c)). The mental rotation stimuli were adapted from the Mental Rotation Stimulus Library established by Peters and Battista [117] with rotational angle difficulty. Figure 4 shows simplified examples.

4 DATA ANALYSIS APPROACH

In this section, we present the mathematical analyses applied to fMRI, fNIRS, and eye-tracking data. Where applicable, we applied a *false discovery rate* (FDR) threshold at $q < 0.05$ to control for false positives as a result of multiple comparisons.

Notation. We use the neuroimaging notation A > B to refer to the *contrast* (or difference) between two tasks. For example, Sequence > Tree refers to the comparison of brain activations during sequence vs. tree manipulation. Contrasts are *directional* tests: the aforementioned Sequence > Tree contrast will specifically attempt to identify regions in which average sequence task activity is *greater* than tree manipulation. Critically, this does *not* imply that the inverse contrast (Tree > Sequence) will reveal regions in which tree activity is significantly greater than sequence activity, as differences in the opposite direction may be too small to be statistically meaningful (particularly conservative thresholds were used to guard against false positives).

4.1 Neuroimaging Analysis Approach

Our goal is to localize brain activation from task-related changes in the BOLD response (fMRI) or light absorption (fNIRS). Such analyses pose complicated statistical challenges, involving the interpretation of *hemodynamic* responses across anatomically and functionally diverse participants, which themselves are indirect metabolic proxies for underlying *neuronal* (i.e., molecular/cellular) responses. We used standard preprocessing techniques to identify and remove artifacts, validate model assumptions, and standardize locations of brain regions across participants. We used general linear models to obtain estimates of task-related brain activations within voxels (fMRI) or channels (fNIRS) based on the canonical hemodynamic response function. More specifically, we performed "multivariate pattern analyses" using Gaussian Process Classification (GPC) to determine the extent to which code and prose tasks elicited similar patterns of brain activity. For display purposes, we generate "posterior importance" maps. Each map contains a set of views of the brains including axial[2] and lateral[3] views. We divided the brain into 90 regions of the cerebrum, defined by the Automated Anatomical Labeling (AAL) atlas [161] and determine the total contribution (sum of absolute weights) of all voxels in each region, relative to every other part of the brain (cf. Section A.3). Finally, we performed statistical tests at both individual and group levels to test for significant brain activations, including subsequent correction for false positives. Appendix A provides a detailed summary of the mathematical analyses applied to fMRI and fNIRS.

4.2 Eye-Tracking Analysis Approach

Our eye-tracking analysis involves fixation detection, preprocessing, and the identification of areas of interest.

Fixation Detection. We use Ogama[4] to analyze eye-movement data. Ogama employs a dispersion-type algorithm with a moving window to detect fixations [128].

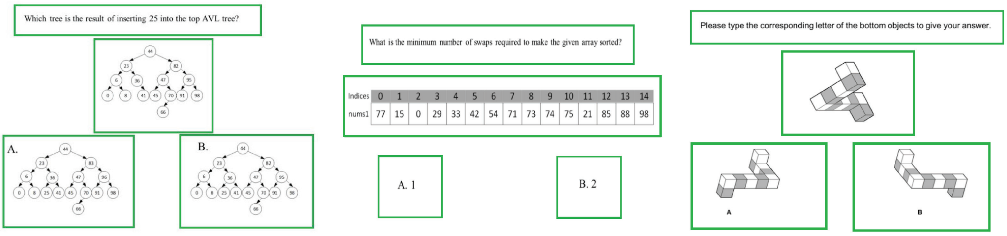
Preprocessing. The first step in preprocessing the eye-gaze data is to remove outliers and identify and fix drifts (offsets). *Drift* is the gradual decrease in time of the accuracy of data, when compared to the true coordinates of the eye movements. Drift happens when a participant moves beyond the capability of an eye tracker to follow or as a result of the deterioration of calibration over time. We use Ogama to manually detect drift by reviewing the captured video of eye movements and replaying the fixations and saccades. In our stimuli, questions are placed at the top of the screen, admitting a high-confidence mapping between the first fixation and the top question line on the screen. If the drift is visible to the researcher and is homogeneous (i.e., at any given time point, it is the same for all fixations), then we correct it by shifting all the fixations uniformly. When this is not possible, we exclude the incriminating stimulus and its captured data from the analysis. For about 10 participants, we corrected the drift on approximately 20% of the stimuli using this process. We also removed the gaze data of four (out of 30) participants from the analysis as around 80% of their stimuli were subjected to uncorrectable drift (i.e., either very few fixations or some located at the bottom of the page).

AOI and Metrics. We use Goldberg and Helfman's guidelines [58] for defining AOIs in terms of size and granularity. We manually divide every stimulus into four AOIs: *Question*, *Graph*, *Correct*, and *Wrong*. Figure 5 shows examples of AOI partitioning for Tree, List, and Mental tasks. The sizes of these areas are roughly the same across all stimuli of the same task. Yet, different AOIs may have different sizes across tasks. Deitelhoff et al. analyzed the impact of AOI sizes and paddings on code comprehension and reported that it can influence results [35]. Thus, we do not directly compare fixation data over two AOIs from different tasks (e.g., a *Graph* AOI in a Tree stimulus vs. a *Graph* in a List stimulus). Instead, we look into the distribution of visual attention over the AOIs while comparing different tasks. The *Question* AOI contains the question that participants

[2] An axial plane is parallel to the ground and divides the brain into the top (superior) and bottom (inferior).

[3] The lateral (sagittal) plane is perpendicular to the ground and goes right through the middle of the brain, dividing it into left and right halves.

[4] http://www.ogama.net/.



Which tree is the result of inserting 25 into the top AVL tree?

What is the minimum number of swaps required to make the given array sorted?

Indices	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
nums1	77	15	0	29	33	42	54	71	73	74	75	21	85	88	98

A. 1

B. 2

Please type the corresponding letter of the bottom objects to give your answer.

(a) AOIs of a Tree stimulus. (b) AOIs of a List stimulus. (c) AOIs of a Mental stimulus.

Fig. 5. Example AOI divisions for our *Data Structure Study*, reduced for presentation space.

answered. The *Graph* AOI represents the original data structure while the *Correct* and *Wrong* AOIs display the question's correct answer and the distractor option, respectively. We did not include the *Question* AOI in the analyses. For all Mental stimuli, we used the same question, while for the other categories, we used various types of questions.

We use the eye-gaze data to analyze and compare the problem-solving strategies of participants. A *strategy* models gaze data and visual attention trends over time throughout a task. We use scanpath metrics to quantify strategies. A *scanpath* is a series of fixations or AOIs in chronological order. Scanpaths are commonly analyzed in eye-tracking studies [58, 77, 138] and are typically presented in terms of diagrams (e.g., heatmaps) [77]. We compute the number of transitions between AOIs along with two standard oculomotor metrics as the components of a scanpath. First, *fixation time* is the duration of all the fixations on an AOI or the stimulus. Higher fixation time indicates difficulty in extracting information and an increased strain on the working memory [59, 75]. Second, *fixation counts* are the total number of fixations and indicate the number of attention shifts required to complete the task [78]. Counts often correlate highly with the time spent on a task.

5 RESULTS AND ANALYSIS

We address the following research questions:

RQ1 What is the difference between code review and prose review in terms of brain activity?
RQ2 What is the role of expertise in code review tasks?
RQ3 Do data structure manipulations use spatial ability?
RQ4 What is the role of task difficulty in data structure tasks?
RQ5 Do developers use different problem solving strategies for data structure tasks?

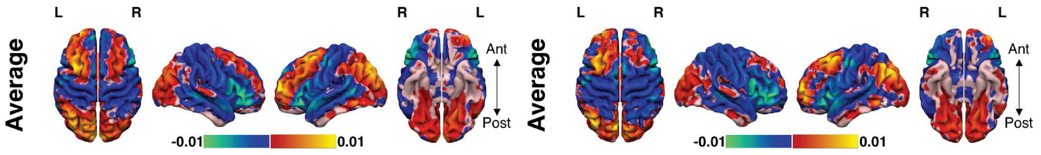
Tablen 2 summarizes the details of research questions and the modalities used. Our *Code Study* answers RQ1 and RQ2 while the remaining research questions are answered by the *Data Structure Study*. We also analyze the impact of participants' gender, age, and programming experience (years of programming) on participants' performance for the *Data Structure Study*.

5.1 RQ1: Task Classification and Regional Inference

We assess if we are able to classify which task a participant is performing based solely on patterns of brain activity using GPC. If code and prose are processed using highly overlapping brain systems, classifier accuracy would be low, reflecting entangled patterns of activity. For more details regarding the classification analysis, please refer to Appendix A. We consider the three tasks (Code Review, Code Comprehension, and Prose Review) pairwise. Median balanced accuracy (*BAC*) was compared for each of the three models using nonparametric Wilcoxon rank-sum tests [10]. There

Table 2. Summary of Research Questions, Modalities Used, and the Analysis Methods in Presented Studies

	Research Question	**Modality**	**Method**
Code Study	RQ1: Task Classification and Regional Inference	fMRI	Assessing the patterns of brain activity using Gaussian Process Classification
			Decoding the neural representations of code and prose from multivariate patterns of brain activity
	RQ2: Expertise	fMRI	Performing the correlation analysis between expertise (measured by GPA) and the accuracy of the brain activity classifier
Data Structure Study	RQ3: Involvement of Spatial Ability	fMRI & fNIRS	Performing contrast analysis: comparing the activation pattern of brain regions
			Computing pairwise comparison of brain activities between tasks
	RQ4: Task Difficulty	fMRI & fNIRS	Performing contrast analysis: comparing the activation pattern of brain regions
		Eye tracking	Performing correlation analysis of cognitive load (measured by fixation time) and data structure size
	RQ5: Problem-Solving Strategies	Eye tracking	Performing align-and-rank factorial analysis of the visual attention distribution across stimuli regions



(a) Code Comprehension vs. Prose Review

(b) Code Review vs. Prose Review

Fig. 6. Average weight maps for task classifiers. For both (a) and (b), from right to left, we have an axial superior view, an axial inferior view, a right lateral view, and a left lateral view. Ant (anterior) represents the front of the brain while Post (posterior) is the back. L and R indicate the left and right sides of the brain. When regions of the brain colored "hot" are active, the decision is pushed toward Code. The left and right subfigures show a high degree of concordance ($r = 0.75$, $p < 0.001$), quantifying how both code tasks are distinguished similarly compared to the prose task.

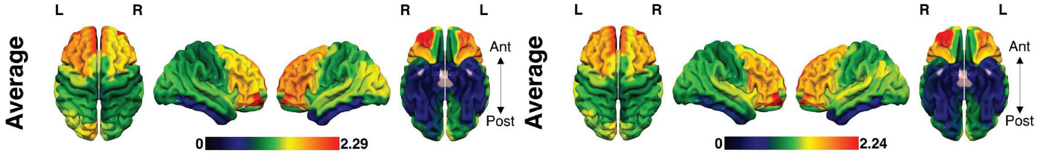
were no significant differences in classification performance for either Review vs. Prose models or Comprehension vs. Prose models, suggesting that GPC's ability to discriminate between code and prose tasks was not driven by the number of prose trials completed. This also held when considering only Prose class accuracy in models. Table 3 presents a full set of summary statistics for classifier performance.

With regard to overall classifier performance, we compared model *BAC* against a null median accuracy of 50% (chance for a binary classifier). For all models, GPC performance was highly significant. These results suggest that Code Review, Code Comprehension, and Prose Review all have distinct neural representations. Inspection of the average weight maps for each Code vs. Prose model (Figure 6) revealed a similar distribution of classifier weights across a number of brain

Table 3. Summary Statistics for Classifier Performance

	Class 1			Class 2			Overall		
Model	Accuracy	*Z*	*p*	Accuracy	*Z*	*p*	*BAC*	*Z*	*p*
Review vs. Prose	58.33%	1.85	0.064	87.50%	4.20	2.63E-05	70.83%	4.00	6.34E-05
Comprehension vs. Prose	72.73%	2.74	0.006	95.83%	4.61	3.94E-06	79.17%	4.51	6.38E-06
Review vs. Comprehension	66.67%	3.68	2.32E-04	58.33%	0.90	0.366	61.84%	3.45	5.70E-04

Median accuracies are given across participants per task (Class 1 and Class 2) and all together (Overall). For example, with Review vs. Prose, Class 1 is Review and Class 2 is Prose; test statistics and probabilities are derived from non-parametric Wilcox on signed-rank tests.



(a) Code Comprehension vs. Prose Review

(b) Code Review vs. Prose Review

Fig. 7. Average regional importance maps for task classifiers. For both (a) and (b), from right to left, we have an axial superior view, an axial inferior view, a right lateral view, and a left lateral view. Ant (anterior) represents the front of the brain while Post (posterior) is the back. L and R indicate the left and right sides of the brain. "Hot" colors indicate areas containing a greater proportion of the total classification weight (over all 90 AAL-defined regions). These proportions are directly interpretable, such that extreme red regions are twice as "important" as light green regions. The left and right subfigures show a near-perfect correlation $r = 0.99$, $p < .001$: the same brain regions as important for both code tasks in general vs. the prose task.

regions (here, "hot" voxels push the decision function toward Code with greater activation, while "cool" voxels indicate the reverse). Correlating the voxelwise values confirmed a high degree of concordance ($r = 0.75$, $p < 0.001$), indicating that (on average) similar patterns of activity distinguished between code and prose regardless of which code task was being performed.

Moreover, we investigate the relationship between tasks and particular brain regions. We look at the brain areas most involved in the GPC and examine their traditional roles and importance. As with the average multivariate weight ("posterior importance") maps, average regional importance maps for both Code vs. Prose classifiers demonstrated remarkable overlap (Figure 7). A correlation between importance maps yielded a near-perfect correspondence: $r = 0.99$, $p < 0.001$. For both classifiers, a wide swath of prefrontal regions known to be involved in higher-order cognition (executive control, decision-making, language, conflict monitoring, etc.) were highly weighted, indicating that activity in those areas strongly drove the distinction between code and prose processing. We also observed fairly large contributions from voxels near Wernicke's area in the temporoparietal cortex—a region classically associated with language comprehension. Together, these results suggest that language-sensitive areas of the brain were differentially recruited when processing code vs. prose. Thus, on average, programming and natural languages exhibit unique neural representations.

We employed a data-driven machine learning approach to decode the neural representations of code and prose from multivariate patterns of brain activity. Binary Gaussian Process classifiers significantly predicted when a participant was performing code-related tasks relative to prose review; it also distinguished between the two code tasks, though to a lesser extent. This latter observation is consistent with the remarkable spatial overlap, both qualitatively and quantitatively, between multivariate classifier weights in Code vs. Prose models, suggesting that the code tasks were largely represented similarly on average. This was confirmed by nearly identical a posteriori

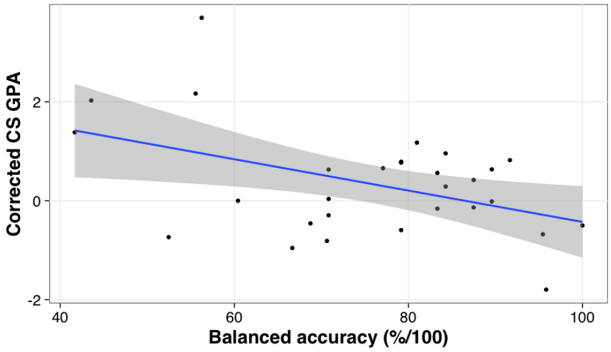


Fig. 8. Negative relationship between classifier performance (x-axis) and expertise (GPA), shaded 95% confidence interval.

estimates of regional importance: a number of prefrontal regions reliably distinguished between the code and prose tasks, accounting for most of the weight in the whole-brain classification models. Importantly, however, the extent to which these tasks were separable depended on one's expertise in programming—in the brains of experienced programmers, code and prose were nearly indistinguishable.

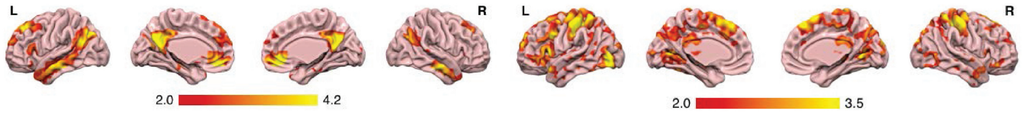
Code Review, Code Comprehension, and Prose Review are largely distinct in their neural representation. Activities in brain regions that are related to higher-order cognition, especially language comprehension, exhibit fairly large distinctions in prose and code tasks.

5.2 RQ2: Expertise

We examine the relationship between classifier accuracy and participant expertise. In light of the observed variability in classification performance across individuals, coupled with stark differences in multivariate weight maps between the highest and lowest performing models (not shown), we tested whether *BAC* predicted one's programming expertise. As a proxy for expertise, we obtained undergraduate GPAs counting only courses from the Computer Science department. These were corrected by the total number of CS credits taken (a 4.0 GPA with 8 credits presumably does not indicate equal expertise to a 4.0 GPA with 32 credits): a simple linear regression was specified predicting computer science GPA from completed credits, and the residualized GPAs were extracted for subsequent analysis. This allowed us to consider GPA as a skill indicator *independent* of the number of credits completed.

We then computed the correlation between expertise and classifier accuracy for both of the Code vs. Prose models. Discriminability performance in Code Review vs. Prose models was not related to expertise ($r = -0.25$, $p = 0.184$). However, the extent to which classifiers distinguished between Code Comprehension and Prose significantly predicted expertise ($r = -0.44$, $p = 0.016$) (see Figure 8). The inverse relationship between accuracy and expertise suggests that, as one develops more skill in coding, the neural representations of code and prose are less differentiable. That is, programming languages are treated more like natural languages with greater expertise.

The neural distinctions between prose and code tasks are mitigated by programmers' expertise: in the brains of experienced programmers, code and prose were nearly indistinguishable.



(a) Significant clusters of activity: Mental > Tree (b) Significant clusters of activity: Sequence > Mental

Fig. 9. "Hotter" colors indicate regions showing a larger magnitude difference between the two tasks. (a) More activity during mental rotation relative to tree manipulation. (b) More activity during difficult sequence manipulation trials relative to difficult mental rotation trials.

5.3 RQ3: Involvement of Spatial Ability

We began with a broad examination of mental Mental > Datastructure tasks, independent of task difficulty: this would allow us to determine whether there were reliable differences between Mental task and the two data structure tasks on average.

fMRI Results. Given that mental rotation reliably activated several regions commonly associated with the brain's "default mode network" (DMN) more than the two code tasks, we applied more focal contrasts to determine whether there were specific differences between Mental > Tree and Mental > Sequence. This revealed that the Mental > Code effect was primarily driven by Mental > Tree (Figure 9(a)). While Mental > Sequence yielded significant differential activations in midline DMN regions, these clusters had relatively minimal spatial extent. Patterns of activity related to Mental > Tree, however, were nearly identical to those observed in the comprehensive Mental > Code contrast (Pearson's $r = 0.97, p < 0.001$). As with the omnibus Code > Mental contrast above, the inverse contrasts (Tree > Mental and Sequence > Mental) also had no voxels survive FDR thresholding.

fMRI results suggest more similarities than differences during mental rotation vs. software engineering tasks. A number of DMN regions involved in mental simulation were recruited more heavily during mental rotation; nevertheless, 95% of voxels were statistically indistinguishable between Mental and Tree tasks.

fNIRS Results. We first examined brain activations comparing each task to a rest condition. The columns Sequence, Mental, and Tree show the Brodmann Areas (BA), a standard classification of neural locations, that are significantly activated during the task categories ($p < 0.01$ and $q < 0.05$). The t-values range from 8 (much stronger activation) to -8 (much weaker). We observe that the three categories of tasks all involve significant activations in exactly the same brain regions: BA 6–9, 17–19, 39 and 46. Table 4 summarizes the fNIRS results. In the frontal lobe, the premotor cortex and supplementary motor cortex (BA 6), and the frontal eye field (BA 8) showed activation. In the parietal lobe, the part which is associated with visuomotor coordination presented activation (BA 7) and part of Wernicke's area showed activation (BA 39). We also observed strong activation in the primary, secondary, and associative visual cortex (BA 17–19). Finally, regions of the dorsolateral prefrontal cortex (BA 9, 46) showed activations for all tasks. All the brain areas listed in the table passed FDR correction ($q < 0.05$).

Having established a broad similarity in how the three tasks each differ from a rest state, we narrowed the investigation by examining how the tasks differ from each other. In Table 4, the column Sequence > Mental shows the brain activation results when comparing sequence tasks and mental rotation tasks. Areas related to vision (BA 17–19), Wernicke's area (BA 39), and the prefrontal cortex (BA 46) showed very different patterns of activation between the data structure task and mental rotation. In addition, areas related to language processing (BA 41, 44–45, and 47,

Table 4. Summary of fNIRS Results

Sequence		Mental		Tree		Sequence > Mental		Mental > Tree		Sequence > Tree	
BA	t-value range	BA	t-value range	BA	t-value range	BA	t-value range	BA	t-value range	BA	t-value range
6	2.5–5.0	6	2.8–4.3	6	3.8–4.6					6	2.7–2.7
7	4.7–5.5	7	5.9–6.4	7	5.1–7.2						
8	2.6–5.1	8	2.9–5.5	8	2.5–5.6						
9	2.6–5.1	9	5.5–5.5	9	2.7–5.3						
17	3.1–4.9	17	3.2–6.2	17	2.6–5.3	17	−2.4–−2.4				
18	3.8–5.2	18	5.3–6.9	18	4.2–5.3	18	−2.4–−2.4	18	2.6–2.6		
19	4.0–6.6	19	5.3–9.1	19	4.2–7.3	19	−4.3–−3.2	19	2.4–4.3		
39	3.7–7.1	39	4.1–7.9	39	4.4–7.9	39	−3.3–−3.3	39	2.4–2.4		
						41	−2.3–−2.3				
						44	−3.3–−2.6	44	2.6–3.4		
						45	−5.0–−2.4	45	3.5–3.5		
46	3.8–4.1	46	3.5–4.6	46	4.7–5.6	46	−5.9–−2.4	46	2.7–4.3	46	−2.6–−2.6
						47	−5.9–−5.0	47	3.4–4.3		

Each column corresponds to a particular task. Each row corresponds to a particular Brodmann Area used during that task along with the range of t-values measured by all fNIRS channels on that BA. Positive t-values indicate stronger activation while negative t-values indicate weaker activation. We report all t-values with $p < 0.01$: all reported results are significant.

which include Broca's Area) strongly distinguish the two. As we observe here, an area (e.g., BA 41) may not significantly distinguish Sequence from a rest state or Mental from a rest state, but may significantly distinguish them *from each other.*

However, the Mental > Tree and Sequence > Tree distinctions are far less compelling. In a comparison, t-values near to either 8 or −8 are relevant. While Sequence > Mental features three areas that reach a magnitude of 5 or more, the other two contrasts never reach a magnitude of 5 and involve fewer regions and channels. In an fNIRS analysis [73, 152], contrasts of that strength result in a conclusion that Mental and Tree, as well as Sequence and Tree, are similar tasks.

fNIRS results demonstrate that mental rotation and data structure tasks involve activations to the same brain regions. However, while Sequence > Mental may be a compelling contrast, the fNIRS evidence does not support the claim that the other tasks are distinct.

The fMRI and fNIRS results suggest a nuanced relationship between medical imaging, spatial ability, and software engineering tasks. Further investigation is warranted. For example, recently, Krueger et al. studied code *writing* using fMRI; previous fMRI studies, including those presented here, had focused on reading or interpreting static stimuli. They found that "while prose writing entails significant left hemisphere activity [. . .], code writing involves more activations of the right hemisphere, including regions associated with attention control, working memory, planning and spatial cognition." Our results suggest that fNIRS may not be suitable for investigations into certain software engineering tasks (e.g., tree manipulations, code writing) that are currently believed to involve spatial ability.

5.4 RQ4: Task Difficulty

Mental rotation tasks have a natural notion of difficulty, namely, angle of rotation. Shepard and Metzler [140] found that the time required to solve mental rotation tasks is a linearly increasing function of the angular difference between the orientations of the two objects. For Tree and List tasks, we use the size of the graph as a measure of difficulty.

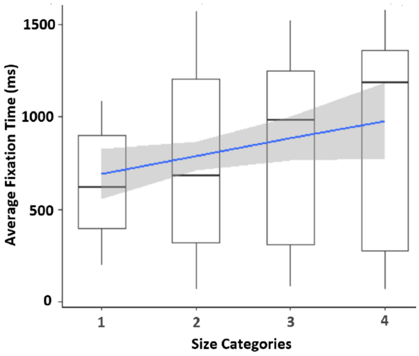


Fig. 10. Positive relationship between the size of the tree and the fixation time required to solve the task. Size category quartiles are 1 (small), 2 (medium), 3 (large), and 4 (very large).

fMRI Results. When we considered the difficulty of each task, we found a significant effect in Sequence > Mental (Figure 9(b)). Larger sequence tasks elicited stronger activations across a wide extent of the brain (FDR-corrected). With the exception of PCC, there was little to no overlap with DMN regions (as seen in the contrasts in Section 5.3). Rather, the largest clusters included bilateral postcentral gyrus (BA 40), left inferior frontal gyrus (IFG; BA 44–45), bilateral dorsomedial PFC (dmPFC; BA 6, 8), bilateral anterior insula (BA 13), and bilateral ventral precuneus extending into visual association cortex (BA 18). The heavy recruitment of frontoparietal regions—particularly in the left hemisphere—suggests an increase in cognitive load [32] scaling with the total size of the stimuli. That is, we found that the brain works measurably "harder" for more difficult problems. Because the relationship between mental rotation difficulty and the BOLD signal is so well-established in psychology and cognitive neuroscience [57, 140], it is particularly compelling that we observe a significantly larger effect (in terms of cognitive load and top-down control rising with more complex stimuli) for sequence data structures in software engineering than for mental rotation.

fNIRS Results. A similar analysis with our fNIRS data revealed no significant findings for the effect of task difficulty on neural activity. This is likely due to fNIRS lacking the penetrative depth and spatial resolution of fMRI.

Eye-Tracking Results. To investigate the impact of task difficulty, we measure participants' visual effort by calculating the average fixation time spent on all stimuli per participant. Fixation time is the duration of all the fixations on the stimulus. The amount of fixation time should increase with the participants' visual effort.

For the Mental task, our results show that the angle of rotation positively correlates with the amount of fixation time spent by participants (Kendall's τ test: $\rho = 0.15, p < 0.05$). The amount of participants' cognitive load to solve the mental rotation task is a linearly increasing function of the angular difference.

We characterize difficulty in the two categories of data structure tasks using the total number of elements in the data structure. List sizes ranged from 10 to 21 elements; trees ranged from 8 to 20. We divide data structures into four size quartiles. We find that the amount of visual effort required to solve tree problems is a significant predictor of difficulty (ordinal logistic regression: $F = 3.7, p = 0.02$) as shown in Figure 10. A higher cognitive load is required to work with larger trees. We find no significant effect of task difficulty on visual effort for the list task.

Table 5. Pairwise Comparisons of Three Tasks Using Non-Parametric Wilcox Tests ($\alpha = 0.05$) for Number of Transitions, Fixation Time, and Fixation Count

	Mean (Standard Deviation)			List vs. Mental	Tree vs. Mental
	List	Mental	Tree	p	p
Number of Transitions	315 (68)	102 (22)	228 (117)	**<0.001**	**<0.001**
Fixation Time (s)	1,361 (1,172)	1,413 (1,095)	1,629 (1,162)	0.7	0.07
Fixation Count	40 (23)	19 (17)	34 (21)	**<0.001**	**<0.001**

Results for each metric are at the right, with significant results (<0.05) bolded.

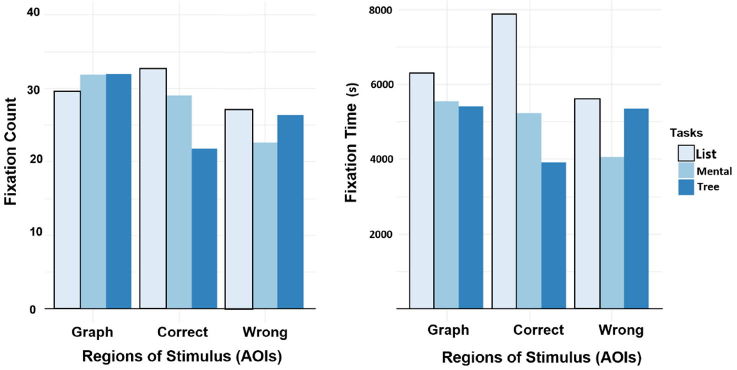


Fig. 11. Comparison of Fixation Time and Fixation Count between AOIs per task. Longer fixation duration and higher number of fixations indicate higher visual effort. Participants working on the Tree task, on average, spent more time and effort on *Wrong* AOI compared to the *Correct* AOI.

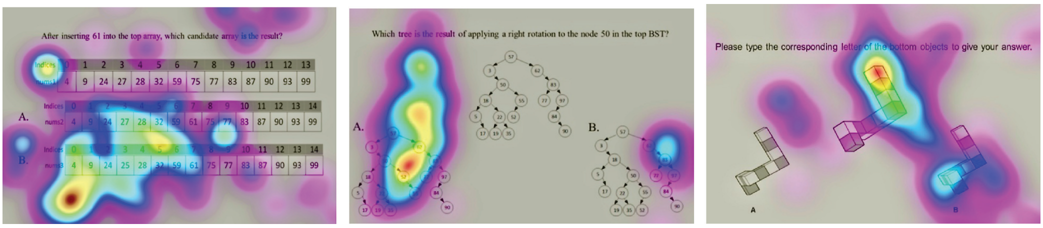
> The brain works measurably harder for more difficult software engineering problems (in terms of cognitive load). Moreover, the regions activated suggest a greater need for effortful, top-down cognitive control when completing challenging sequence manipulation tasks. Similarly, larger tree manipulation tasks required more visual effort.

5.5 RQ5: Problem-Solving Strategies

We use eye-gaze data to analyze and compare the problem-solving strategies of participants. We analyze eye movements globally over the whole stimuli, as well as locally with respect to AOIs. We measure fixation counts and fixation time on the entire stimulus. As shown in Table 5, participants fixated more frequently while working on data structure stimuli, implying a more active scanning pattern and more exploration compared to Mental stimuli.

We also calculate the metrics mentioned above within each AOI to compare the participants' pattern of attention switching across the spectrum of AOIs. We observe more transitions between AOIs for List > Mental and Tree > Mental in a statistically significant manner.

We find that participants have different attention distributions when working on different stimuli. We use a general align-and-rank non-parametric factorial analysis [168] and find that there is a significant interaction between the stimuli categories ($F(4, 204) = 3.58, p < 0.001$ for fixation count and $F(4, 204) = 6.9, p < 0.001$ for fixation time). These results confirm that AOI relevance varies significantly across three categories. Figure 11 presents a qualitative explanation of participants' attention distribution across AOIs. Participants working on the Tree task, on average, spent more time and effort on *Wrong* AOI compared to the *Correct* AOI. Post-hoc comparisons of


After inserting 61 into the top array, which candidate array is the result?
A.
B.
Which tree is the result of applying a right rotation to the node 50 in the top BST?
A.
B.
Please type the corresponding letter of the bottom objects to give your answer.
A
B

(a) Sequence (List or Array) (b) Tree (c) Mental Rotation

Fig. 12. Eye-gaze heatmaps for one participant, working on one stimuli of each task. A heatmap is a color spectrum that represents the intensities of fixations. The colors red, orange, green, and blue indicate the fixation duration from highest (warmer tone, redder) to lowest. While the participant's attention is broadly distributed over the AOIs for Sequence and Mental AOIs, image 12(b) presents long, intense attention on the Wrong AOI (left-hand side).

Table 6. The Impact of Participants' Reported Demographic Information on Performance

	Mean (SD)						**p-value**		
	Gender		Age		Experience		Gender	Age	Experience
Performance	Men	Women	18-22	23-27	<5 years	>5 years			
Time (s)	11.7 (6.4)	12.08 (6.2)	12.19 (6.3)	11.29 (6.3)	12.00 (6.3)	11.55 (6.5)	0.12	**<0.001**	0.1
Accuracy	0.77 (0.41)	0.77 (0.42)	0.76 (0.42)	0.77 (0.41)	0.77 (0.41)	0.76 (0.42)	0.9	0.5	0.6
Avg. Fix. Duration (ms)	191.1 (52.5)	192.3 (51.3)	197.1 (51.5)	190.0 (52)	196.8 (51.1)	189.8 (52.3)	0.3	**0.001**	**0.002**
Fixation Count	30.9 (22)	29.8 (22)	39.4 (23.0)	28.2 (21.2)	39.8 (22.7)	27.7 (21.1)	0.15	**<0.001**	**<0.001**

Pairwise comparisons of performance using Chi-squared test for accuracy and non-parametric Wilcoxon test ($\alpha = 0.05$) for time and visual effort (measured by average fixation duration and fixation count). Significant results ($p < 0.05$) are bolded.

interaction contrasts using a chi-squared test with Holm adjustment reveal an increased fixation time for the *Wrong* AOI compared to the *Correct* AOI for Tree > List ($\chi_r^2(1) = 8.3, p < 0.001$) and Tree > Mental ($\chi^2(1) = 11.8, p < 0.05$). This is consistent with a strategy in which participants tried to rule out the answers believed to be incorrect (see Figure 12) instead of directly solving the problem.

Figure 12 displays eye-gaze heatmaps of one participant working on three different stimuli. A heatmap shows the relative intensity of fixations by assigning each value a color representation. "Hot (warmer)" colors represent those that are highest in their value, fixated more intensely. "Cold" colors correspond to lower values. These heatmaps show distinct patterns of problem-solving behavior while working on various tasks.

Participants use different problem-solving strategies (different attention distributions and scanning pattern) when working on different data structure tasks.

5.6 Performance Differences

In this article, we extend our previous analysis [71] and investigate the impact of participants' reported demographic information (gender, age, and the number of programming years) on performance (See Table 6). We measure participant performances on code review tasks in terms of the amount of visual effort (cognitive) measured by eye-tracking data, the amount of time spent finishing the task, and the number of correct answers (accuracy). The visual effort is measured by fixation count, average fixation duration, and average saccade length [78, 111, 121, 137]. Longer fixation times and saccades indicate more effort and higher cognitive load. No significant effect of provenance was found on eye-tracking metrics in isolation [78, 111, 121, 137].

The test of proportion (Chi-squared test for significance) for accuracy shows no difference between participants regarding gender, age, or experience. However, we observe that age and the total number of years of programming impact the amount of effort participants put into solving the tasks. Our analysis shows that experience (number of years of programming) positively correlates with age (Kendall's τ test: $\rho = 0.7, p < 0.001$). Older and more experienced participants spent less time and effort on the tasks.

While there have been several investigations of prejudicial or systemic bias in software engineering, such as Huang et al.'s study of gender biases in code review [70], Terrell et al.'s study of the acceptance rates of contributions from men vs. women on GitHub [153], or Ford et al.'s study of gender participation barriers on Stack Overflow [52], our analysis is closer to experiments that consider gender as a factor in the performance of software engineering tasks [14, 60, 154]. Our results are in broad agreement with previous work [14, 60, 154] reporting that there is no difference between men and women participants regarding their performance. In addition, in contrast to previous work [14, 60, 70, 154], we did not find any gender differences in problem-solving strategies specifically (i.e., different attention distributions or scanning patterns). While this traditional two-cluster partitioning of participants (i.e., men vs. women) did not reveal any significant behavioral differences, future work may be able to shed further light on this by performing nuanced analyses of individual demographic differences.

6 DISCUSSION

In this section, we discuss the implications of our results and compare various aspects of using three different biological modalities to study software engineering tasks. Based on our collective experience with conducting human studies using neuroimaging and eye tracking, we capture our assessment of the essentials and summarize the suitability of various modalities discussed in Table 7.

6.1 Neuroimaging and Eye-Tracking Agreement

Our fMRI and fNIRS measurements and analyses both support the claim that mental rotation and data structure tasks recruit the same brain regions. However, while fMRI evidence supports a very robust Mental > Tree contrast, the fNIRS evidence is insufficient to support that same claim. This is sensible when we consider the regions yielding the largest differences in fMRI: they largely correspond to structures (e.g., the medial prefrontal cortex and posterior cingulate) that fNIRS cannot measure. Very informally, the parts of the brain that distinguish mental rotation from tree manipulations are too far "inside the skull" for fNIRS to see: its near-infrared light cannot penetrate deeply beyond regions near the cortical surface.

However, while fMRI is more spatially resolved, its restrictive and alien environment can also be more daunting for participants. Previous work also investigated the impact of fMRI isolated environments and scanner noise on participants' mood and performance [76, 101]. We compared participant performance (i.e., whether or not they gave the correct answer and how long it took) for fMRI and fNIRS; such information was available for 30 fMRI and 40 fNIRS participants. Recall that the questions were identical and the participants were drawn from the same pool. The average accuracy of fNIRS participants, 92%, was significantly higher than the 85% accuracy of fMRI participants ($t = 4.5$, $p < 0.01$) with no significant difference in response time. We also investigate the impact of problem difficulty on participants' performance while comparing fMRI and fNIRS. We characterize difficulty using the total number of working elements (for data structures) and rotation angle magnitude (for mental rotation) for the two categories tasks. The average

Table 7. Summary Assessment of the Suitability of Modalities

	Modality		
Suitability	fMRI	fNIRS	Eye tracking
Research conclusion			
Investigating patterns of brain activity	●	●	○
Analyzing the visual attention trends	○	○	●
Analyzing the impact of task difficulty	◕	○	●
Ecological validity			
Providing a realistic setup and environment	○	●	●
Being time and cost effective	○	●	●
Ease of data analysis	◐	◐	◐
Ease of recruitment	◐	◕	●

● = Good, ◐ = Average, ● = Poor or None.

accuracy of fNIRS participants for easy, moderate, and hard problems (92%, 93%, and 91%), was higher than the (81%, 88%, and 84%) accuracy of fMRI participants. The difference was significant for easy ($W = 82.5$, $p < 0.01$) and hard ($W = 113.5$, $p = 0.05$) problems. This could be a very relevant concern for neuroimaging studies of productivity, expertise, accuracy, or similar software engineering issues.

Eye trackers only measure eye movements, so they are inherently incapable of measuring brain structure and functions. However, eye movements are associated with cognitive processes and visual effort [75, 78, 121, 137]. The association between the task difficulty and the amount of cognitive load required to accomplish the task was manifested by both our fMRI and eye-tracking measurement and analyses. fNIRS data produced no significant findings for the effect of task difficulty on neural activity. In this article, we present a standalone, post factum eye-tracking analysis. Previous work in neuroscience has successfully demonstrated the use of fixations as markers (onsets) for calculating electrophysiological brain potentials (e.g., hemodynamic responses) [125, 160]. Peitek et al. [116] also discussed the feasibility and challenges of the simultaneous fMRI and eye-tracking analysis.

We also believe that a simultaneous eye tracking and fMRI recording and analysis (e.g., identifying fixation-related fMRI activation) is a promising direction for enhancing the explanatory power of fMRI. Eye tracking has the potential to provide insights into developers' cognitive processes at a very fine level of granularity (e.g., associated with particular source code features). The idea is to use fixations as markers for calculating hemodynamic brain responses measured by fMRI. We argue that the time of first fixation (e.g., for a graph in a stimulus) is a more valid start signal for cognitive processes than the time the stimulus first appears on the screen. Also, the sensitivity of a fixation-related fMRI analysis could be investigated to determine brain activity differences between different data structures (e.g., Tree vs. List) or tasks (e.g., insertion or deletion).

6.2 Self-Reporting

In the *Data Structure Study*, we also conducted a qualitative analysis of survey data focusing on the correlation between explanations provided by participant and neuroimaging data. At a high level, we find that self-reporting often subtly contrasts with analyses from fMRI and fNIRS data.

In one question, participants were asked to compare and contrast a mental rotation task with a BST rotation task. Of the 72 responses, 70% reported *no similarity* between the two tasks—which does not align with measured observation that the same brain regions are recruited to solve both tasks. Even if mental rotation and tree rotation feel subjectively different, changes to brain regions and brain region connectivity have been shown to correlate with learning rates and expertise [97, 131].

This finding reinforces a considerable body of work on unreliable self-reporting (both in psychology [96, 120] and in computer science, including fields such as security [126], human-computer interaction [36], and software maintenance [54]). As previous studies have relied on self-reporting to study mental processes associated with data structures [5, 6], this evidence informs future research of the importance of neuroimaging (or similar techniques) when studying the cognitive processes underlying software engineering tasks.

6.3 Implications for Reproducible Research

In this subsection, we discuss the actual costs of carrying out these types of studies in the hope that other researchers may carry out similar studies in the future. We have made our IRB protocol, experimental materials, and raw, de-identified scan and survey data publicly available. This allows other researchers to conduct alternate analyses or produce more refined models without the expense of producing this raw data.

Recruiting. All three modalities constrain recruiting. Most directly, remote participation (such as via Amazon's Mechanical Turk crowdsourcing, cf. [54, Sec. 3.5]) is not feasible.

In addition, there are specific filters. For example, fMRI typically requires corrected-to-normal vision (because of the mirror/projection setup) and is not approved for pregnant women or those with medical implants or head tattoos, and so forth. In some cases, participants may not be able to finish a fMRI scanning due to claustrophobia. On the other hand, fNIRS may place significant practical restrictions on the use of participants with dark, thick hair. In practice, we found the fNIRS restrictions to be less onerous (resulting in 0 unusable applicants compared to 4 for fMRI).

Despite this, we found recruiting to be quite straightforward. We recruited our participants with brief advertisements in CS classes, reimbursements, and offering participants high-resolution scans of their brains (that can be 3D-printed). For eye-tracking analysis, we discarded the gaze data of four (out of 30) participants as they include missing data or invalid samples. The retention rate of (87%) is higher than those reported in stand-alone eye-tracking studies [106, 138]. Peitek et al. [116] investigated the feasibility of adding simultaneous eye tracking to fMRI measurement and reported the partial data loss of 50% for eye-gaze data due to the limitation imposed by the fMRI environment. In comparison, 59 of 71 participants' fMRI data and all of the 40 participants' fNIRS data were included in the data analysis here.

Time and Cost. Experiment time and cost are significant concerns for fMRI studies. One hour is about the maximum time that a participant can comfortably remain in the device. With pre- and post-screening, each participant thus takes about 90 minutes, and each participant must be separately supervised by one or two researchers. In addition, fMRI scan time is expensive—about $500 per hour at our institution. This is a significantly higher monetary cost than the usual software engineering human study. The data acquisition of the code review study alone represents a participant and machine cost of $21,000 and 52.5 hours of graduate student time.

There is almost no extra effort or cost when incorporating eye tracking into an fNIRS or fMRI study if an eye-tracking camera is already available and installed. However, the very brief time cost for calibration and validation of the eye tracker is not consequential compared to other setup time costs (e.g., participants completing forms, fNIRS cap fittings).

Research Questions. The nature of the BOLD signal measured by fMRI and fNIRS influences experiment design. Notably, tasks in which participants are performing the same activities at the same time intervals are favored. Similarly, the contrasting, subtractive nature of these analysis forces certain experimental controls. Informally, these modalities cannot illuminate X directly: researcher must formulate tasks Y and Z such that X is the difference between them. In addition, the limited range of participant actions available restricts the range of tasks: for example, without a keyboard, no new coding is possible. In general, however, we found research question design to be fairly direct given consultations with psychology researchers who had medical imaging experience.

IRB. An Institutional Review Board or Ethics Board governs acceptable human study research at a university. While eye-tracking studies are usually "exempted" (a lightweight variant of "approved"), IRBs often distinguish between medical research and other (e.g., social or behavioral) research. fMRI and fNIRS studies fall into the heavily regulated medical category. The medical IRB paperwork necessary for our studies involved 236 questions in the cover sheet alone, and the main protocol was 33 pages (compared to 13 for non-medical protocols). In addition, since brain scans are HIPAA protected data, a four-page data protection and privacy plan was required.

Ecological Validity. One of the more challenging aspects of experimental design was balancing ecological validity (i.e., are the activities undertaken by the participants indicative of the real world) with the constraints of neuroimaging and eye tracking.

Eye tracking and fNIRS both stand out for their portability, admitting experiments in more realistic environments. The development of new eye-tracking tools like iTrace [135] to support scrolling in, and switching between, files paves the way for performing experiments on larger software artifacts, including source code [2, 80], bug reports, and requirements documents. A few previous efforts successfully combined eye tracking and fNIRS to study software engineering tasks [47, 48].

The small sizes of tasks with highly controlled stimuli and the artificial setting are two main factors compromising the ecological validity of fMRI studies [82, 162]. Tasks are designed to isolate one specific skill and must be repeated many times to account for neuroimaging signals' noisy nature. This repetition increases session length. The scan duration and data analysis prevent multi-minute tasks. Also, the sterile, medical environment of medical imaging labs in which participants lie in a narrow magnetic tube (for fMRI) can be intimating and far from naturalistic. The standard MRI-safe button press device precludes scrolling, requiring all stimuli to fit on one screen. While fMRI has the best penetration power, all these issues make the setting and tasks an imperfect fit with software developers' real-life working conditions. These limitations introduce challenges in translating the insights generated in the lab experiments to real-life brain function [82].

Implication Summary. If past history is to be our guide, we believe that many of these limitations will lessen or disappear as newer technologies are invented in the years to come [100]. Recently, Krueger et al. [85] successfully performed the first code writing fMRI study. They employed a bespoke keyboard while moving all metal and control logic to a separate room to use the keyboard inside the bore safely.

Portable devices such as eye trackers and fNIRS are moving research to more naturalistic settings. An eye tracker provides additional insights into participants' cognitive processes and the intentions that motivate their actions. Compared to fMRI and fNIRS, eye tracking provides a cost-effective, less intrusive solution to objectively measure software engineering tasks without conscious filtering. However, eye trackers come with intrinsic limitations [59, 121, 137], including the inaccuracy between actual and measured gaze data, the gradual decrease of accuracy (drift) over time, and not capturing the extrafoveal vision that accounts for up to 98% of the human visual field.

While we acknowledge these limitations and the extra cost and effort required to gather data compared to traditional methods (e.g., screen and audio recordings and surveys), we believe that eye tracking provides a cost-effective way of objectively assess participants' cognitive load. It can be used to perform preliminary studies prior to a neuroimaging study to evaluate the hypotheses, research questions, and materials.

Overall, while costly fMRI studies of software engineering remain limited, we expect the availability of hand-held fNIRS scanners, off-the-shelf eye-tracking cameras, and open-source data analysis tools for both eye tracking and fNIRS data to result in the growth of studies that those modalities in more natural (e.g., sitting down at a standard computer) settings.

7 THREATS TO VALIDITY

One threat to validity associated with generalizability is that our stimuli may not be indicative. For example, all code stimuli were in C and all prose stimuli were in English. Also, due to the inherent limitations of fMRI and fNIRS (see Section 2.1.1), we explicitly used static stimuli that took no longer than 30 seconds to finish. Thus, by focusing only on relatively short source code and data structure tasks in which participants were not able to interact with the computer (no scrolling, no code navigation, etc.), our results may not generalize to real-world software engineering tasks. This emphasis on tasks that are much shorter than many of those performed by practicing software developers is a significant limitation of the current use of neuroimaging in software engineering [24, 41, 48, 51, 74, 104, 116, 142, 143]. We mitigate this threat slightly by choosing stimuli from the representative resources. While our examples are not multi-language, we approach generality by choosing code changes at random from real-world projects and using established standardized test questions. We select data structure tasks from college-level courses which commonly focus on associated fundamental skills.

Our use of GPA as a proxy for expertise introduces an additional threat. Measuring participant expertise is difficult, and the metrics used are often domain-specific. For example, in specification mining the number of edits to version control repositories has been used as a proxy for expertise [88], while research related to Community Question Answering sites may be used as a proxy of expertise based on counting or profiling [170]. GPA correlates with learning and academic aptitude (e.g., [61, 147]).

One potential threat to internal validity concerns whether or not our tasks measure what they claim to be measuring (i.e., "data structure manipulation," "code comprehension," or "code review"). The thought processes that participants used when answering may not be identical: indeed, there is significant inter-participant variance in the neural representation of this problem solving. For the code review study, we mitigate this threat by posing the types of questions known to be asked by programmers during software evolution tasks [144] and presenting code review as it would appear to a remote GitHub user. While the particular data structures and tasks we chose are not representative of all of software engineering (e.g., skip lists, tries, heaps, maps, and so forth are not considered), we mitigate this somewhat by considering fundamental structures (linear sequential structures and branching trees). For eye-tracking data, calibrating the eye tracker for each participant and using well-documented, standard measures can mitigate the conclusion validity threat. However, it is important not to generalize our results far beyond what was directly measured.

Our use of mental rotation tasks as a baseline for spatial ability is one potential threat to external validity, as mental rotation and data structure manipulations differ in their *rigidity*. In spatial ability tasks, rigid transformations are those where distances between every pair of points on an object is preserved [11]. However, operations such as insertion, tree rotation, and merging may be more amenable to comparison with non-rigid transformations. Yet, we believe that mental rotation serves as a useful baseline (see Section 2.2.3). Mental rotation is a paradigm case of spatial ability, and has been classified on the basis of difficulty both with and without medical imaging [27, 30, 65].

A correction of eye-tracking data by manually moving fixations to fit the stimulus is a common procedure [112]. This manual procedure to correct drift may have introduced some biases into the results. We mitigate this threat by (1) assigning one of our authors to apply the same consistent strategy, and (2) fixing the questions on top of the stimuli as a reference point and only manually moving fixations if their locations deviate from the question line.

Another issue is related to our area of interest definitions. Due to inaccuracies in both eye trackers and the human visual system, fixations may fall outside the targeted object. To alleviate this issue, we follow accepted best practices [58, 137] and add extra padding around AOIs. Another factor that may impact the quality of the eye-tracking data is the deterioration of calibration over time due to a participant's fatigue or head movements. The use of video-based eye trackers reduces this instrument bias because participants can move their heads without decalibration.

We used a long-range camera-based eye tracker, placed outside the scanner bore. Integrating electronic devices with fMRI is an open challenge [64, 85]. Electronic or metal devices usually cannot be safely placed near magnetic resonance scanners. Not only can metal objects distort the MRI image, but fMRI also interferes with device measurement and reporting. Compared to eye trackers built in to the head coil, our setup provides lower precision of eye-tracking data. However, this design increases fMRI data quality by removing a source of noise for fMRI data.

The high dimensionality of fMRI and the complex mathematical analyses often necessitate conservative corrections for false positives and/or strong assumptions about the underlying data (that may or may not be met by reality) [16]. In a highly popularized article, Eklund et al. found that fMRI data often fail to meet the assumptions required for a certain "cluster-based" approach to multiple comparisons correction—this method, offered by nearly all common software packages for fMRI analysis, can therefore result in false-positive rates of up to 70% [43]. A key advantage to our multivariate approach is that all voxels are considered simultaneously, precluding the need for voxelwise multiple comparisons correction. However, this approach does preclude the sort of directed regional inference of standard GLM-based tests (cf. [142]).

A final threat to external validity is the pool from which we selected participants. By only recruiting undergraduate and graduate students, our results may not generalize beyond university programming experience and education.

8 RELATED WORK

In this section, we discuss previous work related to computer science and neuroscience, as well as studies in a wider range of domains that have used both fMRI and fNIRS. Additionally, we briefly discuss previous research on eye-tracking studies in software engineering.

Siegmund et al. introduced the study of software engineering tasks with fMRI, focusing on code comprehension [142]. Their analyses identified five brain regions with distinct activation patterns, all of which are relevant to working memory, attention, and language processing. Newer work has explored the relationship between comprehension, code, and prose review with expertise [51], bug detection and brain activities [24, 41], code comprehension with eye tracking [116], and the effects of beacons (semantic cues) on code comprehension [143]. Our study applies Siegmund et al.'s innovative use of neuroimaging, and adopts these previously identified brain regions as an established basis for verbal processing in software engineering. We identify relevant brain regions associated with verbal processing; their work focused on expertise and classification. Also, unlike previous work, we examine data structures and their correlation with spatial ability.

Similar to fMRI, fNIRS has been used to study the relationship between program comprehension and brain activity. Researchers used NIRS signals and found an increase in cerebral blood flow when analyzing obfuscated code and code that requires variable memorization [74, 104]. Subsequent research studied the effect of code readability on cognitive load [48, 143]. Using over 70 participants and multiple modalities, our study supports the feasibility of using fNIRS to study software engineering. Besides fMRI and fNIRS, researchers have tried other medical imaging tools to study software engineering. Crk et al. used electroencephalography (EEG) to investigate the role of expertise in programming language comprehension. Their study found that the brain's electrical activity can indicate both prior programming and self-reported experience levels [29]. Lee et al. used EEG in a similar setting [90] to Floyd et al.'s work [51]. Parnin used electromyography (EMG) to explore the roles of subvocalization for different programming [114]. Researchers have explored the link between programming tasks and cognitive load [53, 89] using EEG, EMG, and eye tracking.

Beyond neuroimaging, Parnin proposed a model focused on how a programmer manages task memory, specifically during multi-tasking and interruptions [113]. Of the previous studies combining neuroimaging or cognitive neuroscience with software engineering, none has investigated the effect of data structures on brain activity or explicitly investigated the relationship between data structures and spatial ability. In addition, no previous study has compared fMRI to fNIRS in the domain of software engineering. Over the last 20 years, the software engineering community has benefited from the uses of eye trackers. The results of eye-tracking studies add to the existing body of knowledge on how developers perform different software engineering tasks and how they use different models and representations along with source code to understand software systems. However, eye trackers are not without shortcomings and unlike neuroimaging, they do not provide insight into the brain activities. As a result, a handful of previous studies researchers started to use eye tracking simultaneously with EEG [53], fNIRS [48], and fMRI [116]. To the best of our knowledge, only Peitek et al. [116] performed a conjoint study to simultaneously use eye tracking and fMRI while providing a comprehensive analysis of the combined data.

However, care must be taken when designing fNIRS studies that involve activities in regions more distal from the scalp. Beyond raw signal-level correlations, our work finds that the resulting models of the two modalities do not draw identical conclusions on low-level explorations in software engineering (see Section 6.1), a relevant concern for future software engineering imaging research.

9 SUMMARY AND CONCLUSION

We present a systematic approach to objectively measure and foundationally understand the cognitive processes of programming activities. In particular, our two studies use fMRI, fNIRS, and eye tracking to investigate (1) the mental relationship between code review, prose review, and code comprehension, and (2) the neural correlation between data structure manipulations and spatial ability. We also examine the influence of problem difficulty and programmers' expertise levels by objectively measuring both programmers' brain activation and eye movements. We recruited 112 students in total to participate in these two studies.

Our first study is a controlled experiment involving 29 participants in which code comprehension, code review, and prose review tasks are contrasted against each other using fMRI. Siegmund et al. asked whether, following Dijkstra, good programmers need good native language skills, and explored code but not language or expertise [142]. Hindle et al. found that most software admits the same statistical properties and modeling as natural language [69]. Our findings in some sense bridge that gap, explicitly relating software, natural language, and expertise. All of our participants are students, and we use GPA as a proxy for expertise. Thus, our results should be considered preliminary, and may not be generalized to other settings. Our second study involved 70 participants in which we hypothesized that data structures are related to spatial ability. Our two key insights, with regard to the neural representation, were the use of multiple medical imaging approaches and the use of the mental rotation paradigm to serve as a baseline for measuring spatial ability.

The contributions of this article are as follows:

- We report on two human studies involving 112 student participants and three modalities, the largest such studies we are aware of for software engineering. We make available our study materials and de-identified dataset of raw participant brain scans and eye-gaze data.
- We find different patterns of neural activity between code comprehension and prose review tasks: the **the neural representations of programming languages and natural languages are distinct**.
- We demonstrate that **the neural representations of programming languages and natural languages are modulated by expertise.** Greater skill predicts less-differentiated representation. That is, expert brains treat code and prose tasks more similarly.
- We find that **data structure and spatial operations are related but distinct neural tasks**: they use the same focal regions of the brain but to different degrees.
- We demonstrate that **problem difficulty matters at a neural level** in computing, with complex stimuli inducing a relatively higher cognitive load in both data structure and mental rotation tasks. This claim is supported by both neural activity data and eye-movement data.
- We observe that participants' overall **problem-solving strategies differed across tasks** through their visual attention trends. Participants used elimination strategies and ruled out the incorrect options for tree tasks but directly solved mental rotation and list tasks.
- We present best practices and describe tradeoffs between fMRI, fNIRS, eye tracking, and self-reporting for software engineering research. We also discuss the barriers to conducting software engineering studies with objective measures using neuroimaging and eye tracking.

To the best of our knowledge, this is the first article that directly compares these three different objective measurements (fMRI, fNIRS, and eye tracking) in software engineering. We thus elaborate on both measurement and performance issues (Section 6.1), as well as monetary, protocol, and recruitment issues (Section 6.3). We argue, at a high level, that neural imaging and eye tracking in computer science have the potential to shed light on multiple unresolved problems (e.g., unreliable self-reporting, pedagogy, retraining aging developers, technology transfer, expertise, and the relationship between software and natural language). We acknowledge that the work presented here is still quite exploratory: a full quantitative theory of relating code, prose, and expertise remains distant. We also acknowledge the time and material costs of such studies, and make our materials and data available, inviting collaboration on future work.

APPENDIX

A NEUROIMAGING MATHEMATICAL ANALYSIS

In this Appendix, we present mathematical details associated with the analysis of medical imaging data from our software engineering experiments.

A.1 Preprocessing

A critical first step in the analysis of data is *preprocessing*, which serves to correct systematic sources of noise.

fMRI. We employed a number of standard preprocessing procedures using the Statistical Parametric Mapping 12 (SPM12, Wellcome Trust Centre for Neuroimaging, London) software in Matlab. First, we computed *voxel displacement maps* (VDMs) using images from the fieldmap sequence. We then realigned the functional scans after accounting for head motion over time; the VDMs were used to "unwarp" geometric distortions from motion. Next, the anatomical scans were segmented, skull-stripped, and spatially coregistered to the functional data. All images were then transformed into a standard space according to the Montreal Neurological Institute (MNI152) template [98]. Finally, we computed a brain mask using gray and white matter segments of the anatomical scans—this was applied in subsequent statistical analyses to prevent identification of false-positive signals within ventricles or outside of brain space.

fNIRS. The raw fNIRS data are light signals transmitted through the channels between emitters and adjacent detectors on the fNIRS cap. The light signals were converted to a measure of the optical density[5] change over time that results from hemodynamic responses.

A.2 First-Level Analysis

fMRI. Functional MRI analyses are *multi-level*. First-level models are specified on individual participant data—the results are then combined in a group-level model to assess average task-related changes in brain activity. We specified two first-level general linear models (GLMs) per participant. Briefly, these analyses require us to *predict* the BOLD response to each condition—voxels whose timeseries align with the predicted response are “task-sensitive.” In each GLM, we specified regressors for Sequence, Tree, and Mental stimuli across all runs. The duration of each event was curtailed to participant response times. These were convolved with the canonical hemodynamic response function (HRF) and high-pass filtered ($\sigma = 128$ s) to remove low-frequency noise. In one model, we additionally specified a *parametric modulator* for each condition to determine whether the magnitude of the BOLD response scaled linearly with trial difficulty. All models were fit using robust weighted least squares (rWLS) [37], which first obtains estimates of the error variance at each timepoint and reweights the images by a factor of 1/variance to reduce the influence of noisy scans (e.g., due to head motion). This procedure homogenizes the residual timeseries and obtains optimal parameter estimates for each condition.

fNIRS. Statistical analyses for fNIRS follow the same general principles as fMRI. We specified within-subject, first-level GLMs to model fNIRS optical density measurements in all the channels that were statistically related to the timing of the hemodynamic responses (as determined by convolving timeseries of stimulus events with the canonical HRF). In fNIRS, systemic physiology and motion-induced artifacts are major sources of noise and false positives. We therefore fit our models using autoregressive-whitened robust regression [13], which controls for such confounds and affords optimal parameter estimation. Then, we applied t-tests to the regression coefficients describing the task-related brain activations modeled for every participant. We additionally separated tasks into three difficulty levels and constructed GLMs to analyze the effect of task difficulty on neural activity.

A.3 Contrasts and Group-Level Analysis

fMRI. Following first-level model estimation, we computed pairwise contrasts to determine mean differences in activity between conditions. These were estimated on a within-participant basis (i.e., on first-level models). We applied a 5 mm^3 full-width at half maximum (FWHM) Gaussian smoothing kernel to each contrast map and carried them upward into group-level *random effects* analyses. A GLM in this context allows us to assess average activity across *all* participants, accounting for inter-individual variance to make some population-level inference. The end result is a *statistical parametric map* of t-values describing clusters of significant activity for a given task-related comparison. Importantly, all models and tests described here were done *voxelwise*—that is, a GLM was specified and estimated for each of nearly 73,000 voxels in brainspace. We therefore applied a *false discovery rate* (FDR) threshold at $q < 0.05$ to control for false positives as a result of multiple comparisons.

[5] The degree to which a refractive medium retards transmitted rays of light.

fNIRS. As with the fMRI analysis, we computed pairwise contrasts to determine mean differences in activity between conditions, estimated on a within-participant basis. Next, we conducted a group-level analysis to summarize the first-level regression coefficients. A mixed effects model was used to examine the average group-level response, with individual participants treated as random effects. Finally, we applied an FDR threshold at $q < 0.05$ to control for false positives from multiple comparisons.

A.4 Gaussian Process Classification

Multivariate Pattern Analyses. We used GPC to determine the extent to which code and prose tasks elicited similar patterns of brain activity. If code and prose are processed using highly overlapping brain systems, classifier accuracy would be low, reflecting entangled patterns of activity. These so-called *multivariate pattern analyses* were implemented in Matlab using the Gaussian Processes for Machine Learning software, v3.5 [122]. Classification is performed in a two-step procedure: the machine is first trained to identify patterns of activity corresponding to two stimulus types (code or prose), and learning performance is then tested using new images without class labels.

Inputs and Feature Selection. The extremely large dimension of fMRI data is a major obstacle for machine learning—we commonly have tens of thousands of voxels but only a few dozen training examples. This can be solved using a simple linear map (a *kernel function*) that reduces the dimensionality of the feature space [87, 130, 139]. To begin, training inputs (*features*) were given as a set of vectors $\{\mathbf{x}_n\}_{n=1}^N$, with corresponding binary (+1/−1) class labels, $\{y_n\}_{n=1}^N$ (where N is the number of beta images for a given participant across both classes). Because any given beta image is a 3D matrix of voxels, we can easily reshape it into an input vector, $\mathbf{x}_n$. The dimensionality of the feature vector is equal to the number of voxels used for pattern identification: for these analyses, we reduced the feature set to 47,187 voxels contained across 90 regions of the cerebrum, defined by the Automated Anatomical Labeling (AAL) atlas [161]. The AAL atlas allowed us to probe whole brain patterns across the same voxels for all participants. For additional feature reduction, we computed a simple $N \times N$ linear kernel whose elements indicated the degree of similarity between all pairs of input images.

GPC. Gaussian Processes treat the classification problem as an extension of the multivariate Gaussian, defined by a covariance function that is used to make predictions for new data (conditioned on a training set). We elected to use GPC over other common methods (e.g., the support vector machine) for several reasons: (1) predictions are made by integrating over probability distributions (vs. hard linear decisions); (2) model hyperparameters and regularization terms are learned directly from the data (vs. costly nested cross-validation routines); and (3) maximum likelihood is robust to potential imbalances in class size, which otherwise bias linear classifiers toward predicting the more common class. GPs have also been successfully used in previous neuroimaging work to decode distinct cognitive states (as we do here), to distinguish healthy individuals from clinical populations, and even to predict subjective experiences of pain [26, 99, 133].

The technical minutiae of GPC analysis have been described in detail previously [86, 122]. Prior to training, a GP is defined entirely by its mean vector, μ, and covariance function, $\mathbf{K}$. The covariance function is parameterized as

$$\mathbf{K} = \frac{1}{l^2}\mathbf{X}\mathbf{X}^T,$$

where l^2 is a learned scaling parameter and $\mathbf{X}\mathbf{X}^T$ gives the linear kernel. The goal of GP-based machine learning is then to identify optimal covariance parameters that allow for accurate predictions of new data. However, because binary classification is by nature non-Gaussian (all $y \in \{+1, -1\}$), we adopt a *function space* view of GPs that models a latent distribution over functions, $f(\mathbf{x})$, given the data, $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$. This distribution is used to estimate relationships between the training data and make predictions for new examples.

To learn such a mapping, we employ a cumulative Gaussian (or *probit*) likelihood and specify the posterior conditional over $\mathbf{f}$ using Bayes' rule:

$$p(\mathbf{f}|\mathcal{D}, \theta) = \frac{\mathcal{N}(\mathbf{f}|0, \mathbf{K})}{p(\mathcal{D}|\theta)} \prod_{n=1}^{N} \phi(y_n f_n),$$

where $\mathcal{N}(\mathbf{f}|0, \mathbf{K})$ is a zero-mean prior, $\phi(y_n f_n)$ is a factorization of the likelihood over training examples, and $p(\mathcal{D}|\theta)$ gives the model evidence (or the marginal likelihood of the data given a vector of hyperparameters, θ). Training therefore involves finding the optimal form of $\mathbf{K}$ by scaling model hyperparameters and maximizing the (log) model evidence.

Expectation Propagation. Class predictions for new images were made using *expectation propagation* (EP). This was necessary because the probit likelihood and the posterior are both non-Gaussian, making exact inference analytically intractable. EP algorithms allow us to reformulate the posterior as a Gaussian and approximate the distribution of the latent function at a new test point, $\mathbf{x}_*$:

$$\begin{aligned} p(y_* = +1|\mathcal{D}, \theta, \mathbf{x}_*) &= \int \theta(f_*) q(f_*|\mathcal{D}, \theta, \mathbf{x}_*) df_* \\ &= \phi\left(\frac{\mu_*}{\sqrt{1 + \sigma_*^2}}\right), \end{aligned}$$

where $q(f_*|\mathcal{D}, \theta, \mathbf{x}_*)$ gives the EP approximation to a Gaussian. Importantly, we still obtain a true probabilistic inference by integrating over the latent posterior. The obtained class probability is converted to a binary class label by inverting the logarithm:

$$t_* = e^p \begin{cases} t_* > 0.50, & y_* = +1 \\ t_* \leq 0.50, & y_* = -1. \end{cases}$$

The 0.50 threshold is non-arbitrary, owed to the symmetry of the cumulative Gaussian.

Testing and Training. We mitigated overfitting via careful cross validation and estimated unbiased measures of classification performance. Together, these offered a robust means of testing the extent to which GPC could distinguish between code and prose-related patterns of activity. Ultimately three binary GPC models were trained and tested for each participant: Code Review vs. Prose Review, Code Comprehension vs. Prose Review, and Code Review vs. Code Comprehension. Predictive performance was assessed using a leave-one-run-out cross-validation (LORO-CV) procedure. For each fold of LORO-CV, the data from one scanning run were removed from the kernel. The kernel was then centered according to the remaining training examples, the model was fit, and class predictions were made for the left-out data. Given that all participants did not necessarily have equal numbers of code/prose examples, average performance across all CV folds was estimated as the balanced accuracy (BAC), or the arithmetic mean of the two class accuracies.

Regional Inference. We next sought to determine which regions of the brain were most involved in discriminating between code and prose. This involved projecting kernel weights back onto the 3D brain—for display purposes, we present weight maps that were averaged across CV folds and participants. It is worth emphasizing, however, that such multivariate maps do not lend themselves to simple regional inference: because the final classification decision depends on information across *all* voxels, it is incorrect to assume voxels with high weight are the "most important." Nevertheless, we may estimate a posteriori the total contribution of each anatomical area in the aforementioned AAL atlas [161].

In this procedure, the absolute values of all voxel weights within a brain region were summed and divided by the total number of voxels in the region. Then, each region's "contribution strength" was divided by the sum of strengths for all regions, yielding a proportion that is directly interpretable as regional importance—a larger value indicates more total weight represented within a region [132]. These importance maps are also presented as a group average.

ACKNOWLEDGMENTS

We gratefully acknowledge the contribution of our colleagues, particularly the co-authors of the conference papers on which this article is based: Benjamin Floyd, Xiaosu Hu, Ryan Krueger, Xinyu Liu, and Tyler Santander all made aspects of this work possible, directly or indirectly. We are also thankful to Stephanie Forrest for comments, guidance, and extensive feedback on earlier drafts. Finally, the authors would like to thank the anonymous reviewers for their insightful comments and suggestions.

REFERENCES

[1] Rasmus Aamand, Thomas Dalsgaard, Yi-Ching Lynn Ho, Arne Moller, Andreas Roepstorff, and Torben Lund. 2013. A NO way to BOLD?: Dietary nitrate alters the hemodynamic response to visual stimulation. *NeuroImage* 83 (July 2013).

[2] Nahla J. Abid, Bonita Sharif, Natalia Dragan, Hend Alrasheed, and Jonathan I. Maletic. 2019. Developer reading behavior while summarizing java methods: Size and context matters. In *Proceedings of the 41st International Conference on Software Engineering*. IEEE Press, 384–395.

[3] Bianca P. Acevedo, Elaine N. Aron, Arthur Aron, Matthew-Donald Sangster, Nancy Collins, and Lucy L. Brown. 2014. The highly sensitive brain: An fMRI study of sensory processing sensitivity and response to others' emotions. *Brain and Behavior* 4, 4 (2014), 580–594.

[4] A. Frank Ackerman, Lynne S. Buchwald, and Frank H. Lewski. 1989. Software inspections: An effective verification process. *IEEE Software* 6, 3 (May 1989), 31–36. DOI: https://doi.org/10.1109/52.28121

[5] Dan Aharoni. 2000. Cogito, ergo sum! Cognitive processes of students dealing with data structures. *ACM SIGCSE Bulletin* 32, 1 (2000), 26–30.

[6] Dan Aharoni. 2000. What you see is what you get: The influence of visualization on the perception of data structures. In *PME International Conference*, Vol. 11. 1–8.

[7] Marwa Abdul-Monem Al-Shandawely. 2010. *Impacts of Data Structures and Algorithms on Multi-Core Efficiency*. Skolan för datavetenskap och kommunikation, Kungliga Tekniska högskolan, Leganes, Spain.

[8] Maizam Alias, Thomas R. Black, and David E. Gray. 2002. Effect of instruction on spatial visualization ability in civil engineering students. *International Education Journal* 3, 1 (2002), 51–71.

[9] Serkan Alkan and Kursat Cagiltay. 2007. Studying computer game learning experience through eye tracking. *British Journal of Educational Technology* 38, 3 (2007), 538–542.

[10] Andrea Arcuri and Lionel Briand. 2014. A hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering. *Software Testing, Verification and Reliability* 24, 3 (2014), 219–250.

[11] Kinnari Atit, Thomas F. Shipley, and Basil Tikoff. 2013. Twisting space: Are rigid and non-rigid mental transformations separate spatial skills? *Cognitive Processing* 14, 2 (2013), 163–173.

[12] Yonatan Aumann and Michael A. Bender. 1996. Fault tolerant data structures. In *Foundations of Computer Science*. IEEE, Burlington, VT, 580–589.

[13] Jeffrey W. Barker, Ardalan Aarabi, and Theodore J. Huppert. 2013. Autoregressive model based algorithm for correcting motion and serially correlated errors in fNIRS. *Biomedical Optics Express* 4, 8 (2013), 1366–1379.

[14] Laura Beckwith, Margaret Burnett, Susan Wiedenbeck, Curtis Cook, Shraddha Sorte, and Michelle Hastings. 2005. Effectiveness of end-user debugging software features: Are there gender issues? In *Proceedings of the 2005 SIGCHI Conference on Human Factors in Computing Systems (CHI'05)*. ACM, New York, NY, 869–878. DOI: https://doi.org/10.1145/1054972.1055094

[15] Roman Bednarik. 2012. Expertise-dependent visual attention strategies develop over time during debugging with multiple code representations. *International Journal of Human-Computer Studies* 70, 2 (Feb. 2012), 143–155. DOI: https://doi.org/10.1016/j.ijhcs.2011.09.003

[16] Craig M. Bennett, M. B. Miller, and G. L. Wolford. 2009. Neural correlates of interspecies perspective taking in the post-mortem atlantic salmon: An argument for multiple comparisons correction. *Neuroimage* 47, Suppl. 1 (2009), S125.

[17] Alice A. Black. 2005. Spatial ability and earth science conceptual understanding. *Journal of Geoscience Education* 53, 4 (2005), 402–414.

[18] David A. Boas, Clare E. Elwell, Marco Ferrari, and Gentaro Taga. 2014. Twenty years of functional near-infrared spectroscopy: Introduction for the special issue. *Neuroimage* (2014), 1–15. DOI: 10.1016/j.neuroimage.2013.11.033

[19] Ruven Brooks. 1977. Towards a theory of the cognitive processes in computer programming. *International Journal of Man-Machine Studies* 9, 6 (1977), 737–751.

[20] Raymond P. L. Buse and Thomas Zimmermann. 2012. Information needs for software development analytics. In *Proceedings of the 34th International Conference on Software Engineering (ICSE'12)*. IEEE Press, 987–996.

[21] Teresa Busjahn, Roman Bednarik, Andrew Begel, Martha Crosby, James H. Paterson, Carsten Schulte, Bonita Sharif, and Sascha Tamm. 2015. Eye movements in code reading: Relaxing the linear order. In *Proceedings of 22th International Conference on Program Comprehension (ICPC'15)*. IEEE, 255–265.

[22] Richard B. Buxton, Kâmil Uludağ, David J. Dubowitz, and Thomas T. Liu. 2004. Modeling the hemodynamic response to brain activation. *Neuroimage* 23 (2004), S220–S233.

[23] Susan Caminiti. 2018. AT&T's $1 Billion Gambit: Retraining Nearly Half its Workforce for Jobs of the Future. Retrieved February 2020 from https://www.cnbc.com/2018/03/13/atts-1-billion-gambit-retraining-nearly-half-its-workforce.html.

[24] João Castelhano, Isabel C. Duarte, Carlos Ferreira, João Duraes, Henrique Madeira, and Miguel Castelo-Branco. 2018. The role of the insula in intuitive expert bug detection in computer code: An fMRI study. *Brain Imaging and Behavior* (May 2018).

[25] Center for Diagnostic Imaging. 2016. I'm Getting an MRI, So What's a Coil? Retrieved February 2020 from https://www.mycdi.com/viewpoints/im_getting_an_mri_so_whats_a_coil_103.

[26] E. Challis, P. Hurley, L. Serra, M. Bozzali, S. Oliver, and M. Cercignani. 2015. Gaussian process classification of Alzheimer's disease and mild cognitive impairment from resting-state fMRI. *Neuroimage* 122 (2015), 232–243.

[27] Mark S. Cohen, Stephen M. Kosslyn, Hans C. Breiter, Gregory J. DiGirolamo, William L. Thompson, A. K. Anderson, S. Y. Bookheimer, Bruce R. Rosen, and J. W. Belliveau. 1996. Changes in cortical activity during mental rotation a mapping study using functional MRI. *Brain* 119, 1 (1996), 89–100.

[28] Michael C. Corballis. 1997. Mental rotation and the right hemisphere. *Brain and Language* 57, 1 (1997), 100–121.

[29] Igor Crk, Timothy Kluthe, and Andreas Stefik. 2016. Understanding programming expertise: An empirical study of phasic brain wave changes. *Transactions on Computer-Human Interaction* 23, 1 (2016), 2.

[30] Jody C. Culham and Nancy G. Kanwisher. 2001. Neuroimaging of cognitive functions in human parietal cortex. *Current Opinion in Neurobiology* 11, 2 (2001), 157–163.

[31] Chip Cutter. 2019. Amazon to retrain a third of its U.S. workforce. Retrieved February 2020 from https://www.wsj.com/articles/amazon-to-retrain-a-third-of-its-u-s-workforce-11562841120.

[32] D. Tomasi, L. Chang, E. Caparelli, and T. Ernst. 2007. Different activation patterns for working memory load and visual attention load. *Brain Research* 1132, 1 (2007), 158–165.

[33] Bert De Smedt, Daniel Ansari, Roland H. Grabner, Minna M. Hannula, Michael Schneider, and Lieven Verschaffel. 2010. Cognitive neuroscience meets mathematics education.*Educational Research Review* 5, 1 (2010), 97–105.

[34] Lionel E. Deimel, Jr. 1985. The uses of program reading. *SIGCSE Bulletin* 17, 2 (1985), 5–14. DOI: https://doi.org/10.1145/382204.382524

[35] Fabian Deitelhoff, Andreas Harrer, and Andrea Kienle. 2019. The influence of different AOI models in source code comprehension analysis. In *Proceedings of the 6th International Workshop on Eye Movements in Programming (EMIP'19)*. IEEE Press, 10–17. DOI: https://doi.org/10.1109/EMIP.2019.00010

[36] Nicola Dell, Vidya Vaidyanathan, Indrani Medhi, Edward Cutrell, and William Thies. 2012. "Yours is better!": Participant response bias in HCI. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'12)*. Association for Computing Machinery, New York, NY, 1321–1330. DOI: https://doi.org/10.1145/2207676.2208589

[37] Jörn Diedrichsen and Reza Shadmehr. 2005. Detecting and adjusting for artifacts in fMRI time series data. *NeuroImage* 27, 3 (2005), 624–634. DOI: https://doi.org/10.1016/j.neuroimage.2005.04.039

[38] Tyrone Donnon, Jean-Gaston DesCôteaux, and Claudio Violato. 2005. Impact of cognitive imaging and sex differences on the development of laparoscopic suturing skills. *Canadian Journal of Surgery* 48, 5 (2005), 387.

[39] Alastair Dunsmore, Marc Roper, and Murray Wood. 2003. Practical code inspection techniques for object-oriented systems: An experimental comparison. *IEEE Software* 20, 4 (July 2003), 21–29. DOI: https://doi.org/10.1109/MS.2003.1207450

[40] Lyn Dupré. 1995. *Bugs in Writing, a Guide to Debugging Your Prose.* ACM Press/Addison-Wesley Publishing Co..

[41] J. Duraes, H. Madeira, J. Castelhano, C. Duarte, and M. C. Branco. 2016. WAP: Understanding the brain at software debugging. In *International Symposium on Software Reliability Engineering*. IEEE, 87–92.

[42] Ann-Christine Ehlis, Sabrina Schneider, Thomas Dresler, and Andreas J. Fallgatter. 2014. Application of functional near-infrared spectroscopy in psychiatry. *Neuroimage* 85 (2014), 478–488.

[43] Anders Eklund, Thomas E. Nichols, and Hans Knutsson. 2016. Cluster failure: Why fMRI inferences for spatial extent have inflated false-positive rates. *Proceedings of the National Academy of Sciences* 113, 28 (2016), 7900–7905. DOI: https://doi.org/10.1073/pnas.1602413113 arXiv:http://www.pnas.org/content/113/28/7900.full.pdf.

[44] James L. Elshoff and Michael Marcotty. 1982. Improving computer program readability to aid modification. *Communications of the ACM* 25, 8 (1982), 512–521. DOI: https://doi.org/10.1145/358589.358596

[45] Anneli Eteläpelto. 1993. Metacognition and the expertise of computer program comprehension. *Scandinavian Journal of Educational Research* 37, 3 (1993), 243–254.

[46] M. E. Fagan. 1999. Design and code inspections to reduce errors in program development. *IBM Systems Journal* 38, 2-3 (June 1999), 258–287. DOI: https://doi.org/10.1147/sj.382.0258

[47] Sarah Fakhoury. 2018. Moving towards objective measures of program comprehension. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE'18)*. Association for Computing Machinery, New York, NY, 936–939. DOI: https://doi.org/10.1145/3236024.3275426

[48] Sarah Fakhoury, Yuzhan Ma, Venera Arnaoudova, and Olusola Adesope. 2018. The effect of poor source code lexicon and readability on developers' cognitive load. In *Proceedings of the 26th Conference on Program Comprehension (ICPC'18)*. Association for Computing Machinery, New York, NY, 286–296. DOI: https://doi.org/10.1145/3196321.3196347

[49] Quyin Fan. 2010. *The Effects of Beacons, Comments, and Tasks on Program Comprehension Process in Software Maintenance*. Ph.D. Dissertation. University of Maryland, Baltimore County, Catonsville, MD.

[50] Nuno Faria, Rui Silva, and Joao L. Sobral. 2013. Impact of data structure layout on performance. In *Proceedings of the 2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP'13)*. IEEE Computer Society, 116–120. DOI: https://doi.org/10.1109/PDP.2013.24

[51] Benjamin Floyd, Tyler Santander, and Westley Weimer. 2017. Decoding the representation of code in the brain: An FMRI study of code review and expertise. In *Proceedings of the 39th International Conference on Software Engineering (ICSE'17)*. IEEE Press, 175–186. DOI: https://doi.org/10.1109/ICSE.2017.24

[52] Denae Ford, Justin Smith, Philip J. Guo, and Chris Parnin. 2016. Paradise unplugged: Identifying barriers for female participation on stack overflow. *Foundations of Software Engineering*. 846–857. DOI: https://doi.org/10.1145/2950290.2950331

[53] Thomas Fritz, Andrew Begel, Sebastian C. Müller, Serap Yigit-Elliott, and Manuela Züger. 2014. Using psycho-physiological measures to assess task difficulty in software development. In *Proceedings of the 36th International Conference on Software Engineering (ICSE'14)*. Association for Computing Machinery, New York, NY, 402–413. DOI: https://doi.org/10.1145/2568225.2568266

[54] Zachary P. Fry, Bryan Landau, and Westley Weimer. 2012. A human study of patch maintainability. In *Proceedings of the 2012 International Symposium on Software Testing and Analysis (ISSTA'12)*. Association for Computing Machinery, New York, NY, 177–187. DOI: https://doi.org/10.1145/2338965.2336775

[55] William D. Gaillard, Bonnie C. Sachs, Joseph R. Whitnah, Zaaira Ahmad, Lyn M. Balsamo, Jeffrey R. Petrella, Suzanne H. Braniecki, Christopher M. McKinney, Kevin Hunter, Ben Xu, et al. 2003. Developmental aspects of language processing: fMRI of verbal fluency in children and adults. *Human Brain Mapping* 18, 3 (2003), 176–185.

[56] Gary H. Glover. 2011. Overview of functional magnetic resonance imaging. *Neurosurgery Clinics* 22, 2 (2011), 133–139.

[57] Andrea Gogos, Maria Gavrilescu, Sonia Davison, Karissa Searle, Jenny Adams, Susan L Rossell, Robin Bell, Susan R. Davis, and Gary F. Egan. 2010. Greater superior than inferior parietal lobule activation with increasing rotation angle during mental rotation: An fMRI study. *Neuropsychologia* 48, 2 (2010), 529–535.

[58] Joseph H. Goldberg and Jonathan I. Helfman. 2010. Comparing information graphics: A critical look at eye tracking. In *Proceedings of the 3rd BELIV'10 Workshop: BEyond Time and Errors: Novel evaLuation Methods for Information Visualization (BELIV'10)*. ACM, New York, NY, 71–78. DOI: https://doi.org/10.1145/2110192.2110203

[59] J. H. Goldberg and X. P. Kotval. 1999. Computer interface evaluation using eye movements: Methods and constructs. *International Journal of Industrial Ergonomics* 24, 6 (1999), 631–645.

[60] V. Grigoreanu, J. Brundage, E. Bahna, M. Burnett, P. ElRif, and J. Snover. 2009. Males and females script debugging strategies. *End-User Development* (2009), 205–224.

[61] Wayne A. Grove, Tim Wasserman, and Andrew Grodner. 2006. Choosing a proxy for academic aptitude. *Journal of Economic Education* (2006), 131–147.

[62] Peter Hallam. 2016. *What Do Programmers Really Do Anyway?* Technical Report. Microsoft Developer Network.

[63] Nuzhat J. Haneef. 1998. Software documentation and readability: A proposed process improvement. *SIGSOFT Software Engineering Notes* 23, 3 (1998), 75–77. DOI: https://doi.org/10.1145/279437.279470

[64] Michael Hanke, Nico Adelhöfer, Daniel Kottke, Vittorio Iacovella, Ayan Sengupta, Falko R. Kaule, Roland Nigbur, Alexander Q. Waite, Florian J. Baumgartner, and Jörg Stadler. 2016. Simultaneous fMRI and eye gaze recordings during prolonged natural stimulation-a studyforrest extension. *BioRxiv* (2016), 046581.

[65] Irina M. Harris, Gary F. Egan, Cynon Sonkkila, Henri J. Tochon-Danguy, George Paxinos, and John D. G. Watson. 2000. Selective right parietal lobe activation during mental rotation: A parametric PET study. *Brain* 123, 1 (2000), 65–73.

[66] Mary Hegarty and Maria Kozhevnikov. 1999. Types of visual–spatial representations and mathematical problem solving. *Journal of Educational Psychology* 91, 4 (1999), 684.

[67] Rik N. A. Henson, Cathy J. Price, Michael D. Rugg, Robert Turner, and Karl J. Friston. 2002. Detecting latency differences in event-related BOLD responses: Application to words versus nonwords and initial versus repeated face presentations. *Neuroimage* 15, 1 (2002), 83–97.

[68] Frouke Hermens, Rhona Flin, and Irfan Ahmed. 2013. Eye movements in surgery: A literature review. *Journal of Eye Movement Research* 6, 4 (Nov. 2013). DOI: https://doi.org/10.16910/jemr.6.4.4

[69] Abram Hindle, Earl T. Barr, Zhendong Su, Mark Gabel, and Premkumar Devanbu. 2012. On the naturalness of software. In *Proceedings of the 34th International Conference on Software Engineering (ICSE'12).* IEEE Press, 837–847.

[70] Yu Huang, Kevin Leach, Zohreh Sharafi, Nicholas McKay, Tyler Santander, and Westley Weimer. 2020. Investigating gender bias and differences in code review: Using medical imaging and eye-tracking. In *International Symposium on the Foundations of Software Engineering (ESEC/FSE'20).* ACM/SIGSOFT.

[71] Yu Huang, Xinyu Liu, Ryan Krueger, Tyler Santander, Xiaosu Hu, Kevin Leach, and Westley Weimer. 2019. Distilling neural representations of data structure manipulation using FMRI and FNIRS. In *Proceedings of the 41st International Conference on Software Engineering (ICSE'19).* IEEE Press, 396–407. DOI: https://doi.org/10.1109/ICSE.2019.00053

[72] Dorota Huizinga and Adam Kolawa. 2007. *Automated Defect Prevention: Best Practices in Software Management* (1st ed.). Wiley.

[73] Theodore J. Huppert. 2016. Commentary on the statistical properties of noise and its implication on general linear models in functional near-infrared spectroscopy. *Neurophotonics* 3 (March 2016), 010401.

[74] Yoshiharu Ikutani and Hidetake Uwano. 2014. Brain activity measurement during program comprehension with NIRS. *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*. IEEE, 1–6.

[75] Robert J. K. Jacob and Keith S. Karn. 2003. Eye tracking in human-computer interaction and usability research: Ready to deliver the promises. *Mind* 2, 3 (2003), 4.

[76] Shawna N. Jacob, Paula K. Shear, Matthew Norris, Matthew Smith, Jeff Osterhage, Stephen M. Strakowski, Michael Cerullo, David E. Fleck, Jing-Huei Lee, and James C. Eliassen. 2015. Impact of functional magnetic resonance imaging (fMRI) scanner noise on affective state and attentional performance. *Journal of Clinical and Experimental Neuropsychology* 37, 6 (2015), 563–570.

[77] Halszka Jarodzka, Kenneth Holmqvist, and Marcus Nyström. 2010. A vector-based, multidimensional scanpath similarity measure. In *Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications (ETRA'10).* ACM, New York, NY, 211–218. DOI: https://doi.org/10.1145/1743666.1743718

[78] Marcel A. Just and Patricia A. Carpenter. 1980. A theory of reading: From eye fixations to comprehension. *Psychological Review* 87, 4 (1980), 329.

[79] Niall Kennedy. 2006. Google Mondrian: Web-based code review and storage. Retrieved February 2020 from https://www.niallkennedy.com/blog/2006/11/google-mondrian.html.

[80] K. Kevic, B. M. Walters, T. R. Shaffer, B. Sharif, D. C. Shepherd, and T. Fritz. 2017. Eye gaze and interaction contexts for change tasks observations and potential. *Journal of System Software* 128, C (June 2017), 252–266.

[81] Seong-Gi Kim and Seiji Ogawa. 2012. Biophysical and physiological origins of blood oxygenation level-dependent fMRI signals. *Journal of Cerebral Blood Flow & Metabolism* 32, 7 (2012), 1188–1206.

[82] Alan Kingstone, Daniel Smilek, and John D. Eastwood. 2008. Cognitive ethology: A new approach for studying human cognition. *British Journal of Psychology* 99, 3 (2008), 317–340.

[83] John C. Knight and E. Ann Myers. 1993. An improved inspection technique. *Communications of the ACM* 36, 11 (Nov. 1993), 51–61. DOI: https://doi.org/10.1145/163359.163366

[84] Donald E. Knuth. 1984. Literate programming. *Computer Journal* 27, 2 (1984), 97–111.

[85] Ryan Krueger, Yu Huang, Xinyu Liu, Tyler Santander, Westley, and Kevin Leach. 2020. Neurological divide: An fMRI study of prose and code writing. In *Proceedings of the 2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE'20).* IEEE Press, 13.

[86] M. Kuss and C. E. Rasmussen. 2005. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research* 6 (2005), 1679–1704.

[87] S. LaConte, S. Strother, V. Cherkassky, and X. Hu. 2005. Support vector machines for temporal classification of block design fMRI data. *Neuroimage* 26 (2005), 317–329.

[88] Claire Le Goues and Westley Weimer. 2012. Measuring code quality to improve specification mining. *IEEE Transactions on Software Engineering* 38, 1 (2012), 175–190.

[89] Seolhwa Lee, Danial Hooshyar, Hyesung Ji, Kichun Nam, and Heuiseok Lim. 2017. Mining biometric data to predict programmer expertise and task difficulty. *Cluster Computing* (2017), 1–11.

[90] S. Lee, A. Matteson, D. Hooshyar, S. Kim, J. Jung, G. Nam, and H. Lim. 2016. Comparing programming language comprehension between novice and expert programmers using EEG analysis. In *Proceedings of the International Conference on Bioinformatics and Bioengineering*. IEEE, 350–355. DOI: https://doi.org/10.1109/BIBE.2016.30

[91] Daniel Richard Leff, Felipe Orihuela-Espina, Clare E. Elwell, Thanos Athanasiou, David T. Delpy, Ara W. Darzi, and Guang-Zhong Yang. 2011. Assessment of the cerebral cortex during motor task behaviours in adults: A systematic review of functional near infrared spectroscopy (fNIRS) studies. *Neuroimage* 54, 4 (2011), 2922–2936.

[92] Martin A. Lindquist, Ji Meng Loh, Lauren Y. Atlas, and Tor D. Wager. 2009. Modeling the hemodynamic response function in fMRI: Efficiency, bias and mis-modeling. *Neuroimage* 45, 1 (2009), S187–S198.

[93] Marcia C. Linn and Anne C. Petersen. 1985. Emergence and characterization of sex differences in spatial ability: A meta-analysis. *Child Development* (1985), 1479–1498.

[94] Jia Liu, Alison Harris, and Nancy Kanwisher. 2010. Perception of face parts and face configurations: An fMRI study. *Journal of Cognitive Neuroscience* 22, 1 (2010), 203–211.

[95] Sarah Lloyd-Fox, Anna Blasi, and C. E. Elwell. 2010. Illuminating the developing brain: The past, present and future of functional near infrared spectroscopy. *Neuroscience & Biobehavioral Reviews* 34, 3 (2010), 269–284.

[96] Paul A. Mabe and Stephen G. West. 1982. Validity of self-evaluation of ability: A review and meta-analysis. *Journal of Applied Psychology* 67, 3 (1982), 280.

[97] E. Maguire, D. Gadian, I. Johnsrude, Catriona Good, John Ashburner, Richard Frackowiak, and Christopher Frith. 2000. Navigation-related structural change in the hippocampi of taxi drivers. *Proceedings of the National Academy of Sciences of the United States of America* 97, 8 (2000), 4398–4403.

[98] J. B. Antoine Maintz and Max A. Viergever. 1998. A survey of medical image registration. *Medical Image Analysis* 2, 1 (1998), 1–36.

[99] A. Marquand, M. Howard, M. Brammer, C. Chu, S. Coen, and J. Mourao-Miranda. 2010. Quantitative prediction of subjective pain intensity from whole-brain fMRI data using Gaussian processes. *Neuroimage* 49 (2010), 2178–2189.

[100] Pawel J. Matusz, Suzanne Dikker, Alexander G. Huth, and Catherine Perrodin. 2019. Are we ready for real-world neuroscience?

[101] A. Mazard, B. Mazoyer, O. Etard, N. Tzourio-Mazoyer, S. M. Kosslyn, and Emmanuel Mellet. 2002. Impact of fMRI acoustic noise on the functional anatomy of visual mental imagery. *Journal of Cognitive Neuroscience* 14, 2 (2002), 172–186.

[102] Scott D. Moffat, Elizabeth Hampson, and Maria Hatzipantelis. 1998. Navigation in a "virtual" maze: Sex differences and correlation with psychometric measures of spatial ability in humans. *Evolution and Human Behavior* 19, 2 (1998), 73–87.

[103] Martin M. Monti, Audrey Vanhaudenhuyse, Martin R. Coleman, Melanie Boly, John D. Pickard, Luaba Tshibanda, Adrian M. Owen, and Steven Laureys. 2010. Willful modulation of brain activity in disorders of consciousness. *New England Journal of Medicine* 362, 7 (2010), 579–589.

[104] Takao Nakagawa, Yasutaka Kamei, Hidetake Uwano, Akito Monden, Kenichi Matsumoto, and Daniel M. German. 2014. Quantifying programmers' mental workload during program comprehension based on cerebral blood flow measurement: A controlled experiment. In *Companion Proceedings of the 36th International Conference on Software Engineering (ICSE Companion'14)*. Association for Computing Machinery, New York, NYA, 448–451. DOI: https://doi.org/10.1145/2591062.2591098

[105] NASA Software Reuse Working Group. 2005. Software reuse survey. Retrieved on February 2020 from http://www.esdswg.com/softwarereuse/Resources/library/working_group_documents/survey2005.

[106] Unaizah Obaidellah, Mohammed Al Haek, and Peter C.-H. Cheng. 2018. A survey on the usage of eye-tracking in computer programming. *ACM Computing Surveys* 51, 1 (Jan. 2018), Article 5, 58 pages. DOI: https://doi.org/10.1145/3145904

[107] Hellmuth Obrig. 2014. NIRS in clinical neurology A 'promising' tool? *Neuroimage* 85 (2014), 535–546.

[108] John P. O'Doherty, Alan Hampton, and Hackjin Kim. 2007. Model-based fMRI and its application to reward learning and decision making. *Annals of the New York Academy of Sciences* 1104, 1 (2007), 35–53.

[109] Seiji Ogawa, Tso-Ming Lee, Alan R. Kay, and David W. Tank. 1990. Brain magnetic resonance imaging with contrast dependent on blood oxygenation. *Proceedings of the National Academy of Sciences* 87, 24 (1990), 9868–9872.

[110] Masako Okamoto, Mari Matsunami, Haruka Dan, Tomoko Kohata, Kaoru Kohyama, and Ippeita Dan. 2006. Prefrontal activity during taste encoding: An fNIRS study. *Neuroimage* 31, 2 (2006), 796–806.

[111] Paul Oman and Jack Hagemeister. 1992. Metrics for assessing a software system's maintainability. In *Proceedings of Conference on Software Maintenance*. IEEE, 337–344.

[112] Christopher Palmer and Bonita Sharif. 2016. Towards automating fixation correction for source code. In *Proceedings of the 9th Biennial ACM Symposium on Eye Tracking Research & Applications (ETRA'16)*. Association for Computing Machinery, New York, NY, 65–68. DOI: https://doi.org/10.1145/2857491.2857544

[113] Chris Parnin. 2010. A cognitive neuroscience perspective on memory for programming tasks. *Programming Interest Group* (2010), 27.

[114] Chris Parnin. 2011. Subvocalization—Toward hearing the inner thoughts of developers. In *ICPC*. IEEE Computer Society, 197–200. http://dblp.uni-trier.de/db/conf/iwpc/icpc2011.html#Parnin11.

[115] Roy D. Pea and D. Midian Kurland. 1984. On the cognitive effects of learning computer programming. *New Ideas in Psychology* 2, 2 (1984), 137–168.

[116] Norman Peitek, Janet Siegmund, Chris Parnin, Sven Apel, Johannes C. Hofmeister, and André Brechmann. 2018. Simultaneous measurement of program comprehension with FMRI and eye tracking: A case study. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM'18)*. Association for Computing Machinery, New York, NY, Article 24, 10 pages. DOI: https://doi.org/10.1145/3239235.3240495

[117] Michael Peters and Christian Battista. 2008. Applications of mental rotation figures of the Shepard and Metzler type and description of a mental rotation stimulus library. *Brain and Cognition* 66, 3 (2008), 260–264.

[118] Laura-Ann Petitto, Melody S. Berens, Ioulia Kovelman, Matt H. Dubins, K. Jasinska, and M. Shalinsky. 2012. The "perceptual wedge hypothesis" as the basis for bilingual babies' phonetic processing advantage: New insights from fNIRS brain imaging. *Brain and Language* 121, 2 (2012), 130–143.

[119] Susan J. Pickering and Paul Howard-Jones. 2007. Educators' views on the role of neuroscience in education: Findings from a study of UK and international perspectives. *Mind, Brain, and Education* 1, 3 (2007), 109–113.

[120] Philip M. Podsakoff and Dennis W. Organ. 1986. Self-reports in organizational research: Problems and prospects. *Journal of Management* 12, 4 (1986), 531–544.

[121] Alex Poole and Linden J. Ball. 2005. Eye tracking in human-computer interaction and usability research: Current status and future. In *Prospects",* C. Ghaoui (Ed.), Encyclopedia of Human-Computer Interaction. Idea Group, Inc., Pennsylvania. Information Science Reference - Imprint of IGI Publishing, Hershey, PA, 1–5.

[122] Carl Edward Rasmussen and Hannes Nickisch. 2010. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research* 11 (Dec. 2010), 3011–3015.

[123] Darrell R. Raymond. 1991. Reading source code. In *Proceedings of the 1991 Conference of the Centre for Advanced Studies on Collaborative Research (CASCON'91)*. IBM Press, 3–16.

[124] K. Rayner. 1978. Eye movements in reading and information processing. *Psychological Bulletin* 85, 3 (1978), 618–660.

[125] Fabio Richlan, Benjamin Gagl, Stefan Hawelka, Mario Braun, Matthias Schurz, Martin Kronbichler, and Florian Hutzler. 2013. Fixation-related fMRI analysis in the domain of reading research: Using self-paced eye movements as markers for hemodynamic brain responses during visual letter string processing. *Cerebral Cortex* 24, 10 (May), 2647–2656. DOI: https://doi.org/10.1093/cercor/bht117 arXiv:http://oup.prod.sis.lan/cercor/article-pdf/24/10/2647/868083/bht117.pdf.

[126] Shannon Riley. 2006. Password security: What users know and what they actually do. *Usability News* 8, 1 (2006), 2833–2836.

[127] Spencer Rugaber. 2000. The use of domain knowledge in program understanding. *Annals of Software Engineering* 9, 1–4 (2000), 143–192.

[128] Dario D. Salvucci and Joseph H. Goldberg. 2000. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the 2000 Symposium on Eye Tracking Research & Applications (ETRA'00)*. ACM, New York, NY, 71–78. DOI: https://doi.org/10.1145/355017.355028

[129] Hanan Samet. 1990. *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

[130] Bernhard Scholkopf and Alexander J. Smola. 2001. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA.

[131] J. Scholz, M. Klein, T. Behrens, and H. Johansen-Berg. 2009. Training induces changes in white matter architecture. *Nature Neuroscience* 12, 11 (2009), 1370–1371.

[132] J. Schrouff, J. Cremers, G. Garraux, L. Baldassarre, J. Mourão Miranda, and C. Phillips. 2013. Localizing and comparing weight maps generated from linear kernel machine learning models. In *Proceedings of the 2013 International Workshop on Pattern Recognition in Neuroimaging (PRNI'13)*. IEEE Computer Society, 124–127. DOI: https://doi.org/10.1109/PRNI.2013.40

[133] Jessica Schrouff, Caroline Kussé, Louis Wehenkel, Pierre Maquet, and Christophe Phillips. 2012. Decoding semi-constrained brain activity from fMRI using support vector machines and Gaussian processes. *PLoS One* 7, 4 (04 2012), 1–11. DOI: https://doi.org/10.1371/journal.pone.0035860

[134] Scicurious. 2012. IgNobel Prize in Neuroscience: The Dead Salmon Study. *Scientific American Blog Network* (Sept. 2012). Retrieved on February 2020 from https://blogs.scientificamerican.com/scicurious-brain/ignobel-prize-in-neuroscience-the-dead-salmon-study/.

[135] Timothy R. Shaffer, Jenna L. Wise, Braden M. Walters, Sebastian C. Müller, Michael Falcone, and Bonita Sharif. 2015. iTrace: Enabling eye tracking on software artifacts within the IDE to support software engineering tasks. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*. ACM, 954–957.

[136] Zohreh Sharafi, Timothy Shaffer, Bonita Sharif, and Yann-Gaël Guéhéneuc. 2015. Eye-tracking metrics in software engineering. In *2015 Asia-Pacific Software Engineering Conference (APSEC'15)*. IEEE, 96–103.

[137] Zohreh Sharafi, Bonita Sharif, Yann-Gaël Guéhéneuc, Andrew Begel, Roman Bednarik, and Martha Crosby. 2020. A practical guide on conducting eye tracking studies in software engineering. *Empirical Software Engineering* (2020), 1–47.

[138] Zohreh Sharafi, Zéphyrin Soh, and Yann-Gaël Guéhéneuc. 2015. A systematic literature review on the usage of eye-tracking in software engineering. *Information and Software Technololgy* 67, C (Nov. 2015), 79–107. DOI: https://doi.org/10.1016/j.infsof.2015.06.008

[139] John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

[140] Roger N. Shepard and Jacqueline Metzler. 1971. Mental rotation of three-dimensional objects. *Science* 171, 3972 (1971), 701–703.

[141] Forrest Shull, Ioana Rus, and Victor Basili. 2001. Improving software inspections by using reading techniques. In *Proceedings of the 23rd International Conference on Software Engineering (ICSE'01)*. IEEE Computer Society, 726–727.

[142] Janet Siegmund, Christian Kästner, Sven Apel, Chris Parnin, Anja Bethmann, Thomas Leich, Gunter Saake, and André Brechmann. 2014. Understanding understanding source code with functional magnetic resonance imaging. In *Proceedings of the 36th International Conference on Software Engineering (ICSE'14)*. Association for Computing Machinery, New York, NY, 378–389. DOI: https://doi.org/10.1145/2568225.2568252

[143] Janet Siegmund, Norman Peitek, Chris Parnin, Sven Apel, Johannes Hofmeister, Christian Kästner, Andrew Begel, Anja Bethmann, and André Brechmann. 2017. Measuring neural efficiency of program comprehension. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering (ESEC/FSE'17)*. Association for Computing Machinery, New York, NY, 140–150. DOI: https://doi.org/10.1145/3106237.3106268

[144] Jonathan Sillito, Gail C. Murphy, and Kris De Volder. 2006. Questions programmers ask during software evolution tasks. In *Proceedings of the 14th ACM SIGSOFT International Symposium on Foundations of Software Engineering (SIGSOFT'06/FSE-14)*. Association for Computing Machinery, New York, NY, 23–34. DOI: https://doi.org/10.1145/1181775.1181779

[145] Kerri Smith. 2012. fMRI 2.0: Functional magnetic resonance imaging is growing from showy adolescence into a workhorse of brain imaging. *Nature* 484, 7392 (2012), 24–27.

[146] Stephen M. Smith, Peter T. Fox, Karla L. Miller, David C. Glahn, P. Mickle Fox, Clare E. Mackay, Nicola Filippini, Kate E. Watkins, Roberto Toro, Angela R. Laird, et al. 2009. Correspondence of the brain's functional architecture during activation and rest. *Proceedings of the National Academy of Sciences* 106, 31 (2009), 13040–13045.

[147] Andrea Solimeno, Minou Ella Mebane, Manuela Tomai, and Donata Francescato. 2008. The influence of students and teachers characteristics on the efficacy of face-to-face and computer supported collaborative learning. *Computers & Education* 51, 1 (2008), 109–128. DOI: https://doi.org/10.1016/j.compedu.2007.04.003

[148] Ian Sommerville. 2010. *Software Engineering* (9th ed.). Vol. 137035152. Pearson.

[149] Catherine J. Stoodley, Eve M. Valera, and Jeremy D. Schmahmann. 2012. Functional topography of the cerebellum for motor and cognitive tasks: An fMRI study. *Neuroimage* 59, 2 (2012), 1560–1570.

[150] Robert E. Strom and Shaula Yemini. 1986. Typestate: A programming language concept for enhancing software reliability. *IEEE Transactions on Software Engineering* SE-12, 1 (1986), 157–171.

[151] Veronica Sundstedt. 2010. Gazing at games: Using eye tracking to control virtual characters. In *ACM SIGGRAPH 2010 Courses (SIGGRAPH'10)*. ACM, New York, NY, Article 5, 160 pages. DOI: https://doi.org/10.1145/1837101.1837106

[152] Sungho Tak and Jong Chul Ye. 2014. Statistical analysis of fNIRS data: A comprehensive review. *NeuroImage* 85 (2014), 72–91. DOI: https://doi.org/10.1016/j.neuroimage.2013.06.016 Celebrating 20 Years of Functional Near Infrared Spectroscopy (fNIRS).

[153] Josh Terrell, Andrew Kofink, Justin Middleton, Clarissa Rainear, Emerson Murphy-Hill, Chris Parnin, and Jon Stallings. 2017. Gender differences and bias in open source: Pull request acceptance of women versus men. *PeerJ Computer Science* 3, e111 (2017).

[154] Zohreh Sharafi, Zéphyrin Soh, Yann-Gaël Guéhéneuc, and Giuliano Antoniol. 2012. Women and men—Different but equal: On the impact of identifier style on source code reading. In *International Conference on Program Comprehension.*

[155] The College Board. 2016. *The Official SAT Study Guide (Redesigned SAT)*. College Board.

[156] Vadim Tkachenko. 2016. How Three Fundamental Data Structures Impact Storage and Retrieval. Retrieved February 2020 from https://dzone.com/articles/how-three-fundamental-data-structures-impact-stora.

[157] Dyanne M. Tracy. 1987. Toys, spatial ability, and science and mathematics achievement: Are they related? *Sex Roles* 17, 3–4 (1987), 115–138.

[158] C. Triantafyllou, R. D. Hoge, G. Krueger, C. J. Wiggins, A. Potthast, G. C. Wiggins, and L. L. Wald. 2005. Comparison of physiological noise at 1.5 T, 3 T and 7 T and optimization of fMRI acquisition parameters. *Neuroimage* 26, 1 (2005), 243–250.

[159] Alexia Tsotsis. 2011. Meet Phabricator, the Witty Code Review Tool Built Inside Facebook. Retrieved February 2020 from https://techcrunch.com/2011/08/07/oh-what-noble-scribe-hath-penned-these-words/.

[160] Kristian Tylén, Micah Allen, Bjørg Kaae Hunter, and Andreas Roepstorff. 2012. Interaction vs. observation: Distinctive modes of social cognition in human brain and behavior? A combined fMRI and eye-tracking study. *Frontiers in Human Neuroscience* 6 (2012), 331.

[161] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, and M. Joliot. 2002. Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain. *NeuroImage* 15, 1 (2002), 273–289. DOI:https://doi.org/10.1006/nimg.2001.0978

[162] Nienke van Atteveldt, Marlieke T. R. van Kesteren, Barbara Braams, and Lydia Krabbendam. 2018. Neuroimaging of learning and development: Improving ecological validity. *Frontline Learning Research* 6, 3 (2018), 186.

[163] Martijn P. Van Den Heuvel and Hilleke E. Hulshoff Pol. 2010. Exploring the brain network: A review on resting-state fMRI functional connectivity. *European Neuropsychopharmacology* 20, 8 (2010), 519–534.

[164] Jonathan Wai, David Lubinski, and Camilla P. Benbow. 2009. Spatial ability for STEM domains: Aligning over 50 years of cumulative psychological knowledge solidifies its importance. *Journal of Educational Psychology* 101, 4 (2009), 817.

[165] Cathrin Weiß, Rahul Premraj, Thomas Zimmermann, and Andreas Zeller. 2007. How long will it take to fix this bug? In *Workshop on Mining Software Repositories*. IEEE, 1–1.

[166] Robin Williams. 1971. A survey of data structures for computer graphics systems. *ACM Computing Surveys* 3, 1 (March 1971), 1–21. DOI:https://doi.org/10.1145/356583.356584

[167] Leigh Williamson. 2008. IBM rational software analyzer: Beyond source code. In *Rational Software Developer Conference*.

[168] Jacob O. Wobbrock, Leah Findlater, Darren Gergle, and James J. Higgins. 2011. The aligned rank transform for nonparametric factorial analyses using only anova procedures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 143–146.

[169] Eun-Mi Yang, Thomas Andre, Thomas J. Greenbowe, and Lena Tibell. 2003. Spatial ability and the impact of visualization/animation on learning electrochemistry. *International Journal of Science Education* 25, 3 (2003), 329–349.

[170] Reyyan Yeniterzi and Jamie Callan. 2015. Moving From Static to Dynamic Modeling of Expertise for Question Routing in CQA Sites. Retrieved on February 2020 from https://www.aaai.org/ocs/index.php/ICWSM/ICWSM15/paper/view/10548.

[171] Zuoning Yin, Ding Yuan, Yuanyuan Zhou, Shankar Pasupathy, and Lakshmi Bairavasundaram. 2011. How do fixes become bugs? In *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering (ESEC/FSE'11)*. Association for Computing Machinery, New York, NY, 26–36. DOI:https://doi.org/10.1145/2025113.2025121

[172] Motahareh Bahrami Zanjani, Huzefa H. Kagdi, and Christian Bird. 2016. Automatically recommending peer reviewers in modern code review. *IEEE Transactions on Software Engineering* 42, 6 (2016), 530–543. DOI:https://doi.org/10.1109/TSE.2015.2500238

[173] Zutao Zhang and Jiashu Zhang. 2010. A new real-time eye tracking based on nonlinear unscented Kalman filter for monitoring driver fatigue. *Journal of Control Theory and Applications* 8, 2 (2010), 181–188.

Received March 2020; revised August 2020; accepted November 2020