Simultaneous Measurement of Program Comprehension with fMRI and Eye Tracking: A Case Study

Norman Peitek
Leibniz Institute for Neurobiology
Magdeburg, Germany

Janet Siegmund
University of Passau
Passau, Germany

Chris Parnin
NC State University
Raleigh, North Carolina, USA

Sven Apel
University of Passau
Passau, Germany

Johannes C. Hofmeister
University of Passau
Passau, Germany

André Brechmann
Leibniz Institute for Neurobiology
Magdeburg, Germany

ABSTRACT

Background Researchers have recently started to validate decades-old program-comprehension models using functional magnetic resonance imaging (fMRI). While fMRI helps us to understand neural correlates of cognitive processes during program comprehension, its comparatively low temporal resolution (i.e., seconds) cannot capture fast cognitive subprocesses (i.e., milliseconds).
Aims To increase the explanatory power of fMRI measurement of programmers, we are exploring in this methodological paper the feasibility of adding simultaneous eye tracking to fMRI measurement. By developing a method to observe programmers with two complementary measures, we aim at obtaining a more comprehensive understanding of program comprehension.
Method We conducted a controlled fMRI experiment of 22 student participants with simultaneous eye tracking.
Results We have been able to successfully capture fMRI and eye-tracking data, although with limitations regarding partial data loss and spatial imprecision. The biggest issue that we experienced is the partial loss of data: for only 10 participants, we could collect a complete set of high-precision eye-tracking data. Since some participants of fMRI studies show excessive head motion, the proportion of full and high-quality data on fMRI and eye tracking is rather low. Still, the remaining data allowed us to confirm our prior hypothesis of semantic recall during program comprehension, which was not possible with fMRI alone.
Conclusions Simultaneous measurement of program comprehension with fMRI and eye tracking is promising, but with limitations. By adding simultaneous eye tracking to our fMRI study framework, we can conduct more fine-grained fMRI analyses, which in turn helps us to understand programmer behavior better.

CCS CONCEPTS

• **Human-centered computing → HCI design and evaluation methods**; **Empirical studies in HCI**;

KEYWORDS

program comprehension, functional magnetic resonance imaging, eye tracking

ACM Reference Format:
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 *ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM) (ESEM '18), October 11–12, 2018, Oulu, Finland.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3239235.3240495

1 INTRODUCTION

Program comprehension is the cognitive process of understanding program code. Since programmers spend most of their time with comprehending existing code [23, 39], researchers have been trying to unravel its underlying cognitive (sub)processes for decades. Initially, conventional research methods have been used (e.g., think-aloud protocols, interviews, comprehension summaries) to observe programmers. These observations are the foundation of some widely accepted program-comprehension models (e.g., top-down or bottom-up comprehension) [6, 14, 29, 38]. However, conventional research methods are limited when it comes to isolating the role of specific cognitive subprocesses, such as attention, memory, or language processing — an issue we are addressing with functional magnetic resonance imaging (fMRI). fMRI has been extensively used in neuroscience, as it allows researchers to identify neural correlates of cognitive processes [18]. In software-engineering research, specifically to study program comprehension, there are currently five studies that have applied fMRI, which all identified a largely overlapping network of activated brain areas, involved in working memory, attention, language comprehension, and semantic integration [8, 12, 16, 34, 36].

Despite these successes, fMRI is also inherently limited: First, the sequence of mental processes for each individual task and participant during an fMRI session cannot be directly observed. While we can assure that participants are fulfilling the task by checking their responses, we cannot analyze *how* they completed the task. Moreover, program comprehension consists of several simultaneously active cognitive subprocesses (e.g., attention, working memory, problem solving), each of which contributes to the overall process with varying intensity. Previous studies have proposed theories on program-comprehension phases [22, 25], which could explain the behavior of participants. However, so far, we lack information on brain activation during program comprehension to accurately

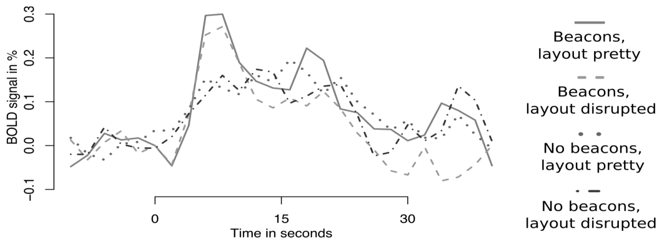


Figure 1: BOLD response in Brodmann area 21 from our study on how code influences top-down comprehension [36]. In the study, we manipulated meaningfulness of identifiers (beacon versus no beacon) and code formatting (layout pretty versus layout disrupted).

identify individual program-comprehension phases. Second, the temporal resolution of fMRI is rather low (i.e., depending on the protocol, 1 to 2 seconds) and is delayed by about 5 seconds [9, 21]. Thus, it does not allow us to immediately observe more rapid cognitive subprocesses. We may miss some cognitive subprocesses of program comprehension, this way assuming a uniform fMRI activation across the whole period of understanding a source code.

For example, Figure 1 shows the blood oxygenation level dependent (BOLD) response for Brodmann area (BA[1]) 21 of four conditions from our previous fMRI study on top-down comprehension [36]. The BOLD response shows a higher activation strength between 5 to 10 seconds for both conditions, which include a meaningful identifier (i.e., a *beacon* [6]). A possible explanation for the increased activation strength is that, if participants recognize a beacon, they recall an appropriate programming plan [6, 41]. However, as fMRI alone gives only limited insight into a participant's strategy during a task, this interpretation of a semantic recall is speculation.

In this paper, we report and evaluate a new methodology for conducting studies of program comprehension with fMRI by including eye tracking. With a focus on methodology, we are not analyzing the results in terms of program comprehension (i.e., how participants understood the source code), but we focus here on evaluating the methodology of combining fMRI and eye tracking. With the framework of fMRI and simultaneous eye tracking in place, we will be able to conduct a more fine-grained analysis of cognitive subprocesses of program comprehension in future studies.

1.1 Simultaneous fMRI and Eye Tracking

By simultaneously recording eye movements, we hope to identify when a participant is dealing with which part of the code [1], thereby connecting program-comprehension phases to the resulting brain activation [27, 28]. For the example of Figure 1, we may find participants fixating on a beacon shortly before an increased BOLD response. In this case, eye tracking may provide evidence for the suspected fixation on a beacon, which would support the theory of semantic recall of programming plans during top-down comprehension [36, 38].

[1]Brodmann areas are an anatomical classification system. The entire brain is split into several areas on the basis of cytoarchitectonic differences suggested to serve different functional brain processes [5].

With eye tracking, we can observe visuo-spatial attention by collecting eye-movement data [19], from which we can infer what the programmer is focusing on. Thus, eye tracking allows us to better understand the behavior of a programmer during a program-comprehension task. Furthermore, due to the high temporal resolution of eye tracking (i.e., depending on the model, 50 – 2000 Hz), we can capture even rapid eye movements. This allows us to infer details of ongoing cognitive processes during program comprehension. For example, Duchowksy et al. have shown that patterns of high saccadic amplitudes after fixations indicate that the participants are scanning for a feature in a presented stimulus [11]. However, longer fixations in relation to shorter saccades indicate a pursuit search, suggesting that a participant is trying to verify a task-related hypothesis, rather than overviewing a stimulus [11].

Eye tracking alone is a popular measure to observe visual attention during program comprehension [1, 7, 31]. It may also offer a way to test hypotheses about program comprehension. However, unlike neural measures, eye tracking is limited regarding insights into higher-level cognitive processes (e.g., language comprehension, decision making, working memory). Thus, researchers have begun to combine eye tracking with neural measures, for example, simultaneous recording of electroencephalography (EEG) [17] or functional near-infrared spectroscopy (fNIRS) [15]. While an fMRI experiment design is more difficult, fMRI offers a higher spatial resolution than EEG and fNIRS, which was our motivation to integrate fMRI and eye tracking as simultaneous measures.

Eye tracking has been used in fMRI studies, but mostly as indicator for whether participants are fulfilling the task (and not sleeping) [19]. Duraes et al. studied debugging with simultaneous measurement of fMRI and eye tracking, but did not report how successful and precise the recorded eye-tracking data were in their experiment. Furthermore, they did not appear to use the eye-movement data [12]. In a follow-up experiment, they analyzed the eye-tracking data separately from the brain-activation data [8]. In the area of software engineering, there has been no study combining fMRI and eye tracking for a conjoint, more comprehensive analysis [28].

In summary, combining fMRI and eye tracking is promising for program-comprehension research, because the two methods complement each others' strengths. The high temporal resolution of eye tracking, in combination with information about which part of the code a participant is focusing on, allows us to identify the origin of neural activations more precisely in time. By combining the two methods, we can reason about causal relationships: What part of a program gives rise to the activation of a specific brain area or triggers a certain cognitive process?

1.2 Results and Contributions

The present study is a non-exact replication of our previous fMRI study [36] on the contrast of top-down [6] and bottom-up comprehension [29, 32, 38]. We simplified the experiment design to increase statistical power and added eye tracking (cf. Section 3). We conducted the study in a 3-Tesla fMRI scanner at the Leibniz Institute for Neurobiology in Magdeburg, Germany.

Our results demonstrate that it is indeed possible to simultaneously observe program comprehension with fMRI and eye tracking.

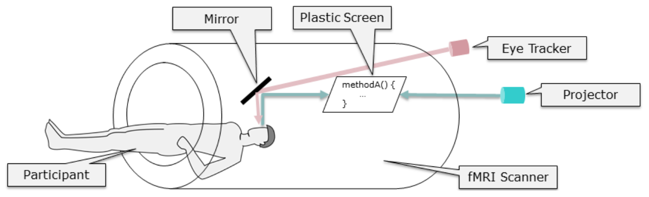


Figure 2: Setup of long-range camera-based eye tracking in an fMRI scanner

Moreover, the data that we collected provide new insights into program comprehension. However, there are several caveats. First, for only 40% of our participants, the eye tracker could continuously capture the eye movements, which is much lower than outside an fMRI environment. Second, for our purpose of detecting fixations on beacons, the vertical spatial imprecision of 30 – 50 pixels that we experienced can be an issue when not optimizing the code display with large fonts and line spacing. Third, there is a spatial imprecision that slowly grows over time (i.e., *drift*); it is small enough though that it can be ignored for our experiment lengths and study purposes.

In summary, we make the following contributions:

- We present the first study exploring simultaneous measurement and analysis of fMRI and eye-tracking data in software engineering, supporting our prior hypothesis of semantic recall in BA21 [36].
- We show that a more fine-grained fMRI analysis with simultaneous eye tracking is feasible, especially when carefully designing the code display for the fMRI environment.
- We devise an extension of our fMRI study framework [33], which has been the base for most follow-up fMRI experiments [16, 34, 36], with eye tracking.
- We provide all materials as a replication package, and publicly share all developed eye-tracking analysis scripts to support future combined studies. We also share the raw and processed eye-tracking data.

Our long-term research goal is to gain a comprehensive understanding of program comprehension, for which we need to conjointly analyze fMRI and eye-tracking data (e.g., analyze brain activation after a fixation on a beacon). For this purpose, we have to be confident in the spatial precision of our collected eye-tracking data. In this paper, we investigate how much we can rely on the eye tracker's spatial precision and how we can optimize our experiment design for future fMRI studies.

2 RESEARCH OBJECTIVES

Simultaneous fMRI and Eye Tracking. First, we need to evaluate whether the delicate experiment setup allows us to collect stable and precise eye-tracking data during a 30-minute fMRI experiment:

RQ1: Can we simultaneously observe program comprehension with fMRI and eye tracking?

While the simultaneous measurement and analysis of fMRI and eye-tracking data would open the door to a novel perspective on program comprehension, the strict non-magnetic environment around an fMRI scanner poses a challenge for the use of an eye tracker. We used a long-range camera-based eye tracker mounted outside the scanner bore, which is roughly 90 centimeters away from a participant's head (cf. Figure 2). From this position, the eye tracker and infrared light reflect through the small mirror, over which participants see the source code, and then hits a participant's eye through a small opening of the head coil (cf. Figure 2).

Precision of Eye Tracking inside an fMRI Scanner. Second, for our goal of an eye-tracking-informed fMRI analysis, we do not only require both data streams, but we need to confidently recognize and match specific events of program comprehension with spatial precision (e.g., fixation on a beacon). Thus, for our purposes, we need, at least, a word-level spatial precision. To evaluate whether we can expect such precision from our setup, we pose our second research question:

RQ2: Is eye tracking sufficiently precise for fMRI studies of program comprehension?

Answering this question is crucial, because there is an inherent trade-off between the font size and the amount of text we can display on an fMRI screen. The screen is restricted in size, such that with a font size of 32, we can show 20 lines of text. A bigger font size lets the eye tracker more reliably capture the elements that participants were looking at, but results in fewer lines of text on the fMRI screen. Although scrolling is possible, it comes with new challenges, as it may induce movement (which further reduces the number of usable fMRI datasets) and continuously changes the visual input for each individual and task, adding complexity to the eye-tracking analysis.

Eye-Tracking Drift. Third, we need to evaluate whether the eye-tracking data are consistently precise throughout an experiment:

RQ3: Is there a drift throughout the experiment?

After an initial calibration, eye-tracking accuracy is expected to *drift* (a spatial imprecision worsening over time) [19]. Participant movements and change in positioning challenge an eye tracker to consistently capture a precise result [19]. In conventional eye-tracking experiments in front of a computer, the calibration can be repeated, if the eye-tracking accuracy is falling below a threshold [13, 20]. Such on-demand experiment interruption is not possible in an fMRI study, because the functional measurement of program comprehension has a pre-specified fixed length. A possible split in multiple short sections (as done by Floyd et al. [16]), with intermittent eye-tracking re-calibrations, increases the experiment length and the head movement of participants, decreasing fMRI data quality.

Our eye tracker's recommended time for continuous eye tracking without a repeated calibration and validation procedure is 20 minutes. A common length for fMRI experiments is around 30 minutes, exceeding this time limit. This raises the question of how precise eye tracking will be towards the end of an experiment. Two aspects of fMRI experiments suggest that a stable measurement throughout a 30-minute experiment is possible: First, a participant's head is fixated with cushions during the fMRI session, so that head movement, which is a common cause for impaired eye-tracking precision, is small. Second, the fMRI protocol includes a rest condition, which displays a fixation cross in the screen center which participants should fixate on. The precision and accuracy of the eye tracker during the fixation-cross condition, which will be displayed around every minute, is a clear indicator of the eye-tracking stability throughout an experiment. Answering RQ3 will help us to decide whether we need intermittent eye-tracker calibrations.

3 EXPERIMENT DESIGN

We based our study design on our previous fMRI studies of program comprehension [34, 36]. All participants completed the experiment in the same order. We contrasted tasks of bottom-up comprehension, top-down comprehension, and finding syntax errors. We induced bottom-up comprehension by removing semantics from code identifiers (e.g., a variable name of `reversedWord` versus `result`), which requires a line-by-line mental execution. We triggered top-down comprehension by using code snippets with rich semantics, which enables participants to quickly hypothesize a snippet's function. Furthermore, we familiarized participants with some of the snippets in a prior training to ensure sufficient domain knowledge for employing top-down comprehension.

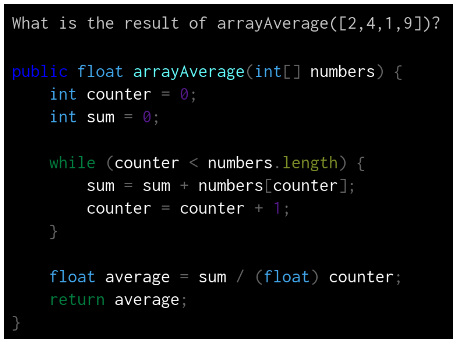
As one of our long-term research goals is to understand how beacons [6] influence top-down comprehension, we created two versions of top-down comprehension by manipulating the meaningfulness of a snippet's identifiers (e.g., a function name of `arrayAverage` versus a scrambled `beebtBurebzr`). We did not familiarize participants with all snippets to understand the effect of our training. All top-down and bottom-up comprehension snippets were part of our previous top-down comprehension study [36], with a length of 7 to 14 lines of code (LOC).

In summary, we manipulated the following independent variables: comprehension mode (bottom-up, top-down), the presence of beacons (yes, no), and prior training (yes, no).

3.1 Refinement of Study Framework

The goal of the present study is to improve the measurement framework based on the experience from our previous fMRI studies, where we experienced some limitations regarding the control and rest conditions. In a nutshell, the success of an fMRI analysis depends on how suitable the chosen conditions are. fMRI analysis is based on computing contrasts between appropriately different conditions to carefully exclude cognitive processes from the brain-activation data that are unrelated to a research question. In the previous studies, we used finding syntax errors as *control tasks*, which we contrasted with comprehension tasks to obtain brain activation that is specific for program comprehension (e.g., working memory, but not unrelated visual attention). Furthermore, a rest condition as brain-activation baseline is necessary, in which participants think about programming as little as possible.

Improved fMRI Contrasts. In our previous study [36], we found that participants, in fact, partially comprehended code when finding syntax errors, reducing contrast strength. Furthermore, they reflected on the comprehension tasks during the rest condition, also disturbing the brain-activation baseline. Both reduce the statistical power of our fMRI analysis. Thus, for this experiment, we attempted to mitigate these problems by adding a new distractor task[2] prior to the rest condition. This way, we intend to block snippet-related cognitive processes during the rest condition and also provide a better control task (as it is not related to programming).


```
What is the result of arrayAverage([2,4,1,9])?

public float arrayAverage(int[] numbers) {
    int counter = 0;
    int sum = 0;

    while (counter < numbers.length) {
        sum = sum + numbers[counter];
        counter = counter + 1;
    }

    float average = sum / (float) counter;
    return average;
}
```

Figure 3: Example snippet as visible in the fMRI scanner.

Integration of Eye Tracking. We extended the study framework by integrating eye tracking, which required us to make several changes at the technical level. First, we calibrated and validated the eye tracker, which was scheduled after the anatomical pre-measurements, but before the functional fMRI scan. Second, we adapted the used Presentation® software[3] scripts to synchronize the eye tracker and the presented source code. We connected the Presentation software and the EyeLink eye tracker with the *PresLink* plugin. More details on the experiment design, including materials and used scripts are available on the project's Web site.[4]

3.2 Experimental Conditions

Overall, we had the following design: The fMRI session consisted of five trials, each containing different comprehension conditions, a syntax-error-finding task, a distractor task, and intermittent rest conditions, resulting in the following sequence in one trial:

- Top-down comprehension (Trained, with beacons [Tr-B], 30 sec.)
- Distractor task (15 sec.), rest condition (20 sec.)
- Bottom-up comprehension (BU, 30 sec.)
- Distractor task (15 sec.), rest condition (20 sec.)
- Top-down comprehension (Trained, no beacons [Tr-NB], 30 sec.)
- Distractor task (15 sec.), rest condition (20 sec.)
- Top-down comprehension (Untrained, with beacons [NTr-B], 30 sec.)
- Distractor task (15 sec.), rest condition (20 sec.)
- Finding syntax errors (SY, 30 sec.)
- Distractor task (15 sec.), rest condition (20 sec.)

Each program-comprehension task lasted 30 seconds, plus a 2-second grace period to submit a late response. They were each followed by a 15-second distractor task and a 20-second rest condition. This results in an experiment execution time of $5\ trials \cdot 335\ sec. \approx 28\ minutes$.

3.3 Task Design

As tasks, participants should determine the output of a given Java method call. Figure 3 shows an exemplary snippet inducing top-down comprehension as visible in the fMRI scanner. We kept the

[2]For this purpose, we used a d2 task, which is a psychological test of attention in which participants scan through a row of letters and decide for each letter whether it is a *d* with two marks [4]

[3]Version 19.0, Neurobehavioral Systems, Inc., Berkeley, CA, USA, https://neurobs.com
[4]https://github.com/brains-on-code/simultaneous-fmri-and-eyetracking

Table 1: Participant demographics

Characteristic		N (in %)
Participants		22
Gender	Male	20 (91%)
	Female	2 (9%)
Pursued academic degree	Bachelor	9 (41%)
	Master	13 (59%)
Age in years ± SD		26.70 ± 6.16
Programming experience	Years of experience ± SD	6.14 ± 4.57
	Experience score [35] ± SD	2.73 ± 0.75
	Java experience [35] ± SD	1.93 ± 0.33

computational complexity of the snippets low (e.g., square root of 9 or 25), so that participants can focus on program comprehension. The participants responded for each task via a two-button response device.

During the finding-syntax-error task, we asked participants to click the response button whenever they found a syntax error. Each snippet contained three syntax errors, which did not require comprehension of the snippet (e.g., missing semicolon).

3.4 Study Participants

We recruited 22 students from the University of Magdeburg via bulletin boards. Requirements for participating in the study were experience in object-oriented programming and the ability to participate in an fMRI experiment. We invited left-handed participants for a second fMRI session to determine their language-dominant hemisphere [2]. Every participant completed a programming experience questionnaire [35], which showed that our participants are a fairly homogeneous group in terms of programming experience. Table 1 shows details regarding the participants' programming experience and their demographics. The participants were compensated with 10 Euro per hour.

3.5 Experiment Execution

Upon arrival, we informed participants about the goals of our study, the risks of fMRI, and asked for their informed consent. Then, they completed the programming experience questionnaire and a brief training, directly followed by the fMRI measurement and eye tracking. Afterward, we conducted a short debriefing interview.

3.6 fMRI Imaging Methods

We carried out the imaging sessions on a 3-Tesla scanner,[5] equipped with a 32-channel head coil. The heads of participants were fixed with a cushion with attached ear muffs containing fMRI-compatible headphones.[6] Participants wore earplugs to reduce scanner noise by 40 to 60 dB overall. We obtained a T1-weighted anatomical 3D dataset with 1 mm isotropic resolution of the participant's brain before the functional measurement. To capture a whole-head fMRI, we acquired 878 functional volumes in 28 min using a continuous EPI sequence.

[5]Philips Achieva dStream, Best, The Netherlands
[6]MR Confon GmbH, Magdeburg, Germany

3.7 Eye-Tracking Methods

We used an MRI-compatible EyeLink 1000[7] eye tracker for our study. It offers 1000 Hz temporal resolution, <0.5° average accuracy, and 0.01° root mean square (RMS).

We tracked a participant's left eye[8] on a display with a resolution of 1280 by 1024 pixel. We calibrated the eye tracker with a randomized 9-dot grid, and we conducted a 9-dot validation to identify possible issues with the calibration. If the error during validation exceeded the EyeLink's recommended thresholds, we repeated the calibration and validation process (this was necessary 7 times).

We calibrated and validated the eye tracker after the pre-measurements, but before the functional fMRI scan. After successful calibration and validation, we started the functional fMRI scan.

We used vendor and customized scripts to extract and convert the obtained eye-tracking data. We imported the data into Ogama for further analysis [40]. We provide the complete work flow, all custom scripts, raw and processed eye-tracking data on the project's Web site.[4]

4 RESULTS

In this section, we present the results of our study, following our three research questions.

RQ1: Can we simultaneously observe program comprehension with fMRI and eye tracking?

Data Recording. We successfully calibrated, validated, and recorded eye-tracking data for 20 out of 22 invited participants. We could not gather eye-tracking data from 2 participants, where a calibration was not possible (once due to Strabismus [3], once due to a technical failure). Due to the delicate setup of the eye-tracking camera outside the bore, the angle to a participant's eyes is not ideal. When calibration initially could not be completed, we had to ask 8 out of 20 participants to widely open their eyes during calibration. The validation with widely opened eyes showed good precision. For participant comfort, we let participants return to the natural state of their eyes during the experiment. However, when the request to widely open their eyes during calibration was necessary, the eye tracker was unable to consistently capture eye movements during the experiment, which resulted in incomplete eye-tracking data (for all cases, we obtained eye-movement data for less than 30% of the experiment time). Overall, for 10 out of 20 participants, the eye tracker was not able to consistently capture eye movements.

Data Quality. An important aspect of eye-tracking data quality is the number of frames captured by the eye tracker. In general, 100% of recorded frames is not realistic because of a high blink rate: In an fMRI environment, participants look via a mirror into a bright, projector-backlit screen, and ventilation with dry air is necessary to sustain a comfortable environment for participants; both increase the blink rate. For 8 out of 20 (40%) participants, the number of recorded frames was excellent (more than 85%). For 10 out of 20 (50%) participants, the eye tracker captured, at least, 65% of all frames. For 8 out of 20 (40%), the eye tracker captured

[7]SR Research Ltd, Ottawa, Ontario, Canada, http://www.sr-research.com
[8]We focused on the left eye due to technical constraints. It is also possible to track the dominant eye or both eyes, but that requires additional equipment.

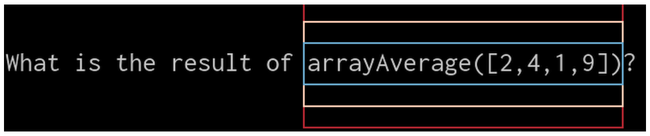


Figure 4: Visualization of Inner, 25-px Extra, and 50-px Extra area-of-interests (AOIs) around a task identifier for the analyses of RQ2

Table 2: Summary of all fixation counts and lengths within AOIs around task identifiers. Number in brackets is the overall percentage.

	Inner AOI	25-px Extra AOI	50-px Extra AOI
Fixation Count	1 266 (46%)	2 244 (81%)	2 772 (100%)
Fixation Length (sec)	2 215 (48%)	3 885 (83%)	4 661 (100%)

less than 10% of all frames. To obtain representative, meaningful eye-tracking data, we need to capture, at least, 65% of frames.

RQ1: Our study indicates that it is, in principle, feasible to simultaneously measure program comprehension with fMRI and eye tracking. However, the high failure rate of eye tracking due to the fMRI environment requires further setup improvements and has to be considered when designing future experiments.

RQ2: Is eye tracking sufficiently precise for our fMRI studies of program comprehension?
For our goal of an eye-tracking-informed fMRI analysis, it is critical not only to record eye movements (e.g., to analyze generic metrics such as fixation counts or average saccade lengths), but also to provide eye-tracking data with high spatial precision (e.g., to detect fixations on beacons). When eye tracking is successful, are the spatial errors small enough to confidently detect fixations on an individual identifier? For our stimuli, one line of code was 40 pixels high and a single character around 20 pixels wide. Thus, we would require a spatial error of smaller than 40 pixels.

Data Selection. To obtain an accurate result, we will only analyze the eye-tracking data for all participants that are suitable (at least, 65% of recorded frames, $n = 10$) for RQ2.

Calibration Validation. At the beginning of every experiment, we conducted a calibration validation to estimate the spatial error, which showed an average error of 22 pixels horizontally and 26 pixels vertically (i.e., 0.99°). The estimated horizontal spatial error of 22 pixels during calibration validation indicates that the eye tracker is precise enough to detect fixations on words, but not on single characters. The estimated vertical spatial error of 26 pixels during calibration validation is problematic for us, as it is close to the used line height of 40 pixels. With such a spatial error, we might erroneously categorize a fixation on an incorrect line of code.

Area-of-Interest Analysis Over All Snippets. To analyze whether the assumed vertical spatial imprecision can lead us to incorrectly categorize a fixation, we conducted an area-of-interest (AOI) analysis on the task description at the top of each presented code snippet. We added three AOIs with different heights around the instructed

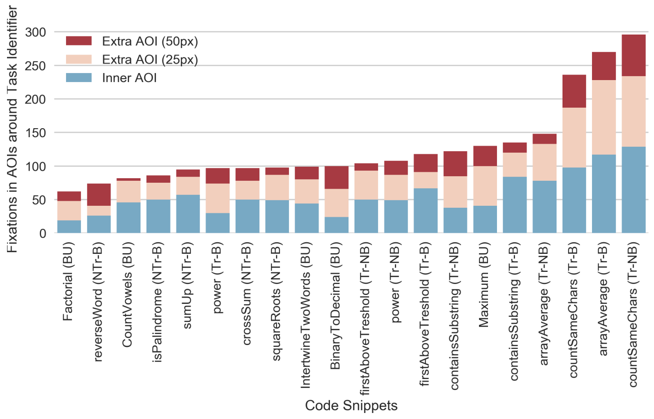


Figure 5: Fixation count for Inner, 25-px Extra, and 50-px Extra area-of-interests (AOIs) and code snippet

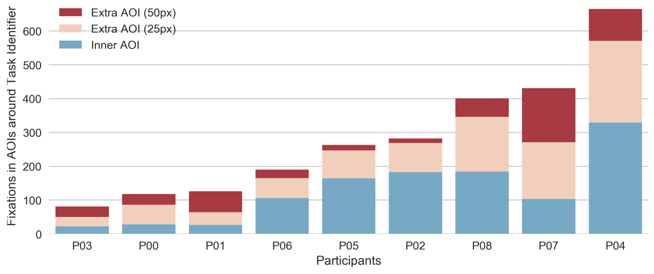


Figure 6: Fixation count for Inner, 25-px Extra, and 50-px Extra area-of-interests (AOIs) and participant

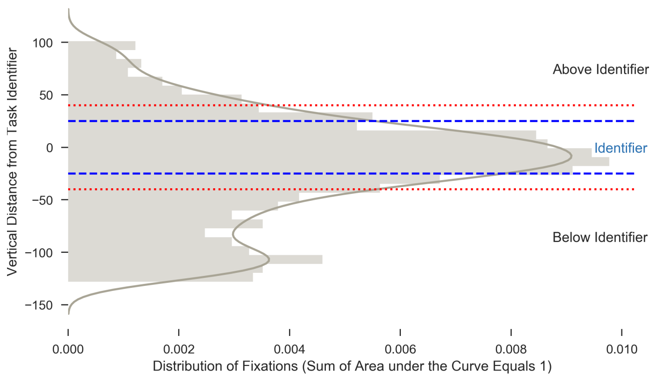


Figure 7: Distribution of vertical distance from 3390 fixations on task identifiers. - - - - indicates current line height. ········ indicates suggested line height

method call, which we visualize in Figure 4. The inner AOI includes only the line of the method call in question, while the extra AOIs span, respectively, 25 and 50 vertical extra pixels. Because there is nothing above or below the task line, the eye tracker should not record more fixations in the extra AOIs than the inner AOI. Assuming there is no human error and no technical spatial error, all fixations should be on the inner AOI directly on the task line. Every fixation on one of the extra AOIs could be later interpreted as fixation on an incorrect line, and thus on a wrong identifier. We applied the same AOI sizes to all 20 comprehension snippets and, again, analyzed all eye-tracking datasets with, at least, 65% of recorded frames ($n = 10$).

In Table 2, we summarize the results of the AOI analysis. Over all comprehension snippets and participants, there are 1 266 fixations, spanning, in total, over 2 215 seconds on the inner, correctly sized AOI. On the two extra AOIs, which may lead to incorrectly categorized fixations, there are, respectively, 2 244 and 2 772 fixations. Thus, only less than half of the fixations around the task identifier are actually detected on the actual task identifier line. 978 extra fixations are detected within 25 pixels, and another 528 fixations within the next 25 pixels. That is, at the used line height of 40 pixels, we cannot confidently detect fixations at the level of identifiers, because the vertical spatial error is too high.

AOI Analysis for Each Snippet. To examine whether a snippet affected the combined AOI analysis, we also looked at the AOI fixations for each snippet. In Figure 5, we show the number of fixations on all three AOIs for each snippet. While there are large differences in absolute fixation count, the relative sizes between the three AOIs are similar, showing that the precision is not affected by the type of snippet.

AOI Analysis for Each Participant. In a next step, we separately analyzed each participant to eliminate the chance that an individual participant distorts the result of our AOI analysis. Figure 6 shows that there are differences between our participants regarding fixation count and eye-tracking precision. Even for participants for which we captured precise eye-tracking data, we observe a notable amount of vertical spatial error. This supports the conclusion that 40 pixels line height is not sufficient to prevent incorrect classification of fixations.

Optimal Line Height. Since all analyses point toward a need for lines higher than 40 pixels, we need to find an optimal line height. Larger font sizes and line spacings are common in eye-tracking experiments to create more vertical distance between lines. This would let us be more confident when detecting fixations, but at the cost of fewer lines of code to fit on the screen without scrolling. To find a balance, we analyzed the vertical distance from each fixation around the task identifier. The maximum vertical distance considered in this analysis was three line heights in each direction (i.e., 120 pixels above and below the center of the identifier).[9] We analyzed the same 10 participants with more than 65% of recorded frames. Figure 7 reveals that the current line height catches a lot of fixations, but there is a significant number of detected fixations right below and above the line. Based on this result, a line height and spacing of 80 pixels would allow the eye tracker to correctly capture fixations.

RQ2: Without correction, an estimated vertical spatial error of 30 – 50 pixels is too high with the used line height of 40 pixels to confidently detect fixations at the level of individual identifiers. An increase to 80 pixel line height would allow us to do so.

[9]Note that the increased fixations of 80 pixels and more below the identifier is due to fixations on actual code. For some snippets, there was only an 80 pixel distance between task instruction and code.

RQ3: Is there a drift throughout the experiment?
The analyses of RQ2 showed that there is a significant spatial error that has to be addressed in future experiments (e.g., by increasing line heights). But does that spatial error further increase throughout an experiment? To answer RQ3, we analyzed the spatial error from the fixation cross during each of the 25 rest conditions throughout our experiment.

To obtain an accurate picture of the drift over time, we included only participants for which the eye tracker was able to consistently track the eyes (≥85% of frames, $n = 8$). We excluded the first half second of fixations during the rest condition, because participants were still concentrated on the previous task and needed some time to move their gaze to the fixation cross. We also excluded all fixations off the screen (e.g., participants looking above the screen at the eye-tracking camera) or with an absolute spatial error of larger than 300 pixels (e.g., participants looking around).

Figure 8 shows the spatial error for the x-axis and y-axis over time. The observed spatial error based on the fixation cross largely confirms the estimated error of our validation (cf. RQ2). A horizontal spatial error of around 25 pixels substantiates previous estimates during the analysis of RQ2. The vertical spatial error of initially around 55 pixels is higher than the general validation error, but consistent with the validation error at the center middle (average error of 47 pixels). Overall, the spatial error is slowly growing, but largely stable throughout the experiment. The estimated spatial error is worse at the end of the experiment, but the eye-tracking data would still usable with an appropriately increased line height.

The vertical yellow lines in Figures 8 to 10 mark the eye tracker's recommended maximum experiment length of 20 minutes. The data indicate that the eye-tracking data quality does not significantly deteriorate after 20 minutes, which supports the conclusion that we can conduct our experiments without intermittent re-calibrations.

Figures 9 and 10 respectively show the spatial errors on x-axis and y-axis for each participant over time. For most participants, the horizontal spatial error is stable throughout the entire experiment. We believe some individual outliers (e.g., rest condition 15) can be attributed to participants looking around and not due to technical errors. However, the vertical y-axis reveals a different result. For some participants, the spatial error is consistently small enough to provide a useful dataset, but for some participants, we observe a large spatial error of more than 100 pixels from the beginning, which was not evident during the calibration validation.

RQ3: The drift throughout the experiment is negligibly small in comparison to the problematic general spatial error.

5 DISCUSSION

Having presented our results, we now discuss their implications.

5.1 Eye-Tracking Optimizations

When we fine-tuned the eye-tracking setup in several pilot sessions, we knew that a perfect 100% output of the eye tracker is out of reach. Nevertheless, that the eye tracker was only able to reliably capture eye movements for 40% of our participants was unexpectedly low. In the future, we will investigate further optimizations of the eye-tracking camera vendor. We will also evaluate whether we save the camera video feed and estimate eye gazes even if the eyes are not

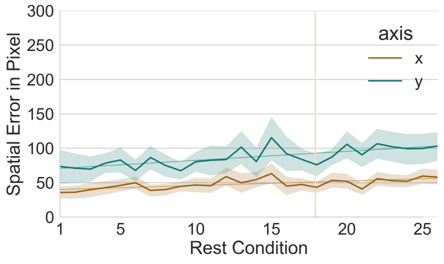


Figure 8: Spatial error of the rest condition's fixation cross over time. Standard deviation is shown as shades.

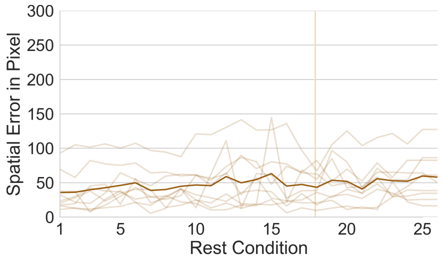


Figure 9: Spatial error on x-axis over time for 8 participants with complete eye-tracking data

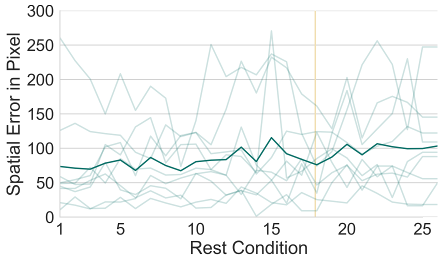


Figure 10: Spatial error on y-axis over time for 8 participants with complete eye-tracking data

widely opened for the automated tracking. If so, we would be able to almost double the success rate.

In addition to the issue of the low recording success rate, our data showed a significant spatial error, especially on the vertical axis. In future fMRI studies, we plan to evaluate a refined calibration procedure, for example, by using a 16-point calibration particularly focusing on the code snippet area. We expect an improvement, because spatial errors are smaller around the calibrated points [19]. This way, we may be able to increase spatial precision. Moreover, we will decrease the validation thresholds for when we accept a calibration as successful to further improve spatial precision [19].

While the refinements of our setup and calibration may improve spatial precision, we still see the need for optimizing the display of our code snippets. A significant point of action are adjusting font size and line spacing. When we designed our study, we based it on our previous successful fMRI study design, which did not optimize the code-snippet display for the eye-tracking modality. Most snippets still have empty space around them; we could have used larger fonts and line spacing. For our snippets, we should be able to increase line height to 80 pixels without the need for scrolling. This way, we can mitigate inevitable spatial errors and be more confident when detecting fixations, for example, on specific identifiers.

In our study, the drift appears small enough to reject the need for multiple eye-tracking re-calibrations.

5.2 Challenges and Limits

Our study also exemplified a typical problem of multi-modal experiments. Both of our measures, fMRI and eye tracking, have exclusion rates, where some participant's data cannot be used, for example, due to motion artifacts (fMRI) or half-closed eyes (eye tracking). In our study, only 17 of 22 (77%) fMRI datasets and 10 of 20 (50%) of eye-tracking datasets were fully usable. When considering both measures, we obtained only for 7 of 22 (32%) participants a complete dataset (i.e., fMRI and eye-tracking data are both usable). If a study requires both measures (e.g., an fMRI study that uses eye tracking as input), this high data drop-out needs to be kept in mind during planning, such that more participants need to be invited.

5.3 Potential Benefits

The complete datasets, including both fMRI and eye-tracking data, provide substantial insights. We can test individual hypotheses, for example, whether we connect participants' fixations on a beacon (detected with eye tracking) leads to a semantic recall (increased brain activation detected with fMRI). In our previous study, we could not explain the higher peak in BA21 during the conditions when beacons were present [36]. In the present study, we could replicate the effect, and with the simultaneous eye tracking, we now could confirm our suspicion that participants fixated on beacons prior to the peak in the BOLD response. Initially, we replayed the eye-tracking data with Ogama to qualitatively detect fixations on task identifiers. Then, we used an AOI to identify fixations specifically for each participant and snippet. We can now feed the detected fixation timestamps into our fMRI analysis for a more fine-grained result. Thus, eye tracking is the basis to advance from a coarse block-based fMRI analysis to a more detailed event-related analysis where we can distinguish fine-grained effects of program comprehension (such as semantic recall after fixation on beacons, as discussed in Peitek et al. [28]). Overall, we were able to confirm our hypothesis of semantic recall in BA21, which illustrates the feasibility of our setup.

Simultaneous fMRI and eye tracking offers possibilities beyond testing of particular hypotheses, such as understanding behavior of programmers better. For example, Figure 5 shows that, for all trained top-down comprehension conditions, participants fixate more often on the task instruction, indicating that they focus more on the result computation than comprehending a code snippet. Moreover, we can generate new hypotheses by exploring the data (e.g., mental loop execution leads to increased working memory activation in BA6) [28]. Both measures combined offer a powerful way to observe and understand program comprehension.

This approach of observing participants with simultaneous eye tracking and fMRI is also applicable to research questions beyond program comprehension, such as location bugs [8] or emotional user experience during human-technology interaction [30]. While eye tracking allows us to observe visual attention, fMRI captures an accurate representation of ongoing mental processes. Together, they can give substantial insights for many research questions with the human factor in mind.

5.4 Enhanced fMRI Study Framework

Overall, our study showed that simultaneous fMRI and eye tracking is challenging. Nevertheless, we demonstrated that it can be feasible and already provides insightful data. Therefore, we propose to enhance our fMRI framework for program-comprehension studies [33, 34] by adding eye tracking as an additional modality. Based on the experience of our study, we can encourage future fMRI studies to also include eye tracking, as it can notably increase insights from a study.

From a participant’s perspective, there is almost no extra effort with integrated eye tracking in comparison to the basic fMRI study framework. The calibration and validation at the beginning usually takes only an extra minute.

Nevertheless, there is a large extra effort for the investigators to design and analyze an fMRI study with integrated eye tracking. The selection of an appropriate eye-tracking solution, technical setup, stimuli preparation, and implementation of an analysis pipeline is time-consuming. To support future endeavors of the research community, we share all of our materials and scripts.

6 THREATS TO VALIDITY

6.1 Construct Validity

We operationalized eye-tracking precision by identifying fixations on code stimuli from previous fMRI studies. It is likely that a study specifically targeted at testing eye tracking in an fMRI environment, say with only small visual inputs all across the screen, would have lead to a more reliable and accurate answer. However, the results may not have been applicable to our study purposes (e.g., detecting fixations on code identifiers, finding the correct line height of code).

A post-processing correction of eye-tracking data is common, where the fixations are manually changed to fit the stimuli [10, 26]. We did not apply such correction, as it may have a substantial influence on the results and insights of this paper. However, it is possible that the imprecision explored in RQ2 can be reduced with such a correction, and thus our result, a recommended line height of, at least, 80 pixels, is excessive. Nevertheless, we prefer to be on the safe side.

6.2 Internal Validity

The nature of controlled fMRI program-comprehension experiments lead to a high internal validity, while reducing external validity [37]. For the presented technical analysis, we only see the number of participants ($n = 22$) as a threat to internal validity.

6.3 Conclusion Validity

We conducted the analyses for RQ2 and RQ3 on a single area/point of the screen (identifier on top of the screen and fixation cross in the middle center, respectively). While the spatial precision slightly changes throughout the screen, we estimate the chances for a significantly different result on other parts of the screen as low.

6.4 External Validity

We conducted the fMRI study at a single location. It is possible that another environment (e.g., different fMRI scanner or eye-tracking solution) will lead to better (or worse) results. For example, eye tracking built-in to the head coil[10] offers a higher precision than a long-range camera-based eye tracker outside the scanner bore, but, in turn, may decrease fMRI data quality. Thus, we have to decide for which modality we optimize the experiment setup.

[10]For example, Real Eye™, Avotec, Inc, http://avotecinc.com

Independent of equipment, our experience reported in this paper, and the developed analysis scripts, may still be valuable to researchers attempting such a study with fMRI and eye tracking.

7 RELATED WORK

Program comprehension is an established research field in software engineering. Nevertheless, neuro-imaging and psycho-physiological measures are still novel methods in this field. Closest to our study regarding the used methods is the study by Duraes et al., who observed debugging with fMRI and eye tracking [12]. It is unclear, though, what kind of eye-tracking data were recorded, and the authors did not appear to use the eye-tracking data. In a follow-up fMRI experiment, Castelhano et al. collected fMRI and eye-tracking data and separately analyzed them to study expert bug detection [8]. Our study differs by specifically focusing on the simultaneous use of fMRI and eye tracking to pave the way for future combined experiments. There are multiple other fMRI studies of program comprehension without eye tracking: We conducted two studies on bottom-up comprehension [34] and top-down comprehension [36]. Floyd et al. contrasted reviewing code and prose in an fMRI scanner [16].

There have been numerous studies observing program comprehension with eye tracking exclusively. Bednarik and Tukiainen were first to propose eye tracking as a tool to analyze cognitive processes during program comprehension [1]. Sharif and Maletic showed differences in comprehension of camelCase and under_score identifier styles, also using an eye tracker [31]. In the same vein, Busjahn et al. collected evidence for a difference in reading styles between novices and experts using eye tracking [7]. These studies show that eye tracking is a valuable measure to observe program comprehension, but is limited in explaining *why* programmers behave in a certain way and thus could benefit from the use of a simultaneous neural measure, such as fMRI.

First studies demonstrate that the combination of visual attention-capturing eye tracking and other neural measures besides fMRI is promising. Fritz et al. used EEG, eye tracking, and electrodermal activity to predict task difficulty [17]. Lee et al. used eye tracking and EEG to predict expertise and task difficulty [24]. Fakhoury et al. used fNIRS and eye tracking to show that the quality of identifier names is linked to the developers’ cognitive load when comprehending code [15].

8 CONCLUSION

Observing programmers with simultaneous fMRI and eye tracking would open the door to a more holistic understanding of program comprehension. In this paper, we reported that simultaneous measurement of program comprehension with fMRI and eye tracking is challenging, but promising. In a first study of its kind, we were able to gather *simultaneous* fMRI and eye-tracking data, although the overall success rate was moderate and we need to design snippets properly to confidently detect fixations on identifiers.

Nevertheless, a conjoint analysis of both data streams offers new observations of program comprehension. This way, we replicated a previous result of a stronger activation in BA21 when beacons were available. With simultaneous eye tracking, we now were able to confirm a prior fixation on beacons. This confirms our hypothesis of a semantic recall of programming plans in BA21. Overall, our gathered data encourage us to explore combined fMRI and eye-tracking data in more detail. Can we relate programmer behavior (based on eye movements) to the resulting brain-activation data? Can we find an increase in working memory activation in BA6 when programmers are mentally executing loops? In the long term, we are convinced that integrating eye tracking in all future fMRI studies is worth it.

ACKNOWLEDGMENTS

We thank all participants of the fMRI study. Furthermore, we thank Andreas Fügner, Anke Michalsky, and Jörg Stadler for their technical support during pilot studies and fMRI data acquisition.

Siegmund's and Brechmann's work is supported by DFG grants SI 2045/2-1 and BR 2267/7-1. Siegmund's work is further funded by the Bavarian State Ministry of Education, Science and the Arts in the framework of the Centre Digitisation.Bavaria (ZD.B). Parnin's work is supported by the National Science Foundation under grant number 1755762. Apel's work has been supported by the German Research Foundation (AP 206/6).

REFERENCES

[1] Roman Bednarik and Markku Tukiainen. 2006. An Eye-Tracking Methodology for Characterizing Program Comprehension Processes. In *Proc. Symposium on Eye Tracking Research & Applications (ETRA)*. ACM, 125–132.

[2] Anja Bethmann, Claus Tempelmann, Ria De Bleser, Henning Scheich, and André Brechmann. 2007. Determining Language Laterality by fMRI and Dichotic Listening. *Brain Research* 1133, 1 (2007), 145–157.

[3] Francis A. Billson. 2003. *Strabismus*. BMJ Books.

[4] Rolf Brickenkamp, Lothar Schmidt-Atzert, and Detlev Liepmann. 2010. *Test d2-Revision: Aufmerksamkeits-und Konzentrationstest*. Hogrefe Göttingen.

[5] Korbinian Brodmann. 2006. *Brodmann's Localisation in the Cerebral Cortex*. Springer.

[6] Ruven Brooks. 1983. Towards a Theory of the Comprehension of Computer Programs. *Int'l Journal of Man-Machine Studies* 18, 6 (1983), 543–554.

[7] 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 *Proc. Int'l Conf. Program Comprehension (ICPC)*. IEEE, 255–265.

[8] 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).

[9] B. Chance, Z. Zhuang, C. UnAh, C. Alter, and Lipton L. 1993. Cognition-Activated Low-Frequency Modulation of Light Absorption in Human Brain. *Proc. Nat'l Academy Sciences of the United States of America (PNAS)* 90, 8 (1993), 3770–3774.

[10] Andrew L. Cohen. 2013. Software for the Automatic Correction of Recorded Eye Fixation Locations in Reading Experiments. *Behavior Research Methods* 45, 3 (2013), 679–683.

[11] Andrew T. Duchowski. 2017. *Eye Tracking Methodology – Theory and Practice, Third Edition*. Springer.

[12] João Duraes, Henrique Madeira, João Castelhano, Isabel C. Duarte, and Miguel Castelo-Branco. 2016. WAP: Understanding the Brain at Software Debugging. In *Proc. Int'l Symposium Software Reliability Engineering (ISSRE)*. IEEE, 87–92.

[13] Wolfgang Einhäuser, Ueli Rutishauser, and Christof Koch. 2008. Task-Demands Can Immediately Reverse the Effects of Sensory-Driven Saliency in Complex Visual Stimuli. *Journal of Vision* 8 (2008), 2–2.

[14] Christopher Exton. 2002. Constructivism and Program Comprehension Strategies. In *Proc. Int'l Workshop Program Comprehension (IWPC)*. IEEE, 281–284.

[15] Sarah Fakhoury, Yuzhan Ma, Venera Arnaoudova, and Olusola Adesope. 2018. The Effect of Poor Source Code Lexicon and Readability on Developers' Cognitive Load. In *Proc. Int'l Conf. Program Comprehension (ICPC)*. IEEE, 11 pages.

[16] 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 *Proc. Int'l Conf. Software Engineering (ICSE)*. IEEE, 175–186.

[17] 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 *Proc. Int'l Conf. Software Engineering (ICSE)*. ACM, 402–413.

[18] Michael S. Gazzaniga, Richard B. Ivry, and George R. Mangun. 2013. *Cognitive Neuroscience: The Biology of the Mind*. Norton & Company.

[19] Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost Van de Weijer. 2011. *Eye Tracking: A Comprehensive Guide to Methods and Measures*. OUP Oxford.

[20] Anthony J. Hornof and Tim Halverson. 2002. Cleaning Up Systematic Error in Eye-Tracking Data by Using Required Fixation Locations. *Behavior Research Methods, Instruments, & Computers* 34, 4 (2002), 592–604.

[21] Scott Huettel, Allen Song, and Gregory McCarthy. 2014. *Functional Magnetic Resonance Imaging*. Vol. 3. Sinauer Associates.

[22] Jürgen Koenemann and Scott Robertson. 1991. Expert Problem Solving Strategies for Program Comprehension. In *Proc. Conf. Human Factors in Computing Systems (CHI)*. ACM, 125–130.

[23] Thomas D. LaToza, Gina Venolia, and Robert DeLine. 2006. Maintaining Mental Models: A Study of Developer Work Habits. In *Proc. Int'l Conf. Software Engineering (ICSE)*. ACM, 492–501.

[24] 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.

[25] Russell Mosemann and Susan Wiedenbeck. 2001. Navigation and Comprehension of Programs by Novice Programmers. In *Proc. Int'l Workshop Program Comprehension (IWPC)*. IEEE, 79–88.

[26] Christopher Palmer and Bonita Sharif. 2016. Towards Automating Fixation Correction for Source Code. In *Proc. Symposium on Eye Tracking Research & Applications*. ACM, 65–68.

[27] Norman Peitek, Janet Siegmund, and André Brechmann. 2017. Enhancing fMRI Studies of Program Comprehension with Eye-Tracking. In *Proc. Int'l Workshop on Eye Movements in Programming*. Freie Universität Berlin, 22–23.

[28] Norman Peitek, Janet Siegmund, Chris Parnin, Sven Apel, and André Brechmann. 2018. Toward Conjoint Analysis of Simultaneous Eye-Tracking and fMRI Data for Program-Comprehension Studies. In *Proc. Int'l Workshop on Eye Movements in Programming*. ACM, 1:1–1:5.

[29] Nancy Pennington. 1987. Stimulus Structures and Mental Representations in Expert Comprehension of Computer Programs. *Cognitive Psychology* 19, 3 (1987), 295–341.

[30] Kathrin Pollmann, Mathias Vukelić, Niels Birbaumer, Matthias Peissner, Wilhelm Bauer, and Sunjung Kim. 2016. fNIRS as a Method to Capture the Emotional User Experience: A Feasibility Study. In *Human-Computer Interaction. Novel User Experiences*. Springer International Publishing, 37–47.

[31] Bonita Sharif and Johnathon Maletic. 2010. An Eye Tracking Study on camelCase and under_score Identifier Styles. In *Proc. Int'l Conf. Program Comprehension (ICPC)*. IEEE, 196–205.

[32] Ben Shneiderman and Richard Mayer. 1979. Syntactic/Semantic Interactions in Programmer Behavior: A Model and Experimental Results. *Int'l J. Parallel Programming* 8, 3 (1979), 219–238.

[33] Janet Siegmund, André Brechmann, Sven Apel, Christian Kästner, Jörg Liebig, Thomas Leich, and Gunter Saake. 2012. Toward Measuring Program Comprehension with Functional Magnetic Resonance Imaging. In *Proc. Int'l Symposium Foundations of Software Engineering–New Ideas Track (FSE-NIER)*. ACM, 24:1–24:4.

[34] 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 *Proc. Int'l Conf. Software Engineering (ICSE)*. ACM, 378–389.

[35] Janet Siegmund, Christian Kästner, Jörg Liebig, Sven Apel, and Stefan Hanenberg. 2014. Measuring and Modeling Programming Experience. *Empirical Softw. Eng.* 19, 5 (2014), 1299–1334.

[36] 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 *Proc. Europ. Software Engineering Conf./Foundations of Software Engineering (ESEC/FSE)*. ACM, 140–150.

[37] Janet Siegmund, Norbert Siegmund, and Sven Apel. 2015. Views on Internal and External Validity in Empirical Software Engineering. In *Proc. Int'l Conf. Software Engineering (ICSE)*, Vol. 1. IEEE, 9–19.

[38] Elliot Soloway and Kate Ehrlich. 1984. Empirical Studies of Programming Knowledge. *IEEE Trans. Softw. Eng.* 10, 5 (1984), 595–609.

[39] Thomas Standish. 1984. An Essay on Software Reuse. *IEEE Trans. Softw. Eng.* SE–10, 5 (1984), 494–497.

[40] Adrian Voßkühler, Volkhard Nordmeier, Lars Kuchinke, and Arthur M. Jacobs. 2008. OGAMA (Open Gaze and Mouse Analyzer): Open-Source Software Designed to Analyze Eye and Mouse Movements in Slideshow Study Designs. *Behavior Research Methods* 40, 4 (2008), 1150–1162.

[41] Susan Wiedenbeck. 1991. The Initial Stage of Program Comprehension. *International Journal of Man-Machine Studies* 35, 4 (1991), 517–540.