Toward Gaze-assisted Developer Tools

Peng Kuang
Dept. of Computer Science
Lund University
Lund, Sweden
peng.kuang@cs.lth.se

Emma Söderberg
Dept. of Computer Science
Lund University
Lund, Sweden
emma.soderberg@cs.lth.se

Diederick C. Niehorster
Humanities Lab
Dept. of Psychology
Lund University
Lund, Sweden
diederick_c.niehorster@humlab.lu.se

Martin Höst
Dept. of Computer Science
Lund University
Lund, Sweden
martin.host@cs.lth.se

***Abstract*—Many crucial activities in software development are linked to gaze and can potentially benefit from gaze-assisted developer tools. However, despite the maturity of eye trackers and the potential for such tools, we see very few studies of practitioners. Here, we present a systematic mapping study to examine recent developments in the field with a focus on the experimental setup of eye-tracking studies in software engineering research. We identify two gaps regarding studies of practitioners in realistic settings and three challenges in existing experimental setups. We present six recommendations for how to steer the research community toward gaze-assisted developer tools that can benefit practitioners.**

***Index Terms*—eye tracking, gaze behavior, developer tools, software engineering, mapping study**

I. Introduction

One of the main activities of a software developer is reading code [1], [2], an activity driven by gaze. Still, research using eye trackers to study the gaze behavior of software developers during their day-to-day work is not common [3], [4]. Gaze-driven tools become even more attractive when considering the increase in remote work and online collaboration [5]. There are a number of activities in software engineering where we believe that gaze analysis and eye-tracking data could contribute to research, e.g., in cooperative work, both in the same location and geographically distributed, and in code comprehension.

Eye trackers used to be expensive and required trained personnel to operate, effectively confining their use to lab environments. But this is not the case anymore. In the last decades, rapid technical development has greatly increased the feasibility of in-situ acquisition of eye-tracking data, both by the availability of cheap dedicated eye trackers and eye tracking via commodity webcams. It is reasonable to assume eye trackers will soon be ubiquitous at developers' desks, integrated into monitors, laptops, and cell phones [6]–[8].

Considering research in software engineering, the possibility of using eye tracking in software development has been identified [9], [10] and we see an increase in studies that utilize eye trackers to understand e.g., program comprehension [3], [4]. Also, it can be utilized as a means for data collection in empirical software engineering research [11]. However, when considering the experimental setup of eye-tracking studies so far, we see a dominance of lab studies, with students and educational material [3]. We see very few studies considering the day-to-day activities performed in professional software engineering settings. Yet, there is evidence that experts and novices have different gaze patterns [12], [13]. How well can we build tools for practitioners with gaze data from students in an educational lab setting?

In this paper, we systematically review recent studies utilizing eye trackers in software engineering research to investigate recent trends (Section II). We find a persistent lack of eye-tracking experiments conducted outside of the lab environment and a dominance of student participants (Section III). With these results in mind, we discuss challenges in utilizing results and possible paths forward (Section IV).

TABLE I
Summary of steps in mapping study

	Steps
1.	Search in academic database, resulting in 509 papers.
2.	All authors review 10% of papers (titles and abstracts), with Kappa showing strong inter-rater agreement.
3.	Updated search in academic database, resulting in 513 papers.
4.	First author reviews all papers (titles and abstracts), resulting in 204 papers.
5.	All authors review 10% of papers (full content) in pairs and agree to keep all papers.
6.	All authors develop a data collection scheme together.
7.	Joint decision to focus on papers from 2018 and later, resulting in 136 papers.
8.	First author reviews remaining papers (full content), resulting in 86 papers.
9.	First author extracts data according to the data collection scheme, resulting in 71 papers.
10.	Joint analysis of data set and extraction of summaries by the first and second author.

II. Method

To develop a thorough overview of current use of eye-tracking in software engineering research and to address the question: *how is eye-tracking used in software engineering research?*, we conducted a mapping study [14] summarized in Table I. We explain the steps and the limitations of the study

This work was partly funded by the Swedish Foundation for Strategic Research (grant nbr. FFL18-0231), the Swedish Research Council (grant nbr. 2019-05658), and Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

below. Supplementary material with our data set is available online[1].

Mapping Study Steps In step 1, we conducted a search in Scopus[2], an "abstract and citation database" with references to articles from journals and conferences from established publishers (e.g., Elsevier, Springer, and IEEE). Scopus was chosen as a source because we deemed it to provide a representative view of available literature in the studied research area. We used the following search string (with line numbers added):

```
SUBJAREA ( comp ) AND                                    1
TITLE-ABS-KEY (                                          2
   (  {debugging} OR {programming} OR                    3
      {source code navigation} OR                        4
      {code browsing} OR {code search} OR                5
      {code review} ) AND                                6
   (  {eye tracking} OR {eyetracking} OR                 7
      {gaze} OR {eye movement} OR                        8
      {gaze estimation} ) ) AND                          9
( EXCLUDE ( DOCTYPE , "cr" ) )                           10
```

The first line focuses on the search in the area of computer science and the last line removes irrelevant documents (e.g., book reviews). Line 2 states that we search in titles, keywords, and abstracts, and the two main areas we require in each article are stated on lines 2–6 (programming) and 7–9 (eye tracking). The search resulted in 509 papers (April 22, 2022).

In step 2, we conducted a pilot review of the titles and abstracts of the first 10% of the found papers (51 papers) to develop an approach for sorting out irrelevant papers based on only titles and abstracts. We selected the 51 most recent papers to ensure this approach is based on the most recent research. All authors then reviewed these separately. The inclusion criteria we applied were: First, the stimuli must be code (studies solely with pseudocode stimuli were excluded); Second, the study must contain experiments with humans (but studies with children were excluded); Third, the study must be generating some data rather than entirely reusing preexisting data. The exclusion criteria we employed were: the paper is not in English, peer-reviewed, or available in full text. We then performed a Light's Kappa analysis of the inter-rater reliability of multiple raters, and got a Kappa score of 0.86, indicating that the magnitude of agreement is "strong" [15]. With the strong agreement, one author could proceed to review the rest of the papers.

Next, we re-ran the search (step 3, on 27 June 2022) since some time had passed, and this resulted in 4 additional papers, i.e., 513 papers in total. Then the first author proceeded with selecting the relevant papers (step 4), ending up with 204 potentially relevant papers.

In step 5, we selected the first 20 of the included papers (approximately 10%) to conduct a quality assurance study with the full text of each paper. We assigned two co-authors to cross-review each paper. We had six unique pairs which covers 18 of the 20 papers by repeating three times. For the remaining two papers, we deliberately assigned them to the pairs who had a relatively low agreement in the prior pilot review. After this process, we discussed our assessment of the quality of the papers we read. We unanimously agreed on the inclusion of 17 papers. For the remaining three papers, we used a majority voting mechanism to decide on the inclusion of two papers; we also included a paper when there was a tie. After review, all 20 examined papers were included. Since we had an agreement on which papers to remove and which to keep, we concluded that one author could review the rest of the papers.

In steps 6-9, we designed a data collection form (step 6) and used it to conduct pilot data extraction on the aforementioned papers. The first author also recorded a time estimate for reading each paper. We discussed and revised the form after this procedure. Based on review time and the existence of a previous study covering literature up until 2017 [3], we decided to limit the investigation to papers from 2018 and later (step 7). After that the first author reviewed the remainder of the papers (step 8), resulting in 86 papers, and also extracted data from the papers (step 9).

In the final step, we coded details about the experimental setup (the participants, stimuli/artifacts, devices, environment, and methods). We only considered distinct primary studies. That is, publications on the same data set were counted only once and it was the first or original study we considered. In total, we excluded 15 papers with this criterion, leaving 71 papers to be included in the reported results.

Limitations There are some limitations and threats to validity in this type of research (e.g., [16]). Even if a structured process for selecting papers is applied there is a risk, e.g., that some papers are rejected falsely. We have carefully reviewed papers and followed a process where we could reach a consensus about papers, which we believe increased the quality of the selection.

Also, it is not possible to get complete coverage of primary studies. However, we selected a well-known database and carefully designed the search string, which we believe increased the completeness. If very few primary studies were identified that could be seen as an indication that the selection was too narrow. However, in this case, we identify not so few relevant sources (in the magnitude of 15 per year). We compare our results to trends, but it is impossible to get a complete list of trends in the broader area. We base our analysis on our experience from research in the fields and we believe this experience to be sufficient to identify future paths. To not miss important perspectives, we spent effort on identifying and obtaining consensus in the review process.

III. Results

Fig. 1(a) shows the summary of roles of participants in the studies we found, split into students, practitioners, mixed, and unspecified. A study was labeled as having mixed participants when it included participants with more than one role, e.g., students and researchers [17], [18]. The data show that while there was no change in the number of studies using only students during the years covered by our review, there was a notable increase in the number of studies using participants

[1]https://portal.research.lu.se/files/137278187/ICSE_NIER_2023_KuangEtAl_artifact.xlsx

[2]https://www.scopus.com

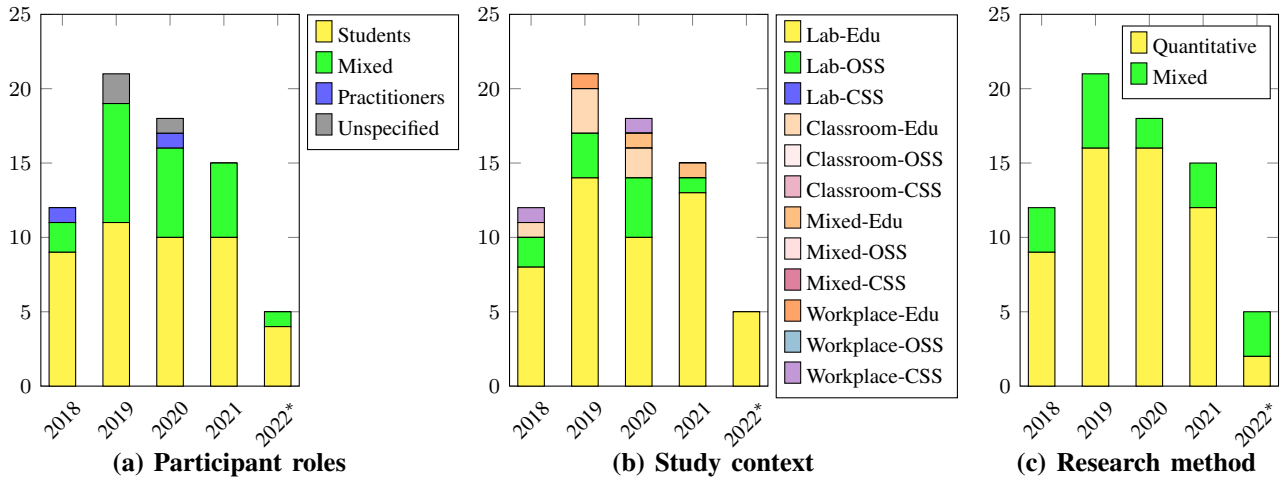

Fig. 1. Overview of experiment setup in selected papers split by year. The legends list coding categories for participant roles (a), study context (b) with a combination of location and material used, and study method. Note Edu=Educational, OSS=Open-Source Software, and CSS=Closed-source software. Mixed is used in all plots to indicate a combination of roles, contexts, or methods. *2022 includes research only up to June 27, 2022.

with more than one role in 2019–2021, compared to 2018. Across the years, the vast majority of studies (92.96%) used students. These studies used either only student participants (61.97%) or used students in combination with researchers (8.45%, e.g., [17], [18]), practitioners (12.68%, e.g., [19]–[21]) or both (8.45%, e.g., [22], [23]). Conversely, only 22.54% of studies included practitioners as participants.

Finding 1: The vast majority of studies (92.96%) used students in eye-tracking experiments, with 61.97% using only students. 22.54% of studies used practitioners.

Fig. 1(b) shows in what context (setting & material) the reviewed studies were performed, split into 12 combinations of setting and material. The categories with the "mixed" prefix indicate studies that were conducted in more than one setting, usually involving different participant roles, e.g., Lab for Students and Workplace for Researchers/Practitioners [21], [24].

We notice that laboratory studies using open-source software as the source of stimuli were more common in 2019 (e.g., [25], [26]) and 2020 (e.g., [18], [27], [28]) than the other years. Only two studies were done in the workplace with closed-source software [29], [30].

Finding 2: A majority of studies (84.51%) were conducted in a laboratory setting, either with educational materials (70.42%) or with materials adapted from open-source software (14.09%).

Fig. 1(c) shows what research methods were used by studies across the time period we reviewed. As the studies we examined all utilize eye tracking, the research method they adopted was either quantitative or mixed. Among these mixed studies, 68.75% (15.49% of all studies) used post-hoc/retrospective interviews to complement the quantitative analysis of eye movement data; the rest used either think-aloud, verbal Q&A, or a combination of the two.

Moreover, ten distinct studies adopted other sensors/devices, e.g., fMRI (functional Magnetic Resonance Imaging, e.g., [27], [31]), fNIRS (functional Near Infrared Spectroscopy, e.g., [28], [31]), EEG (electroencephalogram, e.g., [32], [33]) and HRV (heart rate variability, e.g., [22], [34]), either simultaneously or subsequently with eye trackers. Two studies used VR eye trackers instead of screen-based eye trackers to collect gaze data [35], [36].

Finding 3: A majority of studies (77.46%) used Quantitative research method while 22.54% adopted a Mixed approach.

IV. Discussion

Our data combined with previous data [3], suggests a clear increase in research studies involving eye-tracking in the area of software engineering. We ended up with 71 papers in our study, which constitutes an average of 15.78 papers per year (71/4.5), while in the previous survey by Obaidellah et al. [3] they found 2.33 papers per year over the range 1990-2017, with 6.4 papers per year for the last 5 years in their data set (2012-2017). We see this as an encouraging development toward the utilization of gaze in developer tooling, but we also see gaps in the areas covered by this research and challenges in how to conduct research in this space.

Gap: Gaze behavior of practitioners. The study participants are predominately students (**Finding 1**), a finding that aligns with previous surveys of the field [3], [4]. Although the argument can be made that students in some cases can be seen as representatives of junior engineers [4], this still leaves a gap regarding the gaze behavior of more experienced practitioners.

Gap: Gaze behavior in realistic setups. Studies primarily take place in a lab or classroom setting, with mainly educational material and occasionally with material from open-source (**Finding 2**). While we acknowledge efforts to bring in more realism, with stimuli from open-source, and of course the few gems studying practitioners in the workplace (e.g., [29], [30]), there is generally little knowledge about gaze behavior in realistic software development settings. This aligns with

previous surveys where *"most of the experiments reported in relation to programming were conducted in an institution for higher learning"* [3] (Section 3.4) with 75% of studies being carried out with students, teachers or researchers with a connection to programming courses and often with *"source code, texts, or models that can fit on one screen"* [4] (Section 5.5.3).

Challenge: Task incentive. Besides the realism of the setting and the stimuli, there is also something to be considered regarding the incentives of participants, which is not mentioned in earlier work [3], [4]. The incentive for doing a task may have an impact on the gaze behavior and thus create bias in findings. It is reasonable to assume that there could be a big gap between knowing that not finding a bug will have no severe consequences vs. knowing that not finding a bug (and fixing it) will affect thousands of users.

Challenge: Characterizing participant expertise. While it is plausible to divide participants into different groups and contrast them for studies, we observe the line researchers use to draw between novices and experts is not consistent in the literature. While freshman students are typically treated as novices and practitioners as experts, senior and graduate students may be treated as intermediate, advanced, or novices. Variation in role assignment makes it difficult to compare and replicate studies. While previous surveys show that studies of differences between novices and experts are common [3], [4], they do not list this challenge.

Beyond the immediate challenges in the current practice in expertise characterization, the notion of expertise is by nature very difficult to measure and context-dependent. Expertise, proficiency, and task difficulty are intertwined and interact with each other. Expertise can vary between languages; an expert in C can be a novice in JavaScript. Past language experience matters; being a novice in your fifth language is not the same as the first time you learn to program [37]. Expertise may also vary within a language; solving the same problem for the fifth time is not the same as the first time. Lastly, expertise can also degrade over time; 20 years of experience in C may not be the same after spending 5 years developing in Python.

Challenge: Building cross-study knowledge. Eye tracking research deals with massive amounts of data [4] and a high degree of attention and rigor is required when using eye trackers to generate valid data. In addition, processing and analyzing eye movement data is also challenging. There has been a lack of guidance in the field [3], [4], resulting in variation in experimental setups and problems with replication (e.g., [38], see also [39]). Variation in the meta-data of gathered data sets further makes it difficult to compare results and to combine results into larger data sets, to open up for automated pattern recognition via machine learning.

A. Toward gaze-assisted developer tools

With the presented gaps and challenges in mind, we present recommendations for how to drive the research in this area toward gaze-driven developer tools for practitioners.

Recommendation: Find paths to practitioners. While we acknowledge that it may be difficult for researchers to get access to practitioners [4], we still advocate for more studies with practitioners and preferably conducted in their natural work environment, a point also made by others [40]. We encourage researchers to take on the challenge of finding paths to practitioners, perhaps by considering other research methods or adaptions to their experimental setups. Are there ways to move to practitioners rather than moving practitioners to the lab? Are there ways to find a mutual benefit to their participation?

Recommendation: Let expertise be more complex. The current practice of characterizing expertise is to use years of programming, years of using a certain programming language, self-reported proficiency/confidence, and perceived task difficulty. While all these metrics are valid and relevant, we encourage researchers to consider alternative ways of characterizing expertise, capturing more of the nuance. Perhaps by capturing experience along more dimensions, for instance, past experience in different programming languages, and recent development activities.

Recommendation: Provide realistic incentives. In order to increase the validity it has been argued that not only expertise is important, but also the incentives for conducting tasks in experiments [41]. Incentives are probably clearer in an industrial setting, but incentives can be seen as orthogonal to subject experience, e.g. there may, to some extent, be clear incentives also in tasks that are part of student projects. We encourage researchers to obtain realistic incentives also with students as subjects.

Recommendation: Tee up for machine learning. As the field matures, practical guides for how to conduct eye-tracking experiments are emerging [40], as well as efforts to share data sets [42], [43]. High-quality data sets with gaze data open the door for machine learning and the incorporation of such techniques into developer tools for improved developer assistance. To enable this development, we encourage researchers to take on the challenge of creating larger data sets with gaze data from practitioners in realistic settings.

Recommendation: Characterize gaze in development. While shared gaze data sets enable the training of machine learning models, an important factor that makes such data sets useful lies in the richness of the meta-data that describe them (see also [44]). Describing the space where gaze plays a role in software development, is more complex than describing the space of an introductory programming course. While a course is simplified (e.g., typically focuses on one language and construction of small applications), software development is multi-faceted and complex (e.g., editing of multiple languages, remote collaboration with pair programming, code review, and whiteboard design discussions). We encourage researchers to take on the challenge of characterizing this "gaze space" in software development to enable knowledge building and gaze-driven tooling for software developers.

Recommendation: Involve participants in design. Software development is not only a cognitively-demanding task

but can also be an emotion-draining task. Gaze data, along with other bio-sensors, open up possibilities to detect part of a developer's emotional state [40]. This data can potentially enrich the existing services of developer tools and also create new ones focused on emotional user experiences [45]. However, there is a risk that the cost of sharing bio data may outweigh the benefit of the assistance. We recommend researchers consider participatory design methods [46] in exploring such services.

V. Conclusions

To investigate how eye-tracking is used in software engineering research, we carried out a systematic mapping study. We focused on the experimental setups in the last 5 years and found that the majority of experiments are carried out with students, in a lab setting and with educational material as stimuli. We identify gaps in terms of studies with practitioners in realistic settings, and we identify challenges in existing studies regarding task incentives, characterization of expertise, and variation in the practice around data gathering. To address these gaps and challenges, we present six recommendations aimed at steering the research community toward gaze-assisted developer tools useful for practitioners.

References

[1] R. Minelli, A. Mocci, and M. Lanza, "I Know What You Did Last Summer - An Investigation of How Developers Spend Their Time," in *Int. Conf. on Program Comprehension (ICPC)*, 2015, pp. 25–35.

[2] X. Xia, L. Bao, D. Lo, Z. Xing, A. E. Hassan, and S. Li, "Measuring Program Comprehension: A Large-Scale Field Study with Professionals," *Transactions on Software Engineering*, vol. 44, no. 10, 2018.

[3] U. Obaidellah, M. Al Haek, and P. C.-H. Cheng, "A survey on the usage of eye-tracking in computer programming," *ACM Comput. Surv.*, vol. 51, no. 1, 2018.

[4] Z. Sharafi, Z. Soh, and Y.-G. Guéhéneuc, "A systematic literature review on the usage of eye-tracking in software engineering," *Information and Software Technology*, vol. 67, pp. 79–107, 2015.

[5] L. Yang, D. Holtz, S. Jaffe, S. Suri, S. Sinha, J. Weston, C. Joyce, N. Shah, K. Sherman, B. Hecht, and J. Teevan, "The effects of remote work on collaboration among information workers." *Nature Human Behaviour*, vol. 6, pp. 43–54, 2022.

[6] Tobii, "Eye tracking fully integrated and baked right into the very latest high performance gaming devices from alienware, acer and msi," https://gaming.tobii.com/products/laptops/ (Oct 3, 2022).

[7] EIZO, "Eye Tracking Leader Tobii Pro Choose EIZOs Monitors for Behavioral Research Solution," https://www.eizoglobal.com/solutions/casestudies/tobii-pro/ (Oct, 2022).

[8] N. Valliappan, N. Dai, E. Steinberg, J. He, K. Rogers, V. Ramachandran, P. Xu, M. Shojaeizadeh, L. Guo, K. Kohlhoff, and V. Navalpakkam, "Accelerating eye movement research via accurate and affordable smartphone eye tracking," *Nature Communications*, vol. 11, 2020.

[9] B. Sharif and T. Shaffer, "The use of eye tracking in software development," in *Foundations of Augmented Cognition*, D. D. Schmorrow and C. M. Fidopiastis, Eds., 2015, pp. 807–816.

[10] B. Sharif, B. Clark, and J. I. Maletic, "Studying developer gaze to empower software engineering research and practice," in *Int. Sym. on Foundations of Software Engineering (FSE)*, 2016, p. 940–943.

[11] B. Sharif and N. Mansoor, "Humans in empirical software engineering studies: An experience report," in *Int. Conf. on Software Analysis, Evolution and Reengineering (SANER)*, 2022, pp. 1286–1292.

[12] T. Busjahn, R. Bednarik, A. Begel, M. E. Crosby, J. H. Paterson, C. Schulte, B. Sharif, and S. Tamm, "Eye movements in code reading: relaxing the linear order," in *Int. Conf. on Program Comprehension (ICPC)*, 2015, pp. 255–265.

[13] J. A. Saddler, C. S. Peterson, P. Peachock, and B. Sharif, "Reading behavior and comprehension of C++ source code - a classroom study," in *Augmented Cognition*, 2019.

[14] K. Petersen, S. Vakkalanka, and L. Kuzniarz, "Guidelines for conducting systematic mapping studies in software engineering: An update," *Information and Software Technology*, vol. 64, pp. 1–18, 2015.

[15] A. J. Conger, "Integration and generalization of kappas for multiple raters," *Psychological Bulletin*, vol. 88, 1980.

[16] D. Budgen, P. Brereton, S. Drummond, and N. Williams, "Reporting systematic reviews: Some lessons from a tertiary study," *Information and Software Technology*, vol. 95, pp. 62–74, 2018.

[17] J. Costa, R. Gheyi, M. Ribeiro, S. Apel, V. Alves, B. Fonseca, F. Medeiros, and A. Garcia, "Evaluating refactorings for disciplining #ifdef annotations: An eye tracking study with novices," *Empirical Software Engineering*, vol. 26, 2021.

[18] B. De Oliveira, M. Ribeiro, J. A. S. Da Costa, R. Gheyi, G. Amaral, R. De Mello, A. Oliveira, A. Garcia, R. Bonifacio, and B. Fonseca, "Atoms of Confusion: The Eyes Do Not Lie," *Int. Conf. Proc. Series*, p. 243 – 252, 2020.

[19] T. Busjahn and S. Tamm, "A Deeper Analysis of AOI Coverage in Code Reading," in *Eye Tracking Research and Applications Sym. (ETRA)*, 2021.

[20] C. S. Peterson, "Investigating the effect of polyglot programming on developers," in *Sym. on Visual Languages and Human-Centric Computing (VL/HCC)*, 2021, pp. 1–2.

[21] S. N. Emhardt, E. M. Kok, H. Jarodzka, S. Brand-Gruwel, C. Drumm, and T. van Gog, "How experts adapt their gaze behavior when modeling a task to novices," *Cognitive Science*, vol. 44, no. 9, 2020.

[22] H. Hijazi, J. Cruz, J. Castelhano, R. Couceiro, M. Castelo-Branco, P. de Carvalho, and H. Madeira, "ireview: an intelligent code review evaluation tool using biofeedback," in *Int. Sym. on Software Reliability Engineering (ISSRE)*, 2021, pp. 476–485.

[23] F. Hauser, S. Schreistter, R. Reuter, J. H. Mottok, H. Gruber, K. Holmqvist, and N. Schorr, "Code reviews in C++: Preliminary results from an eye tracking study," in *Eye Tracking Research and Applications Sym. (ETRA)*, 2020.

[24] S. Aljehane, B. Sharif, and J. Maletic, "Determining differences in reading behavior between experts and novices by investigating eye movement on source code constructs during a bug fixing task," *Eye Tracking Research and Applications Sym. (ETRA)*, vol. PartF169257, 2021.

[25] M. Ahrens, K. Schneider, and M. Busch, "Attention in software maintenance: An eye tracking study," in *Int. Workshop on Eye Movements in Programming (EMIP)*, 2019, p. 2 – 9.

[26] N. J. Abid, B. Sharif, N. Dragan, H. Alrasheed, and J. I. Maletic, "Developer reading behavior while summarizing java methods: Size and context matters," in *Int. Conf. on Software Engineering (ICSE)*, 2019, p. 384 – 395.

[27] Y. Huang, K. Leach, Z. Sharafi, N. McKay, T. Santander, and W. Weimer, "Biases and differences in code review using medical imaging and eye-tracking: Genders, humans, and machines," in *European Software Engineering Conf. and Sym. on the Foundations of Software Engineering (ESEC/FSE)*, 2020, p. 456 – 468.

[28] S. Fakhoury, D. Roy, Y. Ma, V. Arnaoudova, and O. Adesope, "Measuring the impact of lexical and structural inconsistencies on developers' cognitive load during bug localization," *Empirical Software Engineering*, vol. 25, no. 3, p. 2140 – 2178, 2020.

[29] H. Vrzakova, A. Begel, L. Mehtätalo, and R. Bednarik, "Affect recognition in code review: An in-situ biometric study of reviewer's affect," *Journal of Systems and Software*, vol. 159, p. 110434, 2020.

[30] A. Begel and H. Vrzakova, "Eye movements in code review," in *Workshop on Eye Movements in Programming (EMIP)*, 2018.

[31] Z. Sharafi, Y. Huang, K. Leach, and W. Weimer, "Toward an objective measure of developers' cognitive activities," *Transactions on Software Engineering and Methodology*, vol. 30, no. 3, 2021.

[32] Y.-T. Lin, Y.-Z. Liao, X. Hu, and C.-C. Wu, "Eeg activities during program comprehension: An exploration of cognition," *IEEE Access*, vol. 9, p. 120407 – 120421, 2021.

[33] S. Lee, D. Hooshyar, H. Ji, K. Nam, and H. Lim, "Mining biometric data to predict programmer expertise and task difficulty," *Cluster Computing*, vol. 21, no. 1, p. 1097 – 1107, 2018.

[34] K. Mangaroska, K. Sharma, D. Gašević, and M. Giannakos, "Multimodal learning analytics to inform learning design: Lessons learned from computing education," *Journal of Learning Analytics*, vol. 7, no. 3, p. 79 – 97, 2020.

[35] J. C. Hung and C.-C. Wang, "The influence of cognitive styles and gender on visual behavior during program debugging: A virtual reality

eye tracker study," *Human-centric Computing and Information Sciences*, vol. 11, 2021.

[36] C.-C. Wang, J. C. Hung, S.-C. Wang, and Y.-M. Huang, "Visual attention analysis during program debugging using virtual reality eye tracker," *Lecture Notes in Computer Science*, vol. 11937, p. 97 – 106, 2019.

[37] G. Simhandl, P. Paulweber, and U. Zdun, "Design of an executable specification language using eye tracking," in *Workshop on Eye Movements in Programming (EMIP)*, 2019, pp. 37–40.

[38] N. Peitek, J. Siegmund, and S. Apel, "What drives the reading order of programmers? an eye tracking study," in *Int. Conf. on Program Comprehension (ICPC)*, 2020, p. 342–353.

[39] K. Holmqvist, S. L. Örbom, I. T. C. Hooge, D. C. Niehorster, ..., and R. S. Hessels, "Eye tracking: empirical foundations for a minimal reporting guideline," *Behavior Research Methods*, 2022.

[40] Z. Sharafi, B. Sharif, Y.-G. Guéhéneuc, A. Begel, R. Bednarik, and M. Crosby, "A practical guide on conducting eye tracking studies in software engineering," *Empirical Software Engineering*, vol. 25, p. 3128–3174, 2020.

[41] M. Höst, C. Wohlin, and T. Thelin, "Experimental context classification: incentives and experience of subjects," in *Int. Conf. on Software Engineering (ICSE)*, 2005, pp. 470–478.

[42] I. McChesney and R. Bond, "Eye tracking analysis of code layout, crowding and dyslexia - an open data set," in *Sym. on Eye Tracking Research and Applications*, 2021.

[43] R. Bednarik, T. Busjahn, A. Gibaldi, A. Ahadi, M. Bielikova, ..., and I. van der Linde, "Emip: The eye movements in programming dataset," *Science of Computer Programming*, vol. 198, p. 102520, 2020.

[44] D. C. Niehorster, M. Hildebrandt, A. Smoker, H. Jarodzka, and N. Dahlströhm, "Towards eye tracking as a support tool for pilot training and assessment," in *Int. Workshop on Eye-Tracking in Aviation (ETAVI)*, 2020, pp. 17 – 28.

[45] K. Pollmann, M. Vukelić, N. Birbaumer, M. Peissner, W. Bauer, and S. Kim, "fnirs as a method to capture the emotional user experience: A feasibility study," in *Human-Computer Interaction. Novel User Experiences*, 2016.

[46] T. Robertson and J. Simonsen, "Challenges and Opportunities in Contemporary Participatory Design," *Design Issues*, vol. 28, no. 3, pp. 3–9, 2012.