Data Analysis Tools Affect Outcomes of Eye-Tracking Studies

Timon Dörzapf
Saarland University,
Saarland Informatics Campus
Saarbrücken, Germany

Norman Peitek
Saarland University,
Saarland Informatics Campus
Saarbrücken, Germany

Marvin Wyrich
Saarland University,
Saarland Informatics Campus
Saarbrücken, Germany

Sven Apel
Saarland University,
Saarland Informatics Campus
Saarbrücken, Germany

ABSTRACT

Background: Eye-tracking studies in software engineering offer insights into the thought processes of developers and their interaction with visual information. However, the analysis of eye-tracking data lacks established standards. Particularly concerning for the comparability of study findings is the large variety of tools for evaluating eye-tracking data, all of which use different algorithms and preset parameters, often undisclosed.
Aims: Our study has three objectives: (1) characterizing the analysis tool landscape for eye-tracking data within software-engineering research; (2) analyzing justifications by study authors for their choice of a particular analysis tool; and (3) investigating whether the choice of the analysis tool affects study results and conclusions.
Method: First, we conducted a systematic mapping study to identify reports of eye-tracking studies in software engineering, from which we extracted analysis tools used and authors' justifications. Second, we reproduced the statistical analyses on three publicly available eye-tracking data sets from the original studies using two different analysis tools.
Results: We found that the most frequently used analysis tools are Tobii Software, iTrace, and Ogama, although around a third of the papers did not mention the used analysis tool at all. The choice of analysis tool is also rarely justified. Most importantly, our case studies revealed that the choice of analysis tool significantly affects study results.
Conclusions: The lack of standardization in analyzing eye-tracking data poses a significant threat to the comparability and reproducibility of eye-tracking studies in software engineering. Greater importance should be attached to this aspect in existing reporting guidelines for eye-tracking studies.

CCS CONCEPTS

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

ESEM '24, October 24–25, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-1047-6/24/10.
https://doi.org/10.1145/3674805.3686672

KEYWORDS

Eye tracking, data analysis tools, software engineering

ACM Reference Format:
Timon Dörzapf, Norman Peitek, Marvin Wyrich, and Sven Apel. 2024. Data Analysis Tools Affect Outcomes of Eye-Tracking Studies. In *Proceedings of the 18th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '24), October 24–25, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3674805.3686672

1 INTRODUCTION

Staring at their screen and looking carefully for a bug in the code, software developers unconsciously pay attention to things that are difficult for a human observer to recognize. Some fixations are barely longer than a few milliseconds. Yet, insights into the locations on which a developer's eyes linger for some time during code comprehension and into the locations between which they frequently jump back and forth would be of great importance to identify complex code locations. Fortunately, eye tracking helps with this task.

The number of software-engineering studies that use eye tracking to collect data has risen sharply in recent years [25, 29]. In the field of code comprehension research, for example, the methodology is one of several promising psycho-physiological measurement methods that are expected to provide more profound insights into the thought processes of developers [9, 27, 29]. Moreover, the cost of an eye tracker is no longer a major barrier to its use, as the cheapest devices cost around a hundred euros [24].

This, however, does not mean that the design and conduct of eye-tracking studies are straightforward. There are numerous decisions to be made, such as the choice of the specific eye tracker, data analysis tool, and algorithms (e.g., fixation classification). While guidelines attempt to provide clarity, there is no commonly agreed way of performing and analyzing eye-tracking experiments in software engineering [23–25]. A lack of knowledge about the consequences of certain design decisions affects the comparability of study results, though. In particular, there is a large variety of tools for analyzing eye-tracking data, some of which do not even disclose their analysis algorithms [24, 25].

In this paper, we investigate the effect of the choice of the analysis tools on the outcome of an eye-tracking study. To this end, we formulate three research questions:

- **RQ1:** Which analysis tools are used to analyze eye-tracking data in software-engineering studies?

- **RQ2:** How do authors justify their choices of analysis tools, and what consequences do they discuss?
- **RQ3:** How does using different analysis tools on the same experiment data affect experimental results and conclusions?

In a first step, we unravel how authors of past publications have viewed the importance of the chosen tool: We conduct a systematic mapping study and find that only a minority reports and discusses their chosen tool. Since we further suspect that the choice of an analysis tool for eye-tracking data could affect the results and implications of a study, we conduct three case studies in a second step in which we use different eye-tracking data analysis tools on publicly available data sets for statistical analysis and compare the outcomes. We find that different analysis tools can substantially influence not only the numerical results, but even statistical conclusions of empirical eye-tracking studies.

In summary, we make the following contributions:

- A systematic survey of 97 eye-tracking studies in software engineering regarding the used analysis tools and the authors' justification for the choice of these tools;
- A comparative data analysis with common eye-tracking data analysis tools on three publicly available data sets;
- Recommendations for the research community on how best to deal with the sometimes necessary heterogeneity in the eye-tracking tool landscape to ensure comparable study results in the future;
- An online replication package[1] including extracted data and executable analysis scripts.

2 BACKGROUND ON EYE TRACKING

There is a wide variety of eye trackers available for scientific and commercial purposes [25]. Depending on the eye tracker, they may have a different form or use different methods to track eye gaze. An eye tracker normally consists of the following hardware and software components [24]:

- One or more (usually infrared) cameras
- One or more (usually infrared) light sources
- Image-processing software that detects and locates the eyes and the pupils and maps eye motion and the stimulus
- Data collection software to collect and store real-time eye-gaze data
- Real-time display showing the location of the eyes' focus.

Eye trackers are used to observe the eye movements of a person. This enables monitoring a person's visual attention, for example, when working on a task involving a visual stimulus. For a study in software engineering, such stimulus may be a code snippet with the task of comprehending it.

Typically, raw eye-movement data comprise the timestamp of the measurement and the x and y coordinates of the eye's position on the screen. Depending on the used camera or software, supplementary parameters may be recorded, including positions for both eyes, pupil dilation, and other pertinent data.

Eye-gaze data, obtained by processing the raw eye-movement data, can be classified into the following, which are the most important categories for our study [7]:

- *Fixation*: eye movement that stabilizes the retina, over a stationary object of interest. They range in duration from 150 to 600 ms, but this changes depending on the task and the characteristics of the person.
- *Saccade*: rapid eye movements that are used to reposition the fovea, the central pit of the retina of the eye, to a new location in the visual environment. Saccades range in duration from 10 to 100 ms, which renders the person effectively blind during the transition. They occur between fixations.

The interpretation of eye-tracking data is based on two fundamental assumptions [15]: a person tries to interpret a stimulus as soon as they see it (*immediacy assumption*) and a person fixates their attention on a stimulus until they understand it (*eye-mind assumption*). Saccades and fixations then often form the basis for the analysis of more complex visual effort metrics.

2.1 Visual Effort Metrics

Several metrics quantify a person's visual effort, which are representative of the task and stimuli being assessed [23]. The most common metrics are fixation-based metrics and saccade-based metrics.

Researchers may investigate the fixation count, that is the total number of fixations on a stimulus or in specific *areas-of-interest* (AOIs). For example, if code comprehension is the task of interest, a relevant AOI could be a function name indicating the purpose of the function. For example, a higher number of fixations on a specific stimulus may indicate that the search for relevant information is inefficient [10]. Another metric is the average fixation time, which is the sum of the durations of all fixations divided by the number of fixations. Finally, there is the fixation time, which is the sum of the durations of all fixations.

Similarly, researchers may analyze the saccade data. For example, they may count the number of saccades as well as their average and total duration to understand the visual attention of participants.

2.2 Eye-Tracking Analysis Tools

Eye-tracking analysis tools are used to process and analyze the data captured by eye trackers. There are several aspects to consider when choosing an analysis tool. First, these tools can be commercial or open-source software. Commercial software is often provided by the manufacturer of the eye tracker. Second, there is a difference of implemented preprocessing and analyses. Most analysis tools offer an integrated algorithm to detect fixations and saccades from the raw data. However, while some tools only provide basic visual effort metrics, some tools also allow for the computation of more advanced metrics (e.g., scanpath analyses [13]) or to visualize the data in different ways (e.g., heatmaps [13]).

Fixation and Saccade Algorithms. In the context of fixation and saccade analysis, various algorithms are used to differentiate between fixations and saccades based on velocity and dispersion criteria. The identification by velocity threshold algorithm, introduced by Salvucci and Goldberg [21], employs a predefined velocity threshold to classify data into fixations and saccades. Fixations are segments with point-to-point velocities below this threshold, while saccades exceed it. Another commonly used algorithm is the identification by dispersion-threshold algorithm, which operates on both x and

[1] https://github.com/brains-on-code/eye-tracking-tools-influence

y coordinates, requiring data segments to meet minimum duration and dispersion thresholds to be classified as fixations. The identification by minimal spanning tree algorithm constructs a "tree" representation of data, minimizing branching to distinguish samples from distinct clusters and effectively identify saccades. These algorithms provide foundational methods for analyzing eye movement data and are crucial for various research applications. New algorithms, such as I2MC [12], are continuously developed to improve the accuracy and efficiency of eye movement analysis [3].

3 RELATED WORK

Although our research on the impact of the selection of eye-tracking tools is novel, some studies exist on related topics. Sharafi et al. performed a systematic literature review (SLR) in 2015 on the usage of eye tracking as a methodology in software engineering [25]. They analyzed 36 papers between 1990 and 2014 to evaluate the current state of the art of eye tracking. An identified limitation of current eye-tracking studies is using several different tools to analyze eye-tracking data. The authors named Taupe and Ogama as open-source software working with many eye trackers.

In a follow-up paper, Sharafi et al. commented further on the lack of standardized protocols and tools, making comparisons across studies difficult [24]. Commercial eye-tracker suppliers would typically provide their closed-source data analysis tools.

In recent years, efforts have been made to create more open-source software to analyze eye-tracking data [6, 11, 30]. These tools can be used with frequently used eye trackers, which makes them good alternatives to the paid closed-source software of commercial eye-tracker suppliers. Zyrianov et al. present `Deja Vu` [30], which offers recording eye tracking and all telemetry data such that high-speed, high-quality eye trackers can overcome their real-time limitations of mapping screen coordinates to lines and columns, which in turn facilitates code comprehension studies.

Andersson et al. provided an overview of the currently available algorithms to parse eye-movement data [3]. They tested ten different algorithms on the same data set to provide a comparison between those algorithms and to evaluate which algorithms perform best. They found that depending on the algorithm, the resulting event duration showed a great variance. For static stimuli, fixation and saccade detection were working relatively well, but for all other measures or for dynamic stimuli, the algorithms did not provide good results. They also found that the LNS algorithm [16] was the best-performing algorithm for saccade detection.

The issue of reproducibility has been widely discussed within the field of software engineering [17, 20, 26]. Current approaches to empirical replications are problematic, to the point that some studies may even not be reproducible at all [26]. Additionally, multiple factors undermine the credibility of the results [20]. Moreover, only a restricted number of studies are replicated, which is a missed opportunity to independently verify the results [17].

Clearly, a lack of reproducibility and therefore standardized data analysis procedure has been a longstanding issue. The growing number of open-source analysis tools makes it difficult to choose the right tool for a study. In addition, available algorithms may further impact the results of a study. Thus, understanding which analysis tools are used in eye-tracking studies in software engineering and how their results compare is crucial.

4 PART I: EYE-TRACKING TOOL LANDSCAPE

In the first part of our work, we conduct a systematic mapping study [19] to address RQ1 and RQ2. We structure published primary eye-tracking studies in software engineering regarding several predefined characteristics. Our approach is largely based on that of Sharafi et al. [25], but we describe all the details of our procedure in the following subsections. The objective of this first part is to answer RQ1 and RQ2, that is, which eye-tracking analysis tools have been used in the past and how this was justified by the authors of the primary studies. Figure 1 provides a high-level schematic overview of our two-part research approach.

4.1 Search Process

We searched for eye-tracking studies that used stimuli related to software engineering through the digital libraries of IEEEXplore[2] and ACM[3]. For both libraries, we defined three sets of keywords, one of each had to be present in the paper:

```
(("eye-track*" OR "eye track*" OR "eyetrack*") AND
(code OR program* OR representation*) AND
(comprehen* OR understand* OR debug* OR read* OR scan*))
```

These are almost identical to those of Sharafi et al. [25], except for the additional stipulations that papers using restricted focus viewer (RFV)[4] are excluded, and that eye-tracking experiments must be performed in front of a screen.

4.2 Inclusion and Exclusion Criteria

After collecting papers through the digital libraries, we filtered them based on predefined inclusion and exclusion criteria. Again, we used nearly the same inclusion criteria as Sharafi et al. [25] with two additional exclusion criteria (E5 and E6):

- I1: Primary studies published in journals or conference and workshop proceedings in the form of experiments, surveys, case studies, reports, and observation papers using eye tracking to investigate software-engineering activities.
- I2: Primary studies that present more detailed and complete results if there is more than one published version of a specific study.

The exclusion criteria are the following:

- E1: Papers that do not use an eye tracker or that use a RFV.
- E2: Papers that are not related to software engineering
- E3: Papers that are not published in English.
- E4: Papers where eye-tracking experiments were not conducted in front of a screen.
- E5: Papers re-reporting the results, or doing a re-analysis of a previously published experiment.
- E6: Papers not reporting an empirical study, but, for example, only propose a proof of concept.

[2] https://ieeexplore.ieee.org
[3] https://dl.acm.org/
[4] RFV is a technique in which researchers limit participants' attention to a narrow view, which is not relevant for our research goals.

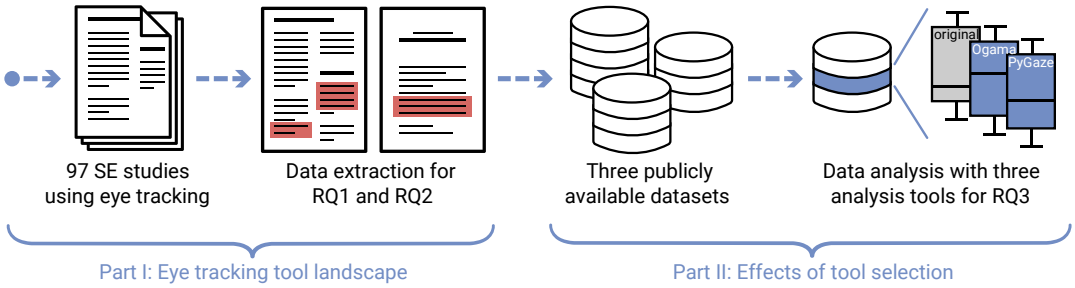

Figure 1: Schematic representation of our two-part research approach

4.3 Data Extraction

After applying the inclusion and exclusion criteria, we extracted several data items for each included paper. Data items are summarized in Table 1 and further described in our supplemental materials.

Table 1: Extracted data items for our systematic mapping study (RQ1 and RQ2)

Category	Data Items
Included papers	Title, DOI, citation (APA), publication year, venue
Analysis tools	Analysis tool used, analysis tool justifications and consequences
Eye tracker	Eye tracker used, eye-tracker configuration
Study design	Task of interest, language, scrolling
Participants	Number of participants, demographics of participants
Replication available	Replication package

To ensure that we extracted the data consistently, we created a data-extraction form and let the first author extract all included papers. A copy of the form is provided in the replication package.

We extracted most of the data items as direct data points from the papers. For the items *study task*, and *demographics* a few categories were defined beforehand and this selection was expanded where necessary, to which data from the papers was then mapped. Data on *analysis tool justifications and consequences* were inductively coded and classified using thematic analysis [5]. To this end, we first extracted the relevant passage as a quote. From this quote, we created labels that described the quote as accurately as possible. Finally, we derived abstract categories from the individual labels. The extraction was performed by the first author.

4.4 Results and Discussion for RQ1 & RQ2

Our search produced 173 results for IEEEXplore and 136 for ACM. We added the 36 papers of Sharafi et al. receiving a total of 348 related papers. After performing duplicate removal and applying our inclusion/exclusion criteria, we analyzed a total of 97 papers.

> **RQ1** Which analysis tools are used to analyze eye-tracking data in software-engineering studies?

We found that, overall, 32 out of the 97 papers did not mention the analysis tool used to analyze the eye-tracking data. Of the papers that named an analysis tool, the most used analysis tools are Tobii software[5] (n=15), followed by iTrace (n=11) and Ogama (n=10). In four cases, no analysis tool was used, and in three cases, custom tools, which the authors created themselves, were used to analyze the eye-tracking data. Not all of those custom tools were publicly available. The most-used eye tracker was by far the Tobii eye tracker (n=56), followed by EyeLink (n=8) and GazePoint (n=7). Eleven studies applied only specific algorithms to analyze the eye-tracking data instead of an analysis tool.

As Tobii eye trackers are the most used eye trackers, we also compared the analysis tools used to analyze the eye-tracking data for Tobii eye trackers. We found that Tobii software (n=15) was the most used analysis tool to analyze the eye-tracking data of Tobii eye trackers, followed by Ogama (n=10) and Taupe (n=4). Notably, 23 studies (≈ 41%) did not name the analysis tool used to analyze the eye-tracking data of Tobii eye trackers.

We also categorized the most-used analysis tools, Ogama, Tobii Software, and iTrace depending on which study task was performed. Tobii software and Ogama were only used for program comprehension and debugging tasks, whereas iTrace has more variety as it also covered non-code comprehension and traceability tasks.

When comparing the different eye-tracking metrics that were analyzed, we found that the most used metrics were fixations (n=44), followed by saccades (n=19) and AOIs (n=19). In addition, a lot more different metrics were analyzed, but most were only used a few times.

> **RQ1** The most frequently used analysis tool is Tobii software (n=15), followed by iTrace (n=11) and Ogama (n=10). There are 32 papers (33%) that do not mention the analysis tool used.

Discussion of the Results for RQ1. The high number of studies (≈ 33%) that did not name the analysis tool used to analyze the eye-tracking data is concerning. Different analysis tools may lead to different results, as we will investigate in RQ3, which impacts the reproducibility and validity of the results. There is a clear need for a comprehensive reporting structure that includes the used analysis

[5]The term Tobii software encompasses every analysis tool developed by Tobii, including multiple versions and changed names over time.

tool. Holmqvist et al. [14] proposed reporting guidelines for eye-tracking studies. Among the strongly recommended aspects to be reported were the adequate description of the data processing and analysis steps with all relevant parameters and the reporting of firmware and software versions (where applicable).

In the same vein, around 41% of the studies did not name the analysis tool they used in combination with a Tobii eye tracker. We assume that most of these studies used the proprietary Tobii analysis tool, but did not name it in the publication, as the authors may have considered it unnecessary. Even though it may make sense to them, it is not clear for other researchers and, therefore, should still be named in any case. We discuss the implications for reporting in more detail in Section 5.6.2.

It is not surprising that Tobii software is the primary tool for Tobii eye-tracker data analysis, given its association with the eye tracker's manufacturer. It is likely favored for its seamless integration and lack of need for external software exploration. iTrace emerges as the second most-used tool, likely due to its unique scrolling support during eye-tracking measurements. This facilitates more natural participant task execution, which is especially beneficial in software-engineering studies where scrolling is prevalent. Ogama's popularity, although declining, is attributed to its open-source nature and wide compatibility. It has been established in the eye-tracking community since its creation in 2007. However, its usage has decreased due to aging algorithms and discontinued support since 2016. Twelve studies relied solely on algorithms without analysis tools and thus risk implementation-specific variations that could potentially influence results. On the other hand, five studies used custom tools, which face reproducibility challenges and limited applicability beyond the scope of a single study.

For both Ogama and Tobii Software, the categories of study tasks were program comprehension and debugging. Only iTrace also covered non-code comprehension and traceability. This is probably due to missing features in Ogama and Tobii Software, which are needed for specific studies. For example, traceability is a study task not supported by Ogama and only partially by Tobii Software while being fully supported by iTrace.

> **RQ2** How do authors justify their choices of analysis tools and what consequences do they discuss?

During the data collection process, we gathered the justifications and consequences that the authors named for using a specific analysis tool. Then, we mapped them to labels, which we compiled into categories. Out of 97 collected studies, 66 studies did not name any justifications or consequences. We analyzed the remaining 31 studies and present the results in Table 2 and Table 3.

In total, we identified four different categories for justifications, which we show in Table 2. Of these categories, the feature category was mentioned the most and contained the highest number of labels. The most named label of the feature category was the ability to support scrolling during the eye-tracking measurements with five mentions. The second most named label was the ability to map the eye gaze to meaningful elements on the screen with four mentions. The non-functional characteristics relate to some characteristics of the analysis tool, such as simplicity of use, ease of modifying the analysis tool for its own needs, or that the analysis tool is open source. All labels that were identified were unique. The visualization category mostly consisted of the advantage of having different results from the eye-tracking measurements visualized, such as fixations. In the support category, the most common label was the support of several eye trackers. The other labels were support for a Macintosh computer and synchronization of the eye-tracking data and the data of a galvanic skin response sensor.

Table 2: Categories of authors' justification for their analysis tool selection

Category	Mentions	Labels	Example Description
Features	16	8	Specific abilities of the analysis tool
Non-functional characteristics	8	8	Characteristics of the analysis tool that are not of functional nature
Visualization	5	4	Visualization techniques
Support	4	3	Different ways the analysis tool supports other hardware/software

Table 3: Categories of authors' stated consequences for their analysis tool selection

Category	Mentions	Labels	Example Description
Features	7	6	Specific inabilities of the analysis tool
Non-functional characteristics	4	3	Characteristics of the analysis tool that are not of functional nature
Support	1	1	Different ways the analysis tool does not support other hardware/software

For the discussed consequences, we identified three different categories (see Table 3). As with the justifications categories, the feature category was the most mentioned one and had the highest number of labels. Only one label was mentioned twice, which is the inability to collect eye-tracking data from outside the IDE window. The other labels were all naming missing features that would have been beneficial for their analysis of the eye-tracking data. The non-functional characteristics category had one label, the inaccuracy of the analyzed data, mentioned twice. The other two labels were about the difficulty of finding optimal parameters to analyze the eye-tracking data correctly, and about the analysis tool being closed source. The support category has only one label, which is that there is no synchronization of the measured data between functional magnetic resonance imaging (fMRI) and an eye tracker.

> **RQ2** We found that the majority of papers (66 of 97) did not justify their choice of analysis tools. When justified, authors typically described functional requirements as their reason for a specific tool for analyzing their eye-tracking data. Authors also rarely discuss the consequences of choosing a specific tool.

Discussion of the Results for RQ2. Most studies did neither justify their choice of an analysis tool nor discuss its potential consequences. This could be because the authors may have thought that the analysis tool they used would not make a difference to the results. It could also be that they did not know any other analysis

tools and therefore did not have a choice. Still, we would like to stress that it is important to name the justifications or consequences for the choice of analysis tool, as different analysis tools may have different algorithms to analyze the eye-tracking data and thus may come to different results, as evident in the results of RQ3.

A total of 33 justifications and only 12 consequences were named. Researchers mostly discuss why a specific analysis tool was chosen, but did not explicitly consider the consequences of their choice often.

5 PART II: EFFECTS OF TOOL SELECTION

In Part I, we established that different analysis tools are used to analyze eye-tracking data. Furthermore, authors are not always reporting the used analysis tool and the reasoning for their choice. In this section, we aim to understand the effect of using different tools and pose the following research question:

> **RQ3** How does using different analysis tools on the same experiment data affect experimental results and conclusions?

To answer this research question, we conduct three case studies. We re-analyze existing data sets from published studies with different analysis tools to understand the effect on their results.

5.1 Choice of Data Sets

We selected three case studies with the following process: Of the 97 analyzed studies from Part I, 28 studies provided openly accessible replication packages. These were mentioned either in the paper or in the supplemental materials section on the publisher's website. Of these 28 studies with replication packages, 11 were unavailable or inaccessible to us, probably due to their age. We ranked the remaining replication packages with the following criteria in order of importance:

(1) Data contain raw eye-tracking data
(2) Eye-tracking results are important for the insights of the study
(3) Eye-tracking results can be used to compare different analysis tools
(4) Lowest amount of data that cannot be used for analysis
(5) Greatest number of data points
(6) The used analysis tool is not one of the analysis tools that are used in our data analysis.

Most data sets were excluded because the data were unavailable or did not contain raw eye-tracking data. After applying the first three criteria, we were left with the two papers: Peitek et al. [18] and Sharafi et al. [22]. Those also fulfilled the remaining criteria.

Additionally, we considered the public EMIP data set [4]. [6] While the EMIP data set was used for several publications, it does not contain an analysis in itself. Nevertheless, it is an interesting candidate as it offers a comprehensive set of raw eye-tracking data that can be used for a range of analytical purposes. The availability of the raw eye-tracking data allows for a straightforward comparison of studies using these data. For our purpose, we therefore focus on the study of Aljehane et al. [2], which used the EMIP data set but was not part of the mapping study, as it was excluded due to exclusion criterion E5, and reported specific results that allow for a comparison with results from an analysis with different analysis tools.

[6] https://www.emipws.org/dataset/

5.2 Data Analysis

In Section 4, we established that there is a plethora of analysis tools available. In this part, we focus on openly available analysis tools. We analyzed the previously chosen data sets with the following analysis tools and compare our results to the original conclusions:

- Ogama version 5.1[7]
- PyGaze version 0.6.0[8] with PyGazeAnalyzer version 0.1.0[9]

We used the default parameters for these two tools. The tools further used the following algorithms at the time of our analysis:

- Ogama: fixation-detection algorithm from LC Technologies, which is a dispersion-type algorithm with window.
- PyGaze: crude algorithm by PyGazeAnalyser, a submodule of PyGaze, not further described.

When necessary, the data were converted to the format needed for the analysis tools. A slight modification of the code was necessary to be able to read the data from PyGaze[10]. This has no impact on the results, since the data were not altered in any way.

All scripts used to calculate and compare the eye-tracking metrics are available in the replication package.

5.3 Case Study on EMIP Dataset

For our first case study, we performed the same data analysis as Aljehane et al. [2] on the EMIP dataset [4] to compare the results of different analysis tools. Aljehane et al. [2] investigated the impact of expertise on the eye-movement measurements of participants. To achieve that, they addressed three research questions, two of which we will investigate in this paper.

5.3.1 *RQ1: Correlation Between Expertise and Eye-Tracking Metrics.*

Aljehane et al. [2] found that using years of programming experience is the best choice to explain expertise based on the correlation between eye-movement measurements and expertise. Specifically, they found small ($0.1 < |r_s| \leq 0.3$) to medium ($|r_s| > 0.3$) correlations between the years of programming experience and the eye-movement measurements, except for saccade duration and line coverage (vehicle task).

In our re-analysis, we computed the same eye-movement measures as the original analysis (i.e., number of fixations, sum of fixation duration, saccade duration, and saccade length) and compared our results to the original results (see Table 4).

When replicating the analysis with Ogama, we found no such correlation, as every eye-movement measurement had a correlation coefficient of less than 0.1 for the years of programming experience. The highest correlation in this case would be the self-estimation of the participants in their Java knowledge, where the correlation coefficients of the eye-movement measurements were between 0.09

[7] http://www.ogama.net/
[8] http://www.pygaze.org/
[9] https://github.com/esdalmaijer/PyGazeAnalyser
[10] The function remove_missing of the fixation and saccade-detection algorithm returned an array where missing values were removed, but the index of the array was not reset and thus not continuous, therefore causing a crash. This was remedied by changing remove_missing such that the indexes of the array were continuous again.

Table 4: Study of Aljehane et al. [2]: Comparison of the correlation coefficients ρ between eye-movement measurements and expertise: original values and our re-analysis with Ogama and PyGaze. Cells in bold are, at least, weak correlations ($\rho > 0.1$).

		# of Fixations	Sum of Fixation Duration	Saccade Length	Saccade Duration
Original	Vehicle Task	**-0.02**	0.01	**0.17**	0.07
	Rectangle Task	0.08	0.07	-0.02	-0.01
	Self-Estimation Java	-0.03	0.01	0.08	0.06
	Self-Estimation Progr.	-0.07	-0.09	0.02	0.07
	Years of Programming	**-0.12**	-0.09	0.03	**0.13**
Ogama	Vehicle Task	0.00	-0.02	0.01	0.04
	Rectangle Task	0.09	**0.14**	0.07	-0.03
	Self-Estimation Java	0.09	**0.11**	**0.11**	**0.11**
	Self-Estimation Progr.	0.05	0.08	-0.02	-0.00
	Years of Programming	-0.02	0.01	-0.06	-0.07
PyGaze	Vehicle Task	-0.01	-0.01	0.01	0.01
	Rectangle Task	0.10	**0.15**	-0.02	-0.01
	Self-Estimation Java	**0.11**	**0.12**	0.04	0.04
	Self-Estimation Progr.	0.06	0.06	0.02	0.02
	Years of Programming	-0.02	0.02	**-0.14**	**-0.11**

and 0.12. When replicating the analysis with PyGaze, we found that the years of programming experience had a small to medium correlation with the eye-movement measurements for the saccade measurements, but not for the fixation measurements. There was also no expertise metric which could be described as the best choice to explain expertise, as none distinguished themselves from the others.

5.3.2 RQ2: Metrics Between Experts and Novices. In addition to correlations, Aljehane et al. [2] found that eye-tracking metrics provide strong statistical evidence in assessing the expertise of participants. Novice developers had a statistically significant (based on a non-parametric Mann-Whitney test) higher number of fixations and a longer sum of fixation duration than expert developers while having a medium effect size. Additionally, experts had statistically significant longer saccade duration and longer saccade lengths than novices, but those results had only a small effect size.

We also calculated the average of the actual data in fixation and saccade metrics of novices and experts and their respective significance with the non-parametric Mann-Whitney test, as Aljehane et al. [2] did for their RQ2, and present them in Table 5.

When replicating the analysis with Ogama, we found similar results as Aljehane et al. [2]. Novices showed a higher number of fixations and a longer sum of fixation duration than experts, which also have a medium effect size. Contrary to the original results, the saccade length was slightly shorter for experts, and experts also only had around half the saccade duration of novices. We also found vastly different times for the saccade duration, which may be due to using different definitions of the saccade duration. The saccade duration and saccade length have a small to medium effect size.

Table 5: Study of Aljehane et al. [2]: Comparison of Mann-Whitney test results of experts and novices in fixation-related and saccade-related metrics: Original results and our re-analysis with Ogama and PyGaze. Cells in bold are statistically significant ($p < 0.05$) or, at least, a medium effect size ($d > 0.3$).

		Z	p	Cohen's d
Original	Number of Fixations	-3.74	**<0.01**	**0.31**
	Total Fixation Duration	-3.70	**<0.01**	**0.31**
	Saccade Length	-2.44	**0.01**	0.09
	Saccade Duration	-1.03	0.30	0.09
Ogama	Number of Fixations	-3.62	**<0.01**	**0.32**
	Total Fixation Duration	-3.00	**<0.01**	0.23
	Saccade Length	-1.88	0.06	0.11
	Saccade Duration	-1.50	0.13	0.19
PyGaze	Number of Fixations	-2.86	**<0.01**	0.25
	Total Fixation Duration	-1.80	0.07	0.13
	Saccade Length	-3.34	**<0.01**	0.24
	Saccade Duration	-2.84	**<0.01**	0.04

When replicating the analysis with PyGaze, we again found that novices had a higher number of fixations and a longer total fixation duration than experts, but the effect is no longer significant for total fixation duration. Further, novices had again a longer saccade duration and a longer saccade length than experts. The relative difference in saccade duration between novices and experts was a bit smaller than in the original study. The effect size of the saccade length increased from 0.09 to 0.24.

5.4 Case Study on Peitek et al.

Peitek et al. [18] investigated the impact of programmer efficacy on program comprehension in a combined EEG and eye-tracking study. Here, we focus on their research question on the reading behavior based on several eye-tracking metrics of programmers. Peitek et al. [18] provided a replication package for their study with Jupyter Notebooks to recalculate their results. We used these to compare the influence of different analysis tools on the results. We compare the original results, which were calculated with the I2CM algorithm [12], with our re-analysis in Table 7.

Overall, Peitek et al. found that increased efficacy leads to shorter (first) fixations, shorter gaze durations, a much lower chance that a token is revisited, and a higher probability that a token is skipped.

When replicating the analysis with Ogama, our results are similar to the original results, but with smaller effect sizes than the original results. When replicating the analysis with PyGaze, we found that the correlation of the first fixation duration, fixation duration, and gaze duration are slightly positive, which implies that decreased efficacy leads to shorter fixations and gaze durations. This is contrary to the original results.

Peitek et al. [18] also found that increased efficacy leads to a lower code element coverage, as shown in Table 6. The results of the fixations calculated by Ogama and PyGaze lead to the same result as the original results. The difference increases throughout the task, which is also the same as the original results. However,

Ogama and PyGaze both lead to smaller correlation coefficients than the original results.

5.5 Case Study on Sharafi et al.

Sharafi et al. [22] investigated the different problem-solving strategies used by participants for data-structure tasks. They split the data-structure tasks into list, mental, and tree-rotating tasks, with the mental task being the mental manipulation of spatially presented information. Additionally, they analyzed the impact of participants' reported demographic information on performance. The measurements were analyzed with Ogama.

Sharafi et al. [22] provided a replication package for their study with the data of the study, which we used to compare the influence of different analysis tools on the results. We used the formatted raw data of the replication package to calculate the eye-tracking metrics with Ogama and PyGaze. We had to make a few assumptions during the re-analysis, as the original paper and replication package were not always clear on how the results were originally calculated. Specifically, we assumed that the raw data provided had already been cleaned of the removed participants. Additionally, we also assumed that the statistical tests were performed after grouping the results of the participants. While we tried to replicate the study to the best of our knowledge, the assumptions that we had to make may have led to a difference in magnitude of the fixation time.

Overall, Sharafi et al. found that participants fixated more frequently while working on data structure stimuli. This is also the case for the data calculated with Ogama and PyGaze, which we present in Table 8. Additionally, while the fixation time was not correlated in the original results, it was for the data calculated with Ogama and PyGaze.

In the original analysis, Sharafi et al. also performed an exploratory analysis of the data collected in their study. They correlated the participants' age and gender to the fixation count and average fixation duration. While Sharafi et al. did not provide conclusions made from these results, we still investigated whether different analysis tools would yield different results. Sharafi et al. found no difference between participants regarding gender, while the age of the participants had a significant impact on the eye-tracking metrics.

When replicating the analysis with Ogama, we found no significant correlation between the age for fixation count and average fixation duration, which contradicts the original results. When replicating the analysis with PyGaze, we also found no significant correlation between the age for fixation count and average fixation duration, which also contradicts the original results (see Table 9).

The combined results of all three case studies let us answer RQ3:

RQ3 The choice of analysis tools can have a substantial influence on the experimental results of a study. In our re-analysis, we found different and partially contradicting results in all three case studies with PyGaze. With Ogama, we found different results in two case studies. These differences are not only of numerical nature, but even affect statistical significance, effect sizes, and conclusions.

5.6 Discussion of the Results for RQ3

With our re-analysis of study data in three case studies, we found that the choice of the analysis tool can have a significant influence on the quantitative and qualitative results and conclusions of an eye-tracking study. In all three case studies, we found different and partially contradicting results when using PyGaze. With Ogama, we found different results in two case studies. This shows that the choice of the eye-tracking analysis tool can have a significant influence on the results and conclusions of a study. Additionally, two of the three considered studies did not provide all the necessary information to exactly replicate their results, including missing parameters for algorithms and thresholds for fixation detection.

5.6.1 Influence of Different Tools on Study Conclusions. Our results can be explained by the fact that different analysis tools use different algorithms and configurations to analyze eye-tracking data. PyGaze and Ogama use different algorithms to calculate eye-tracking metrics compared to each other and the algorithms used in the original studies. We also only used the standard configurations of the tools (as the original analyses likely did), which may not be the best configuration for the data set used in the original studies. Therefore, the choice of the analysis tool can have a significant influence on the results and conclusions of an eye-tracking study, even if the same data set is used.

The difference in the results can also vary significantly within a case study. With Ogama, we came to a different conclusion for Aljehane et al.'s study on the EMIP data set: for their first research question our results were different, but not for our re-analysis of their second research question. Therefore, if only parts of a study are replicated and the replicated parts confirm the original results, the remaining part of the study may still yield different results for different analysis tools.

So what should we do? As shown in RQ2, researchers provide different reasons for selecting an analysis tool. Consequently, adopting an approach where everyone has to use the same tool to ensure comparability of results does not appear to be particularly fruitful. Some tools simply appear to be better suited to certain studies than others due to their functional capabilities, which justifies a certain heterogeneity in the tool landscape. However, we must be aware of the consequences of tool selection on the comparability of study results and counteract these consequences, for example through better reporting.

5.6.2 Extending Reporting. The results of our case studies confirm that some studies do not provide all the necessary information to exactly replicate the results. Two of the three studies did not provide the parameters for the algorithms used to analyze the eye-tracking data. As the choice of the parameters can have a significant effect on the results, this information is crucial for replicating the results. Additionally, some analysis tools may not have the same parameters as the original tools. Comparing the results of such different tools proves to be challenging, as the results may be influenced by parameters that cannot be comparably modified in all tools. As one step to address this issue, we recommend extending the minimum reporting guidelines for eye-tracking studies [8] to include the used analysis tool, including the algorithm's parameters.

Table 6: Study of Peitek et al. [18]: Comparison of Spearman's ρ correlation between programmer efficacy and code element coverage over task-completion time: Original results (I2MC) and our re-analysis with Ogama and PyGaze.

	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Original	-0.08	-0.09	-0.11	-0.11	-0.17	-0.21	-0.26	-0.26	-0.28	-0.35
Ogama	0.00	-0.06	-0.07	-0.07	-0.09	-0.09	-0.13	-0.16	-0.16	-0.23
PyGaze	0.02	-0.04	-0.08	-0.08	-0.11	-0.11	-0.15	-0.17	-0.20	-0.24

Table 7: Study of Peitek et al. [18]: Comparison of the Spearman's correlation coefficients ρ between programmer efficacy and different eye-tracking measures based on the original analysis (I2MC) and our re-analysis with Ogama and PyGaze.

Analysis Tool	First Fixation Duration	Fixation Duration	Gaze Duration	Skip Token Probability	Single Fixation Probability	Refixation Probability
Original (I2MC)	-0.14	-0.14	-0.19	0.08	-0.31	-0.37
Ogama	-0.08	-0.06	-0.13	0.00	-0.24	-0.25
PyGaze	0.06	0.05	0.01	0.09	-0.25	-0.26

Table 8: Study of Sharafi et al.: Pairwise comparisons of three tasks using non-parametric Wilcox tests: Original values and our re-analysis with Ogama and PyGaze. Cells in bold indicate statistical significance ($p < 0.05$).

Analysis Tool	Measure	Results: Mean (Standard Deviation)			Statistical Testing	
		List	Mental	Tree	List vs Mental p	Tree vs Mental p
Original (Ogama)	Fixation Time (s)	1 361 (1 172)	1 413 (1 095)	1 629 (1 162)	0.70	0.07
	Fixation Count	40 (23)	19 (17)	34 (21)	**<0.01**	**<0.01**
Ogama	Fixation Time (s)	218 (90)	84 (52)	204 (104)	**<0.01**	**<0.01**
	Fixation Count	38 (21)	17 (11)	36 (19)	**<0.01**	**<0.01**
PyGaze	Fixation Time (s)	181 (97)	72 (48)	171 (109)	**<0.01**	**<0.01**
	Fixation Count	49 (27)	21 (14)	47 (24)	**<0.01**	**<0.01**

Table 9: Study of Sharafi et al.: Comparison of the impact of participants' demographic information on the observed eye-tracking measures: Original values and our re-analysis with Ogama and PyGaze. Cells in bold indicate statistical significance ($p < 0.05$).

Analysis Tool	Measure	Results: Mean (Standard Deviation)				Statistical Testing	
		Gender		Age		Gender	Age
		Men	Women	18-22	23-27	p-value	p-value
Original (Ogama)	Fixation Duration (ms)	191 (53)	192 (51)	197 (52)	190 (52)	0.30	**<0.01**
	Fixation Count	31 (22)	30 (22)	39 (23)	28 (21)	0.15	**<0.01**
Ogama	Fixation Duration (ms)	278 (683)	242 (281)	285 (266)	262 (181)	1.00	0.30
	Fixation Count	28 (24)	32 (23)	29 (17)	30 (17)	1.00	0.62
PyGaze	Fixation Duration (ms)	106 (62)	131 (91)	113 (41)	124 (63)	1.00	0.63
	Fixation Count	35 (29)	40 (29)	38 (21)	38 (20)	1.00	0.63

The available replication packages enabled us to replicate the results of the original studies. However, we encountered some issues during the replication process. For Sharafi et al. [22] we needed to make some assumptions about the data set, as some information was unclear. Peitek et al. [18] provided Jupiter Notebooks to replicate their results. This made it easier to replicate the results, but we encountered some issues with updates of libraries, due to which the replication package was not working out of the box. Additionally, the code contained some errors, which needed to be fixed to replicate the results. Nevertheless, providing full scripts is helpful for replicating the results of a study.

Peitek et al. [18] further used the I2MC algorithm [12] to analyze the eye-tracking data. This algorithm is non-deterministic, which means that it produces slightly different results on the same data set, even if computed on the same machine. This poses a problem for the reproducibility of research results, as the original results may not be exactly replicable, which happened during our replication of the study. Therefore, the analysis of eye-tracking studies leads to a trade-off between the speed and quality of non-deterministic algorithms and the reproducibility of research results with deterministic algorithms.

Despite these criticisms, we would like to praise the fact that the data sets used were among the very few that were available at all, and which additionally met our inclusion criteria.

5.6.3 Influence of Parameters on Results. Most papers did not name the values of the algorithm parameters they used to calculate their fixations and saccades. Exploratively, we also analyzed the influence of the parameters of the analysis tools on the experimental results. We varied the distance between the fixation points for PyGaze for the experimental data of the EMIP data set [1]. We show the results in Table 10.

Table 10: EMIP data set [4]: An analysis on the influence of a parameter (i.e., distances between fixation points) for PyGaze. Cohen's d substantially differs depending on the parameter setting. Cells in bold are, at least, a medium effect size (≥ 0.3).

Measure	Distance Between Fixation Points (px)				
	10	15	25	35	50
Number of Fixations	0.07	0.16	0.25	**0.30**	**0.35**
Total Fixation Duration	0.00	0.06	0.13	0.16	0.19

For low distances between fixation points, the effect size of the number of fixations and the total fixation duration is small or negligible, contrary to the high distances between fixation points, where the effect size is medium. This implies that the variation of parameters can have a significant influence on the experimental results of the study. Therefore, it is important to report the values of the parameters used to ensure the reproducibility of each study.

6 THREATS TO VALIDITY

The methodology of our systematic mapping study (Part I) and the case studies (Part II) is subject to a few limitations that may affect the validity of our results.

6.1 Internal Validity

Part I. We may have missed papers due to insufficient search parameters and not performing snowballing [28] after the keyword search. We tried to mitigate this issue by building our search terms on those used by Sharafi et al. [25] in their literature review. We further do not expect to draw a different conclusion with additional literature, since we already included a large number of 97 papers in our data set.

There may be human error in data extraction and analysis, as it was mainly the first author who performed the data extraction and analysis. However, by regularly consulting the other authors about irregularities and by updating them on the progress, we intended to comply with the recommendations for systematic mapping studies that responsibility does not lie exclusively with one person [19]. Nonetheless, coded labels and categories (e.g., the justifications and consequences of using a specific analysis tool) in the data extraction may have been slightly different if a different person had performed the qualitative analysis.

Part II. The comparison of the analysis tools harbors the threat that we may have used different parameters to calculate the results than the original studies did. This threat cannot be ruled out as the used parameters for most case studies are unknown (which led to the recommendation of extended reporting guidelines).

Another threat to validity is that the data quality of the data sets may have affected the results of RQ3. We minimized this threat by using three data sets selected according to specific inclusion criteria. Additionally, one of them was the EMIP data set, which is an open-source data set used in many software-engineering studies [4].

Finally, we had to make assumptions for the interpretation of some aspects of the replication packages. We tried to mitigate this threat by carefully reading the papers and the documentation of the replication packages, but some aspects remained unclear, as we discussed earlier.

6.2 External Validity

Part I. The generalizability of our study may be limited by the search restriction to recent eye-tracking studies related to software engineering. Paper authors from a different research domain may provide different reasons and justifications for the choice of an analysis tool. They could also discuss them to a different extent.

Part II. Our comparison of study outcomes for different analysis tools was limited to a subset of potential tools. Additional ones could not be compared due to the lack of a standard for importing and exporting data sets. Accordingly, there is a chance that on the same data sets more similar results could be obtained using different analysis tools than was the case in our case studies.

Similarly, we limited ourselves to only a few data sets. A comparable data analysis with the same tools, but on different data sets, could lead to different conclusions. However, due to the very different outcomes we have already observed, we do not believe this to be the case.

7 CONCLUSION

Eye-tracking studies in software engineering have become popular, but they do not yet follow a standardized approach. The findings of our systematic mapping study indicate a growing trend toward using specific analysis tools in papers that cite Tobii Software, iTrace, and Ogama. However, the majority of authors did not provide any justification or consequence for their choice of analysis tool. When justification was provided, it was primarily about the functional requirements of the tool for the study. Consequences of the choice of an analysis tool were rarely discussed by the study authors. Yet, that appears to be necessary: Our case studies revealed that the choice of analysis tool can have a significant influence on the experimental results of a study. This finding may impact the comparability and reproducibility of all studies that used eye tracking, especially those that did not name the analysis tool they used.

Future research should focus on developing a standardized data format for data from eye-tracking studies, as this would enhance the comparability and reproducibility of studies. Furthermore, a study that employs different analysis tools than those used in this paper could help to gain a more profound understanding of the discrepancies in the results between the analysis tools. Finally, and we should not tire of mentioning this, providing replication packages for the (re-)analysis of eye-tracking data should be the norm since it would greatly benefit the health of a research community.

ACKNOWLEDGMENTS

This work has been supported by ERC Advanced Grant 101052182 as well as DFG Grant 389792660 as part of TRR 248 – CPEC.

REFERENCES

[1] Salwa Aljehane, Bonita Sharif, and Jonathan Maletic. 2021. Determining Differences in Reading Behavior Between Experts and Novices by Investigating Eye Movement on Source Code Constructs During a Bug Fixing Task. In *Proc. Symposium on Eye Tracking Research & Applications (ETRA)*. ACM, Article 30, 6 pages.

[2] Salwa Aljehane, Bonita Sharif, and Jonathan Maletic. 2023. Studying Developer Eye Movements to Measure Cognitive Workload and Visual Effort for Expertise Assessment. *Proc. Human-Computer Interaction* 7, ETRA (2023), 1–18.

[3] Richard Andersson, Linnea Larsson, Kenneth Holmqvist, Martin Stridh, and Marcus Nyström. 2017. One Algorithm to Rule Them All? An Evaluation and Discussion of Ten Eye Movement Event-Detection Algorithms. *Behavior Research Methods* 49 (2017), 616–637.

[4] Roman Bednarik, Teresa Busjahn, Agostino Gibaldi, Alireza Ahadi, Maria Bielikova, Martha Crosby, Kai Essig, Fabian Fagerholm, Ahmad Jbara, Raymond Lister, et al. 2020. EMIP: The Eye Movements in Programming Dataset. *Science of Computer Programming* 198 (2020), 102520.

[5] Daniela Cruzes and Tore Dyba. 2011. Recommended Steps for Thematic Synthesis in Software Engineering. In *International Symposium Empirical Software Engineering and Measurement*. IEEE, 275–284.

[6] Benoît De Smet, Lorent Lempereur, Zohreh Sharafi, Yann-Gaël Guéhéneuc, Giuliano Antoniol, and Naji Habra. 2014. Taupe: Visualizing and analyzing eye-tracking data. *Science of computer programming* 79 (2014), 260–278.

[7] Andrew Duchowski. 2017. *Eye Tracking Methodology: Theory and Practice*. Springer.

[8] Matt Dunn, Robert Alexander, Onyekachukwu Amiebenomo, Gemma Arblaster, Denize Atan, Jonathan Erichsen, Ulrich Ettinger, Mario Giardini, Iain Gilchrist, Ruth Hamilton, et al. 2023. Minimal Reporting Guideline for Research Involving Eye Tracking (2023 Edition). *Behavior Research Methods* (2023), 1–7.

[9] Sarah Fakhoury. 2018. Moving Towards Objective Measures of Program Comprehension. In *Proc. Europ. Software Engineering Conf./Foundations of Software Engineering (ESEC/FSE)*. ACM, 936–939.

[10] Joseph Goldberg and Xerxes Kotval. 1999. Computer Interface Evaluation using Eye Movements: Methods and Constructs. *Industrial Ergonomics* 24, 6 (1999), 631–645.

[11] Drew Guarnera, Corey Bryant, Ashwin Mishra, Jonathan Maletic, and Bonita Sharif. 2018. iTrace: Eye Tracking Infrastructure for Development Environments. In *Proc. Symposium on Eye Tracking Research & Applications (ETRA)*. ACM, Article 105, 3 pages.

[12] Roy Hessels, Diederick Niehorster, Chantal Kemner, and Ignace Hooge. 2017. Noise-Robust Fixation Detection in Eye Movement Data: Identification by Two-Means Clustering (I2MC). *Behavior Research Methods* 49, 5 (2017), 1802–1823.

[13] 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.

[14] Kenneth Holmqvist, Saga Lee Örbom, Ignace Hooge, Diederick Niehorster, Robert Alexander, Richard Andersson, Jeroen Benjamins, Pieter Blignaut, Anne-Marie Brouwer, Lewis Chuang, et al. 2023. Eye Tracking: Empirical Foundations for a Minimal Reporting Guideline. *Behavior Research Methods* 55, 1 (2023), 364–416.

[15] Marcel Just and Patricia Carpenter. 1980. A Theory of Reading: From Eye Fixations to Comprehension. *Psychological Review* 87, 4 (1980), 329.

[16] Linnéa Larsson, Marcus Nyström, and Martin Stridh. 2013. Detection of Saccades and Postsaccadic Oscillations in the Presence of Smooth Pursuit. *Transactions on Biomedical Engineering* 60, 9 (2013), 2484–2493.

[17] Zaheed Mahmood, David Bowes, Tracy Hall, Peter Lane, and Jean Petrić. 2018. Reproducibility and Replicability of Software Defect Prediction Studies. *Information and Software Technology* 99 (2018), 148–163.

[18] Norman Peitek, Annabelle Bergum, Maurice Rekrut, Jonas Mucke, Matthias Nadig, Chris Parnin, Janet Siegmund, and Sven Apel. 2022. Correlates of Programmer Efficacy and their Link to Experience: A Combined EEG and Eye-Tracking Study. In *Proc. Joint Meeting on Foundations of Software Engineering (ESEC/FSE)*. ACM, 120–131.

[19] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. 2008. Systematic Mapping Studies in Software Engineering. In *Proc. Conference on Evaluation and Assessment in Software Engineering (EASE)*. 1–10.

[20] Gema Rodríguez-Pérez, Gregorio Robles, and Jesús González-Barahona. 2018. Reproducibility and Credibility in Empirical Software Engineering: A Case Study Based on a Systematic Literature Review of the Use of the SZZ Algorithm. *Information and Software Technology* 99 (2018), 164–176.

[21] Dario Salvucci and Joseph Goldberg. 2000. Identifying Fixations and Saccades in Eye-Tracking Protocols. In *Proc. Symposium Eye Tracking Research & Applications (ETRA)*. 71–78.

[22] Zohreh Sharafi, Yu Huang, Kevin Leach, and Westley Weimer. 2020. Towards an Objective Measure of Developers' Cognitive Activities. *ACM Trans. Softw. Eng. & Methodology* 30, 3 (2020), 1–40.

[23] Zohreh Sharafi, Timothy Shaffer, Bonita Sharif, and Yann-Gaël Guéhéneuc. 2015. Eye-Tracking Metrics in Software Engineering. In *Proc. Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 96–103.

[24] Zohreh Sharafi, Bonita Sharif, Yann-Gaël Guéhéneuc, Andrew Begel, Roman Bednarik, and Martha Crosby. 2020. A Practical Guide on Conducting Eye Tracking Studies in Software Engineering. *Empirical Softw. Eng.* 25, 5 (2020), 1–47.

[25] Zohreh Sharafi, Zéphyrin Soh, and Yann-Gaël Guéhéneuc. 2015. A Systematic Literature Review on the Usage of Eye-Tracking in Software Engineering. *Information and Software Technology* 67 (2015), 79–107.

[26] Martin Shepperd, Nemitari Ajienka, and Steve Counsell. 2018. The Role and Value of Replication in Empirical Software Engineering Results. *Information and Software Technology* 99 (2018), 120–132.

[27] Janet Siegmund, Norman Peitek, André Brechmann, Chris Parnin, and Sven Apel. 2020. Studying Programming in the Neuroage: Just a Crazy Idea? *Commun. ACM* 63, 6 (2020), 30–34.

[28] Claes Wohlin. 2014. Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering. In *Proc. Conference on Evaluation and Assessment in Software Engineering (EASE)*. 1–10.

[29] Marvin Wyrich, Justus Bogner, and Stefan Wagner. 2023. 40 Years of Designing Code Comprehension Experiments: A Systematic Mapping Study. *Comput. Surveys* 56, 4 (2023), 1–42.

[30] Vlas Zyrianov, Cole Peterson, Drew Guarnera, Joshua Behler, Praxis Weston, Bonita Sharif, and Jonathan Maletic. 2022. Deja Vu: Semantics-Aware Recording and Replay of High-Speed Eye Tracking and Interaction Data to Support Cognitive Studies of Software Engineering Tasks––Methodology and Analyses. *Empirical Software Engineering* 27, 7 (2022), 168.