SN Computer Science (2024) 5:729
https://doi.org/10.1007/s42979-024-03045-3

REVIEW ARTICLE

On Eye Tracking in Software Engineering

Lisa Grabinger[1] · **Florian Hauser**[1] · **Christian Wolff**[2] · **Jürgen Mottok**[1]

Received: 8 December 2023 / Accepted: 8 June 2024

Abstract
Eye tracking is becoming more and more important as a research method within the field of software engineering (SE). Existing meta-analyses focus on the design or conduct of SE eye tracking studies rather than the analysis phase. This article attempts to fill this gap; it presents a systematic literature review of eye tracking studies in the field of SE—focusing mainly on the data analysis methods used. From the IEEE Xplore and ACM digital libraries we gather 125 papers up to the first quarter of 2024. Detailed evaluation provides information on the number of papers that use specific methods of analysis (i.e., descriptive or inferential statistics, and gaze visualization) or settings (e.g., sample size, technical setup, and selected aspects of research design). With the data obtained we can infer the popularity of specific analysis methods in the field. Those results enable efficient work on data analysis tools or education of aspiring researchers and can serve as basis for standardization or guidelines within the community—providing for methods to include as well as current inconsistencies.

Keywords Eye tracking · Software engineering · Empirical · Systematic literature review

Introduction

Eye tracking tells us where a person focuses their visual attention—by recording an estimation of their point of gaze over time [52]. Capturing people's visual focus helps to understand how visual artifacts are perceived (i.e., what promotes or hinders processing) and hence also to uncover or mimic certain visual strategies (i.e., what makes an expert). As empirical research method, eye tracking is particularly useful in highly visual fields—such as software engineering (SE), the discipline that deals with the means and practice of developing software systems [13, p.17].

Even though the first eye tracking study.[1] in SE was published more than 30 years ago [35], it was only a decade ago that the research area really began to grow [70, 120]; the late upswing is mostly blamed on the accessibility of eye tracking systems [70, 119]. Today, there is an active and growing community that engages in workshops [11], establishes standards [118, 120], and reflects with meta-analyses [70, 91, 119].

Yet, up to now, most overarching efforts in the field are directed towards planning or conducting a study. Existing systematic literature reviews (SLRs) focus on the design, setup, or sample of the study [70, 91, 119].[2] The data analysis phase is rarely addressed: [118] states that data analysis is either hypothesis-driven or data-driven while [120] outlines data visualization options and gives best practices for statistical data analysis.

The present article starts at this very point: It takes a closer look at *which data analysis methods are actually*

✉ Lisa Grabinger
lisa.grabinger@oth-regensburg.de

Florian Hauser
florian.hauser@oth-regensburg.de

Christian Wolff
christian.wolff@ur.de

Jürgen Mottok
juergen.mottok@oth-regensburg.de

[1] Laboratory for Safe and Secure Systems, Technical University of Applied Sciences Regensburg, Seybothstraße 2, Regensburg 93053, Germany

[2] Faculty of Computer Science and Data Science, University of Regensburg, Bajuwarenstraße 4, Regensburg 93053, Germany

[1] In the context of this paper, we refer to an empirical study that uses eye tracking as its main research method as *eye tracking study*.

[2] Note that [91] focuses on only one aspect of software engineering, namely programming, while [70] only covered the past five years.

Published online: 26 July 2024

being used in eye tracking studies in SE—information that is crucial to work on proper tools, to establish standards, or to train young researchers for the analysis phase of eye tracking studies. To obtain this information, we perform a SLR—collect publications that analyze SE eye tracking studies and report the analysis methods used in them. We also report the sample size, technical set-up, and selected aspects of the research design. This provides a more comprehensive picture of the selected papers as well as comparability with the main findings of the two comprehensive meta-analyses of the field, [119] and [91].

The article is structured as follows. First, we elaborate on the research method itself—from research goals to collection and evaluation procedures. In "Results" we present the evaluation results and compare them, where reasonable, with [119] and [91]. The paper is rounded off by "Discussion", where we answer our research questions discuss the validity of findings as well as implications for future work.

Methods

For this article, we perform a SLR based on the guidelines given in [67]. The detailed methodology is described on the following pages: After starting with the research questions ("Research Questions"), we explain the process followed for selecting ("Selection Procedure") and evaluating ("Evaluation Procedure") papers.

Research Questions

As explained earlier, our research is mainly driven by the question of what methods are used for data analysis in SE eye tracking studies. In view of the different types of analysis methods we expect (i.e., statistics as well as eye tracking specific analyses), we formulate our main research questions (RQs) as follows:

(RQ1) What *descriptive statistics* methods are used for data analysis in SE eye tracking studies?
(RQ2) What *inferential statistics* methods are used for data analysis in SE eye tracking studies?
(RQ3) What *gaze visualization* methods are used for data analysis in SE eye tracking studies?

Beyond that, we are also interested in the setting of SE eye tracking studies. Thereby, we focus on sample size, technical set-up (i.e., eye tracking device and sampling rate), and selected aspects of research design (i.e., materials, grouping variables, and measured variables). This leads to the following additional research questions:

(RQ4) What are the samples sizes in SE eye tracking studies?
(RQ5) What eye trackers are used in SE eye tracking studies?
(RQ6) What are the sampling rates in SE eye tracking studies?
(RQ7) What SE-related artifacts are used in SE eye tracking studies?
(RQ8) What kinds of data grouping are used in SE eye tracking studies?
(RQ9) What data is collected for analysis in SE eye tracking studies?

To shed light on the development of SE eye tracking studies, we do not only present our findings but compare them—as far as possible—with the two comprehensive meta-analyses [119] and [91], both of which date back more than five years.

Selection Procedure

The process of selecting papers within this SLR follows [119] and [91]; the steps are outlined in Fig. 1 and explained in detail in the following subsections.

Data Bases

Sharafi et al. [119] uses the Engineering Village database, but emphasizes that it also searches the IEEE Xplore and ACM digital libraries. Obaidellah et al. [91] uses IEEE Xplore, ACM, and Scopus but points out that Scopus gave a lot of duplicates. Starting from that, we focus on the freely available IEEE Xplore and ACM digital libraries.

Search String

We use a search string based on the ones used in [119] and [91], but with some modifications. We adopt the basic idea of both sources to combine individual search strings for methods (i.e., eye tracking) and materials (e.g., artifacts of SE). We refrain from including a third partial search term targeting typical software engineering activities (e.g., debugging), as is the case in [119] and partially in [91]. Following [119] and [91], the *partial search string on eye tracking* should include the terms for method and device each in the two common spelling variants with and without hyphens (i.e., "eye tracking", "eye-tracking", "eye tracker", and "eye-tracker").[3] Unlike in [119], we do not include the term "RFV" (short for restricted focus viewer); also, departing from both previous

[3] This disagreement goes so far that even two works with identical titles can be found—except for the hyphen in the term "eye(-)tracking" [46, 118].

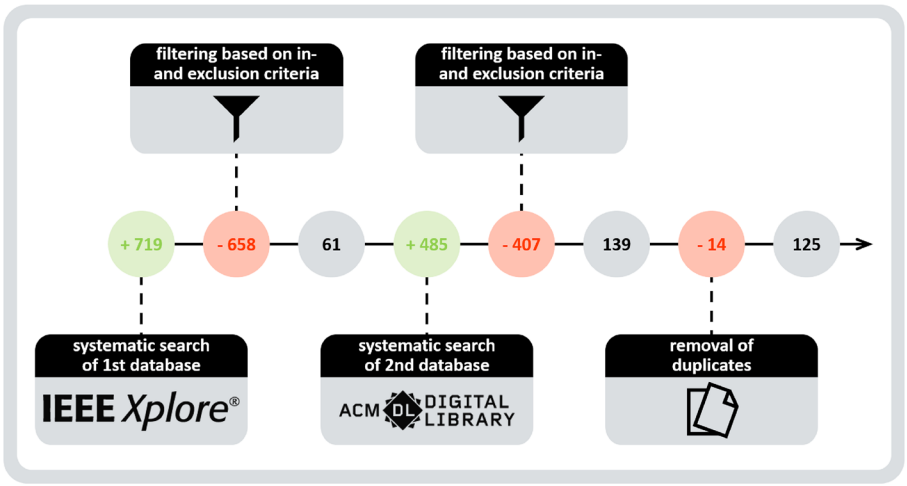


Fig. 1 Paper selection process

SLRs, we add the more general terms "eye movement" or "eye movements". For the *partial search term on SE-related artifacts*, we take the overlap of the two previous SLRs: "code" (from "source code" in [119] and "code" or "pseudo code" in [91]), "program*", and "uml"; deviating from the two works, we add the terms "software" and "requirement".

Individual databases treat the same search string differently. [119] sticks to one database, while [91] has slightly different search terms for different databases—thereby threatening the internal validity of the SLR [119]. To avoid this, we define a common search term that is suitable for both selected databases, given below:

```
(eye-tracking OR eyetracking OR eye-tracker OR eyetracker
OR eye movement OR eye movements) AND (code OR program*
OR uml OR software OR requirement*)
```

Search Queries

For the actual queries, we still need to slightly adjust the common search string. In accordance with [119] we search in titles, abstracts, and keywords, all at the same time; in contrast, [91] searches only in keywords and abstracts, one after the other. These different fields are addressed differently in the two databases—as shown in Fig. 2.

Inclusion and Exclusion Criteria

Entering the queries from Fig. 2 into the corresponding search engines, we obtain 719 results from IEEE Xplore and 485 results from ACM up to the end of the first quarter of 2024. From these more than a thousand search results, we gather work that:

publication ... is found within a journal or conference proceedings published either by ACM or IEEE Press.

contents ... presents the (statistical) analysis of data collected as part of an empirical survey using eye tracking as a research method. Here, we include work that analyzes a previously published study, but with a new method of analysis.

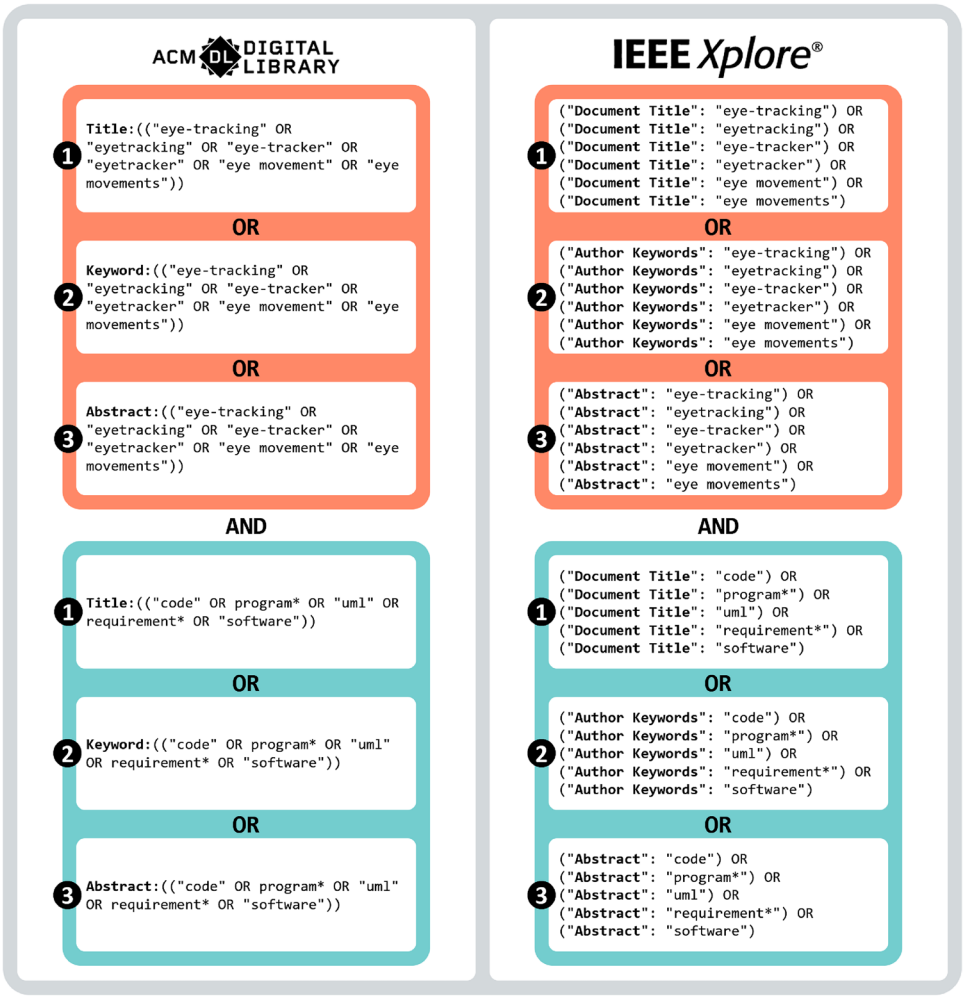
aim ... is of direct interest to SE educators or practitioners, i.e., provides insights for software engineering artifacts, processes, or teaching.

To ensure that only relevant papers are selected according to the above *inclusion criteria*, we review the full paper against predefined *exclusion criteria*. To be precise, we do not consider work that:

publication ... is written in a language other than English or less than two pages. We also do not consider so-called grey literature—a criterion that is unused because we base our search exclusively on the IEEE Xplore and ACM digital libraries, which do not provide such type of work.

contents ... uses some type of head-mounted or integrated eye tracker for data collection. Instead, we limit our analysis to studies conducted with monitor-based eye trackers in a laboratory setting.

Fig. 2 Actual search queries



aim ... evaluates the overall user experience of interfaces, the web, or visual elements—even if the particular system or element is relevant to SE. In addition, we exclude work that addresses the development of a comprehensive approach to analyzing eye tracking data, such as ML models or tools.

Applying these criteria drastically reduces the number of hits, leaving 61 papers from IEEE and 78 papers from ACM. Among these are 14 duplicates, i.e., papers that are listed in both digital libraries. Finally, after manual duplicate removal, 125 papers remain [1–10, 12, 14–40, 42–45, 47–51, 53–66, 68, 69, 71–90, 92–117, 121–128, 130–137].

Snowballing

For the same reason that we agreed to limit our systematic search to two databases, namely IEEE Xplore and ACM digital library, we refrain from any kind of snowballing: The goal of this study is not to provide a comprehensive list of all publications in the designated field, but rather to *understand* what analysis methods or settings are *common* in that field; therefore, it is sufficient to consider the papers published by the two main publishing houses of that area.

Evaluation Procedure

We follow a two-step process to analyze the 125 collected papers. First, we extract the data for each paper in a structured manner, and then we combine the extracted data across all papers to answer our research questions. A detailed explanation is given in the below subsections.

Data Extraction Form

For extracting the relevant information of selected papers, we again follow [119] and define a data extraction form specifically tailored to our needs. The form is listed in the Table 1, along with the possible data values and the mapping to the research questions. We specify the data type, or, where

appropriate, the categories from which to choose, where an empty string (i.e., "") means that the specific criterion is not applicable to or given in that paper. If more than one answer applies to a work, the individual answers are listed jointly, separated by semicolons. If data is not assigned to a specific research question, a hyphen is given instead.

Extracted Data

Regardless of the research questions, we extract information about the publication as such—the number of pages in the published PDF, the year of publication, whether it was published in conference proceedings or a journal, either ACM or IEEE, as well as the abbreviation of the respective conference or journal. We also indicate whether the authors provide access to the study data—not at all, with a now dead link, as raw data, or in structured form.

Based on research questions RQ1 through RQ3, for each paper we list the methods used in that paper. For reasons of clarity, we use more than one list per research question and paper. We differentiate between visualizations (e.g., box plots) and measures (e.g., standard deviation) for *descriptive statistics*; meanwhile, we split *inferential statistics* into hypothesis tests for group differences, correlation analyses, and regression analyses. *Gaze visualizations* are distinguished in visualizations displayed directly on the stimulus[4] (e.g., heat maps) and visualizations of transitions between areas of interest (AOIs).[5] For all statistical methods, we interpret data collected on a Likert scale as metric; for hypothesis tests for group differences and correlation analyses, we do not extract the exact method but the test setting (e.g., "one metric dependent variable between multiple independent groups" or "two metric variables", respectively).

For research question RQ4 through RQ6 we gather data on the experimental set-up itself—the number of participants as well as the sampling rate and manufacturer of the eye tracking device. For the former, we specify the number of individuals whose eye movement data are analyzed; for pair programming tasks, each member of the pair is counted individually. For the latter, we limit the question about eye trackers to manufacturers rather than models, because models change rapidly over the years as [91] points out.

Research Question RQ7 is addressed by data on the SE-related artifacts used for the stimuli. To each paper, we assign one of four categories: computer science (e.g., logic gates [51]), requirements engineering (e.g., social goal models [112]), software development (e.g., UML diagrams [137]), or source code. For the latter, we also record the chosen source code language(s).

For research question RQ8, we track criteria used to group data (e.g., when conducting a hypothesis test for group differences). In doing so, we deviate from the procedure in [119] and [91] in a few ways: First, we track not only what is somewhat explicitly stated as an independent variable, but what is actually used for a sample split. Second, we track whether the sample split is pairwise (e.g., same participants with and without intervention [16]) or not (e.g., two groups of participants in [35]). Third, rather than tracking the specific characteristic, we assign them to categories, such as experience (e.g., novice versus expert in [35]), personal data (e.g., gender in [116]), or setting (e.g., graphic versus textual representations [117]).

Based on research question RQ9, we extract the measured quantities that are investigated by means of statistics or AOI-based visualizations, not distinguishing between mitigating or dependent variables due to different reporting styles across papers. When talking about eye tracking study measurements, it is important to keep in mind that the handling of eye movement metrics is far from consistent [118]. Yet we need definitions and names to assign them. Thereby, we again deviate from [119] and [91]. We use the *naming convention* provided by the eye tracking device manufacturer Tobii [129] rather than the one proposed in [118, 119], for two reasons: First, most readers should be familiar with the Tobii naming convention anyway, since Tobii is the most commonly used eye movement manufacturer (see "Results"); second, the naming convention from [119] resp. [118] does not cover the aspect of *visits* that we consider important. Also, for clarity, we report only the *baseline variable* (e.g., the *number of fixations*) and not a specific metric derived from it (e.g., the *fixation rate on relevant items* in [113]). However, we do not limit ourselves to eye tracking data but also include analysis of *non-eye tracking data* collected as part of the study, with two limitations. First, when multiple studies are reported in an article, we focus only on the eye tracking study (e.g., in [8]). Second, we omit measurements that are used only to describe the sample (e.g., *age* in [121]). We also do not capture *stimuli characteristics* that are used to compute complex metrics (e.g., the *number of words* to obtain the *number of fixations per word* in [53]). However, just as with the data for RQ8, we report what is actually used rather than what is said to be used.

In addition, for each paper we list together the measures, visualizations, hypothesis tests, and procedures used that do not fall into one of the above categories. However, we deliberately omit three aspects: First, the *pre-processing* of eye movement data (i.e., the steps from time series data to fixations and saccades with time stamps), second, the determination of *extreme values or range* as well as calculations

[4] In eye tracking research, a stimulus is the visual material provided to a participant [119].

[5] In eye tracking research, it is common to analyze not only the stimulus as a whole, but also specific regions of the stimulus, referred to as AOIs. AOIs can be included in two ways: Either to restrict the computation of a metric to that region (e.g., number of fixations to a specific region rather than the entire stimulus) or to analyze gaze in a discrete manner. In the latter case, the changes between two AOIs are usually referred to as a *switch* [16] or *transition* [119].

Table 1 Data extraction form

Aspect	Extracted data	Possible data values	RQ
Publishing details	Number of pages	*integer*	–
	Year of publication	[1990; 2023]	–
	Format	{"conference proceedings"; "journal article"}	–
	Publisher	{"ACM"; "IEEE"}	–
	Abbreviated source	*string*	–
	Data set	{""; "dead link"; "raw"; "structured"}	–
Descriptive statistics	Measures	*string*	RQ1
	Visualizations	*string*	RQ1
	Bar plot (group handling)	{""; "grouped"; "singular"; "stacked"}	–
	Bar plot (extras)	{""; "with error bars"; "without error bars"}	–
	Line plot (x-axis)	{""; "qualitative"; "quantitative"}	–
	Line plot (extras)	{""; "with error bars"; "with groups"; "without error bars"; "without groups"}	–
	Scatter plot (x-axis)	{""; "qualitative"; "quantitative"}	–
	Scatter plot (extras)	{""; "with groups"; "with trend line"; "without groups"; "without trend line"}	–
Inferential statistics	Group differences	*string*	RQ2
	Correlation analysis	*string*	RQ2
	Regression analysis	*string*	RQ2
Gaze visualizations	On-stimulus	*string*	RQ3
	AOI transitions	*string*	RQ3
	Heat map (color scheme)	*string*	–
	Heat map (computation)	{""; "duration"; "frequency"}	–
	Heat map (subjects)	{""; "aggregated"; "singular"}	–
Experimental set-up	Number of participants	*integer*	RQ4
	Eye tracking device	*string*	RQ5
	Sampling rate	*integer*	RQ6
Research design	Artifact	{"computer science"; "requirements engineering"; "software development"; "source code"}	RQ7
	Source code language	*string*	RQ7
	Data grouping (unpaired)	*string*	RQ8
	Data grouping (paired)	*string*	RQ8
	Measured quantities	*string*	RQ9
Others	Other measures	*string*	–
	Other visualizations	*string*	–
	Other hypothesis tests	*string*	–
	Other procedures	*string*	–

of *basic arithmetic* (e.g., percentages), and third, whether or not a paper includes *interview responses* by simple citation.

During data extraction, we noted differences in the visual representations of bar plots, line plots, scatter plots, and heat maps that might be of interest for tool development or training. Therefore, we systematically collected data for *bar plots* on group treatment (i.e., singular, grouped, or stacked) and other features (e.g., error bars), for *line or scatter plots* on x-axis scaling (i.e., whether it is qualitative or quantitative) and other features (e.g., error bars, groups, or trend lines), and for *heat maps* on the color scheme (e.g., green to red), the basis of calculation (e.g., frequency or duration), or the treatment of individual subjects (e.g., aggregated or singular).

Analysis Criterion

As we aim to understand *how common* certain analysis methods or settings are, we choose the *number or percentage of papers* using that method or setting as our main evaluation criterion. To account for the fact that we included all papers ranging from two-pagers to extensive articles of more than 30 pages, we divide the papers into three groups based on their page count:

- *short* papers with up to four pages,
- *regular* papers with five to nine pages, and
- *long* papers with at least ten pages.

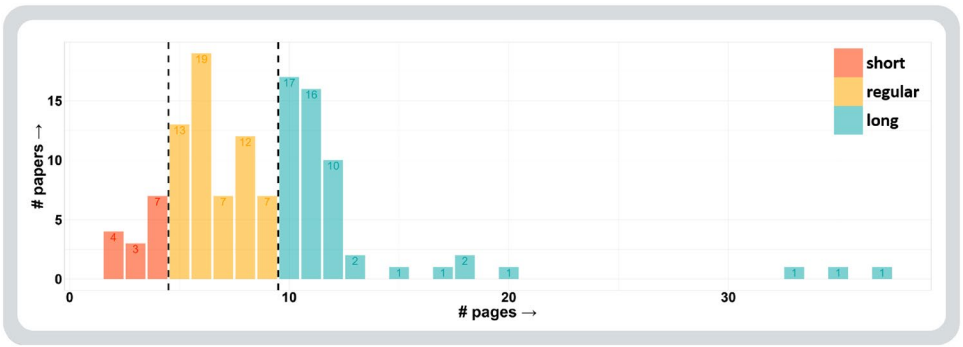


Fig. 3 Number of papers by pages (red: short, yellow: regular, blue: long)

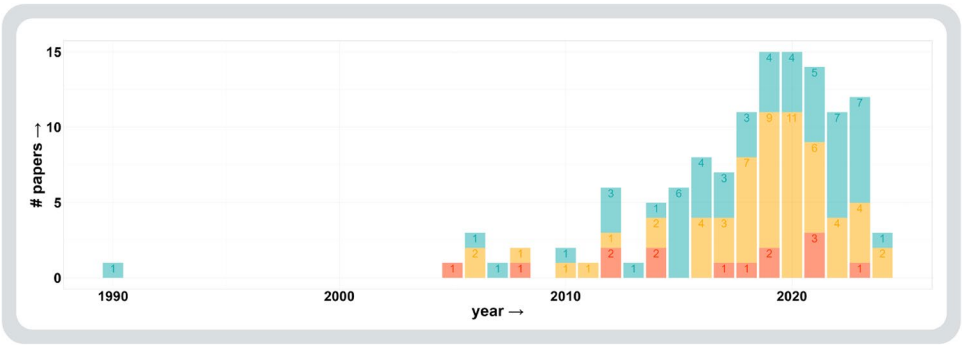


Fig. 4 Number of papers by year of publication (red: short, yellow: regular, blue: long)

Accordingly, we provide four frequencies for each research item: *overall* as well as *separated into short, regular, and long papers*. For example, the total number of papers collected is 125 (14 short, 58 regular, 53 long). Figure 3 shows the distribution of papers over the number of pages, together with the division into the three groups and their respective coloring for the remainder of the paper.

Results

The following pages present the information collected through the SLR—ordered by the grouping aspects from Table 1. Here, the results are presented without direct reference to the research questions—the answer to which can be found in "Discussion".

Publishing Details

The data collected for this aspect are not intended to answer research questions, but to give a better idea of the sample. The first criterion, the *number of pages* is already shown in advance in Fig. 3; it serves for grouping within the sample and is not further discussed.

The *year of publication* is broken down in Fig. 4; thereby, the data for 2024 may be misleading as we were captured the first quarter. Sharafi et al. [119] and Obaidellah et al. [91] report a positive trend in the number of publications over the years—a statement we can at least partially support. Sharafi et al. [119] collected a total of 36 papers by 2015, with a maximum of seven papers per year in 2012; Obaidellah et al. [91] included three more years and collected significantly more papers (i.e., 63), with a maximum of eight papers per year in 2015. Five years later, even with an incomplete search, we found a total of 125 papers—with the strongest years being 2019 and 2020, each with 15 papers per year. Sharafi et al. [119] indicates that only 14% of papers were published up to 2006—in our sample, this number drops to about 4%. According to Obeidellah et al. [91], 62% of articles are published in or after 2012—in our data it is 91%. Nevertheless, we cannot confirm a steady increase, as the number of publications per year declined after 2021 and 2022. In large part, this is probably not due to a waning interest in the field, but to the more difficult conditions for laboratory studies due to the emergence of the COVID-19 pandemic.

Format and *publisher* are jointly presented in Table 2. Overall, the ratio of ACM to IEEE is nearly even with 54–46%—for journal articles alone, the reverse is true with more than twice as many IEEE results. Note that overall, journal articles account for only about 10% of all articles, even less than [119] and [91] report (i.e., 22% or 25%, respectively).

We also recorded the *abbreviation* of the respective publishing conference or journal. The results can be found in Fig. 5. Sources that are unique are grouped as "other conferences" or "other journals". As in [119], ETRA turns out to be the most important conference, followed by ICPC—in

Table 2 Number of papers by format and publisher ("# all (# short, # regular, # long)" papers)

Format	Publisher		Total
	ACM	*IEEE*	
Conference	64 (8, 38, 18)	48 (6, 19, 23)	112 (14, 57, 41)
Journal	4 (0, 0, 4)	9 (0, 1, 8)	13 (0, 1, 12)
Total	68 (8, 38, 22)	57 (6, 20, 31)	125 (14, 58, 53)

[91], these two conferences are also listed among the top three. Note that recurring workshops lead to bias. Take the EMIP workshop as an example: Throughout the years, it published its papers both independently (e.g., in 2022 [20, 69]) or as part of the proceedings of the conference with which it was co-located (e.g., in 2023 [6, 48, 126]).

Regarding *data availability*, we observe that not even a quarter of papers provide links to data repositories. However, of these, some links are unavailable (i.e., 11 (0 short, 1 regular, 10 long) paper); accordingly, the database is actually available in only 14% of the papers. In most cases, raw data is offered (i.e., 11 (0 short, 1 regular, 10 long) paper); only six (0 short, 2 regular, 4 long) papers provide structured data.

Descriptive Statistics

94% of papers (i.e., 117 (13 short, 53 regular, 51 long) paper) use descriptive statistics methods; measures are even more popular than visualizations with 110 (12 short, 48 regular, 50 long) versus 87 (10 short, 38 regular, 39 long) papers applying them. The uses of the former are detailed in Fig. 6, those of the latter in Fig. 7. For both figures, we group the elements that appear only in one work back into the "others" category—leaving only one element in this grouping in Fig. 7, namely error plots. We observe a strong prevalence of the measure *mean*—used in about 86% of the papers—followed by *standard deviation* in more than every second paper and *median* in about every fifth. The most commonly used visualization is the *bar plot* (i.e., in more than one-third of the papers) followed by *box plots* (i.e., in a quarter of papers), *line plots* (i.e., in more than a fifth of papers), and *scatter plots* (i.e., in every tenth paper).

Among the different papers, there are variations in the way descriptive statistics visualizations are used, e.g., one paper [109] uses a secondary axis on its pie chart that shows data in a layered fashion. In what follows, we address such variations in the commonly occurring visualization methods, i.e., bar plots, box plots, line plots, and scatter plots.

In their simplest form, *bar plots* are used to represent the value of a variable across different groups; however, the visualization form can also handle the grouping of data, resulting in so-called grouped or stacked bar plots. Figure 8 shows the number of papers that use certain types of bar plots. Note that the number of single, stacked, and grouped bar plots as shown in the figure does not add up to the total number of papers using bar plots. This is because four papers use multiple bar plots with different grouping [5, 96, 115, 131] while

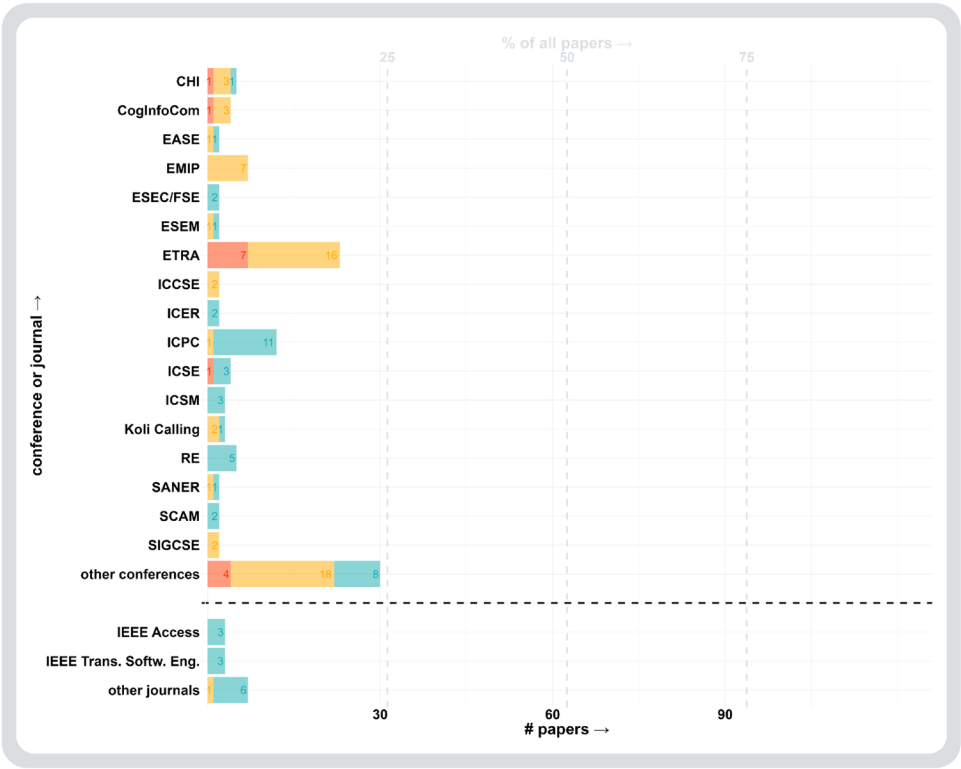


Fig. 5 Number of papers by abbreviated source (red: short, yellow: regular, blue: long)

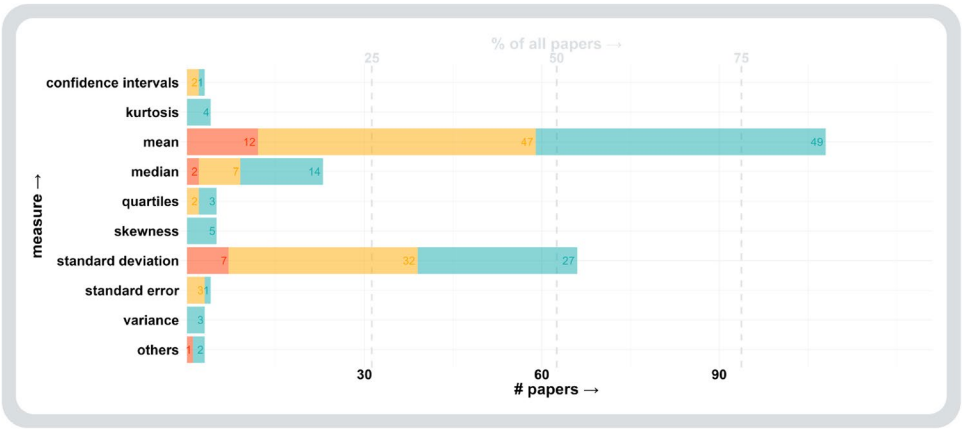


Fig. 6 Number of papers by descriptive statistical measures (red: short, yellow: regular, blue: long)

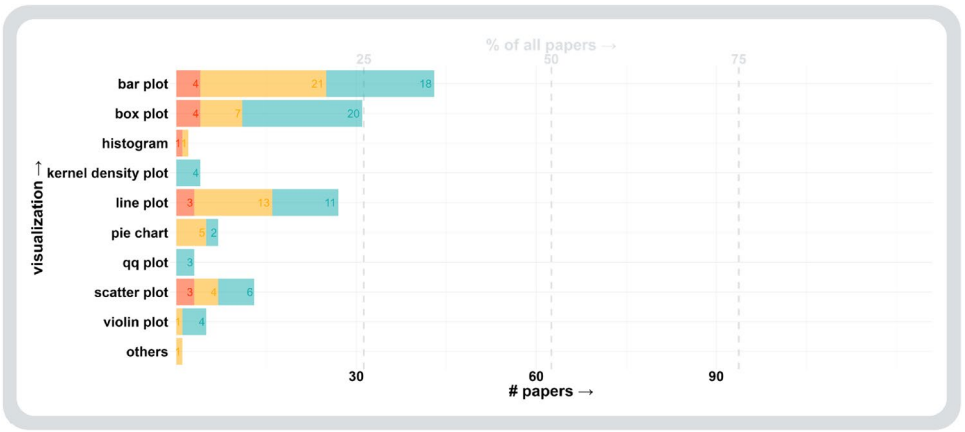


Fig. 7 Number of papers by descriptive statistical visualizations (red: short, yellow: regular, blue: long)

one paper uses a simultaneously grouped and stacked version [134]. Above that, the figure does not take into account the fact that one work [24] uses a two-sided version of a bar chart. However, we note that grouped bar plots are used in more than half of the cases, followed by singular ones in about a third. Error bars are added in almost one in six cases; obviously only for singular (i.e., 4 (1 short, 2 regular, 1 long)) or grouped versions (i.e., 4 (0 short, 3 regular, 1 long)).

For *box plots*, there is little variety in the presentation. The only noticeable aspect is that almost all papers use this visualization method to compare different distributions (e.g., one variable within different groups); only three papers [61, 106, 107] show a single box plot, with the former two [106, 107] also using multiple box plots within one plot elsewhere in the paper.

Against that, for *line plots*, we paid attention to three aspects: the x-axis scaling as well as the presence of grouping or error bars. These variations are summarized in Fig. 9, not accounting for the fact that in one paper [20] the line plot is radial. The figure shows that twice as many papers build their bar plots on qualitative than on quantitative x-axes. Grouping is used in more than three-quarters of cases, six times with quantitative x-axis, 15 times with qualitative x-axis; note that [12, 80, 85] present different versions of line graphs, with and without grouping, all three with qualitative x-axis scaling. Error bars are added in only two papers [12, 87], both times together with grouping and once with qualitative, once with quantitative x-axis scaling.

For *scatter plots*, we used a similar procedure and extracted the scale of the x-axis and the presence of groupings and trend lines. Again, the data are summarized in a figure (i.e., Fig. 10). Regarding the x-axis scaling, we observe an opposite trend as for line plots: All but one paper [35] placed their scatter plots on a qualitative x-axis. Except for this one paper, all added either groups or a trend line—or even both [14, 69]. Again, the figure omits one aspect: two papers [69, 98] use two different versions of scatter plots: with and without groups in [69] as well as with and without a trend line in [98].

Inferential Statistics

About three-quarters of the papers (i.e., 90 (11 short, 33 regular, 46 long) papers) apply inferential statistical procedures;

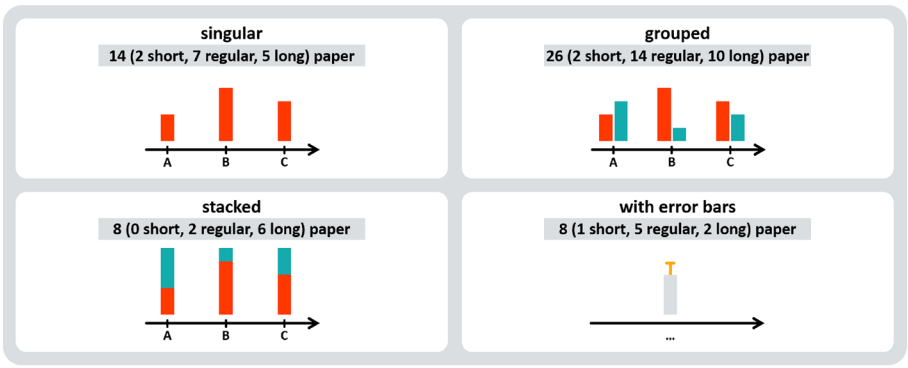


Fig. 8 Number of papers by version of bar plot

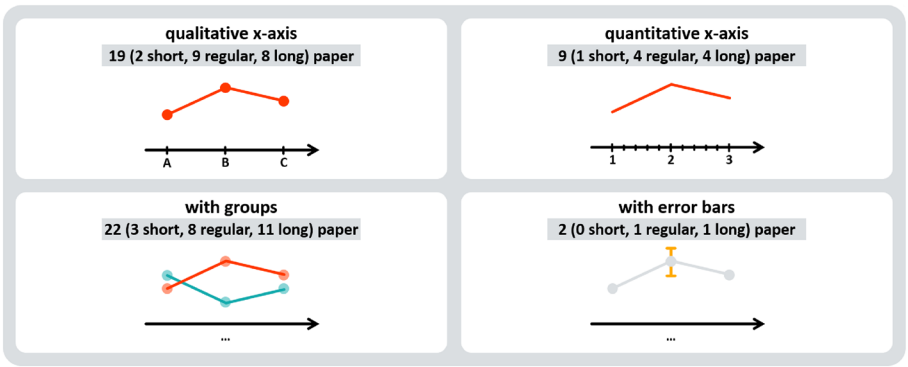


Fig. 9 Number of papers by version of line plot

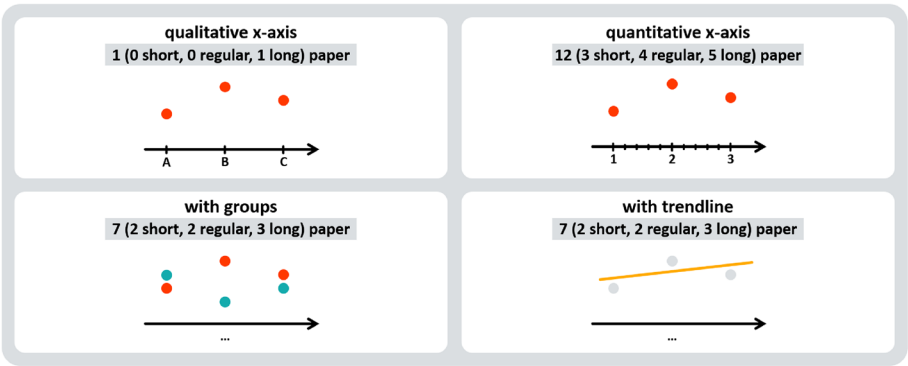


Fig. 10 Number of papers by version of scatter plot

almost all of them, 86 (10 short, 32 regular, 44 long) papers, use hypothesis tests for group differences. About one in five papers reports correlation analysis (i.e., 23 (4 short, 9 regular, 10 long)), while only 11 (2 short, 3 regular, 6 long) paper deal with regression analysis. The latter is fairly mixed; against that, correlation is always computed between two metric variables, except for [3], where a metric and an ordinal variable are correlated.

The individual hypothesis tests for group differences are further broken down in Fig. 11—omitting tests with covariates or multiple dependent variables, as these can only be found in one paper [66]. We see that most hypothesis tests are based on an independent variable and a metric dependent variable. In addition to the data in the figure, note that there is a slight tendency toward paired group settings, as tests for dependent groups are found in 83 (9 short, 32 regular, 42 long) papers, while tests for independent groups are found in only 59 (7 short, 22 regular, 30 long) papers. In addition, almost twice as many papers use a two-group setting rather than looking for differences between multiple groups (i.e., 74 (8 short, 30 regular, 36 long) vs. 39 (4 short, 17 regular, 18 long) papers).

Gaze Visualizations

One in four papers (i.e., 29 (3 short, 11 regular, 15 long)) visualizes transitions between AOIs; one in three papers (i.e., 44 (1 short, 21 regular, 22 long)) visualizes gaze superimposed on the stimulus. Again, we give graphical

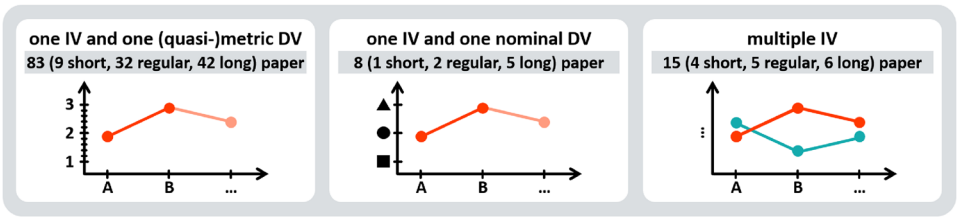


Fig. 11 Number of papers by version of hypothesis test for group differences (IV: independent variable, DV: dependent variable)

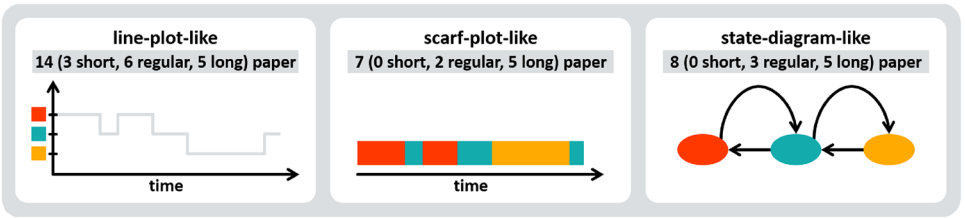


Fig. 12 Number of papers by version of visualization of AOI transitions

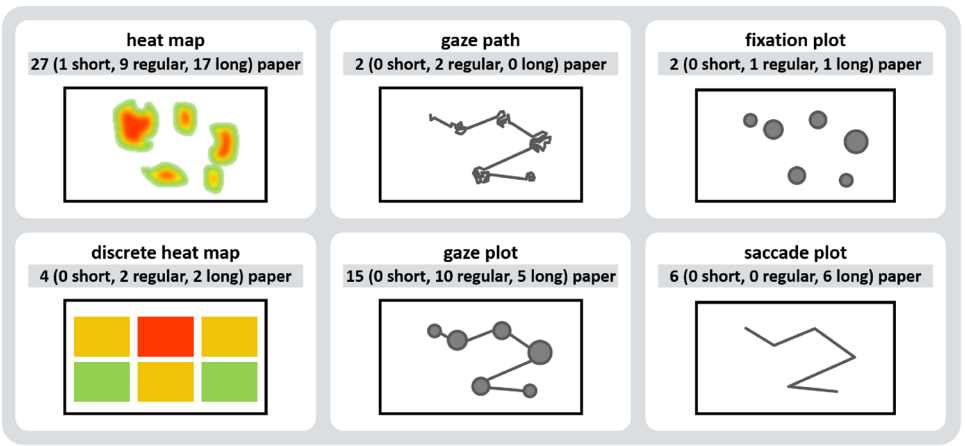


Fig. 13 Number of papers by version of visualization on stimulus

representations when analyzing the variations in visualization (see Fig. 12 and 13). This helps to avoid misunderstandings due to inconsistent naming, e.g., what is called *heat map* in this article is called *fixation map* in [44] or *attention map* in [62]; the term *scan path* may refer to text string [20], to a visualization of coordinates over time [98], as well as to what we call *saccade plot* [40], *gaze plot* [7], *line-plot-like* visualization [21], or *scarf-plot-like* visualization [60].

Figure 12 details the three ways of visualizing transitions between AOIs that occur at least twice in the sample, resembling a line plot, a scarf plot, or a state diagram—omitting the representation as a matrix [7, 102]. In the *line-plot-like* representation, the axes are sometimes changed to show not the AOIs over time, but the fixation number on the x-axis [133] or the source code line number on the y-axis [21, 65, 75, 76, 124, 127, 128, 133]. There is one notable exception to the *scarf-plot-like* visualizations: [114] considers only the sequence of AOIs, regardless of the duration they are viewed. The *state-diagram-like* version is often referred to as *(radial) transition graph* [20, 25, 103, 121]. In it, the arrows are augmented with some sort of metric based on the frequency of the transitions; this is represented by the written number [84] or the thickness of the arrows [20, 25, 103, 121] or both [73, 110]. The order of the nodes is either somewhat arbitrary [73, 84, 110] or radially clockwise [20, 25, 103, 121]. In [121], the size of the nodal area is given by the average duration of fixations in the particular area of interest.

Figure 13 lists the on-stimulus visualizations that are used by at least two different research teams for data analysis purposes (i.e., not for procedural explanations such as AOI placements as in [92]). We distinguish *heat maps* (i.e., continuous or discretized representations of fixation intensity [119]), *gaze paths* (i.e., representations of the sequence of raw data), *gaze plots* (i.e., representations of the sequence of fixations and saccades as circles and lines, where the duration of fixations is reflected as the radius [119]), as well as *fixation* or *saccade plots* as respective subsets of the latter. Here we still have to report some variations within the presented options. Some works restrict the fixations, to at least

300 ms for a gaze plot [105] or 250 ms for a fixation plot [125]. [84] includes the numbering of fixations in a saccade plot; [7] presents gaze plots enhanced with the number and duration of the respective fixations.

For a *(continuous) heat map*, most papers use either a color scheme from green to red (i.e., 14 (1 short, 5 regular, 8 long) paper) as shown in the figure or from purple to red (i.e., 11 (0 short, 3 regular, 8 long) paper); only two papers use other color schemes [40, 136]. The intensity of fixations is equally often computed by the frequency or duration of fixations (i.e., 9 (0 short, 3 regular, 6 long) paper); in the remaining 9 papers, the basis of computation is not specified. While gaze plot, gaze path, fixation plot, and saccade plot may only contain gaze data from one person, about half of the heat maps displays data aggregated from multiple participants (i.e., 13 (0 short, 1 regular, 12 long) papers), with one paper being unclear [27]. For a *discrete heat map*, Crosby and Stelovsky [35] uses numbers instead of colors, while Busjahn et al. [27] changes the font color of individual words rather than the color of regions; when color is used, they adhere to the scheme green to red [2, 27, 69]. The discretization is based on words [27, 69], elements [35], or lines [2]; the calculation is again equally often based on frequency [12, 69] as duration [2, 27], but mostly on one person's data—apart from an undefined case [27].

Overall, we agree with the results of [119] and [91] that the most commonly used gaze visualizations are heat maps, followed by gaze plots.

Experimental Set-Up

We observe *samples sizes* between 1 [33] and 207 participants [10]—a range even more extreme than the ones pointed out by [119] and [91] (i.e., [5; 169] and [2; 82], respectively). [119] reports a total of 1022 participants; with about three times as many contributions, we get about three times as many participants (i.e., 3730)—however, our corresponding mean is about half that of [119] (i.e., 29 (22 short, 26 regular, 34 long) instead of 57). Obeidellah et al. [91] argues for analyzing the frequency of each sample size rather than descriptives—this is shown in Fig. 14. It can be seen from the figure that sample sizes of more than 40 participants occur only occasionally. Note that three papers each report multiple experiments with different samples [34, 66, 85, 86]; accordingly, the number of papers in the figure adds up to 130 instead of 125.

Regarding the *eye tracking devices*, we report similar results to [119] and [91]: An eye tracker from Tobii is used in 58% of the papers within our sample, 47% in [119], and 55% [91]. A detailed listing can be taken from Fig. 15; here, again, we group one-time results into "others".

Sharafi et al. [119] reports *sampling rates* between 30 and 500 Hz; we again get a wider range, up to 1000 Hz, detailed in Fig. 16—again with the "others" grouping and a total other than 125 due to two papers with multiple sampling rates [26, 30]. It can be seen that, in accordance with [119], the majority of studies samples at 60 Hz.

Research Design

As shown in Fig. 17, more than three quarters of the papers rely on source code—even more than in [119] with 64%, but less than in [91] with 79%. Of these papers, 58% use Java, followed by C-family languages (i.e., C, C#, C++, and ANSI C sum to 34%), see Fig. 18. The dominance of Java has decreased compared to [119], which reports 70% of all source-code-related papers, and increased compared to [91] with 48%.

Obeidellah et al. [91] reports that nearly half of the papers use *sample splitting* when analyzing the data (i.e., 48%), with experience being the most common grouping factor (i.e., in 43% of papers that use a sample split). When evaluating sample splits we go a step further—taking into account the pairing of the resulting groups. The results are summarized in Fig. 19. For the sake of clarity we do not list the exact grouping factors, but assign them to categories, such as *personal data* instead of dyslexia [81–83] or gender [66, 69, 90, 116]. Note that experience is almost always measured by background (e.g., years of use, occupation, or course enrollment), while some papers draw on test scores [3, 132] or self-assessment [25]; performance is most often measured via task results, with individual papers additionally weighting with completion time [92, 98]. The figure shows that for unpaired samples, the most common grouping factor is experience, followed by performance, setting, and subjects. A split into paired samples is mainly based on screen areas, settings, or tasks.

Beyond that, we are of course also interested in which *quantities* are *measured* or evaluated, respectively. Sharafiet al. [119] concludes that "the majority of eye-tracking studies use[s] metrics that are calculated using either the numbers or durations of fixations"; in contrast, saccade- or region-based metrics are rather uncommon (i.e., found in only 14% of papers) [119]. Obeidellah et al. [91] also works out that fixation-based metrics are most commonly used, followed by region-based and non-eye-tracking-based measures, namely processing time or some kind of task score. For our sample, Fig. 20 summarizes the categories of quantitatively measured quantities that are found in the papers. In total, quantitative eye tracking data is used in all but two papers [44, 62]. Nearly three-quarters of the papers use data that is not from eye tracking (i.e., 92 (10 short, 38 regular, 44 long));[6]

[6] Since processing time can be measured by the eye tracker in most cases, another number of interest is 83 (9 short, 32 regular, 42 long)—the number of papers that include data in their analysis that cannot be measured by the eye tracker.

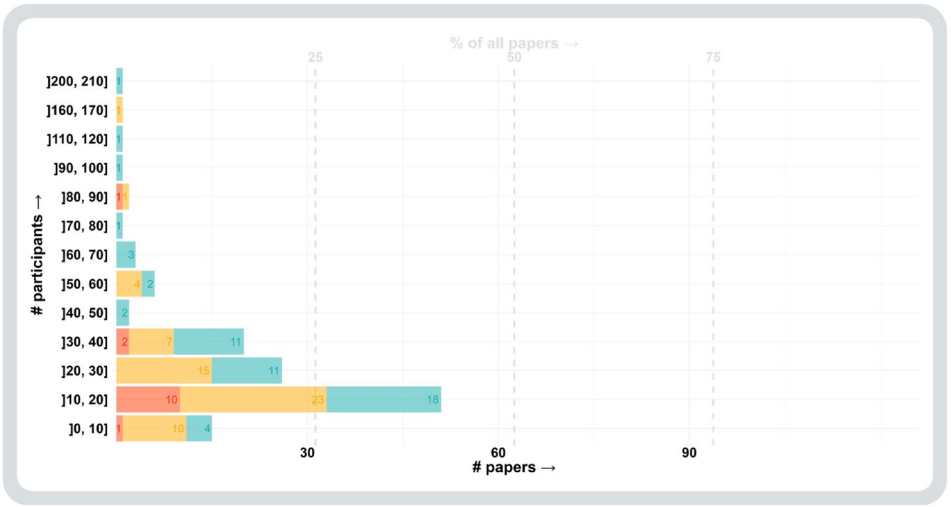


Fig. 14 Number of papers by number of participants (red: short, yellow: regular, blue: long)

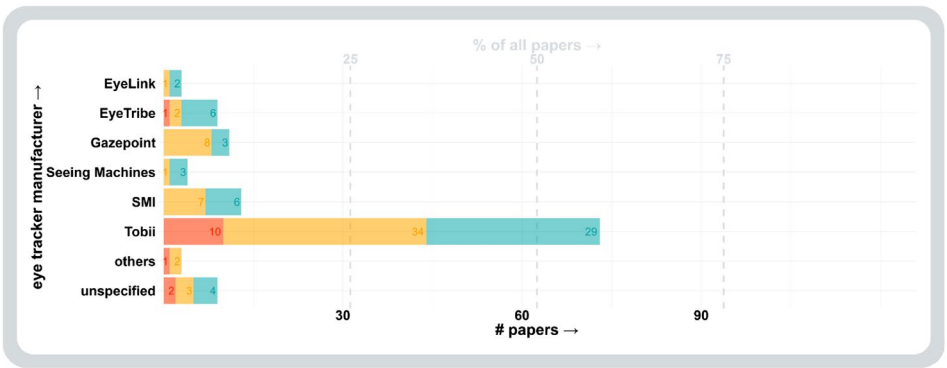


Fig. 15 Number of papers by eye tracker manufacturer (red: short, yellow: regular, blue: long)

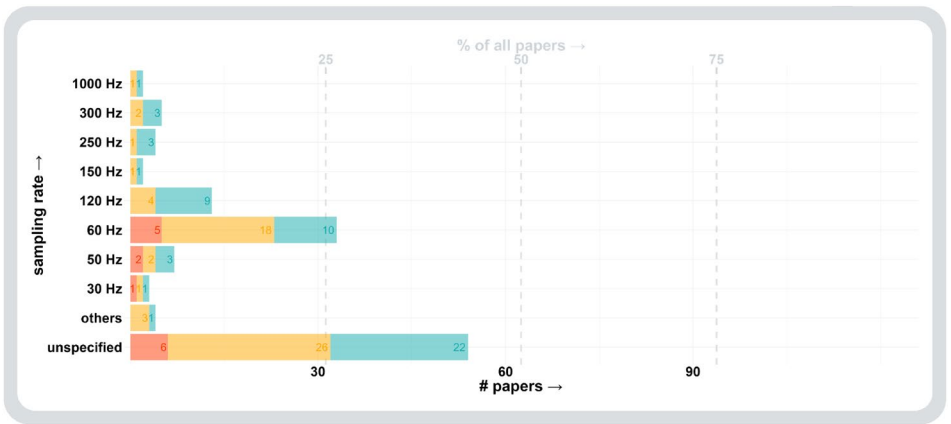


Fig. 16 Number of papers by sampling frequency (red: short, yellow: regular, blue: long)

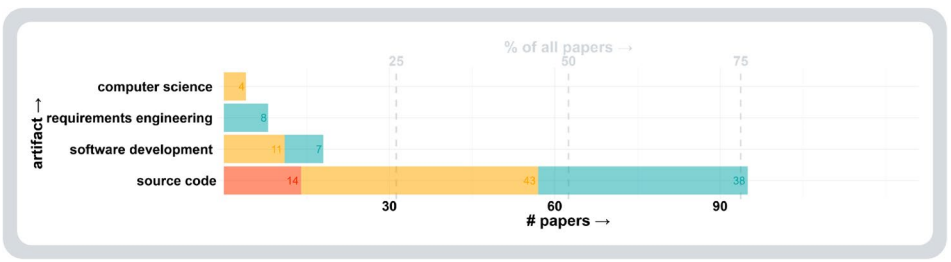


Fig. 17 Number of papers by SE-related artifact (red: short, yellow: regular, blue: long)

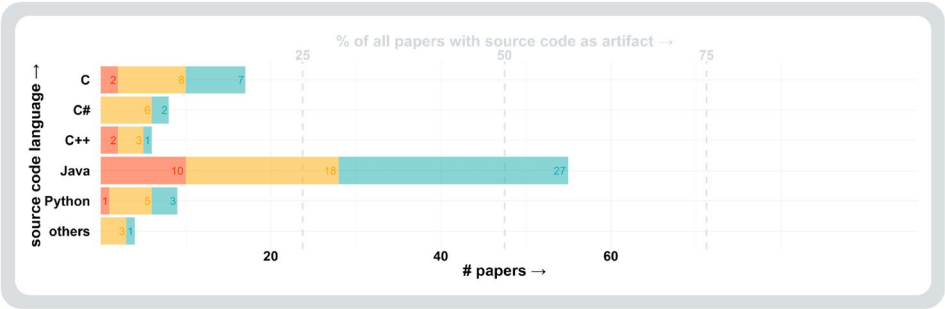


Fig. 18 Number of papers by source code languages (red: short, yellow: regular, blue: long)

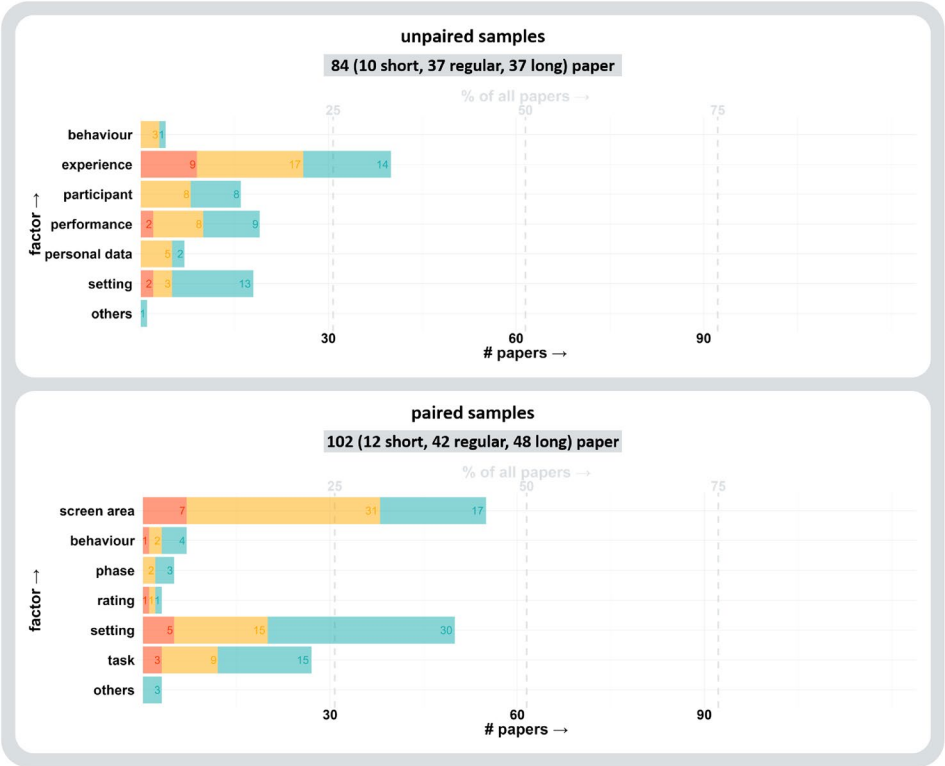


Fig. 19 Number of papers by grouping factor (red: short, yellow: regular, blue: long)

thereby, task scores and processing times are both evaluated in about half of the papers. Above that, conscious decisions (i.e., in the form of ratings) are found in about a quarter of the papers (i.e., 33 (2 short, 11 regular, 20 long)).

The eye tracking metrics used are detailed in Fig. 21—grouped similarly to [118], with the more generic term *region-based* instead of *scan-path-based*. One metric is missing from the figure: the average pupil diameter—used in five papers [3, 4, 10, 23, 56]. However, the figure confirms the dominance of fixation-based metrics in agreement with [119] and [91].

Others

In this section, we collect the methods used in at least two papers that do not fall into one of the above categories—starting with *hypothesis tests* that do not investigate group differences. About one in five papers reports the use of normality tests; four papers mention variance equality tests; two papers each refer to sphericity tests or Chi-squared test of independence. Regarding *visualizations*, in five papers we observe a representation similar to the line plot-like visualization of AOI transitions, but with a quantitatively scaled y-axis (i.e., coordinates) called *gaze transition diagram* [84, 110, 111]. Two papers use bar-plot-like elements within tables. Apart from these, there are two noticeable *measures*: inter-rater reliability in five and Kullback–Leibler divergence in two papers. We also need to report the use of three classification models, two cluster analyses, and five transformations (including three log transformations). Sequential analyses are performed in nine papers,

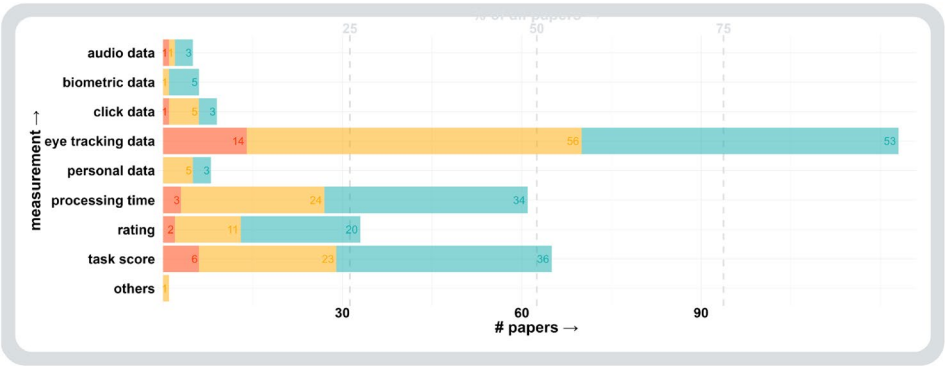


Fig. 20 Number of papers by measured quantity (red: short, yellow: regular, blue: long)

including four times based on the Needleman-Wunsch algorithm and three times using the sequential pattern mining algorithm.

Discussion

This article aims to provide an overview of the analysis methods or settings commonly used in SE eye tracking studies. In this section, the most important findings are summarized together with the respective limitations.

Research Answers

In the following, we summarize the results presented in "Results" with respect to the research questions posed at the beginning of this paper (i.e., "Research Questions"). We underline our statements with the respective percentage of all papers—limiting ourselves to what is used in at least 10% of the papers.

(RQ1) Regarding descriptive statistics, we mainly observe the measures mean (86%), standard deviation (53%), and median (18%) as well as the visualizations bar plot (34%), box plot (25%), line plot (22%), and scatter plot (10%).

(RQ2) As for inferential statistics, hypothesis tests for group differences predominate (69%)—mainly for a metric dependent variable with one independent variable (66%) or for any dependent variable with multiple independent variables (12%). In addition, we see correlation analyses (18%)—almost all between two metric scaled variables.

(RQ3) Gaze visualizations focus mainly on AOI transitions (23%) or are presented in superimposed to the stimulus (35%); for the latter, heat maps are predominantly used (27%).

(RQ4) Sample sizes are often relatively small—83% with up to 40 and 46% with up to 20 participants.

(RQ5) Eye tracking devices are mainly from Tobii (58%), followed by SMI (10%).

(RQ6) Sampling rates are unspecified in 43% of cases, the most commonly reported sampling rate is 60 Hz (26%), followed by 120 Hz (10%).

(RQ7) 76% of the papers use source code for their material—alone 44% the language Java language and 25% C; another 14% rely on elements of software development other than the code itself.

(RQ8) For data grouping that results in unpaired samples, the most common grouping factor is experience (32%), followed by performance (15%), setting (14%), and participants (13%). A split into paired samples is mainly based on screen areas (44%), settings (40%), or tasks (22%).

(RQ9) Quantitatively measured variables primarily include eye tracking data (98%), task scores (52%), processing times (49%), and ratings (26%). The eye tracking data collected are mainly fixation-based (78%) or region-based (57%); only 18% use saccade-based measures.

Threats to Validity

The validity of the presented findings is subject to some limitations regarding the sample, analysis criterion, page count, interpretation, and generalizability. These are explained in more detail below.

- *Sample:* We only systematically search the IEEE and ACM databases and do not perform a snowball search based on the results, which means that we do not have an exhaustive sample of SE eye tracking studies. In addition, based on our search method, we omitted papers with non-SE artifacts that provide results relevant to the SE community, such as [41]. The goal is to get an overview of the methods used rather than sifting through every paper on the topic; IEEE and ACM are the publishers of the major conferences in the field, so we should cover a large proportion of relevant papers.

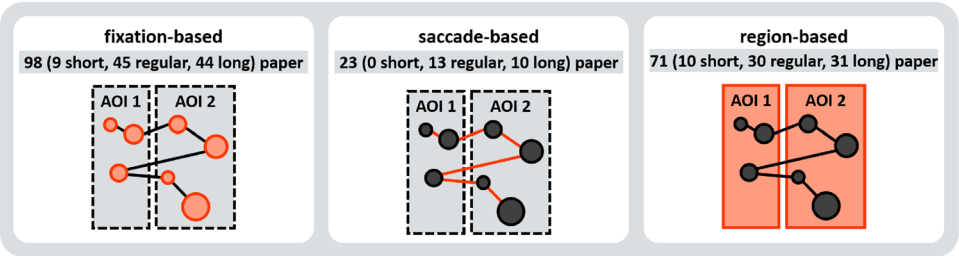


Fig. 21 Number of papers by eye tracking data type

- *Analysis criterion:* For the sake of simplicity, we take the number of papers using a particular method as a measure of its prevalence; it does not matter how often the paper uses the method or how detailed it is. In addition, we do not take into account the research teams behind the papers—for example, all reports on the descriptive statistical measures kurtosis and variance come from the same research team; for skewness, four out of five papers are from the same authors.
- *Page count:* We include papers of all lengths and qualities in our analysis, since some two-page papers turn out to be more extensive in terms of analysis than other papers of ten or more pages; the longer papers often focus more on related work or a broader range of analyses. To compensate for differences in length, we divide the papers into three categories: short, regular, and long. We do not compensate for the quality of research.
- *Interpretation:* The classification of methods is not always entirely clear; the documentation of inferential statistical methods is often incomplete, so that one can only guess whether paired samples are present or not. In addition, there are special cases, such as. [37], which presents a line plot-like representation of what he calls a velocity graph. Since the plot is a automatic report exported from the eye tracker software and is not a visualization of an extracted metric, we classify it as an "other visualization" rather than a "line plot".
- *Generalizability:* Only very few analysis methods are specific to the SE domain (e.g., the AOI visualizations based on the code line number). Most methods are quite general (e.g., statistical methods). Although we cannot guarantee the applicability of the results to other domains, we strongly assume similarities.

To cope with these limitations, the extracted data set is available for downloading at zenodo (www.doi.org/10.5281/zenodo.11279180). It is structured to contain one column for each item from Table 1 and one row for each selected paper.

Conclusion

As part of future work, it would be interesting to analyze the same group of papers in more detail—this time with respect to the evolution of method usage over time or the influence of specific author groups. With this we can, for example, answer the question of whether AOI visualizations in the form of line plots are more popular than state-diagram-like ones whether the latter have only recently emerged and are replacing the former. Apart of that, it might be useful to do the same analysis for an expanded list of papers or for papers from a different field—in order to understand whether the choice of analysis methods is driven by the field (i.e., SE) or the research method (i.e., eye tracking).

As a community, we need to work on consistency of methods. In the end, the choice of method (e.g., whether to use a scarf-plot-like AOI visualization or a gaze plot) will always remain at the discretion of the respective authors. Yet the naming of these methods should be consistent—so that when we talk about a scan path, we all have a common representation in mind, not some of us a scarf-plot-like AOI visualization and others a gaze plot. The results of the present article can serve as basis for such standardization—providing for the elements that should be included as well as current naming conflicts, e.g., the term *scan path* as explained in "Gaze Visualizations". With a consistent naming system, we can then go one step further: We will provide a practical guide to eye tracking research, showing which analysis methods are best suited for which type of investigation.

As authors, we need to be transparent. This includes the use of naming conventions once they are promoted by the community. But we can also make an active contribution to transparency regardless of conventions: with comprehensive reporting. Clearly identify the design and the (in)dependent variables of the study. Be specific when we referring to the applied eye tracking metrics (e.g., which are used for computing a heat map). State the choice of statistical tests, not just the results. This extends to the idea of open science, i.e., making the data sets obtained available—not only for replication, but also for ease of understanding. This will enable a reader of the respective work to really understand the analysis and evaluate it for themselves instead of only taking note of the results.

Acknowledgements The paper is supported by the 'German Federal Ministry of Education and Research' (BMBF) within the funding project HASKI (FKZ: 16DHBKI035).

Author Contributions All authors contributed to the study conception. Literature research and analysis was performed by Lisa Grabinger. The first draft of the manuscript was written by Lisa Grabinger. The other authors reviewed and approved the final manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL. The paper is supported by the 'German Federal Ministry of Education and Research' (BMBF) within the funding project HASKI (FKZ: 16DHBKI035).

Data Availability Statement The extracted data set will be available for downloading via www.zenodo.org at publication.

Declarations

Conflict of interest Not applicable.

Research involving human and/or animals Not applicable.

Informed consent Not applicable.

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Abid N, Maletic J, Sharif B. Using developer eye movements to externalize the mental model used in code summarization tasks. ACM; 2019. pp. 1–9. https://doi.org/10.1145/3314111.3319834.
2. Abid N, Sharif B, Dragan N, et al. Developer reading behavior while summarizing java methods: size and context matters, vol. 2019-May. IEEE; 2019. pp. 384–395, https://doi.org/10.1109/ICSE.2019.00052.
3. Ahmad H, Karas Z, Diaz K, et al. How do we read formal claims? eye-tracking and the cognition of proofs about algorithms. IEEE; 2023. pp. 208–220. https://doi.org/10.1109/ICSE48619.2023.00029.
4. Ahrens M, Schneider K, Busch M. Attention in software maintenance: an eye tracking study. IEEE; 2019. pp. 2–9. https://doi.org/10.1109/EMIP.2019.00009.
5. Ahsan Z, Obaidellah U. Predicting expertise among novice programmers with prior knowledge on programming tasks. IEEE; 2020. pp. 1008–1016.
6. Ahsan Z, Obaidellah U. Is clustering novice programmers possible? Investigating scanpath trend analysis in programming tasks. ACM; 2023. pp. 1–7. https://doi.org/10.1145/3588015.3589193.
7. Alcocer J, Cossio-Chavalier A, Rojas-Stambuk T, et al. An eye-tracking study on the use of split/unified code change views for bug detection. IEEE Access. 2023;11:136195–205. https://doi.org/10.1109/ACCESS.2023.3336859.
8. Ali N, Sharafl Z, Guéhéneuc YG, et al. An empirical study on requirements traceability using eye-tracking. IEEE; 2012. pp. 191–200. https://doi.org/10.1109/ICSM.2012.6405271.
9. Aljehane S, Sharif B, Maletic JI. Determining differences in reading behavior between experts and novices by investigating eye movement on source code constructs during a bug fixing task, vol. PartF169257. ACM; 2021. pp. 1–6. https://doi.org/10.1145/3448018.3457424.
10. Aljehane S, Sharif B, Maletic J. Studying developer eye movements to measure cognitive workload and visual effort for expertise assessment, vol. 7. In: PACM HCI. 2023. pp. 1–18. https://doi.org/10.1145/3591135.
11. Al Madi N, Busjahn T, Sharif B. Summary of the tenth international workshop on eye movements in programming (EMIP 2022). ACM SIGSOFT SEN. 2023;48:79–80. https://doi.org/10.1145/3573074.3573094.
12. Andrzejewska M, Stolińska A. Do structured flowcharts outperform pseudocode? Evidence from eye movements. IEEE Access. 2022;10:132965–75. https://doi.org/10.1109/ACCESS.2022.3230981.
13. Balzert H. Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering. 3rd ed. Heidelberg: Spektrum Akademischer Verlag; 2009.
14. Barik T, Smith J, Lubick K, et al. Do developers read compiler error messages? IEEE; 2017. pp. 575–585. https://doi.org/10.1109/ICSE.2017.59.
15. Bauer J, Siegmund J, Peitek N, et al. Indentation: simply a matter of style or support for program comprehension? vol 2019-May. IEEE; 2019. pp. 154–164. https://doi.org/10.1109/ICPC.2019.00033.
16. Bednarik R, Tukiainen M. Effects of display blurring on the behavior of novices and experts during program debugging. ACM; 2005. pp. 1204–1207. https://doi.org/10.1145/1056808.1056877.
17. Bednarik R, Tukiainen M. An eye-tracking methodology for characterizing program comprehension processes. ACM; 2006. pp. 125–132. https://doi.org/10.1145/1117309.1117356.
18. Bednarik R, Tukiainen M. Temporal eye-tracking data: evolution of debugging strategies with multiple representations. ACM; 2008. pp. 99–102; https://doi.org/10.1145/1344471.1344497.
19. Bednarik R, Schulte C, Budde L, et al. Eye-movement modeling examples in source code comprehension: a classroom study. ACM; 2018. pp. 1–8. https://doi.org/10.1145/3279720.3279722.
20. Beelders T. Eye-tracking analysis of source code reading on a line-by-line basis. ACM; 2022. pp. 1–7. https://doi.org/10.1145/3524488.3527364.
21. Beelders T, Plessis JPD. The influence of syntax highlighting on scanning and reading behaviour for source code, vol. 26–28-September-2016. ACM; 2016. pp. 1–10. https://doi.org/10.1145/2987491.2987536.
22. Begel A, Vrzakova H. Eye movements in code review. ACM; 2018. pp. 1–5. https://doi.org/10.1145/3216723.3216727.
23. Bernard L, Raina S, Taylor B, et al. Minimizing cognitive load in cyber learning materials—an eye tracking study, vol. PartF169257. ACM; 2021. pp. 1–6, https://doi.org/10.1145/3448018.3458617.
24. Bertram I, Hong J, Huang Y, et al. Trustworthiness perceptions in code review: an eye-tracking study. IEEE Computer Society; 2020. pp. 1–6. https://doi.org/10.1145/3382494.3422164.
25. Blascheck T, Sharif B. Visually analyzing eye movements on natural language texts and source code snippets. ACM; 2019. pp. 1–9. https://doi.org/10.1145/3314111.3319917.
26. Busjahn T, Tamm S. A deeper analysis of AOI coverage in code reading. ACM; 2021. pp. 1–7. https://doi.org/10.1145/3448018.3457422.
27. Busjahn T, Schulte C, Busjahn A. Analysis of code reading to gain more insight in program comprehension. ACM; 2011. pp. 1–9. https://doi.org/10.1145/2094131.2094133.

28. Busjahn T, Bednarik R, Schulte C. What influences dwell time during source code reading? Analysis of element type and frequency as factors. ACM; 2014. pp. 335–338. https://doi.org/10.1145/2578153.2578211.
29. Busjahn T, Bednarik R, Begel A, et al. Eye movements in code reading: relaxing the linear order, vol. 2015-August. IEEE; 2015. pp. 255–265. https://doi.org/10.1109/ICPC.2015.36.
30. Busjahn T, Simon, Paterson JH. Looking at the main method—an educator's perspective. ACM; 2021. pp. 1–10. https://doi.org/10.1145/3488042.3488068.
31. Chandrika K, Amudha J, Sudarsan SD. Recognizing eye tracking traits for source code review. IEEE; 2017. pp. 1–8. https://doi.org/10.1109/ETFA.2017.8247637.
32. Cheng G, Poon L, Lau W, et al. Applying eye tracking to identify students' use of learning strategies in understanding program code. ACM; 2019. pp. 140–144. https://doi.org/10.1145/3345120.3345144.
33. Chitalkina N, Bednarik R, Puurtinen M, et al. When you ignore what you see: How to study proof-readers' error in pseudocode reading. ACM; 2020. pp. 1–5. https://doi.org/10.1145/3379156.3391979.
34. Costa JSD, Gheyi R. Evaluating the code comprehension of novices with eye tracking. 2023. pp. 332–341. https://doi.org/10.1145/3629479.3629490.
35. Crosby M, Stelovsky J. How do we read algorithms? A case study. Computer. 1990;23:24–35. https://doi.org/10.1109/2.48797.
36. D'Angelo S, Begel A. Improving communication between pair programmers using shared gaze awareness, vol. 2017-January. ACM; 2017. pp. 6245–6255. https://doi.org/10.1145/3025453.3025573.
37. Davis D, Zhu F. Understanding and improving secure coding behavior with eye tracking methodologies. ACM; 2020. pp. 107–114. https://doi.org/10.1145/3374135.
38. Fakhoury S, Ma Y, Arnaoudova V, et al. The effect of poor source code lexicon and readability on developers' cognitive load. ACM; 2018. pp. 286–296. https://doi.org/10.1145/3196321.3196347.
39. Gorski P, Möller S, Wiefling S, et al. 'I just looked for the solution!' on integrating security-relevant information in non-security API documentation to support secure coding practices. IEEE Trans Softw Eng. 2022;48:3467–84. https://doi.org/10.1109/TSE.2021.3094171.
40. Goswami A, Walia G, McCourt M, et al. Using eye tracking to investigate reading patterns and learning styles of software requirement inspectors to enhance inspection team outcome. ACM; 2016. pp. 1–10. https://doi.org/10.1145/2961111.2962598.
41. Grabinger L, Hauser F, Mottok J. On the perception of graph layouts. J Softw (Malden) Early View 2023;1–18. https://doi.org/10.1002/smr.2599.
42. Gralha C, ao MG, ao Araújo J. Analysing gender differences in building social goal models: a quasi-experiment, vol. 2019-September. IEEE; 2019. pp. 165–176. https://doi.org/10.1109/RE.2019.00027.
43. Gralha C, Pereira R, Goulão M, et al. On the impact of using different templates on creating and understanding user stories. IEEE; 2021. pp. 209–220. https://doi.org/10.1109/RE51729.2021.00026.
44. Gunawan F, Wijaya O, Soewito B, et al. An analysis of concentration region on powerpoint slides using eye tracking. IEEE; 2017. pp. 1–5. https://doi.org/10.1109/EECSI.2017.8239082.
45. Han L, Chen T, Demartini G, et al. A data-driven analysis of behaviors in data curation processes. ACM Trans Inf Syst. 2023;41:1–35. https://doi.org/10.1145/3567419.
46. Hauser F, Mottok J, Gruber H. Eye tracking metrics in software engineering. ACM; 2018. pp. 39–44. https://doi.org/10.1145/3209087.3209092.
47. Hauser F, Schreistetter S, Reuter R, et al. Code reviews in C++: preliminary results from an eye tracking study. ACM; 2020. pp. 1–5. https://doi.org/10.1145/3379156.3391980.
48. Hauser F, Grabinger L, Mottok J, et al. Visual expertise in code reviews: using holistic models of image perception to analyze and interpret eye movements. ACM; 2023. pp. 1–7. https://doi.org/10.1145/3588015.3589189.
49. Hejmady P, Narayanan H. Multiple visualizations and debugging: how do we co-ordinate these? ACM; 2012. pp. 1547–1552. https://doi.org/10.1145/2212776.2223670.
50. Hejmady P, Narayanan H. Visual attention patterns during program debugging with an IDE. ACM; 2012. pp. 197–200. https://doi.org/10.1145/2168556.2168592.
51. Ho HF, Huang DH. Differences of reading processes between and gate and or gate: evidences from eye movement. IEEE; 2014. pp. 123–128. https://doi.org/10.1109/EITT.2014.29.
52. Holmqvist K, Nyström M, Andersson R, et al. Eye tracking: a comprehensive guide to methods and measures. Oxford: Oxford University Press; 2011.
53. Homann A, Grabinger L, Hauser F, et al. An eye tracking study on MISRA C coding guidelines. ACM; 2023. pp. 130–137. https://doi.org/10.1145/3593663.3593671.
54. Hüsing S, Schulte C, Sparmann S, et al. Using worked examples for engaging in epistemic programming projects. 2024. pp. 443–449. https://doi.org/10.1145/3626252.3630961.
55. Iwasa H, Akashi T, Ohno S. A fundamental study on gaze tracking for programming learners based on hierarchical segmentation. 2023. pp. 558–559. https://doi.org/10.1109/GCCE59613.2023.10315517.
56. Jbara A, Feitelson D. How programmers read regular code: A controlled experiment using eye tracking, vol. 2015-August. IEEE; 2015. pp. 244–254. https://doi.org/10.1109/ICPC.2015.35.
57. Jermann P, Nüssli MA. Effects of sharing text selections on gaze cross-recurrence and interaction quality in a pair programming task. ACM; 2012. pp. 1125–1134. https://doi.org/10.1145/2145204.2145371.
58. Jermann P, Sharma K. Gaze as a proxy for cognition and communication. IEEE; 2018. pp. 152–154. https://doi.org/10.1109/ICALT.2018.00043.
59. Kano T, Sakagami R, Akakura T. Modeling of cognitive processes based on gaze transition during programming debugging. IEEE; 2021. pp. 412–413. https://doi.org/10.1109/LifeTech52111.2021.9391940.
60. Karras O, Risch A, Schneider K. Interrelating use cases and associated requirements by links: an eye tracking study on the impact of different linking variants on the reading behavior, vol. Part F137700. ACM; 2018. pp. 2–12. https://doi.org/10.1145/3210459.3210460.
61. Kather P, Duran R, Vahrenhold J. Through (tracking) their eyes: abstraction and complexity in program comprehension. ACM Trans Comput. 2022;22:1–33. https://doi.org/10.1145/3480171.
62. Katona J, Kovari A, Costescu C, et al. The examination task of source-code debugging using GP3 eye tracker. IEEE; 2019. pp. 329–333. https://doi.org/10.1109/CogInfoCom47531.2019.9089952.
63. Katona J, Kovari A, Heldal I, et al. Using eye-tracking to examine query syntax and method syntax comprehension in LINQ. IEEE; 2020. pp. 437–444. https://doi.org/10.1109/CogInfoCom50765.2020.9237910.
64. Kevic K. Using eye gaze data to recognize task-relevant source code better and more fine-grained. IEEE; 2017. pp. 103–105.. https://doi.org/10.1109/ICSE-C.2017.152.

65. Kevic K, Walters B, Shaffer T, et al. Tracing software developers' eyes and interactions for change tasks. ACM; 2015. pp. 202–213. https://doi.org/10.1145/2786805.2786864.
66. Kilic O, Say B, Demirörs O. Cognitive aspects of error finding on a simulation conceptual modeling notation. IEEE; 2008. pp. 1–6. https://doi.org/10.1109/ISCIS.2008.4717930.
67. Kitchenham B. Guidelines for performing systematic literature reviews in software engineering. 2007.
68. Konopka M, Talian A, Tvarozek J, et al. Data flow metrics in program comprehension tasks. ACM; 2018. pp. 1–6. https://doi.org/10.1145/3216723.3216728.
69. Krejtz K, Duchowski A, Wisiecka K, et al. Entropy of eye movements while reading code or text. ACM; 2022. pp. 8–14; https://doi.org/10.1145/3524488.3527365.
70. Kuang P, Söderberg E, Niehorster D, et al. Toward gaze-assisted developer tools. IEEE; 2023. pp. 49–54. https://doi.org/10.1109/icse-nier58687.2023.00015.
71. Li X, Liu W, Wang W, et al. Assessing students' behavior in error finding programming tests: an eye-tracking based approach. IEEE; 2019. pp. 1–6. https://doi.org/10.1109/TALE48000.2019.9225906.
72. Li X, Liu W, Liu H, et al. Task-oriented analysis on debugging process based on eye movements and IDE interactions. IEEE; 2021. pp. 379–384. https://doi.org/10.1109/ICCSE51940.2021.9569438.
73. Lin YT, Wu CC, Hou TY, et al. Tracking students' cognitive processes during program debugging—an eye-movement approach. IEEE Trans Educ. 2016;59:175–86. https://doi.org/10.1109/TE.2015.2487341.
74. Lin YT, Liao YZ, Hu X, et al. EEG activities during program comprehension: an exploration of cognition. IEEE Access. 2021;9:120407–21. https://doi.org/10.1109/ACCESS.2021.3107795.
75. Liu L, Liu W, Li X, et al. Eye-tracking based performance analysis in error finding programming test. IEEE; 2020. pp. 477–482. https://doi.org/10.1109/ICCSE49874.2020.9201882.
76. Liu L, Liu W, Li X, et al. An analysis scheme to interpret students' cognitive process in error finding test. ACM; 2020. pp. 220–225. https://doi.org/10.1145/3425329.3425350.
77. Luo F, Liu R, Awoyemi ID, et al. Novel insights into elementary girls' experiences in physiological computing. 2024. pp. 764–770. https://doi.org/10.1145/3626252.3630900.
78. Al Madi N. How readable is model-generated code? Examining readability and visual inspection of GitHub copilot. ACM; 2022. pp. 1–5. https://doi.org/10.1145/3551349.3560438.
79. Al Madi N, Peterson C, Sharif B, et al. From novice to expert: analysis of token level effects in a longitudinal eye tracking study, vol. 2021-May. IEEE; 2021. pp. 172–183. https://doi.org/10.1109/ICPC52881.2021.00025.
80. Mansoor N, Peterson C, Dodd M, et al. Assessing the effect of programming language and task type on eye movements of computer science students. ACM Trans Comput Educ. 2023. https://doi.org/10.1145/3632530.
81. McChesney I, Bond R. Gaze behaviour in computer programmers with dyslexia: considerations regarding code style, layout and crowding. ACM; 2018. pp. 1–5. https://doi.org/10.1145/3216723.3216724.
82. McChesney I, Bond R. Observations on the linear order of program code reading patterns in programmers with dyslexia. ACM; 2020. pp. 81–89. https://doi.org/10.1145/3383219.3383228.
83. McChesney I, Bond R. The effect of crowding on the reading of program code for programmers with dyslexia, vol. 2021-May. IEEE; 2021. pp. 300–310. https://doi.org/10.1109/ICPC52881.2021.00036.
84. Melo J, Narizo FB, Hansen DW, et al. Variability through the eyes of the programmer. IEEE; 2017. pp. 34–44; https://doi.org/10.1109/ICPC.2017.34.
85. Molina A, Paredes M, Redondo M, et al. Assessing representation techniques of programs supported by GreedEx. IEEE; 2014. pp. 53–58. https://doi.org/10.1109/SIIE.2014.7017704.
86. Molina-Diaz A, Paredes-Velasco M, Redondo-Duque M, et al. Evaluation experiences of the representation techniques of greedy programs: application to the GreedEx tool, vol. 11. In: IEEE-RITA. 2016. pp. 179–86. https://doi.org/10.1109/RITA.2016.2589620.
87. Nakayama M, Harada H. Eye movement features in response to comprehension performance during the reading of programs. ACM; 2020. pp. 1–5. https://doi.org/10.1145/3379156.3391981.
88. Nevalainen S, Sajaniemi J. An experiment on short-term effects of animated versus static visualization of operations on program perception. ACM; 2006. pp. 7–16. https://doi.org/10.1145/1151588.1151591.
89. Nivala M, Hauser F, Mottok J, et al. Developing visual expertise in software engineering: an eye tracking study, vol. 10–13-April-2016. IEEE; 2016. pp. 613–620. https://doi.org/10.1109/EDUCON.2016.7474614.
90. Obaidellah U, Haek MA. Evaluating gender difference on algorithmic problems using eye-tracker. ACM; 2018. pp. 1–8. https://doi.org/10.1145/3204493.3204537.
91. Obaidellah U, Haek MA, Cheng P. A survey on the usage of eye-tracking in computer programming. ACM Comput Surv. 2018;51:1–58. https://doi.org/10.1145/3145904.
92. Obaidellah U, Raschke M, Blascheck T. Classification of strategies for solving programming problems using AOI sequence analysis. ACM; 2019. pp. 1–9. https://doi.org/10.1145/3314111.3319825.
93. Obaidellah U, Blascheck T, Guarnera D, et al. A fine-grained assessment on novice programmers' gaze patterns on pseudocode problems. ACM; 2020. pp. 1–5. https://doi.org/10.1145/3379156.3391982.
94. Oliveira BD, Ribeiro M, Costa JASD, et al. Atoms of confusion: the eyes do not lie. ACM; 2020. pp. 243–252. https://doi.org/10.1145/3422392.3422437.
95. il Park K, Sharif B. Assessing perceived sentiment in pull requests with emoji: evidence from tools and developer eye movements. IEEE; 2021. pp. 1–6. https://doi.org/10.1109/SEmotion52567.2021.00009.
96. il Park K, Weill-Tessier P, Brown N, et al. An eye tracking study assessing the impact of background styling in code editors on novice programmers' code understanding. ACM; 2023. pp. 444–463. https://doi.org/10.1145/3568813.3600133.
97. Peitek N, Siegmund J, Apel S. What drives the reading order of programmers? An eye tracking study. ACM; 2020. pp. 342–353. https://doi.org/10.1145/3387904.3389279.
98. Peitek N, Bergum A, Rekrut M, et al. Correlates of programmer efficacy and their link to experience: a combined EEG and eye-tracking study. ACM; 2022. pp. 120–131. https://doi.org/10.1145/3540250.3549084.
99. Peng F, Li C, Song X, et al. An eye tracking research on debugging strategies towards different types of bugs, vol. 2. IEEE; 2016. pp. 130–134. https://doi.org/10.1109/COMPSAC.2016.57.
100. Peterson C. Investigating the effect of polyglot programming on developers, vol. 2010-October. IEEE; 2021. pp. 1–2. https://doi.org/10.1109/VL/HCC51201.2021.9576404.
101. Peterson C, Abid N, Bryant C, et al. Factors influencing dwell time during source code reading—a large-scale replication experiment. ACM; 2019. pp. 1–4. https://doi.org/10.1145/3314111.3319833.

102. Peterson C, Halavick N, Saddler J, et al. A gaze-based exploratory study on the information seeking behavior of developers on stack overflow. ACM; 2019. pp. 1–6. https://doi.org/10.1145/3290607.3312801.
103. Peterson C, Saddler J, Blascheck T, et al. Visually analyzing students' gaze on C++ code snippets. IEEE; 2019. pp. 18–25. https://doi.org/10.1109/EMIP.2019.00011.
104. Peterson C, il Park K, Baysinger I, et al. An eye tracking perspective on how developers rate source code readability rules. IEEE; 2021. pp. 138–139. https://doi.org/10.1109/ASEW52652.2021.00037.
105. Rodeghero P, McMillan C. An empirical study on the patterns of eye movement during summarization tasks, vol. 2015-November. IEEE; 2015. pp. 11–20. https://doi.org/10.1109/ESEM.2015.7321188.
106. Rodeghero P, McMillan C, McBurney PW, et al. Improving automated source code summarization via an eye-tracking study of programmers. ACM; 2014. pp. 390–401. https://doi.org/10.1145/2568225.2568247.
107. Rodeghero P, Liu C, McBurney P, et al. An eye-tracking study of java programmers and application to source code summarization. IEEE Trans Softw Eng. 2015;41:1038–54. https://doi.org/10.1109/TSE.2015.2442238.
108. Saddler J. Looks can mean achieving: understanding eye gaze patterns of proficiency in code comprehension. ACM; 2019. pp. 1–3. https://doi.org/10.1145/3314111.3322876.
109. Saddler J, Peterson C, Sama S, et al. Studying developer reading behavior on stack overflow during API summarization tasks. IEEE; 2020. pp. 195–205. https://doi.org/10.1109/SANER48275.2020.9054848.
110. Santos D, Sant'Anna C. How does feature dependency affect configurable system comprehensibility?, vol. 2019-May. IEEE; 2019. pp. 19–29. https://doi.org/10.1109/ICPC.2019.00016.
111. Santos D, Sant'Anna C, Ribeiro M. An experiment on how feature dependent variables affect configurable system comprehensibility. ACM; 2023. pp. 61–70. https://doi.org/10.1145/3622748.3622755.
112. Santos M, Gralha C, Goulão M, et al. What is the impact of bad layout in the understandability of social goal models? IEEE; 2016. pp. 206–215. https://doi.org/10.1109/RE.2016.51.
113. Santos M, Gralha C, Goulão M, et al. On the impact of semantic transparency on understanding and reviewing social goal models. IEEE; 2018. pp. 228–239. https://doi.org/10.1109/RE.2018.00031.
114. Schmedes M, Ahrens M, Nagel L, et al. Enriching vision videos with text: an eye tracking study, vol. 2022-August. IEEE; 2022. pp. 77–87. https://doi.org/10.1109/RE54965.2022.00014.
115. Sepasi E, Balouchi K, Mercier J, et al. Towards a cognitive model of feature model comprehension: an exploratory study using eye-tracking, vol. A. ACM; 2022. pp. 21–31. https://doi.org/10.1145/3546932.3546995.
116. Sharafi Z, Soh Z, Guéhéneuc YG, et al. Women and men—different but equal: on the impact of identifier style on source code reading. IEEE; 2012. pp. 27–36. https://doi.org/10.1109/icpc.2012.6240505.
117. Sharafi Z, Marchetto A, Susi A, et al. An empirical study on the efficiency of graphical vs. textual representations in requirements comprehension. IEEE; 2013. pp. 33–42. https://doi.org/10.1109/ICPC.2013.6613831.
118. Sharafi Z, Shaffer T, Sharif B, et al. Eye-tracking metrics in software engineering. IEEE; 2015. pp. 96–103. https://doi.org/10.1109/APSEC.2015.53.
119. Sharafi Z, Soh Z, Guéhéneuc YG. A systematic literature review on the usage of eye-tracking in software engineering. Inf Softw. 2015;67:79–107. https://doi.org/10.1016/j.infsof.2015.06.008.
120. Sharafi Z, Sharif B, Guéhéneuc YG, et al. A practical guide on conducting eye tracking studies in software engineering. Empir Softw Eng. 2020;25:3128–74. https://doi.org/10.1007/s10664-020-09829-4.
121. Sharafi Z, Bertram I, Flanagan M, et al. Eyes on code: a study on developers' code navigation strategies. IEEE Trans Softw Eng. 2022;48:1692–704. https://doi.org/10.1109/TSE.2020.3032064.
122. Sharif B, Maletic JI. An eye tracking study on camelcase and under_score identifier styles. IEEE; 2010. pp. 196–205. https://doi.org/10.1109/ICPC.2010.41.
123. Sharif B, Maletic JI. An eye tracking study on the effects of layout in understanding the role of design patterns. IEEE; 2010. pp. 1–10. https://doi.org/10.1109/ICSM.2010.5609582.
124. Sharif B, Falcone M, Maletic JI. An eye-tracking study on the role of scan time in finding source code defects. ACM; 2012. pp. 381–384. https://doi.org/10.1145/2168556.2168642.
125. Sorg T, Abbad-Andaloussi A, Weber B. Towards a fine-grained analysis of cognitive load during program comprehension. IEEE; 2022. pp. 748–752. https://doi.org/10.1109/SANER53432.2022.00092.
126. Sparman S, Schulte C. Analysing the API learning process through the use of eye tracking. 2023. pp. 1–6. https://doi.org/10.1145/3588015.
127. Spinelli L, Pandey M, Oney S. Attention patterns for code animations: using eye trackers to evaluate dynamic code presentation techniques, vol. Part F137691. ACM; 2018. pp. 99–104. https://doi.org/10.1145/3191697.3214338.
128. Talsma R, Barendsen E, Smetsers S. Analyzing the influence of block highlighting on beginning programmers' reading behavior using eye tracking. ACM; 2020. pp. 1–10. https://doi.org/10.1145/3442481.3442505.
129. Tobii. Pro lab user manual v 1.194, 2022.
130. Turner R, Falcone M, Sharif B, et al. An eye-tracking study assessing the comprehension of C++ and python source code. ACM; 2014. pp. 231–234. https://doi.org/10.1145/2578153.2578218.
131. Uddin MS, Gaur V, Gutwin C, et al. On the comprehension of code clone visualizations: a controlled study using eye tracking. IEEE; 2015. pp. 161–170. https://doi.org/10.1109/SCAM.2015.7335412.
132. Ujbanyi T, Katona J, Sziladi G, et al. Eye-tracking analysis of computer networks exam question besides different skilled groups. IEEE; 2016. pp. 277–281.
133. Uwano H, Nakamura M, Monden A, et al. Analyzing individual performance of source code review using reviewers' eye movement. ACM; 2006. pp. 133–140. https://doi.org/10.1145/1117309.1117357.
134. Villamor M, Rodrigo M. Predicting successful collaboration in a pair programming eye tracking experiment. ACM; 2018. pp. 263–268. https://doi.org/10.1145/3213586.3225234.
135. Šaloun P, Malčik M, Andrešič D, et al. Using eyetracking to analyse how flowcharts are understood. IEEE; 2017. pp. 394–399. https://doi.org/10.1109/INFORMATICS.2017.8327281.
136. Weber T, Winiker C, Hussmann H. A closer look at machine learning code. ACM; 2021. pp. 1–6. https://doi.org/10.1145/3411763.3451679.
137. Yusuf S, Kagdi H, Maletic JI. Assessing the comprehension of UML class diagrams via eye tracking. IEEE; 2007. pp. 1–10. https://doi.org/10.1109/ICPC.2007.10.