The Journal of Systems and Software 220 (2025) 112261

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

An exploratory empirical eye-tracker study of visualization techniques for coverage of combinatorial interaction testing in software product lines[☆]

Kambiz Nezami Balouchi[a], Julien Mercier[b], Roberto E. Lopez-Herrejon[a,*]

[a] Department Software Engineering and IT, École de technologie supérieure, Montreal, Canada
[b] Department of Computer Science, Université du Québec à Montréal, Montréal, Canada

ARTICLE INFO

Keywords:
Software product lines
Combinatorial interaction testing
Covering array
Parallel dimensions plot
Scatter plot
Eye-tracker study

ABSTRACT

Software Product Lines (SPLs) typically provide a large number of configurations to cater to a set of diverse requirements of specific markets. This large number of configurations renders unfeasible to test them all individually. Instead, Combinatorial Interaction Testing (CIT) computes a representative sample according to criteria of the interactions of features in the configurations. We performed an empirical study using eye-tracker technologies to analyze the effectiveness of two basic visualization techniques at conveying test coverage information of ten case studies of varying complexity. Our evaluation considered response accuracy, time-on-task, metacognitive monitoring, and visual attention. The study revealed clear advantages of a visualization technique over the other in three evaluation aspects, with a reverse effect depending on the strength of the coverage and distinct areas of visual attention.

1. Introduction

Testing is an essential activity in any software development project. This activity becomes more challenging when multiple variations of a software system must be considered simultaneously as it is the case of *Software Product Lines (SPLs)*. Several approaches to test SPLs have been proposed over the last years (Lopez-Herrejon et al., 2015). Salient among them are those based on *Combinatorial Interaction Testing (CIT)*, whereby different combinations of selected and non-selected features are considered for testing according to different criteria. These approaches typically yield a large number of combinations, even for SPLs with a small number of features, which are commonly presented to software engineers in textual format. But using this format, tasks such as assessing whether a combination is covered or not by a test suite becomes more error-prone and time consuming.

The motivation of our work is to assess empirically the suitability of basic visualization techniques for conveying the information required to perform simple testing tasks of CIT applied to SPLs. Thus, for our exploratory study, we selected two of the most basic visualization techniques (Ward et al., 2010; Telea, 2015; Ware, 2015), and devised an experiment to discern their relative trade-offs. We chose tasks drawn from various SPL case studies commonly used by the research community (Ferreira et al., 2021). Our analysis focused on: *(i)* the accuracy of responses, *(ii)* how quick were those responses, *(iii)* how confident were participants in their responses, and *(iv)* the visual attention effort required by participants. The latter aspect was analyzed by employing eye-tracker techniques on distinct areas of the visualizations that contained the elements to solve the required tasks. Our study revealed clear advantages of one visualization technique over the other one in three of the evaluated aspects with distinguishable predominant areas of interest in the visualization stimuli.

We start by providing the background of our study, the illustration of the visualization techniques, and the description of the experimental design, followed by a detailed presentation of the results obtained and their analyses. We continue with the description of the threats to validity identified and how we addressed them. We conclude with an extensive related work, a summary of the main findings, and a sketch of further research avenues.

2. Background

In this section, we present the basic background required to describe our empirical study and put in context the results obtained. We start with an overview of Software Product Lines, followed by Combinatorial Interaction Testing applied to this software domain. Then, we present the basics of eye-tracking technologies and the mental models that constitute the theoretical framework of our work from a cognition

[☆] Editor: Laurence Duchien.

[*] Corresponding author.

E-mail addresses: kambiz.nezami-balouchi.1@ens.etsmtl.ca (K. Nezami Balouchi), mercier.julien@uqam.ca (J. Mercier), roberto.lopez@etsmtl.ca (R.E. Lopez-Herrejon).

https://doi.org/10.1016/j.jss.2024.112261
Received 30 January 2024; Received in revised form 7 October 2024; Accepted 18 October 2024
Available online 28 October 2024

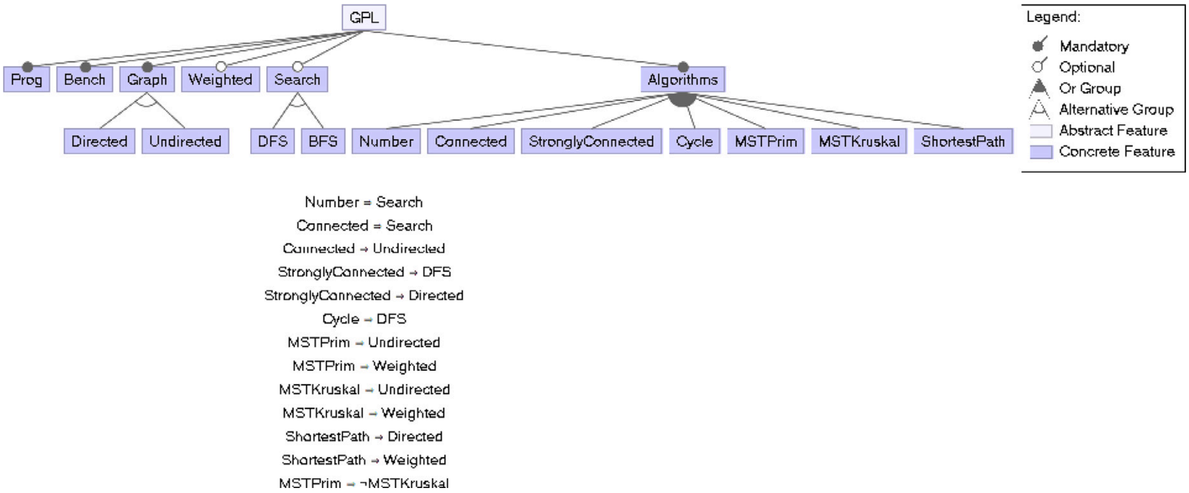


Fig. 1. Feature diagram of the running example Graph Product Line (GPL) rendered with FeatureIDE.

perspective. The last subsection describes the metacognitive aspects (i.e. the capacity of an individual to evaluate his/her own performance) that our study considered. Because of its relevance to our work, we provide a basic introduction to Software Visualization in Section 3, prior to describing the visualization techniques that we evaluated.

2.1. Software Product Lines (SPLs)

Software Product Lines (SPLs) typically provide a large number of possible configurations to meet the needs of specific organizations and users. In this context, the set of valid configurations is expressed by *variability models* of which *feature models* are the de facto standard (Benavides et al., 2010; Heradio et al., 2016). The visual depiction of a feature model is called a *feature diagram*. An example of feature model and its feature diagram is shown in Fig. 1.[1] This tree-like structure represents the features in the configurations as labeled boxes, their relations as lines with annotations, and additional cross-tree constraints with propositional logic.

The *Graph Product Line (GPL)*, depicted in Fig. 1, is a canonical SPL of basic graph types and algorithms that work with them (Lopez-Herrejon and Batory, 2001). We use GPL as a running example throughout the paper to illustrate both the background concepts and the visualization techniques used (see Section 3). This feature diagram indicates that feature GPL is always present (i.e. it is the root of the tree), and, for instance, that all configurations have graphs (mandatory feature Graph), some of them with weights (optional feature Weighted). Similarly, some configurations provide graph searching capability (optional feature Search), which can be either depth-first search (feature DFS) or breadth-first search (feature BFS) but not both (i.e. they form an *alternative group*). All configurations support graph algorithms (mandatory feature Algorithms) in different valid configurations (i.e. form an *or group* whereby at least one feature must be selected). For example, the algorithm that computes connected components (feature Connected) requires searching capability (feature Search), which is denoted as Connected => Search as shown in the figure.

Notice also from this model, that a configuration that contains both features DFS and BFS is not valid because these features are part of an alternative group. Similarly, a configuration with shortest path algorithm (feature ShortestPath) but without weighted graphs (i.e. feature Weighted not selected) is also invalid as it violates one of the cross-tree constraints. This simple feature diagram of GPL contains 18 features and denotes 73 valid configurations. Please refer to Benavides et al. (2010) and Heradio et al. (2016) for further details on feature diagrams and their formal semantics, and to Lopez-Herrejon and Batory (2001) for the GPL case study.

2.2. Combinatorial Interaction Testing (CIT)

Combinatorial Interaction Testing (CIT) is a testing approach whose main objective is computing a representative sample of configurations of a software system (Kuhn et al., 2016). When applied to SPLs, this sample contains configurations that are valid according to the corresponding variability model, commonly expressed as a feature model (Lopez-Herrejon et al., 2015, 2016). A sample is called a *covering array* of strength t, if its configurations contain at least one instance of all possible feature combinations (i.e. valid selected and unselected features) of a given size t.

For instance, when t=2, all the possible valid combinations of two features, or *pairs*, must be considered. In our running example GPL, the pair formed by features Search and DFS, and the pair formed by features MSTPrim and Weighted are two of the 418 total number of pairs that GPL contains. Another example of pair is that formed by features Cycle and !Weighted. Notice that this pair will be present in those configurations that have feature Cycle selected while feature Weighted is not selected, fact denoted by the symbol !.

When t=3, all possible valid combinations of three features, or *triplets*, must be considered. In GPL, two examples of triplets are: *(i)* the combination of features GPL, DFS, and Cycle, and *(ii)* the combinations of features Undirected, !ShortestPath, and !Directed. For GPL, there are 3322 possible triplets.

Over the last decade, multiple approaches have been proposed to compute covering arrays of different t values. They all provide different advantages and trade-offs (e.g., Hervieu et al., 2011; Garvin et al., 2011; Johansen et al., 2012; Lopez-Herrejon et al., 2013; Henard et al., 2014; Lopez-Herrejon et al., 2016; Ahmed et al., 2017). For example, for GPL the optimal[2] covering array for t=2 has 13 configurations, and for t=3 the covering array has 27 configurations.

Even in a small feature model example with a few features, there is a large amount of information regarding the pairs, triplets, covering arrays, etc. that must be conveyed to the testing engineers. Currently,

[1] The figure was rendered using the default layout of FeatureIDE (https://www.featureide.de/), a leading open-source tool in the SPL research community. We use it as is because visualization of feature models is outside the scope of our research work.

[2] A covering array is optimal if it has the minimum number of configurations possible that cover all the required pairs or triplets.

all this information is, at best, made available as plain text. This limitation negatively affects the performance of testing tasks for variable software systems (Lopez-Herrejon et al., 2018; Medeiros et al., 2023). Thus, the driving motivation of our work is exploring the cognitive effort required to perform CIT tasks that use alternative visualization techniques to convey the required information. We performed our analysis based on data captured with eye-tracking technology and self-assessment of metacognition monitoring as will be described in the following subsections.

2.3. Eye-tracking basics

Eye-tracking methodology involves applying techniques that measure eye movements to study visual attention and the cognitive processing of visual information while performing specific tasks (Duchowski, 2017). Eye-trackers use near-infrared lights, invisible to the human eye, that are directed towards the center of the eyes causing reflections on the pupil and the cornea. These reflections are used to gather information about the eye movements and directions (Duchowski, 2017). The premise is that visual attention triggers the cognitive processes required for the comprehension and resolution of tasks. Then, such cognitive processes direct the visual attention to specific locations on the visual field of the participants. Therefore, the allocations involving visual attention can be used as a proxy of the cognitive demands required by a task.

Two assumptions ground this relationship between eye movements and cognitive processes and are foundational in interpreting eye-tracking data (Just and Carpenter, 1980): *(i) eye-mind assumption*, and *(ii) immediacy assumption*. Respectively, these assumptions state that "there is no appreciable lag between what is being fixated and what is being processed" and that "the interpretations at all levels of processing are not deferred; they occur as soon as possible" (Just and Carpenter, 1980).

Consequently, eye-tracking is one of the best and most direct approaches to analyze how visual attention is allocated for performing tasks and for learning in highly visual domains (Peterson et al., 2004; Lai et al., 2013). As such, eye-tracking methodology has led to an extensive and long-standing body of research, both basic and applied (e.g., Duchowski, 2017; Holmqvist and Andersson, 2017), and has been employed to study cognitive constructs such as attention, comprehension, retention in memory, and processing difficulty in tasks involving written text, figures and diagrams (Rayner, 1998, 2009). The works by Obaidellah et al. (2018), and Sharafi et al. (2015a, 2020) summarize the research done with eye-tracking in the field of software engineering.

As defined by Sharafi et al. a *visual stimulus* is any object required to perform a task whose visual perception triggers the participant's cognitive processes to perform some actions related to the task (Sharafi et al., 2020). In our empirical study, the core of the visual stimuli is the visualization of the covering arrays as will be detailed in Sections 3 and 4.

There are numerous types of eye-tracker data and metrics that have been classified among several dimensions and relate to cognitive processing in different ways (Duchowski, 2017; Sharafi et al., 2020; Holmqvist and Andersson, 2017; Albert and Tullis, 2023). One important of data is *fixations* which are eye movements that stabilize the eye on an object of interest from a visual stimulus (Duchowski, 2017). The duration of fixations varies by the task performed and by the participants, with typical values ranging from 100 to 400 milliseconds (Duchowski, 2017; Holmqvist and Andersson, 2017). Commonly, fixations are analyzed on concrete areas of the visual stimulus called *Areas Of Interest (AOIs)* where the researcher is interested in gathering data because they are considered relevant to perform a task (Holmqvist and Andersson, 2017). Another important type of collected data is *saccades*, that are rapid eye movements, ranging from 10 to 100 milliseconds, and occur between two fixations.

The most pertinent metrics for a study must be selected according to the characteristics of the cognitive task under study. For our exploratory study, further details on the metrics used are presented in Section 4.1.

2.4. Theoretical framework

Our study focuses on combinatorial Interaction Testing (CIT) tasks applied to Software Product Lines. From a cognitive perspective, these tasks are considered as comprehension tasks that build, manipulate, and compare mental models with different components that intervene in their resolution (Sepasi et al., 2022; Détienne, 2001; Bidlake et al., 2020).

Mental models are representations of reality constructed by an individual, on which cognition operates through reasoning to make decisions. In the general context of program comprehension, a mental model consists of two parts (Sepasi et al., 2022; Détienne, 2001; Bidlake et al., 2020): *(i)* a *program model* that is constructed by programmers based on the structural knowledge of the code, and *(ii)* a *domain model* that is an abstract representation of the code built using knowledge of the domain and the real-world situation of the program code. Intuitively, the first part corresponds to the syntactic representation and its well-formedness, whereas the second part corresponds to its semantics or meaning in a real-world context.

We posit that two equivalent counterparts can be distinguished in the mental models required to comprehend and manipulate the visual stimuli of the CIT tasks of our study: *(i) visualization model* that is constructed based on the understanding of the visualization technique and its elements (e.g. geometric figures and lines connecting them), and *(ii) domain model* that is also an abstract representation of a domain in the real world, namely the SPL whose elements will be tested.

From this perspective, a CIT task is performed via a sequence of steps to build and operate on these mental models. The cognitive load of these tasks stems from the manipulation of the information necessary to accomplish them successfully, considering the typical limits of working memory in terms of capacity and duration (Chen et al., 2021). Inaccurate answers or unsuccessful performance commonly occurs when such limits are surpassed or when performing the tasks requires updates or repairs of the mental models already built. Do notice that the visual nature of the visualization model (i.e. the first mental model of our CIT tasks) enables the use of eye-tracking technologies in our study.

2.5. Metacognitive monitoring accuracy

Metacognitive monitoring refers to the individual's capacity to self-evaluate the outcome of a task and other various aspects of its performance (Winne, 1996). This capacity is critical for successful task performance because it is the only mechanism to detect and correct inconsistencies or errors while tasks are being carried out and afterwards.

The regulation of cognition has the following main components: planning the task; managing the information and monitoring its comprehension; and evaluating the performance (de Blume, 2022). Global judgements can concern a whole test or a whole performance. These are driven by more superficial information such as self-efficacy and familiarity with the task domain. In contrast, local or item-specific judgments refer to a single test item or subtask and are based on task-specific information. Discrepancies between the judgement and the actual achievement on the task have been termed the *"illusion of not knowing"* in the case of underconfidence and *"illusion of knowing"* in the case of overconfidence (Serra and Metcalfe, 2009).

Two main factors can explain inaccurate answers of participants. The first factor is the incapacity of participants to provide judgments that are more accurate. This is explained by the lack of access to item-specific cues like item difficulty, ease of cognitive processing, or ability to explain meaning. The second factor is the potential lack of motivation from participants to make more accurate judgments because of the effort required or because they simply do not understand the value of doing that. Additionally, participants might be biased because of motivational influences such as wishful thinking (Händel and Bukowski, 2019). In our study, participants perform the metacognitive monitoring after each task using two common self-assessment metrics as described in Section 4.1.

3. Visualization techniques

Let us start by defining some basic terminology. A *visualization* is the visual representation of a domain space using graphics, images, animated sequences, and sound augmentation to present the data, structure, and dynamic behavior of large, complex data sets that represent systems events, processes, objects, and concepts (Williams, 1995). Furthermore, *information visualization* is visualization applied to abstract quantities and relations to get insight in the data (Chi, 2002; Spence, 2014; Ware, 2015). The most common examples of information visualization techniques are trees, maps, and graphs (Telea, 2015). Finally, *software visualization* is the art and science of generating visual representations of various aspects of software and its development process (Diehl, 2007). The driving motivation of software visualization is to help the comprehension of software systems and to improve the productivity of the software development process (Diehl, 2007). Thus, the visualization techniques that we developed for our exploratory study, fall within the scope of software visualization because their goal is to support testing related tasks for Software Product Lines.

The CIT covering arrays, defined in Section 2.2, can be regarded as multidimensional data whereby the strength of the array corresponds to the number of dimensions. Hence, natural choices for visualizing the covering arrays are visualization techniques tailored for multidimensional data (Ward et al., 2010; Telea, 2015; Ware, 2015). We surveyed different alternative visualization techniques (e.g., Ward et al., 2010) and implementation platforms. Our purpose was to select — as starting point — two existing, basic, and commonly-used visualization techniques for multidimensional data that were implemented in an open-source tool capable of generating visualizations in a common and accessible format like HTML. From our survey, we selected for our exploratory study *scatter plots*[3] and *parallel dimensions plots*,[4] implemented using the Python-based library `Plotly`.[5] This library permits the generation of the visualizations in HTML format with basic support for user interaction as described below.

Fig. 2 illustrates the visualization techniques for our running example GPL. We have made available in our public Dataverse repository the four visualizations of this figure in standard HTML format (Lopez-Herrejon and Nezami Balouchi, 2024). Notice that the names of the features have been obfuscated with sequential integer numbers with the format Fi, with $i = 0..N - 1$ where N is the number of features. The symbol `!` is prefixed to a feature to indicate that it has not been selected in a pair or in a triplet. Renaming the features in our experiment prevents any interference with any knowledge that participants may have about the application domain of the feature models. Note as well that the values along the axes are not sorted in any particular order and come from the tool used to compute the CIT covering arrays as explained in Section 4.

Scatter plots. The first visualization technique selected is the scatter plot (Ward et al., 2010; Telea, 2015; Ware, 2015). This type of plot is based on the Cartesian plane with dots located at the intersection of the values shown on the axes. In our study, the values on the axes represent the features (selected and not selected) that form the pairs or triplets (two or three axes) of the covering array that is visualized. Hence, a dot in the plane corresponds to a single pair or triplet of a covering array, located at the intersection of the coordinates that correspond to the features that form such pair or triplet. The color of a dot represents the configuration or *solution* that covers the pair or triplet.[6] Thus, the number of dot colors used in this visualization is the number of solutions in the covering array visualized. Furthermore, this visualization has as navigation aid a small window that pops up while the user hovers over a dot. This window shows the dot's feature names (i.e. the coordinates of the dot) and the name of the solution that covers it.

Let us illustrate this visualization for the case of pairs using our running example GPL. For instance, consider the pink colored dot at the upper right corner of Fig. 2a. When the user hovers over it, the window that pops up indicates that this dot represents the pair `(!F4,F6)` — located at the intersection of `!F4` and `F6` — and that it is covered by solution `S6`. This covering array of GPL has 13 solutions. Consequently, this figure has 13 distinct colors for the dots. The pink dots in this figure depict then all the pairs covered by solution `S6`.

Let us now illustrate scatter plots for CIT covering arrays of $t = 3$. Fig. 2b shows the visualization of such covering array for our running example GPL. The dots are now depicted in a Cartesian plane of three dimensions, where the coordinates represent the values of the three features (selected and not selected) that form a triplet of the covering array. For instance, consider the upper right dot in this figure. When the user hovers over it, the window that pops up indicates that this dot represents the triplet `(F10,!F12,F13)` — located at the intersection of `F10`, `!F12`, and `F13` — and that it is covered by solution `S22`. This covering array has 27 configurations or solutions. Thus, the dots in this figure have a color chosen from a palette of 27 different colors.[7] In the example dot, the pink color corresponds to solution `S22`.

Our scatter plots for $t = 3$ have an additional navigation aid. Consider that a dot in a three-dimensional Cartesian plane sits at the intersection of three two-dimensional planes, one plane intersecting each dimension at the value of the corresponding axis. In our example dot, we can think of a plane intersecting axis `x` at feature `F10`, a second plane intersecting axis `y` at feature `!F12`, and a third plane intersecting axis `z` at feature `F13`. When the user hovers over this dot, our visualization displays black lines that sketch the borders of those three planes, thus facilitating the location of the features along the axes.

Parallel dimensions plots. The second visualization technique selected was the parallel dimensions plot (Ward et al., 2010; Telea, 2015; Ware, 2015). This technique was proposed by Inselberg in 1985 for studying geometries of higher dimensions. This visualization depicts the axes evenly spaced in parallel rather than orthogonal. A data point corresponds to a *polyline* that joints together the corresponding values of each dimension in the order of the axes.

In our plots, the parallel dimensions are structured as follows:

- *Solutions dimension*: the dimension (axis) depicted at the right of the visualization whose values are the names of the solutions (a.k.a configurations) that conform the covering array. Each value in this dimension has a name of the form `Si` and it is represented by a rectangle of a distinct color value whose height is proportional to the number of pairs or triplets covered by the corresponding solution.
- *Feature dimensions*: the two or three dimensions (axes) whose values are the names of the features (selected and not selected) that respectively constitute either the pairs or triplets of the covering array. Each value in these dimensions is visualized by a group of adjacent rectangles that represent the contribution of the feature to each solution. These rectangles are arranged in the same order as in the Solutions dimension. Their color matches the color of the solution they contribute to, and their height is proportional to the number of pair or triples they contribute to a particular solution.

In our parallel dimensions plots, each polyline:

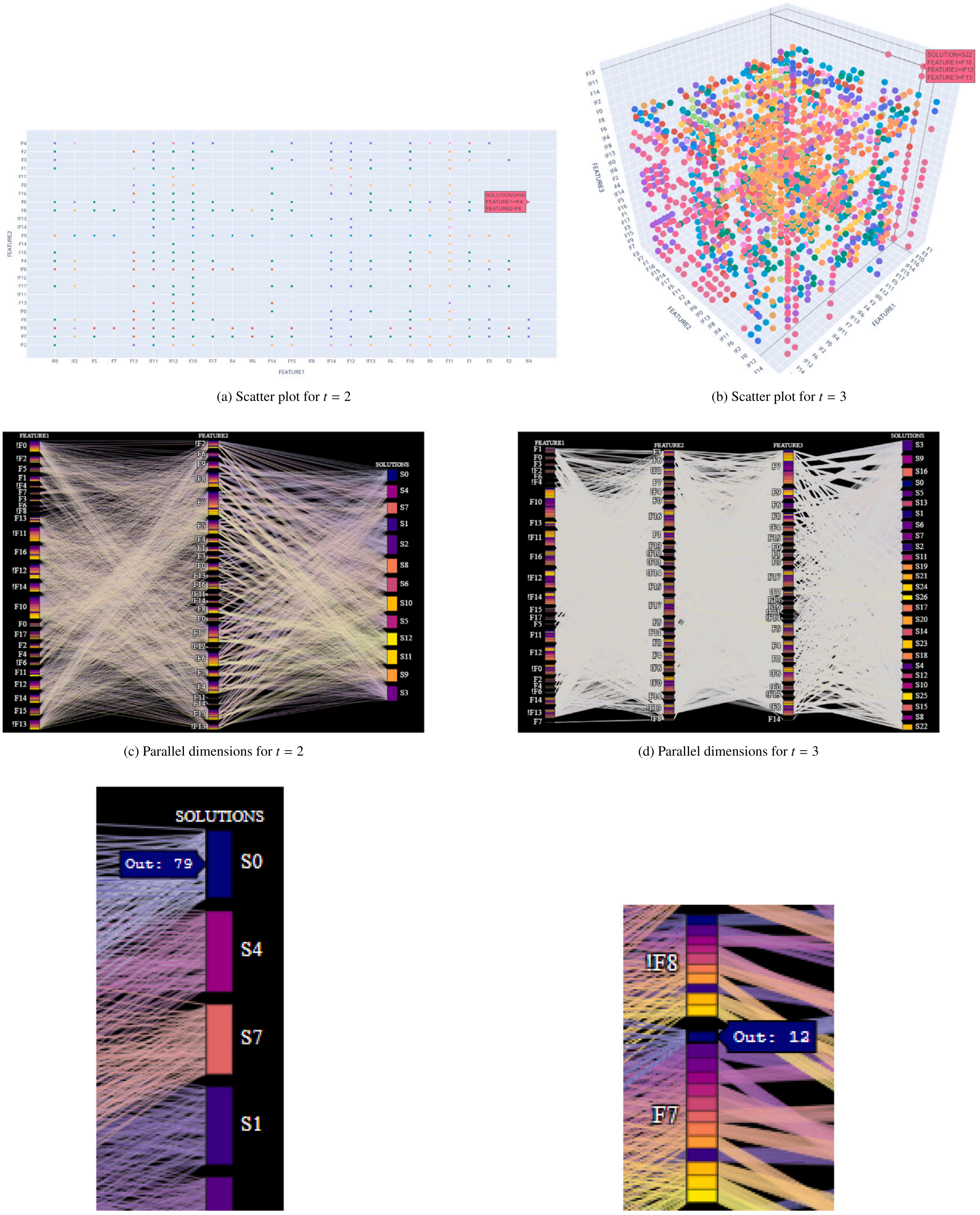
[3] Also known as scattered plots.

[4] Also known as parallel coordinates plots.

[5] Python version 3.10.4 and `Plotly` version 5.7.0, https://plotly.com/.

[6] In the case that a pair or element is covered by more than one solution, one color is assigned.

[7] The scatter plots visualization for $t = 3$ also contains a legend on the right side of the figure with a list of solutions (e.g. `S0`) and their corresponding color. This legend is not shown in the figure for simplicity.



(a) Scatter plot for $t = 2$

(b) Scatter plot for $t = 3$

(c) Parallel dimensions for $t = 2$

(d) Parallel dimensions for $t = 3$

(e) Detail of Solutions dimension

(f) Detail of a Feature dimension

Fig. 2. Illustration of visualization techniques for GPL running example.

- Represents a pair or a triplet covered by a solution in the covering array that is visualized.
- Intersects all the feature dimensions at the corresponding feature name value.
- Has its right extreme ending at the solution that covers the corresponding pair or triplet.
- Has the color of the solution that covers it.

Let us illustrate this visualization with our running example GPL for the case of pairs. Fig. 2c shows this visualization. From the left, the first and second axes are the feature dimensions of the pairs labeled respectively as `Feature1` and `Feature2`. The axis at the right shows the solutions dimension. In this case, the axis has 13 solutions and thus 13 distinct colors for depicting them. As an example, consider the polyline at the top of the visualization. This polyline connects the left dimension at feature `!F0`, the middle dimension at feature `!F2`, and the right dimension at solution `S0`. Thus, this polyline visualizes the pair `(!F0,!F2)` covered by solution `S0`. Notice that the polyline and all the intersecting rectangles share the same solution color. Similarly, Fig. 2d shows our visualization for the triplets of the GPL example. This figure has three feature dimensions and the solution dimension which, for this covering array, has 27 solutions.

Our parallel dimensions visualization has two types of interactions to help the navigation. The first navigation aid highlights the entire polyline while the user hovers over it. For the second aid, when the user hovers a rectangle in an axis, a small window pops up that shows the number of polylines that intersect the rectangle and highlights all those polylines across all the dimensions. For example, Fig. 2e shows 79 as the number of polylines in solution `S0` and highlights all those polylines. As another example, Fig. 2f shows feature `F7`, and its top rectangle intersects with 12 polylines and highlights them across all dimensions.

From the plot description, it can be inferred that the total number of polylines in a plot is the summation of the number of pairs or triples covered by each solution of the covering array depicted. For the GPL example, its 418 pairs are covered by 13 solutions amounting to 1269 polylines. For the case of triplets, the covering array has 27 solutions totaling 9643 polylines. Even though these numbers of polylines may appear to be large numbers, we should keep in mind that the navigation aids help to prune significantly the search space while the users perform the tasks.

For both visualization techniques, we have chosen color palettes provided by default in our implementation library Plotly. The palettes worked well for the number of colors in our case studies and they were well evaluated in our pilot study. Neither experimenting with different color palettes nor assessing accessibility issues, such as color-blindness, are part our of exploratory study.

4. Experimental design

In this section, we present the experimental design of our exploratory study. We followed the *Goal/Question/Metric (GQM)* method (Wohlin et al., 2012). Thus, we first introduce the main goal of our study, its research questions, and their corresponding metrics. Then, we present the study delimitations and describe the feature models selected for computing the covering arrays of our experiment. We close with a detailed description of the entire experiment design, including the definition of the areas of interest of visual attention, and the data processing workflow.

4.1. Goal, questions, and metrics

The goal of our research study is the following:

Goal: Compare the visualization techniques scatter plots and parallel dimension plots in terms of response accuracy, time-on-task, metacognitive monitoring, and visual attention during the execution of a test coverage task in Software Product Lines.

In Section 3 we have described how scatter plots and parallel dimensions plots work. Now, we present and illustrate the test coverage task of our study followed by the research questions and their corresponding metrics.

Task description. Our study focuses on the most basic task concerning test coverage of SPLs. This task has two purposes. First, it aims at asserting whether a pair or triplet is covered or not by a solution (i.e. configuration of a covering array) using one of the two visualization techniques considered for our study. Second, it collects self-assessment information to gather the perception of the participants.

Each task performed in our experiment consists of three parts:

1. A coverage question with the following form:
   Is this set `<SET>` covered by `<SOLUTION>`?
   where `<SET>` and `<SOLUTION>` respectively correspond to a pair or triplet and a named solution. The possible answers to each coverage question are either `Yes` or `No`. As an example, consider the task shown in Fig. 3 which uses the parallel dimensions visualization for the coverage of a pair. The corresponding question is: *Is this set `(!F8,F1)` covered by S3?*
2. A *Certainty Assessment (CA)* self-reported question based on the NASA-TLX index (Hart and Staveland, 1988): *How successful were you in accomplishing what you were asked to do?*. The values range from `Perfect` to `Failure`, which are selected by participants via a slider widget that interactively shows the chosen value on a Likert scale from 0 to 20.
3. A *Difficulty Assessment (DA)* self-reported question based on the NASA-TLX index (Hart and Staveland, 1988): *How mentally demanding was the task?*. The values range from `Very easy` to `Very difficult`, selected also via a slider that shows the value transformed to a Likert scale from 0 to 20.

Research questions and metrics. The research questions of our study are divided in the following categories: response accuracy, time-on-task, metacognition monitoring, and visual attention.

Research Question 1 (RQ1): Is there any effect of the number of elements of a covering array and the visualization technique on the accuracy of task responses?

Metric: Response Accuracy. This metric simply counts the number of accurate and inaccurate responses given by the participants to each coverage question of their tasks.

For a more detailed analysis, we break down RQ1 in two questions, one question for the effect of pairs and one question for effect of triplets.

Research Question 2 (RQ2): Is there any effect of the number of elements of a covering array and the visualization technique on the time-on-task?

Metric: Time-On-Task. This metric is the elapsed time that goes from the moment a coverage question is shown on the screen until the participant selects a response and clicks the space bar to continue. See Fig. 3 for an example.

Like before, we break down RQ2 in two questions, one question for the effect of pairs and one question for the effect of triplets.

Research Question 3 (RQ3): Is the response accuracy related to the metacognition monitoring metrics Certainty Assessment or Difficulty Assessment?

We want to explore the relationship between Certainty Assessment (CA) and Difficulty Assessment (DA) as described in Section 4.1, both for pairs and triplets.

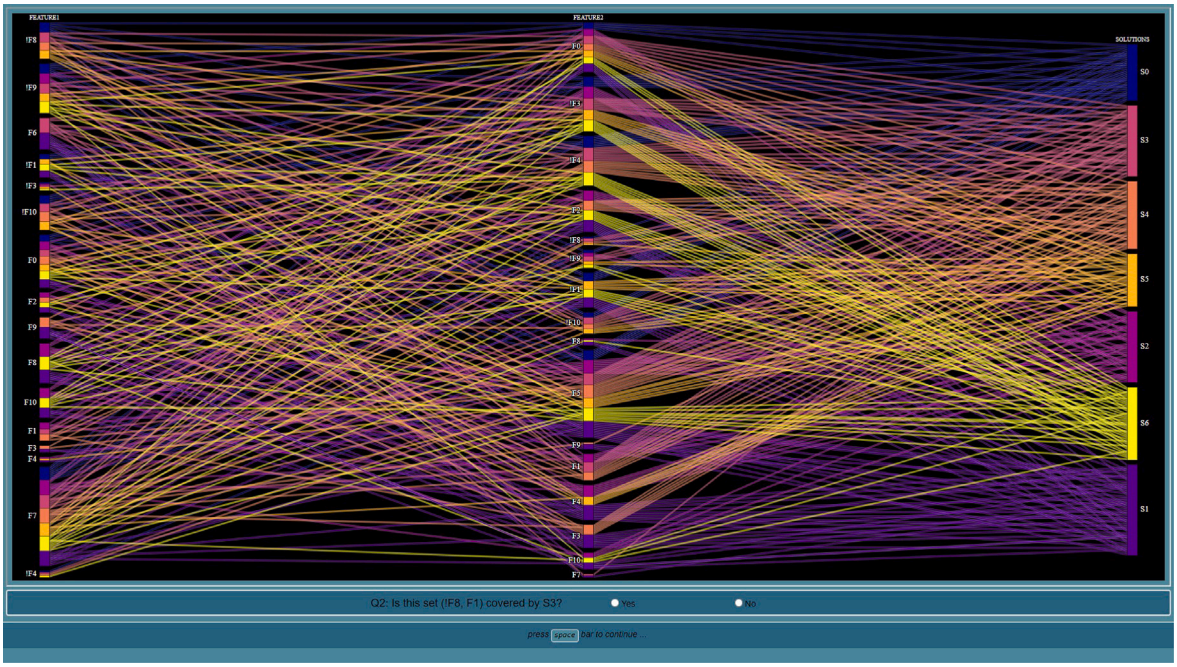
Research Question 4 (RQ4): Is there a relationship between the different AOIs of the visual stimuli in terms of the proportion of visual attention given to them while responding the question of the test coverage tasks?

Recall from Section 2.3 that fixations are movements that stabilize the retina on the visual stimulus and that the Areas Of Interest (AOIs) are regions of the visual stimulus deemed relevant for performing of a task. To answer RQ4, we consider the following two metrics:

Metric: Fixation time. The aggregated duration of all the fixations on a specific Area of Interest (AOI).

Metric: Fixation count. The number of fixations on a specific Area of Interest (AOI).

These two metrics are typically used to infer the mental effort of participants while performing a visual task (Zagermann et al., 2016). More concretely, low fixation time and low fixation counts indicate less effort in cognitive processing of visual information, while long fixation time and high fixation counts indicate more effort (Obaidellah et al., 2018). In Section 4.5, we present further details on the AOIs of our study. Like the previous questions, we broke down RQ4 into two sub-questions, namely for pairs and triplets.



Q2: Is this set (!F8, F1) covered by S3? ○ Yes ○ No

press space bar to continue ...

Fig. 3. Example of pair coverage question of parallel dimensions visualization.

4.2. Exploratory study delimitations

Every experiment has research aspects that are outside its scope. In our case, we explicitly state that two delimitations of our exploratory study are color accessibility and color perception issues. In other words, though they are important aspects of visualization techniques, they are not part of the independent variables considered in our experiment design.

In addition, outside the scope of our study is a comparison between using the visualization techniques versus using only the default textual information of the covering arrays. The large amounts of textual output, the limited eye-tracker distinguishability among text components, and the implied extension of the duration of the participants' experiment time were the reasons behind this decision. However, all the textual information of the covering arrays of all the case studies is available in our Dataverse repository (Lopez-Herrejon and Nezami Balouchi, 2024).

4.3. Selection of feature models and computation of their covering arrays

We based the selection of the feature models used in our study on the community-wide dataset collected in the work of Ferreira et al. (2021). This dataset contains 30 case studies of different domains and complexity that have been extensively used by the research community in the area of t-wise testing of SPLs. We adapted the SPLCAT tool (Software Product Line Covering Array Tool),[8] proposed by Johansen et al. (2012), to compute the pairs, triplets, and covering arrays of our exploratory study.

We generated the visualizations of the 30 case studies using the two visualization techniques of our study, for pairs and triplets.[9] We performed several tests in terms of legibility and selected ten case studies whose visualizations were suitable for the coverage tasks that are the focus of our study. The selected case studies are shown in Table 1. Their number of features ranges from 10 to 21, the number of pairs from 66 to 440, and the number of triplets from 144 to 3583.

Table 1
Selected feature models.

Case study	Features	Pairs	Triplets
ArgoUML-SPL	10	114	476
ATM	10	127	473
Companies	13	209	1143
Email	11	119	463
GPL Modified	21	387	2491
MinePump	11	136	565
Prop4J	17	440	3583
UnionFindSPL	10	66	144
VendingMachine	10	127	499
ZipMe	15	208	1029

The remaining 20 feature models from the original community-wide dataset were excluded from our study because their large number of pairs or triplets produced visualizations that were too crowded or not legible enough to perform the coverage task,[10] hence our selection of the experiment corpus was influenced by these aspects. Exploring alternative visualizations or more sophisticated forms of interaction and navigation for larger number of pairs and triplets is part of our future work.

4.4. Experiment design

We followed a standard approach for the selection of our experiment design (Wohlin et al., 2012; Lazar et al., 2017; Cunningham and Wallraven, 2019) and adhered to the guidelines proposed by Sharafi et al. for using eye-tracking technology in Software Engineering research (Sharafi et al., 2020).

We recruited 26 participants for our study, primarily graduate students from our institution who had no prior familiarity with feature models or combinatorial interaction testing. We randomly selected two

[8] https://martinfjohansen.com/splcatool/.

[9] All the generated visualizations are available in our Dataverse repository (Lopez-Herrejon and Nezami Balouchi, 2024).

[10] The visualizations of the excluded feature models had large numbers of visual components. This fact causes a size reduction of the components which renders them hard to read (e.g., the feature names), even in a large screen. This reduction also made the components hard to distinguish from each other when analyzing the gaze with the eye-tracker.

Table 2
Task visualization distribution, types of plots, type of elements to cover pairs/triples, number of colors, number of polylines.

Scatter plots tasks

Task no.	Case study	Pairs	Triplets	No. colors
Task 5	UnionFindSPL	✓		6
Task 6	VendingMachine	✓		7
Task 7	Prop4J	✓		11
Task 8	Companies	✓		8
Task 13	UnionFindSPL		✓	6
Task 14	Email		✓	37
Task 15	GPL modified		✓	31
Task 16	MinePump		✓	20

Parallel dimensions plots tasks

Task no.	Case study	Pairs	Triplets	No. colors	No. polylines
Task 1	ATM	✓		11	392
Task 2	Email	✓		7	292
Task 3	Prop4J	✓		11	1386
Task 4	MinePump	✓		7	309
Task 9	ArgoUML-SPL		✓	22	1896
Task 10	UnionFindSPL		✓	6	291
Task 11	GPL Modified		✓	31	20 259
Task 12	ZipMe		✓	15	4078

of these students to conduct a pilot study to help us refine the scope and duration of the experiment, and to fine tune the experiment conditions (e.g., lighting, screen and keyboard layout, etc.). Thus, our experiment gathered the data of 24 participants, i.e., $n = 24$.

For our experiment, we chose a *within-subjects*[11] design whereby each participant in the experiment is exposed to all experimental treatments (Lazar et al., 2017). This design is suitable for small samples and isolates the effects of individual differences. By contrast, within-subject designs may produce a learning effect (i.e. participants get better as they learn how to perform a task) and may result in fatigue (i.e. participants get tired or distracted of repeating the same task). We addressed these limitations in two ways. First, to prevent learning effect, we applied a unique and random order of treatments (i.e. tasks) for each participant. Second, informed by our pilot study, we limited the number of treatments to 16 tasks so that participants could finish the entire experiment within a reasonable time.

The structure of each task was already described in Section 4.1 and their visualizations were split into four groups: *(i)* four scatter plot visualizations of pairs, *(ii)* four scatter plot visualizations of triplets, *(iii)* four parallel dimensions plot visualizations of pairs, and *(iv)* four parallel dimensions plot visualizations of triplets. The questions for the tasks were defined as a mix of `Yes` and `No` answers, and a distribution of gaze scan paths across the different regions of the visualizations. Table 2 shows the distribution of the 16 tasks across: visualization techniques, tasks, case studies, pairs or triplets, number of colors, and number of polylines for parallel dimensions plots. The visualizations of the 16 tasks, their questions, and their images with the AOIs are available in our public Dataverse repository (Lopez-Herrejon and Nezami Balouchi, 2024).

Our study was conducted in a dedicated research room for user studies at our institution. During the experiment, participants were sitting alone in the room, while the experimenter monitored them from an adjacent observation room. This setup ensured a consistent and undisturbed environment, with controlled variables such as room lighting and participant positioning relative to the screen, keyboard, and eye-tracker. We used the Tobii Pro Fusion bar eye-tracker and Tobii Pro Lab tool,[12] two commonly used industrial hardware and software tools, for capturing and analyzing gaze data. Our setup consisted on a standard 27-inch screen, with wireless and ergonomic keyboard and mouse.

The session of each participant followed the next sequence of steps:

- Start with a brief introduction of the experiment by the experimenter, describing its objective and the procedure to follow.
- Sign the consent form approved by the ethics committee of our institution.
- Watch a training video (approximately 17 min) and clarify any questions or issues that the participants may have. The training video is also available in our Dataverse repository (Lopez-Herrejon and Nezami Balouchi, 2024).
- Set up of the experiment web interface with the corresponding sequence of tasks for the participant.
- Calibrate of the eye-tracker using Tobii Pro Lab tool, based on the adjusted position of the participant.
- Perform a warm-up practice with two tasks to gain familiarity with the graphical interface and recording the responses. No feedback is given to participants.
- Perform the 16 tasks, each participant in an unique and randomized order.
- Respond to a semi-structured interview recorded to gather further insights from the participant.

4.5. Definition of Areas of Interests (AOIs)

This section explains the Areas of Interest (AOIs) defined for our stimuli. We illustrate this process with the visualization of pairs in a parallel dimensions plot used in the coverage question shown in Fig. 3. To this figure we superimpose orange rectangles to indicate the AOIs as shown in Fig. 4.

All the stimuli in our tasks the AOIs are of the following types:

- Question AOI ①: shows the text of the question.
- Answer AOI ②: contains the radio buttons that permit the participant to answer `Yes` or `No` to the question.
- Axial AOIs ③: contain the axes in the visualization that the participants' gaze must traverse to find the answer to the question. The axes contain the feature names and the solution names. The width of these AOIs were defined based on gaze heat maps during the pilot study and the specifications of the eye-tracker we employed.
- Target AOIs ④: contain the feature names (selected and unselected) mentioned in the question. Target AOIs are located within their corresponding axial AOIs. Notice that nesting AOIs enables to compute the attention metrics of navigating the axes and of locating the feature names within the axes. In Fig. 4, the target AOIs contain feature `!F8` and feature `F1` respectively.
- Solution AOI ⑤: contains the solution name mentioned in the question. The solution AOI is located within the axial AOI that contains the solution names.
- Navigation AOI ⑥: contains the entire area of the visualization. We use this AOI to compute the metrics of navigating the visualization outside the axes (i.e. by subtracting the metrics of the Axial AOIs from the Navigation AOI metrics). In the case of parallel dimensions plots, the navigation metrics corresponds to the data that results from the user following and interacting with the polylines.
- Stimulus AOI ⑦: contains the entire visual stimulus. We use this AOI to compute the metrics of gazes outside all other AOIs and to perform several data consistency checks.

The other three visualizations have a similar scheme of AOIs. For triplets and parallel dimensions plots, the difference is the additional axis to contain the third element of the triplet. Thus, this visualization adds an extra Axial AOI and an extra Target AOI contained within it. For the scatter plots visualizations, the main difference is that

[11] Also known as within-participant of within-group designs.

[12] Further details in: https://www.tobii.com/products/eye-trackers/screen-based/tobii-pro-fusion and https://www.tobii.com/products/software/behavior-research-software/tobii-pro-lab.

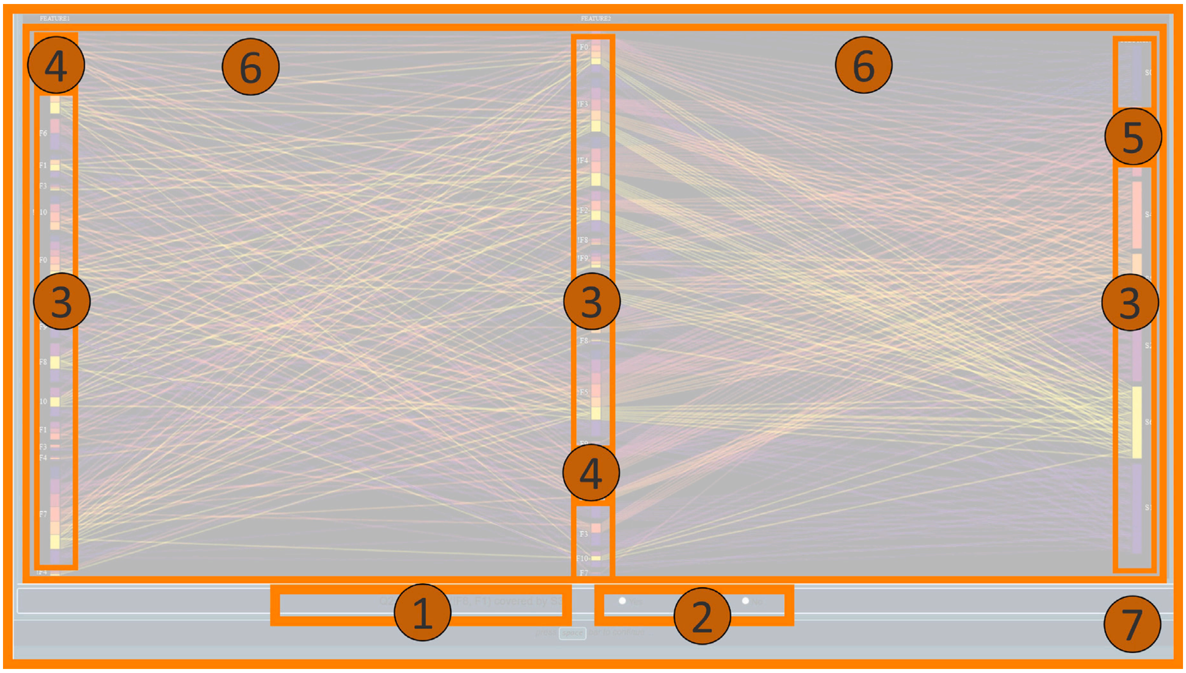


Fig. 4. Example of Areas of Interest (AOIs) for a coverage question in Fig. 3. AOIs: Question AOI ①, Answer AOI ②, Axial AOIs ③, Target AOIs ④, Solution AOI ⑤, Navigation AOI ⑥, Stimulus AOI ⑦.

there is no Axial AOI that contains the Solution AOI. In our Dataverse repository, we share pictures of the actual AOIs for all the questions in our study (Lopez-Herrejon and Nezami Balouchi, 2024). These pictures were exported using the TobiiPro Lab tool.

4.6. Data processing

This section summarizes the data processing workflow that we followed to collect the data of our experiment as illustrated in Fig. 5. We developed an R script to generate random numerical sequences for the tasks assignments of the participants. Each participant has his/her own unique sequence used as the *Experiment Configuration* ① for the set up of the *Experiment Task Manager* ②, a customizable HTML/Javascript application that handles the sequences of the experiment's tasks and the collection of their data. Each participant performs two warm up tasks followed by his/her unique sequence of tasks ③. Once a participant completes all the tasks, the *Experiment Task Manager* generates a *Performance Data* file ④ that contains all the tasks' responses and time-on-task data for the participant.

At the end of the session of the participant, the experimenter carries out an offline manual *Eye-Tracker Data Curation* process ⑤ with the aid of the Tobii Pro Lab tool. During this process, the eye-tracker data is segmented in 16 Excel files, i.e. the *Raw Eye-Tracker Data* ⑥, where each file contains the data of one task[13] We developed a series of R scripts, i.e. *Eye-Tracker Data Aggregation* ⑦, to process the vast amounts of eye-tracker data and aggregate the metrics required by our research questions in a single file per participant, i.e. *Eye-Tracker Data* ⑧. Lastly, the data of all the participants were aggregated via other R scripts, i.e. *Experiment Data Aggregation* ⑨, into a dataset for the entire experiment, i.e. *Complete Experiment Data* ⑩.

In Fig. 5, we add one of the following words in parenthesis to each element: *(a)* `open` when the data is publicly available in our Dataverse repository (Lopez-Herrejon and Nezami Balouchi, 2024), *(b)* `restricted` when the data cannot be openly shared according to the ethics guidelines of our institutions because they contain human subjects information, and *(c)* when there is a manual process tailored to the dataset that cannot be automated or replicated.

5. Results and analysis

In this section, we present the results and analysis for each of the research questions. Our analyses are divided in the specific descriptive and inferential statistics for each questions as well as the interpretation of its results. We conclude the section with a general summary of our findings.

5.1. Response accuracy results (RQ1)

This section presents the descriptive statistics and the analysis of the results of the Research Question 1 regarding accuracy of responses.

5.1.1. Descriptive statistics

In total, out of the 384 responses (24 participants × 16 tasks), 325 (84.64%) were accurate and 59 (15.36%) were inaccurate. Fig. 6 shows the distribution of all responses by participant and by question respectively, while Table 3 summarizes the accurate responses. In the participants' perspective, Fig. 6a, there were six participants that responded all the task questions accurately. Per participant, the average was 13.54 accurate responses with the minimum was eight responses. From the point of view of task questions, Fig. 6b, the average was 20.31 accurate responses, with the minimum of ten (Task 14) and the maximum of 24 accurate responses, in the case of four tasks (Task 4, Task7, Task 8, and Task 15).

Fig. 7 shows the distribution of accurate and inaccurate responses across the strength t of the covering array and the visualization technique. For $t = 2$ and scatter plots, participants got the highest number (94) of accurate responses.[14] In contrast, for $t = 3$ and also scatter plots, participants got the lowest (70) number of accurate responses. The results for parallel dimensions plots were very similar, with 81 and 80 accurate responses respectively for $t = 2$ and $t = 3$.

[13] The results of the warm up questions are neither stored nor analyzed.

[14] Recall that there were 24 participants and 4 tasks per participant for each combination of t value and visualization technique. Thus the maximum value is 96.

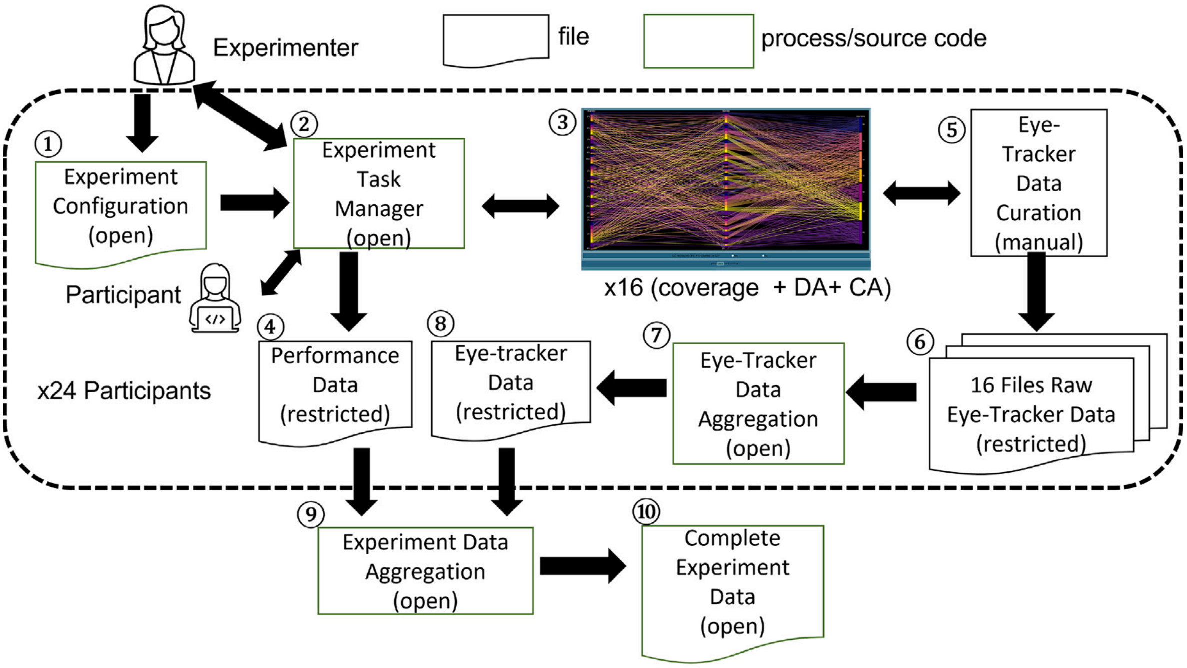
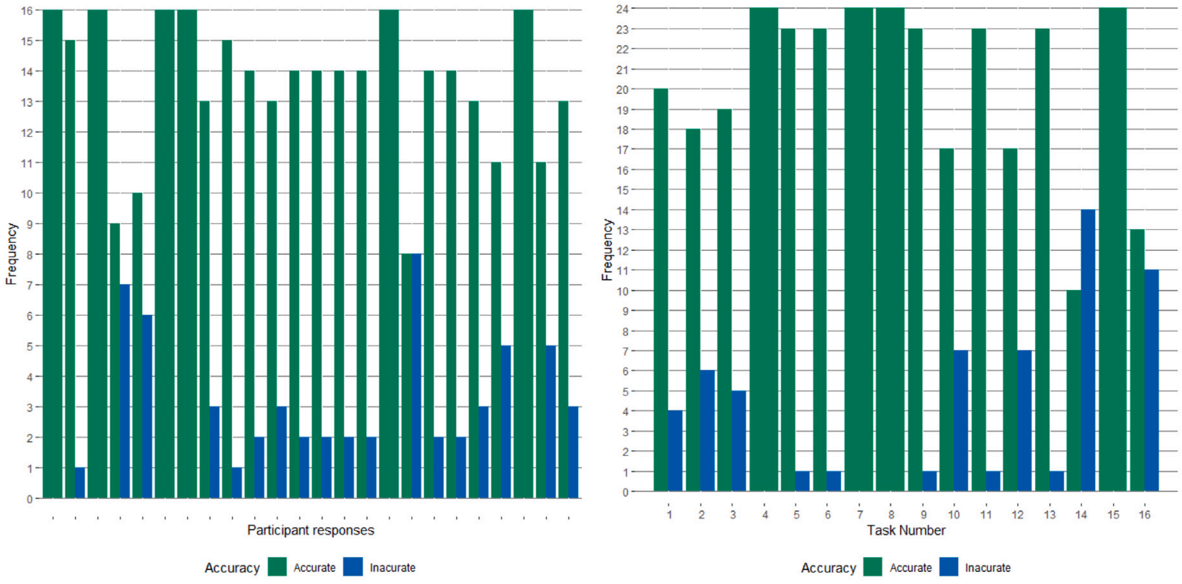

Experimenter
file
process/source code
1
Experiment Configuration (open)
2
Experiment Task Manager (open)
3
x16 (coverage + DA+ CA)
5
Eye-Tracker Data Curation (manual)
Participant
x24 Participants
4
Performance Data (restricted)
8
Eye-tracker Data (restricted)
7
Eye-Tracker Data Aggregation (open)
6
16 Files Raw Eye-Tracker Data (restricted)
9
Experiment Data Aggregation (open)
10
Complete Experiment Data (open)

Fig. 5. Data processing workflow, `open` — publicly available data, `restricted` — private data, `manual` — manual process for curating eye-tracker data.



(a) Response accuracy per participant

(b) Response accuracy per task question

Fig. 6. Response accuracy results per participant and per task question.

Fig. 8 breaks down the accuracy results by showing the number of pairs or triplets for each combination of t value and visualization technique. For $t = 2$ and scatter plots, response accuracy varies slightly as the number of pair increases. Interestingly, here the questions with perfect accurate scores are those with higher number of pairs, followed very closely with 23 accurate responses those that have lower number of pairs. For $t = 2$ and parallel dimensions plots, response accuracy varies regardless of the number of pairs. For $t = 3$, response accuracy varies regardless of the number of triplets in both visualization techniques.

Table 3 also summarizes the response accuracy per participant by covering array strength and visualization technique . Please recall that each participant performed 8 tasks per value of t and 8 tasks per visualization technique. For $t = 2$, participants got between 4 and 8 accurate responses with an average of 7.29. In contrast, for $t = 3$, the minimum value was 3 accurate responses, with maximum value 8 and average value 6.25. In terms of visualization technique, for scattered plots participants obtained between 5 and 8 accurate responses with an average of 6.83. In contrast, for parallel dimensions plots, the range was between 3 and 8 accurate responses, with average of 6.70.

5.1.2. Analysis

Research question RQ1 was answered using multilevel modeling with participants as nesting variable and tasks as repeated measures (Raudenbush and Bryk, 2001). Multilevel modeling has to be used because the assumption of independence of observations is violated

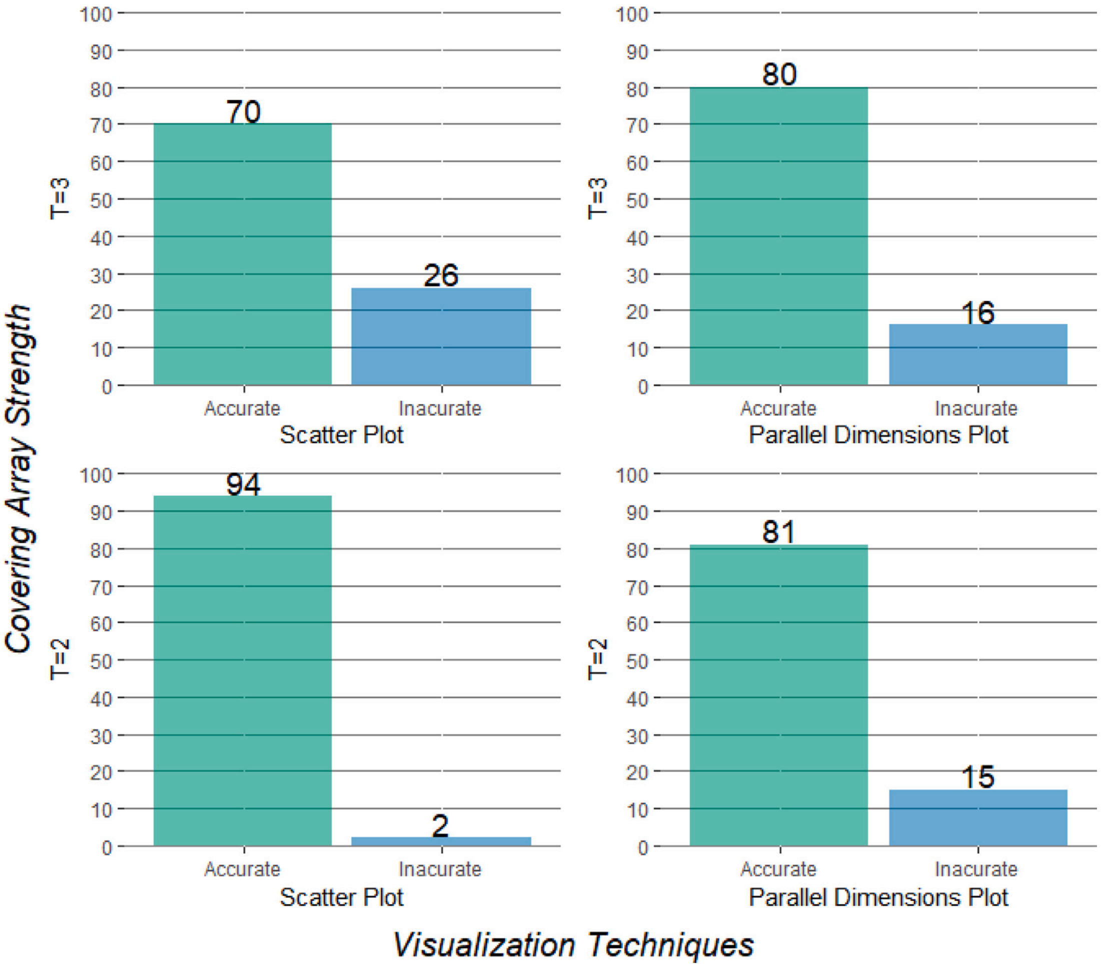


Fig. 7. Response accuracy results summary.

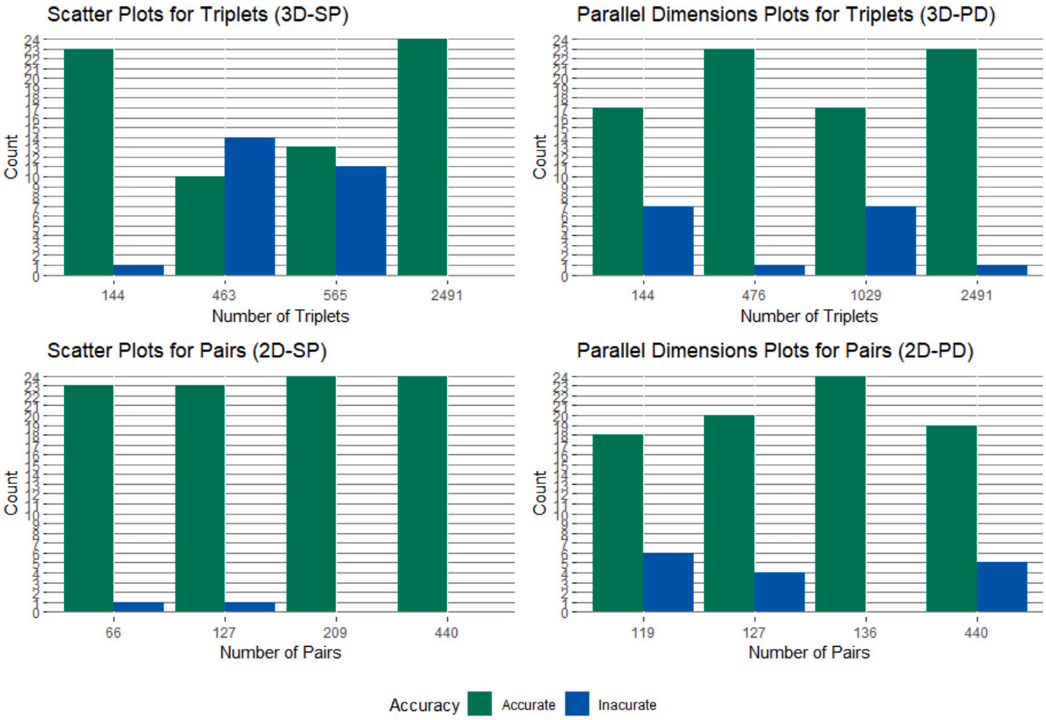


Fig. 8. Accuracy results per number of pairs and triplets with visualization technique.

Table 3
Accurate responses summary.

Per participant				
Criteria	Mean	Min	Max	Std Dev
All 16 tasks	13.54	8	16	2.28
Array strength				
$t = 2$	7.29	4	8	1.08
$t = 3$	6.25	3	8	1.48
Visualization technique				
Scatter plots SP	6.83	5	8	1.09
Parallel dimensions plots PD	6.70	3	8	1.48
Per task				
All participants	20.31	10	24	4.31

Table 4
Time-on-task summary with time in seconds.

Criteria	Mean seconds	Min seconds	Max seconds	Std Dev seconds
All 16 tasks	1047.50	367.50	1497.60	304.41
Accuracy				
Accurate responses	64.30	4.25	465.55	51.03
Inaccurate responses	71.86	16.21	161.98	37.72
Array strength				
$t = 2$	49.38	7.98	208.21	31.46
$t = 3$	81.54	4.25	465.55	57.91
Visualization technique				
Scatter plots SP	69.84	4.48	465.55	59.74
Parallel dimensions plots PD	61.08	4.25	199.13	35.45

because participants contribute observations to a series of questions that are analyzed separately. We performed our analysis with the SPSS Generalized Mixed Linear Models procedure. With this approach, the following two assumptions must be met[15]: *(i)* normality (i.e. that the error terms at every level of the model are normally distributed), and *(ii)* homoscedasticity (i.e. variances of dependent variables are equal between groups—levels of independent variables).

For Questions RQ1a and RQ1b, a logit function was used to predict the binary outcome variable (accurate-inaccurate response) from two factors: visualization technique (categorical), and number of pairs or triplets (ordinal).

Question RQ1a. Effect of number of pairs and visualization technique on response accuracy. There is an effect of visualization technique ($F_{1184} = 48.24$, $p < .001$), and number of element pairs ($F_{5184} = 67.53$, $p < .001$) on response accuracy. The scatter plots visualization technique (2D-SP) produces more accurate responses compared to parallel dimensions plots (2D-PD), 54% and 46% respectively. The interaction effect of visualization technique and number of elements is also significant ($F_{1184} = 12.60$, $p < .001$). The number of accurate responses varies across the number of pairs, and the gain of 2D-SP over 2D-PD is more evident in the lower number of pairs.

Question RQ1b. Effect of number of triplets and visualization technique on response accuracy. There is an effect of visualization technique ($F_{1184} = 14.08$, $p < .001$), and of number of triplets ($F_{5184} = 27.30$, $p < .001$) on response accuracy. The better performance with parallel dimensions plots (3D-PD) compared to scatter plots (3D-SP) is evident across numbers of triplets, and its effect on response accuracy does not show any clear pattern. The interaction effect of visualization technique and number of triplets is borderline ($p < .062$) so it is not further analyzed. Of the accurate responses, 53% were with parallel dimensions plots (3D-PD), while 47% for scatter plots (3D-SP). The number of accurate responses varies across number of triplets, but in no clear pattern.

Discussion for RQ1. The visualization technique of scatter plots seems to improve performance for pairs (2D-SP), while the parallel dimensions plots improves performance for triplets (3D-PD). Considering that the number of pairs and triplets to visualize is similar in both techniques, this finding suggests that the answer accuracy difference lies at how the visualization mental models (see Section 2.4) are built and accessed for each visualization technique. For pairs, there is a clear advantage of 2D-SP for which response accuracy does not appear to be affected as the number of pair increases. In contrast, 2D-PD has more varied response accuracy. This fact suggest that navigating the additional visual elements present in parallel dimensions plots such as lines may cause more cognitive load in detriment of response accuracy. For triplets, the situation seems to be reversed where the additional visual clues provided by 3D-PD seem to have a positive impact on response accuracy.

5.2. Time-on-task results (RQ2)

This section presents the descriptive statistics and the analysis of the results of the Research Question 2 regarding time-on-tasks.

5.2.1. Descriptive statistics

Please recall from Section 4.1 that the time-on-task is the elapsed time from the moment when a task is displayed on the web interface to the submission of the response by clicking the space bar to move on to the next task. In contrast, the time captured by the eye-tracker is analyzed in Section 5.4.

Table 4 summarizes the results of time-on-task. Participants took between 367.5 seconds (6.12 min) and 1497.60 seconds (24.96 min) to respond all the 16 questions. The average per participant was 1047.50 seconds (17.45 min) with a standard deviation of 304.41 seconds (5.07 min). In terms of response accuracy, the accurate responses showed a large range of time-on-task going from 4.25 seconds to 465.55 seconds (7.75 min) with an average of 64.30 seconds (1.07 min) and standard deviation of 51.03 seconds. In contrast, the range of inaccurate responses was from 16.21 seconds to 161.98 seconds (2.69 min) with an average of 71.86 seconds (1.97 min) and a standard deviation of 37.72 seconds.

The distribution of time-on-task over covering array strength t was as follows. For $t = 2$, the range was from 7.98 seconds to 208.21 seconds (3.47 min) with an average of 49.38 seconds, and standard deviation of 31.46 seconds. For $t = 3$, the range went from 4.25 seconds to 465.55 seconds (7.75 min) with an average of 81.54 seconds (1.35 min), and standard deviation of 57.91 seconds.

In terms of visualization techniques, the responses for scatter plots took between 4.48 seconds and 465.55 seconds (7.75 min) with an average of 69.84 seconds (1.16 min), and standard deviation of 59.74 seconds. In the case of parallel dimensions plots, responses took between 4.25 seconds and 199.13 seconds (3.31 min) with an average of 61.08 seconds (1.01 min), and standard deviation of 35.45 seconds.

Fig. 9 summarizes the time-on-task across covering array strength t and visualization techniques. For $t = 2$ and scatter plots, the values of the accurate responses ranged from 7.98 seconds to 208.21 seconds (3.47 min), with an average of 41.83 seconds. In contrast, there were only two observations of inaccurate responses, 81.85 seconds (1.35 min) and 87.39 seconds (1.45 min), averaging 84.62 seconds (1.41 min).

For $t = 2$ and parallel dimensions plots, the accurate responses ranged from 15.77 seconds to 156.55 seconds (2.60 min) with an average of 54.22 seconds, whereas for inaccurate responses the range

[15] There are two other assumptions, linearity and no perfect multicollinearity, that also must be met for this approach but that are not applicable in our case.

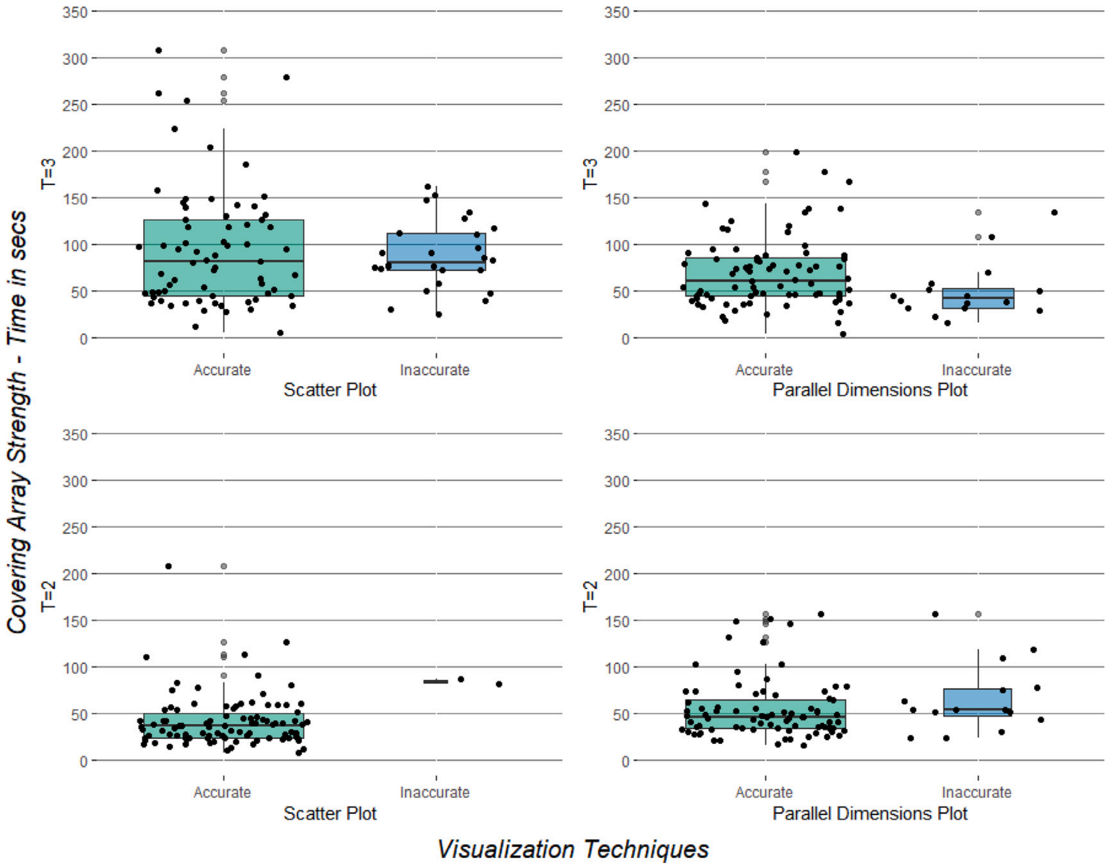


Fig. 9. Time-on-task results summary.

was from 23.24 seconds to 156.70 seconds (2.61 min) with an average of 65.85 seconds (1.09 min).

For $t = 3$ and scatter plots, the accurate responses had values from 4.48 seconds to 465.55 seconds (7.75 min),[16] with an average 100.41 seconds (1.67 min). For inaccurate responses, the values were from 24.86 seconds to 161.98 seconds (2.69 min) with an average of 87.67 seconds (1.46 min).

For $t = 3$ and parallel dimensions plots, the time of accurate responses went from 4.35 seconds to 199.13 seconds (3.31 min), with an average of 69.30 seconds (1.15 min). For inaccurate responses, the values went from 16.20 seconds to 133.98 seconds (2.23 min), with an average of 50.19 seconds.

Fig. 10 breaks down the time-on-task results by the covering array strength t, visualization technique, response accuracy, and number of pairs or triplets. For $t = 2$ and scatter plots, the accurate responses are mostly below the 60 seconds threshold. The two inaccurate responses had 66 pairs and 127 pairs, with time-on-task of 81.85 seconds (1.36 min) and 87.39 seconds (1.45 min) respectively. For $t = 2$ and parallel dimensions plots, the ranges of the boxplots overlap along response accuracy, except for the case of 136 pairs where there were no inaccurate responses.

For $t = 3$ and scatter plots, only one inaccurate response appears at the case of 144 triplets and its time is at in the middle of the range of the accurate responses. For the cases of 463 and 565 triplets, the medians of the inaccurate responses are higher than those of the accurate responses. For the largest case, 2491 triplets, there were no inaccurate responses, and the median time of the accurate responses for this case was higher than any other of the cases. For $t = 3$ and parallel dimensions plots, the cases with 144 and 1029 triplets present each seven inaccurate responses, and their times are within the ranges of the accurate responses for those cases. The cases with 476 triplets and 2491 triplets have each a single inaccurate response that fall within the time range of their accurate response counterpart.

5.2.2. Analysis

Question RQ2 was also answered using multilevel modeling with participants and questions as nesting and repeated measures variables, with SPSS Generalized Mixed Linear Models procedure (Raudenbush and Bryk, 2001). For research questions RQ2a and RQ2b, a linear function was used to predict the continuous outcome variable, time-on-task, from the same 2 factors: visualization technique (categorical), number of pairs or triplets (ordinal). Within the rationale of mental chronometry (Posner, 1993), the analysis was conducted using only the accurate responses.

Question RQ2a. Effect of number of pairs and visualization technique on time-on-task. There is an effect of number of elements ($F_{5184} = 7.43, p < .001$) on time-on-task. The increasing number of elements produces a longer time-on-task, but this tendency is not distributed evenly across the number of elements. The effect of visualization technique is also significant ($F_{1184} = 7.04, p < .001$). The scatter plots (2D-SP) produce shorter time-on-task values. Finally, the effect of the interaction of visualization technique and number of elements is not significant ($p > .05$).

Question RQ2b. Effect of number of triplets and visualization technique on time-on-task. There is an effect of visualization method ($F_{1184} = 11.97, p < .001$), and number of elements ($F_{5184} = 4.10, p < .001$) on time-on-task. The effect of the interaction of visualization technique and number of elements is also significant ($F_{1184} = 5.42, p < .021$). The parallel dimensions plots (3D-PD) produce shorter response times

[16] This value is an outlier which is not shown in the figure to visually facilitate the comparison with same time scale of the other three combinations.

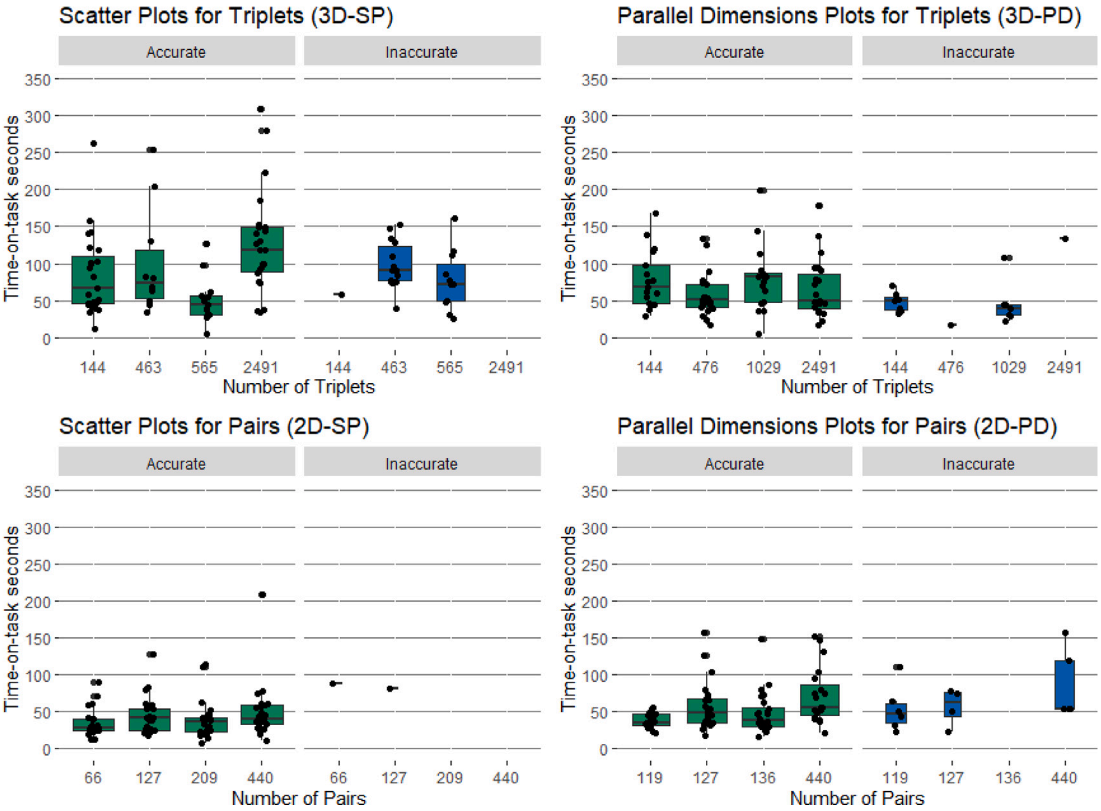


Fig. 10. Time-on-task results per number of pairs or triplets, response accuracy, and visualization technique.

compared with the scattered plots (3D-SP), 66.1 seconds (1.10 min) and 97.0 seconds (1.61 min) respectively. The increasing number of elements produces a somewhat U-shaped curve of time-on-task values.

Discussion for RQ2. In the case of covering arrays of strength $t = 2$, time-on-task relates to the number of pairs considered. This is expected because having larger number of pairs to visualize and navigate through increases the complexity of the mental models built by the participants for answering the tasks. The fact that the visualization techniques of pairs do not affect the time-on-task suggests that both techniques seem equally interpretable by the participants, but it should be noted from RQ1a that the success rate is better with parallel dimensions plots (2D-PD).

In the case of covering arrays of strength $t = 3$, despite comparable success rates in terms of accurate answers (see RQ2b), the scatter plots (3D-SP) induce longer time-on-task values than parallel dimensions plots (3D-PD), indicating an additional difficulty in the construction of the mental models required for the tasks, for which participants compensate by increased cognitive efforts. Parallel dimensions plots (3D-PD) seems the visualization technique of choice, both in terms of accuracy and time-on-task.

5.3. Metacognitive monitoring results (RQ3)

This section presents the descriptive statistics and the analysis of the results of the Research Question 3 regarding the metacognitive monitoring results.

5.3.1. Descriptive statistics

Certainty assessment. Fig. 11a summarizes the data obtained for the certainty self-assessment. Recall that for this aspect, the participants were asked to rank how successful they think they were about their task performance. The values range from 0 = Perfect (completely sure) to 20 = Failure (completely unsure). The most frequent value was 2 with 111 responses, the second most frequent value was 4 with 57 responses, and the third most frequent value was 3 with 56 responses. At the other extreme for value 20, only 7 responses indicated that participants were completely unsure about their performance. The fact that the median was value 3 and that the third quartile was value 5, indicates that, overall, most of the participants felt highly confident about their performance.

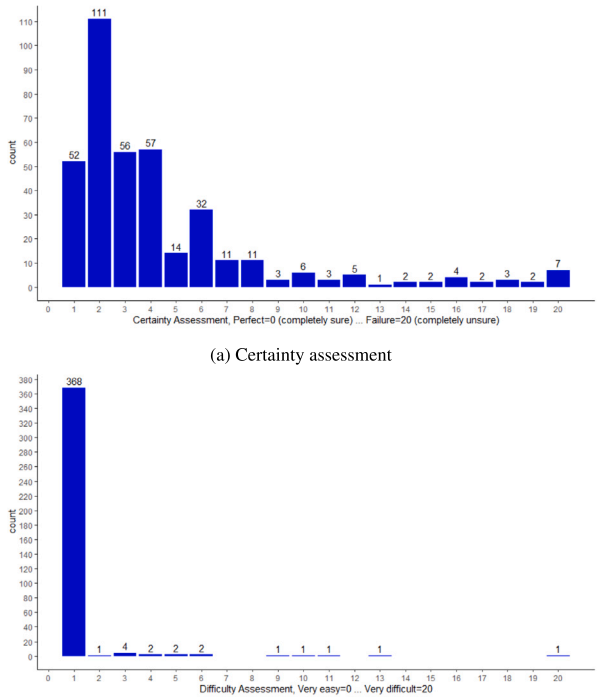
Difficulty assessment. Fig. 11b summarizes the data obtained for the difficulty self-assessment. Recall that participants were also asked to evaluate how difficult the questions were. The values for difficulty range from 0 = very easy to 20 = very difficult. The great majority of responses, 368 out of 384 (95.83%), were ranked with value 1. The rest of self-evaluation was divided as follows: *(i)* four responses for value 3; *(ii)* two responses each for values 4, 5, and 6; and *(iii)* one response for values 2, 9, 10, 11, 13, and 20. Thus, in summary, the majority of participants considered that the majority of tasks were very easy to perform.

5.3.2. Analysis

In this section, we analyze the relationship between Certainty Assessment and response accuracy for pairs and triplets, and if this relationship is modulated by the number of pairs or triplets and the visualization techniques.

A linear mixed model was fitted to the data, with the Certainty Assessment as the dependent variable and the number of pairs/triplets, the visualization technique and the response accuracy as independent variables.

We answered this research question using multilevel modeling with participants as nesting variable and questions as repeated measures. A linear mixed model was fitted to the data, with the Certainty Assessment as the dependent variable and the number of pairs/triplets, the visualization technique, and the response accuracy as independent variables. It should be noted that if a relation is found for each case



(a) Certainty assessment

(b) Difficulty level

Fig. 11. Metacognitive monitoring summary.

(pairs or triplets), i.e. the null hypothesis is rejected, we further analyze whether the number of pairs or triplets and the visualization techniques modulate this relation.

Question RQ3a. Relation between response accuracy and certainty assessment for pairs. There is a relation between certainty assessment and response accuracy ($F_{1179} = 5.11, p < .025$). The participants were more certain of their response when their response was indeed accurate: the metacognitive judgements of certainty are higher for accurate responses (mean value 3.36) than for inaccurate responses (mean value 3.89). It must be noted that the variance of metacognitive judgements of certainty is much greater in the case of inaccurate responses compared to accurate responses. This finding denotes some awareness that there is some probability that the response is inaccurate. However, as we recorded the certainty assessment after the end of each task, it is not possible to assess whether the level of certainty changed while the participants performed the tasks.

Analysis of modulation by visualization technique or number of pairs of the relationship between response accuracy and Certainty Assessment. This relation between certainty assessment and response accuracy is modulated by number of pairs ($F_{3179} = 3.28, p < .022$) and by visualization technique ($F_{1179} = 5.32, p < .022$). Regarding the number of pairs, an increase causes an almost steady decrease in certainty. Our interpretation of this is that the perception of task difficulty drives certainty down. 2D-PD visualizations help participants predict the correctness of their responses; they were more certain when responses are indeed accurate. By contrast, mean levels of certainty are the same whether the response is accurate or not in the case of 2D-SP visualizations. This means that participants cannot predict the likelihood of an inaccurate response when the task is supported by 2D-SP visualizations.

Question RQ3b. Relation between response accuracy and certainty assessment for triplets. Our analysis found no relation between certainty assessment and response accuracy when the task involves triplets of features ($p > .05$). Consequently, we did not do any further analysis (i.e. research sub-question) as we did for pairs.

Table 5
Fixation time and Fixation count summaries.

Fixation time

Array strength	Visualization technique	Mean seconds	Min seconds	Max seconds	Std Dev seconds
$t = 2$	Scatter plot SP	34.15	8.37	171.44	23.11
	Parallel dimensions plot PD	45.09	13.02	133.77	25.93
$t = 3$	Scatter plot SP	79.88	8.92	408.14	61.83
	Parallel dimensions plot PD	53.50	3.65	169.55	30.60

Fixation count

Array strength	Visualization technique	Mean	Min	Max	Std Dev
$t = 2$	Scatter plot SP	131.62	24	596	85.58
	Parallel dimensions plot PD	142.80	48	459	76.43
$t = 3$	Scatter plot SP	266.34	31	987	172.83
	Parallel dimensions plot PD	174.13	9	535	97.68

Discussion for RQ3. For pairs, accurate responses are associated with higher certainty, which is indicative of an adequation between the judgement of the task outcome and the actual performance. The augmentation of number of pairs makes the metacognitive assessment of the task more difficult. Overall, tasks involving pairs yield higher metacognitive monitoring accuracy, in the sense that participants are metacognitively aware that they provided an accurate response or not, and that aspects of the difficulty of the tasks such as the number of pairs modulate their certainty accordingly. For triplets, the fact that metacognitive judgements appear unrelated to response accuracy — and the fact that it was for pairs — indicates that participants may not be able to extract and record the necessary information to judge their performance because the task is too cognitively demanding, or that they are subject to either or both the "illusion of not knowing" and the "illusion of knowing". Overall, tasks involving triplets yield lower metacognitive monitoring accuracy, to the point which the response accuracy was not even statistically related to metacognitive judgements.

5.4. *Visual attention results (RQ4)*

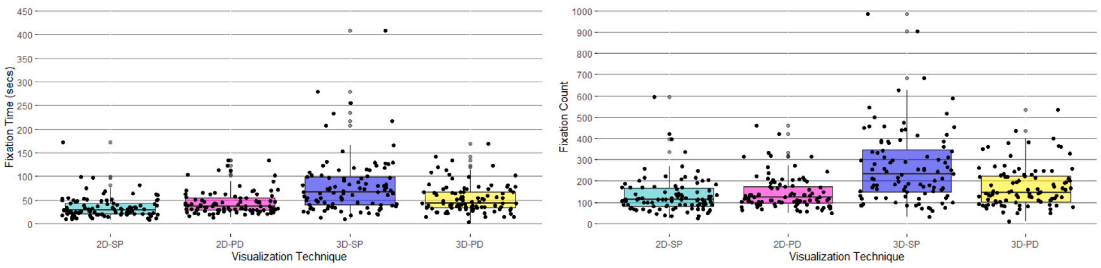
This section presents the descriptive statistics and the analysis of the results of the Research Question 4 regarding the visual attention information captured with the eye-tracker.

5.4.1. *Descriptive statistics*

Table 5 summarizes the descriptive statistics of fixation time and fixation count across covering array strength t and visualization techniques. Fig. 12 complements this information by depicting the corresponding boxplots across both factors.

Analyzing fixation time, for $t = 2$, parallel dimensions plots have median (36.73 seconds) and mean (45.09 seconds) higher than scatter plots (28.92 seconds and 34.15 seconds respectively) although with a narrower value range. Thus, for this combination, participants had on average more fixation time for tasks using parallel dimensions plots than for tasks using scatter plots. Interestingly, this observation is reversed when $t = 3$. Scatter plots exhibit a fixation time with a median (66.87 seconds, 1.11 min) and a mean (79.88 seconds, 1.33 min) significantly higher than for parallel dimensions plot (43.14 seconds and 53.50 seconds respectively).

Concerning fixation count, for $t = 2$, parallel dimensions plots have median 122 and mean 142.80 which is again higher than scatter plots (113 and 131.62 respectively), again in a slightly narrower range. Thus, similarly to fixation time, in this combination participants had on



(a) Fixation time summary

(b) Fixation count summary

Fig. 12. Fixation time and fixation count boxplots where 2D ($t = 2$), 3D ($t = 3$), Scatter Plot (SP), and Parallel Dimensions (PD).

average a larger number of fixations for tasks using parallel dimensions plots than for tasks using scatter plots. Like before, this observation is reversed for $t = 3$ where scatter plots have median (232) and mean (266.34) which is significantly higher than for parallel dimensions plots (146.50 and 174.13 respectively).

Let us start with the proportion of fixation time (see Fig. 13a) for $t = 2$. In scatter plots, the three leading AOIs are: *(1)* `Navigation` with 0.24, *(2)* `Target` with 0.22, and *(3)* `Axial` with 0.21. In contrast, for parallel dimensions plots, there is a higher difference among the three leading AOIs: *(1)* `Axial` with 0.36, *(2)* `Navigation` with 0.19, and *(3)* `Question` with 0.15 but followed shortly by `Target` with 0.13. In summary, participants spent on average more proportion of fixation time traversing the axes in parallel dimensions plots than in scatter plots. This proportion difference can be explained as a result of having more information in the axes (i.e. solutions and their polylines) than just feature names. This increase came with a trade-off reduction in the proportions of `Navigation` and `Target`. In other words, the time spent exploring the solutions in the axes of parallel dimensions plots reduced the time navigating the visualization and the time finding the accurate solution.

We compare now proportion of fixation time for $t = 3$. In scatter plots, the three leading AOIs are: *(1)* `Navigation` with 0.29, *(2)* `Target` with 0.22, and *(3)* `Axial` with 0.17. In contrast, for parallel dimensions plots, there is again a higher difference among the three leading AOIs: *(1)* `Axial` with 0.35, *(2)* `Target` with 0.16, and *(3)* `Navigation` also with 0.16 and followed shortly by `Target` with 0.15. These differences also confirm the trade-off identified in $t = 2$, namely, that participants spend a higher proportion of fixation time exploring the axes of parallel dimensions plots which in turn reduces the time to navigate the visualization and locate the accurate solution.

Our focus now is proportion of fixation count for $t = 2$, see Fig. 13b. For scatter plots, the three leading AOIs are: *(1)* `Navigation` with 0.26, *(2)* `Axial` with 0.24, and *(3)* `Target` with 0.20. For parallel dimensions plots, the leading AOIs are: *(1)* `Axial` with 0.36, *(2)* `Navigation` with 0.24, and *(3)* `Question` with 0.18. These findings show that the same three AOIs have the highest proportion of both fixation time and fixation count for $t = 2$ across both visualization techniques.

Finally, we proceed to the analysis of proportion of fixation count for $t = 3$. For scatter plots, the three leading AOIs are: *(1)* `Navigation` with 0.28, *(2)* `Target` with 0.20, and *(3)* `Axial` with 0.19. For parallel dimensions plot, the leading AOIs are: *(1)* `Axial` with 0.35, *(2)* `Navigation` with 0.21, and *(3)* `Question` with 0.19. These findings show that for scatter plots, the leading AOIs are the same for both proportion of fixation count and proportion of fixation time. However, for the case of parallel dimensions plots, `Axial` and `Navigation` are two of the leading AOIs and `Question` takes the third place instance of `Target`. This suggests that participants look at the question more times but for slightly shorter periods.

5.4.2. Analysis

The fourth research question RQ4 explores the relationships between the stimuli components in terms of visual attention measured with fixation count and fixation time. The seven AOIs we considered are: `Question`, `Answer`, `Axial`, `Target`, `Solution`, `Navigation` and `Stimulus`. We use correlation matrices to analyze the relationships between the AOIs, again divided by pairs and triplets.

Figs. 14 and 15 depict, for pairs and triplets respectively, the correlations between the AOIs using Spearman's rho rank correlation as all the variable distributions were found to be non-normal. The circle contains their ρ value, while their radius and color shade respectively represents the strength and the sign of the correlation.

Question RQ4a. Is there a relation between the visual processing of the AOIs of the visual stimuli for pairs ($t = 2$)*?* As can be seen in Fig. 14a, the majority of correlations have negative values, meaning that paying more attention to certain AOIs is related to looking less at other ones. In particular, in terms of moderate correlations, more time spent looking at the `Target` is associated with less time spent looking at `Axial` ($\rho = -0.50, p < .001$). Also, results show that the more time spent looking at `Question`, the less time spent in `Navigation` ($\rho = -0.43, p < .001$).

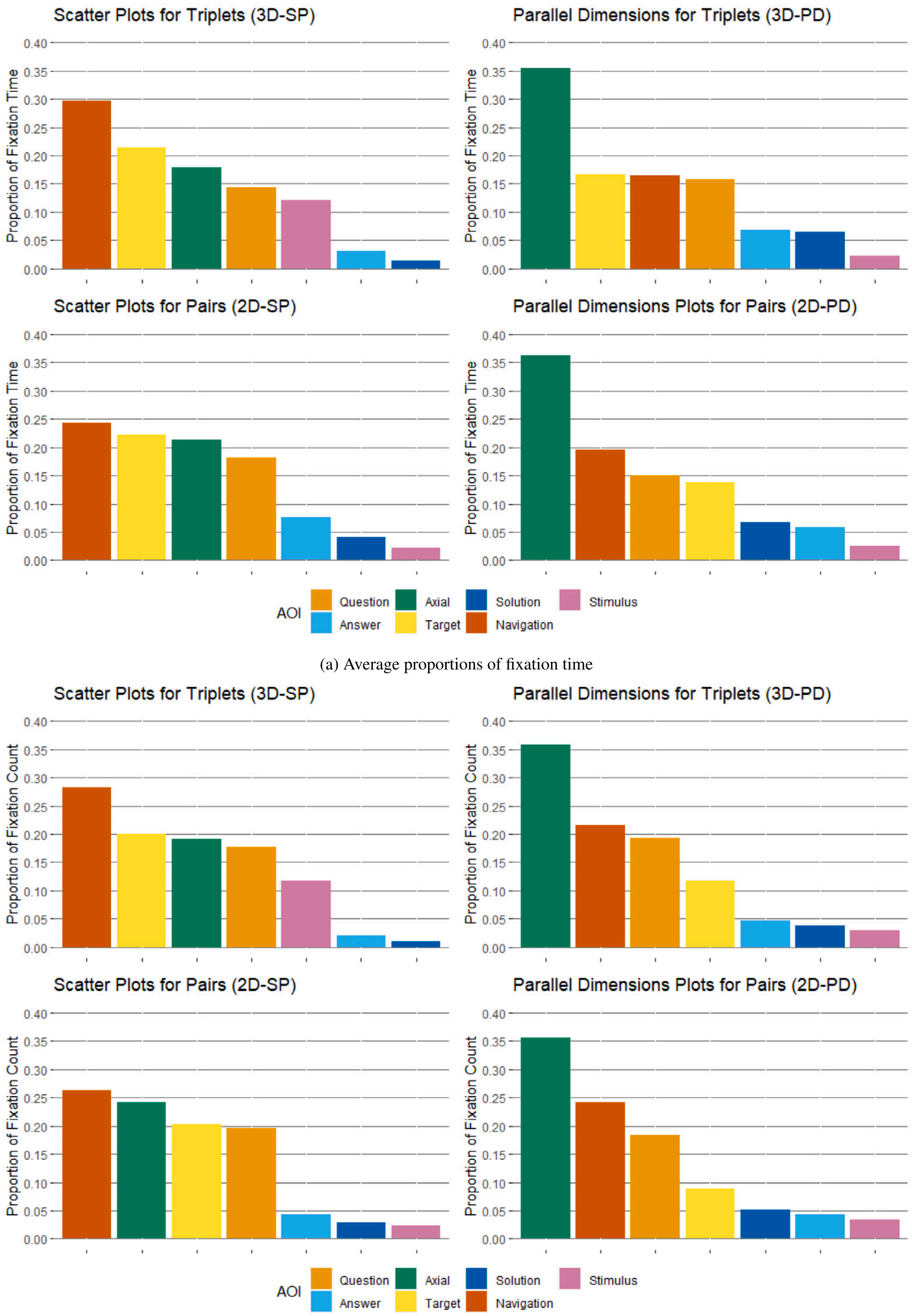
When considering fixation counts, Fig. 14b shows two moderate correlations. In particular, more fixations at the `Question` is associated with less fixations at `Navigation` ($\rho = -0.58, p < .001$). As observed with fixation time, more fixations within the `Target` is associated with less fixations in `Axial` ($\rho = -0.57, p < .001$).

It is expected that spending a higher proportion of time looking at the target is related to a lesser proportion of time scanning the axis to find the target. The relationship between devoting more time looking at the question and less time generally navigating is probably due to a strategy of memorizing part(s) of the question (e.g. the features in the pair) prior to finding the target.

Question RQ4b. Is there a relation between the visual processing of the AOIs of the visual stimuli for triplets ($t = 3$)*?* In Fig. 15a, the more time spent looking at `Axial` is related to less time in `Navigation` ($\rho = -0.50, p < .001$) and less time looking at `Target` ($\rho = -0.48, p < .001$). Also, more time spent on `Navigation` showed less time in `Answer` ($\rho = -0.52, p < .001$).

The results concerning the proportions of fixation counts, shown in Fig. 15b, present moderate correlations between: *(i)* `Axial` and `Target` ($\rho = -0.53, p < .001$), *(ii)* `Axial` and `Stimulus` ($\rho = -0.47, p < .001$), *(iii)* `Axial` and `Navigation` ($\rho = -0.45, p < .001$), *(iv)* `Question` and `Navigation` ($\rho = -0.46, p < .001$), *(v)* `Answer` and `Navigation` ($\rho = -0.41, p < .001$), *(vi)* `Stimulus` and `Solution` ($\rho = -0.45, p < .001$). These results show that `Axial` and `Navigation` AOIs have a relation with most of the other AOIs and like with pairs, more counts in navigating the axes implies lesser counts in the target solution.

Discussion for RQ4. Our analysis of visual attention uncovered the following findings. The first finding is that there are advantages between the visualization techniques. Scatter plots demand overall less visual attention for pairs whereas parallel dimensions plots do for triplets.



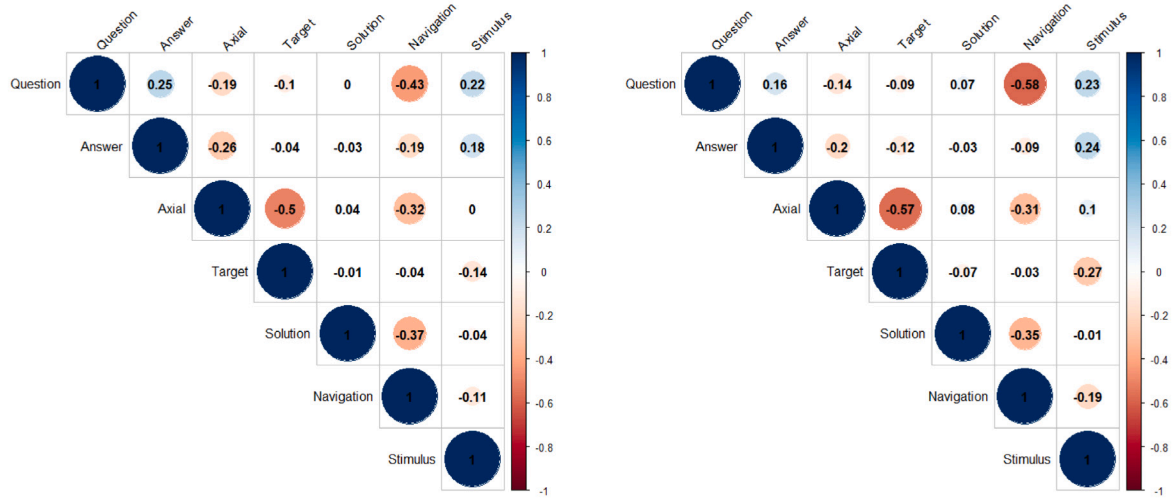
(a) Average proportions of fixation time

(b) Average proportions of fixation count

Fig. 13. Average proportions of fixation time and fixation count per t value and visualization technique.

The second finding is that there is a clear switch of attention focus in the visualization techniques. In scatter plots the highest proportions were all in the Navigation AOI, whereas in parallel dimensions plots the highest proportions all were in Axial AOI. We found a difference between 11% and 16% in the proportions of Axial, and a difference between 2% and 7% in Navigation. We interpret these differences as a materialization of the fact that the navigation mental models created and accessed by participants to solve the tasks are indeed different. In particular, the lines connecting the pairs and triplets in parallel dimensions reduce the effort in navigating the visualization at the price of demanding more proportional effort in searching for the features in the axes.

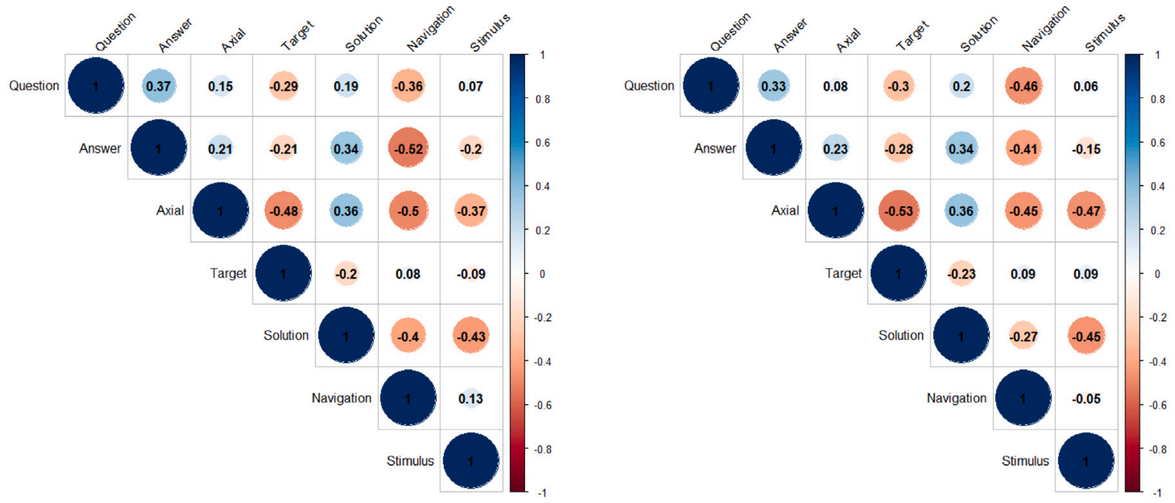
The third finding is that the relative ranks of the proportions of effort of the AOIs in the visualization techniques vary slightly for the three AOIs with most attention devoted to: Navigation, Target and Axial. For example, Navigation got the most attention, proportions of fixation time and count, in scatter plots for pairs and triplets. Similarly, Axial got the most attention in parallel dimensions plots also for both pairs and triplets.



(a) Correlation matrix for proportion of fixation time

(b) Correlation matrix for proportion of fixation count

Fig. 14. AOIs correlation matrices for pairs (RQ4a).



(a) Correlation matrix for proportion of fixation time

(b) Correlation matrix for proportion of fixation count

Fig. 15. AOIs correlation matrices for triples (RQ4b).

The fourth finding is that the analysis of the correlation matrices highlighted the AOIs that have moderate negative correlation values. In the case of pairs, `Question` and `Navigation` exhibited this type of relationship in proportions of fixation time and count, whereas `Target` and `Axial` only in proportion of fixation time. We argue that these relationships help to explain the differences found between scatter plots and parallel dimensions plots. In the former, `Target` and `Axial` are the second and third ranked AOIs whereas in the latter `Axial` is the highest proportion followed by `Navigation` and `Question` on both proportions of fixation time and count. In the case of triplets, `Axial` and `Navigation` have a moderate negative correlation in proportion of fixation time. In contrast, in proportion of fixation count: `Axial` correlates with `Target`, `Stimulus`, and `Navigation`; `Navigation` correlates with `Question` and `Answer`; and `Stimulus` relates with `Solution`. We argue that all these correlations confirm as well the preponderance of `Axial`, `Navigation` and `Target` as the core components of the visualization mental models.

5.5. Summary of results

Our study found advantages, disadvantages and trade-offs between the two visualization techniques across the aspects analyzed. In terms of response accuracy, scatter plots performed better for the case of pairs and seems to be unaffected by the number of pairs considered. In contrast, for the case of triplets, parallel dimensions plots took the lead, though in what appears to be a non-discernible pattern. Regarding time-on-task, scatter plots produced shorter times for pairs while parallel dimensions did for triplets. For both techniques, we found effects with the number of pairs and triplets.

Of the two metrics we used for metacognition monitoring, only certainty assessment yielded interesting results. For pairs, participants were more certain of the answers when they were indeed accurate. Furthermore, they were equally certain for scatter plots whether they were ultimately accurate or not, but they were more able to predict the outcome using parallel dimensions plots. For the case of triplets, we did not find a relationship between response accuracy and certainty assessment.

The eye-tracker data revealed the predominance of three AOIs while solving the CIT tasks, namely: `Navigation`, `Target`, and `Axial`. Hence they together constitute the core components of the mental models that participants built and manipulated to provide their responses. It is important to highlight that fixation time and fixation count are related across the AOIs. This means that a higher proportion of fixation time of an AOI, in relation to other AOIS, more likely entails a higher proportion of fixation count of that AOI.

In these terms, we unveiled two findings. We found a clear difference between visualization techniques. In scatter plots, the highest proportion was in `Navigation` whereas in parallel dimensions plots the highest proportion was in `Axial`. We argue that this switch of focus comes from the fact that parallel dimensions plots ease the required attention of the navigation with the polylines that connect the elements of the pairs and the triples, while increasing the efforts of finding the features in the axis of each dimension.

We also found that both visualization techniques have very similar proportions for pairs and triplets for the same AOIs. These proportion increased in scatter plots for `Navigation` while they decreased slightly (<%5) for the other AOIs. In contrast, for parallel dimensions plots, the proportions of `Axial` were preserved while for the other AOIs they increased or decreased slightly (<%5). This means that the proportions of the visual attention are basically preserved when the value of t increases.

6. Threats to validity

Our empirical study encountered several potential threats to validity, similar to other studies relying on eye-tracker measures (Sepasi et al., 2022; Wohlin et al., 2012; Sharafi et al., 2020). In addressing the robustness of our experimental results, we will delve into the four types of threats proposed by Wohlin et al. (2012), namely internal validity, external validity, construct validity, and conclusion validity.

Internal validity. Our work identified the following potential internal validity threats. The first threat concerns the selection of participants. We addressed this issue by undertaking a recruitment strategy as comprehensive as possible, ensuring a large and diverse participant pool that considered aspects such as gender and technical knowledge in Software Engineering.

The second threat refers to any learning effect or bias resulting from the order in which the participants performed the select CIT tasks. We addressed this threat by assigning a unique and random sequence of tasks to each participant. In addition, we balanced the number of tasks for the factors considered: covering array strength, visualization technique, and number of pairs or triplets.

The third threat relates to issues coming from the instrumentation errors of the experimental design. To mitigate this threat, we devised a meticulous experimental protocol based on guidelines and standards of the research fields. Among the stringent measures taken, there were aspects such as the verification of the eye-tracker data accuracy, correct calibration and adaptation steps of all the equipment for each participant, and adherence to the usage guidelines provided by the eye-tracker manufacturer. Additionally, the implementation of both manual and automated validation processes for the collected data played a crucial role in enhancing the reliability of our measurements.

External validity. Regarding external validity threats, the first threat identified is the selection of the feature models for the tasks. To address this threat, our feature models were chosen from a dataset that was identified as commonly used case studies in the field of SPL CIT testing (Ferreira et al., 2021). We were careful in selecting an adequate balance in terms of number of features and number of pairs or triplets to cover. It is essential to acknowledge that opting for a different set of feature models may yield different results.

The second threat is the choice of visualization techniques and the tools to render them. We employed two of the most basic techniques implemented in a popular visualization tool. Hence, our results may vary in other basic techniques or in other tools that could provide more or different forms of interactions.

The third threat concerns the selection of the color palettes used in the visualization techniques. The scope of our exploratory study did not include color topics such as accessibility as experimental factors. We chose simple and basic palettes that worked well in the pilot study and the actual experiment. Certainly, the selection of different color palettes may produce different results. However, we reiterate that the visualization techniques chosen had additional navigation aids, such that the participants did not rely solely on the color of the visual components to perform their tasks.

The fourth threat is the context where the experiment took place and the limited profiles of expertise studied. As described before, our experiment unfolded in an academic environment, involving participants who lacked prior knowledge in the SPL domain. While this setting allowed us to glean valuable insights, caution must be exercised in generalizing our results to other groups, particularly experienced developers.

Construct validity. Concerning the construct validity threats, we identified the choice of both the measures and the statistical analysis used to answer and interpret our research questions. In terms of measures, we employed the standard measures of cognitive load, response accuracy, time-on-task, and the proportion of fixation counts and proportion of fixation time on different AOIs. In terms of statistical analysis, we employed well-established and up-to-date statistical test procedures. Thus by aligning with recognized cognitive load metrics and employing standard statistical procedures (Gonçales et al., 2021; Holmqvist and Andersson, 2017), we sought to minimize these threats, thus enhancing the robustness and reliability of our analytical framework.

Conclusion validity. Regarding conclusion validity threats, we first identified the experimental environment, which consisted in conducting the experiments in a purpose-designed laboratory, providing identical training materials, and presenting the same information to each participant. To further maintain consistency, participants were not allowed to pose questions after watching the training video, minimizing the potential for technical discussions that could introduce bias. To mitigate the threat, we ensured a consistent experimental environment for all participants. We also must acknowledge as a potential threat factors such as the participants' individual conditions and motivation during the experiment may still exert an influence on the results. Recognizing these aspects helps contextualize the conclusions drawn and underscores the need for cautious interpretation within the specified experimental constraints.

7. Related work

There exists an extensive body of literature in the three main research fields that intersect our exploratory study: Software Product Lines, Software Visualization, and Eye-Tracking in Software Engineering. In this section, we present a succinct summary of the most pertinent publications for our study across these three fields.

Software Product Lines. Extensive and notable benefits such as enhanced product customization and a shortened time to market have attracted the developers' attention to Software Product Lines (SPL) over the past two decades (Czarnecki and Eisenecker, 2000; Clements and Northrop, 2002; Apel et al., 2013; Lopez-Herrejon et al., 2015). However, the extensive and intricate nature of SPL poses a substantial challenge for developers when it comes to testing such systems. In a comprehensive mapping study, Lopez-Herrejon et al. dug into the landscape of SPL testing studies and identified Configuration Interaction Testing (CIT) as a viable solution for effectively testing complex SPL systems (Lopez-Herrejon et al., 2015). More recent work by Ferreira et al. looked at case studies commonly used in SPL CIT research (Ferreira et al., 2021). We used their work as the original dataset for the selection of our tasks as indicated in Section 4.3.

Software Visualization. Broadly construed, Software Visualization is the art and science of generating visual representations of various aspects of software and its development process. The goal of software visualization is to help the comprehension of software systems and to improve the productivity of the software development process (Diehl, 2007). Thus, in contrast with the fields of Information Visualization (Spence, 2014; Ware, 2015) and Data Visualization (Ward et al., 2010; Telea, 2015), the scope of software visualization is focused on software development and software artefacts. In Data Visualization and Information Visualization, there exists multiple sources that provide useful and evidence-based guidelines for the design of successful visualizations also involving multidisciplinary teams (e.g., Walny et al., 2020; Few, 2012), among them are ten principles proposed by Midway that can serve as guidelines for designing data visualizations (Midway, 2020). To the best of our knowledge, there is only incipient work on general guidelines for Software Visualization as described next.

Bedu et al. performed a thorough literature review spanning from 2001 to 2018 that collected 48 software visualization studies (Bedu et al., 2019). Their study revealed the widespread use of Software Visualization across various software domains, notably in Software Architecture Visualization. Interestingly, a noticeable gap in the utilization of visualization techniques within the software testing domain became apparent in studies conducted between 2013 and 2019 (Chotisarn et al., 2020).

Despite the numerous visualization techniques that have been proposed for Software Engineering, a notable gap exists in terms of evaluation techniques for assessing them, as emphasized by Bedu et al. (2019) and Merino et al. (2018). This deficiency in evaluation can be addressed through the implementation of case studies and experiments involving target participants (Merino et al., 2018). Among the diverse methods for collecting participant data in experiments, the most prevalent approach is the use of questionnaires (Merino et al., 2018). While methods such as Think-Aloud, interviews, and surveys can aid in comprehending the experiment process, it is essential to acknowledge their susceptibility to potential influences from participants' memory, communication skills, and subjective judgment that may impact the participants' ability to provide accurate insights into their processes and intentions (Sharafi et al., 2020; Cunningham and Wallraven, 2019; Lazar et al., 2017).

There exists a significant body of research on software visualization applied to different aspects of Software Product Lines (Lopez-Herrejon et al., 2018; Pleuss et al., 2011; Medeiros et al., 2023). The particular challenge in this domain is handling the variability in the artifacts and the development process that can lead to a large number of feature combinations that must be effectively and efficiently managed. An example is analyzing the vast volume of data generated during SPL testing that can be a cumbersome and time-consuming endeavor as highlighted by both Pleuss et al. (2011) and Lopez-Herrejon et al. (2018). Thus, a pivotal objective within the SPL domain is to manage the complexity through the implementation of visual and interactive techniques. The mapping study conducted by Lopez-Herrejon et al. further underscores the need of such techniques, revealing low levels of utilization of visualization techniques in the SPL testing domain (Lopez-Herrejon et al., 2018). The mapping study by Medeiros et al. highlight the diverse and extensive research on visualization techniques for variant-rich systems, and emphasizes the need for more formal empirical evaluations in particular in industrial settings (Medeiros et al., 2023).

Eye-Trackers in Software Engineering. Utilizing eye trackers allows for an in-depth exploration of participants' cognitive processes during dedicated tasks (Sharafi et al., 2015b), particularly in understanding how various stimuli can influence participants' strategies in task completion and their cognitive effort (Sharafi et al., 2020). This recognition of eye trackers as a useful instrument for conducting experiments in Software Engineering dates back to at least 2006, when the first studies were published (Sharafi et al., 2015b). Despite this fact, only a limited number of software visualization studies have incorporated eye tracking, for instance as attested in the literature review of Merino et al. who surveyed works between 2002 and 2017 (Merino et al., 2018). Similarly, another mapping study conducted by Goncales et al. highlighted a lack of eye tracking studies, constituting only 6% of the research conducted from 2009 to 2018 on cognitive load of software engineers (Gonçales et al., 2019, 2021).

Performing studies with eye-trackers in Software Engineering come with several challenges. For instance, designing stimuli for scenarios involving scrolling and traversing between pages. In response to these challenges, researchers have developed stimuli that can fit on a single page (Sharafi et al., 2015b). But also, tools have been developed, as demonstrated by Lankford (2000) and Walters et al. (2013), that address some of the limitations associated with scrolling, thus enhancing the experimental procedures.

Another important challenge is the accuracy of eye-trackers in detecting eye positions that must be considered in the design and selection of AOIs. For example, Sharafi et al. recommend a 1° gap between AOIs, aiming to enhance precision during the eye-tracking process (Sharafi et al., 2020). In addition, researchers must carefully consider various parameters related to the devices themselves. A crucial parameter is the sampling rate, that can typically range from 10 Hz to 2000 Hz (Sharafi et al., 2015b). A basic example choice is 60 Hz according to Poole and Ball (2004).

Other important challenges to keep in mind are the careful selection of other parameters of the experimental design and their interplay with the visualization techniques. Among them are the data collection method, the type of analysis, the number of participants, the number of tasks, the dependent and independent variables, and statistical tests (Merino et al., 2018). Sharafi et al. provide valuable insights and guidelines to aid researchers in designing experiments using this technology (Sharafi et al., 2020). Our exploratory study followed these guidelines.

While numerous studies have individually explored SPL testing, Software Visualization, or using eye trackers, there is a notable absence of studies combining these three subjects. For instance, Burch et al. performed experiments utilizing eye-trackers to identify the optimal visualization technique among traditional, orthogonal, and radial node-link tree layouts (Burch et al., 2011). Their analysis encompassed a comparison of accuracy and time spent on dedicated tasks, taking into consideration the layout, orientation, and the number of marked nodes as independent variables.

Closer to our work, more recently Sepasi et al. (2022) conducted an eye-tracking study to assess participants' cognitive effort during feature model comprehension tasks. Researchers exploring eye movement metrics have considered various parameters to evaluate their studies, with a particular focus on fixation-related metrics, as cognitive processes mainly take place during fixations (Sharafi et al., 2015b). Examining the ratio of fixation time, Sepasi et al. concluded that an extended duration of fixation can lead to an incorrect response (Sepasi et al., 2022). Similarly, they obtained the same results when assessing the ratio of fixation count. Both the fixation count and fixation time were analyzed in terms of the Number of Features (NoF) and Number of Constraints (NoC) of the feature models with the aim of investigating the cognitive load of checking the validity of configurations based on the feature models.

8. Conclusions and future work

We presented an empirical study that evaluated two basic visualization techniques, scatter plots and parallel dimensions plots, applied to Combinatorial Interaction Testing covering arrays of strength pairs ($t = 2$) and triplets ($t = 3$) for Software Product Lines. Each participant in our study performed 16 simple test coverage tasks using both visualizations. We selected case studies from a dataset of feature models commonly used by the research community in that field. Our analysis considered four aspects: response accuracy, time-on-task, metacognition monitoring, and visual attention. The last one employed eye-tracking measures to gauge at the distribution of attention effort among seven selected areas of the visualizations.

We found the following most salient findings. In terms of response accuracy, time-on-task and visual attention, scatter plots exhibited advantages over parallel dimensions plots for the case of covering arrays of pairs. In contrast, for triplets and the same three factors, parallel dimensions plots were the clear winner. Regarding visual attention, each visualization had a predominant area of interest where the highest proportion of fixation time and proportion of fixation counts were found. Concerning the metacognitive monitoring, we found only for the case of pairs that there exists a relation of self-assessment of certainty with response accuracy, this means that participants were more certain of their answer when they were indeed accurate.

As part of our future work, we plan to explore the following points. Our study used static visualizations of data from feature models with a small number of features. We would like to study other visualization techniques that permit richer forms of interactions (e.g. scrolling and zooming) that would allow the depiction and manipulation of larger number of pairs and triplets. The main challenge is developing the adequate and flexible tool chain support that gather the information from the eye-tracker in coordination with the customized user interfaces. We plan to explore the application of more sophisticated statistical analyses and machine learning algorithms to discover and predict patterns of gaze and usage data and their relation with the accuracy and performance of the tasks. We hope our work contributes to building a foundation for the empirical evaluation of user-centered interfaces for relevant software engineering tasks.

CRediT authorship contribution statement

Kambiz Nezami Balouchi: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Julien Mercier:** Writing – review & editing, Writing – original draft, Validation, Supervision, Project administration, Methodology, Investigation, Formal analysis, Conceptualization. **Roberto E. Lopez-Herrejon:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work partially supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) grant RGPIN-2017-05421. We thank Serge Vicente and Sylvie Gervais for their feedback on drafts of the paper and the statistical analyses.

Data availability

Our institutions have constraints regarding what artifacts can be made publicly available when they relate to human participants. Thus, for replication purposes, we have publicly shared in our institutional repository Lopez-Herrejon and Nezami Balouchi (2024), all the data/code that we are allowed to.

References

Ahmed, B.S., Zamli, K.Z., Afzal, W., Bures, M., 2017. Constrained interaction testing: A systematic literature study. IEEE Access 5, 25706–25730. http://dx.doi.org/10.1109/ACCESS.2017.2771562.

Albert, B., Tullis, T., 2023. Measuring the User Experience. Collecting, Analyzing and Presenting UX Metrics, third ed. Elsevier. Morgan Kaufman..

Apel, S., Batory, D.S., Kästner, C., Saake, G., 2013. Feature-Oriented Software Product Lines - Concepts and Implementation. Springer, http://dx.doi.org/10.1007/978-3-642-37521-7.

Bedu, L., Tinh, O., Petrillo, F., 2019. A tertiary systematic literature review on software visualization. In: 2019 Working Conference on Software Visualization (Vissoft). IEEE, pp. 33–44.

Benavides, D., Segura, S., Cortés, A.R., 2010. Automated analysis of feature models 20 years later: A literature review. Inf. Syst. 35 (6), 615–636.

Bidlake, L., Aubanel, E., Voyer, D., 2020. Systematic literature review of empirical studies on mental representations of programs. J. Syst. Softw. 165, 110565. http://dx.doi.org/10.1016/j.jss.2020.110565.

Burch, M., Konevtsova, N., Heinrich, J., Höferlin, M., Weiskopf, D., 2011. Evaluation of traditional, orthogonal, and radial tree diagrams by an eye tracking study. IEEE Trans. Vis. Comput. Graphics 17 (12), 2440–2448.

Chen, O., Paas, F., Sweller, J., 2021. Spacing and interleaving effects require distinct theoretical bases: a systematic review testing the cognitive load and discriminative-contrast hypotheses. Educ. Psychol. Rev. 33 (4), http://dx.doi.org/10.1007/s10648-021-09613-w.

Chi, E.H., 2002. A Framework for Visualizing Information. Springer.

Chotisarn, N., Merino, L., Zheng, X., Lonapalawong, S., Zhang, T., Xu, M., Chen, W., 2020. A systematic literature review of modern software visualization. J. Vis. 23 (4), 539–558.

Clements, P., Northrop, L.M., 2002. Software Product Lines - Practices and Patterns. SEI series in software engineering, Addison-Wesley.

Cunningham, D.W., Wallraven, C., 2019. Experimental Design. From User Studies to Psychophysics. CRC Press. Taylor and Francis Group..

Czarnecki, K., Eisenecker, U.W., 2000. Generative Programming - Methods, Tools and Applications. Addison-Wesley, URL http://www.addison-wesley.de/main/main.asp?page=englisch/bookdetails&productid=99258.

de Blume, A.P.G., 2022. Influence of learning strategy instruction on improving students' accuracy in judgments. J. Educ. Psychol. Am. Psychol. Assoc. (APA) http://dx.doi.org/10.1037/edu0000674.

Détienne, F., 2001. Software Design Cognitive Aspects. Practitioner series, Springer, URL http://www.springer.com/computer/swe/book/978-1-85233-253-2.

Diehl, S., 2007. Software Visualization - Visualizing the Structure, Behaviour, and Evolution of Software. Springer, http://dx.doi.org/10.1007/978-3-540-46505-8.

Duchowski, A.T., 2017. Eye Tracking Methodology - Theory and Practice, Third Edition. Springer, http://dx.doi.org/10.1007/978-3-319-57883-5.

Ferreira, F., Vale, G., Diniz, J.P., Figueiredo, E., 2021. Evaluating T-wise testing strategies in a community-wide dataset of configurable software systems. J. Syst. Softw. 179, 110990. http://dx.doi.org/10.1016/j.jss.2021.110990.

Few, S., 2012. Show Me the Numbers: Designing Tables and Graphs to Enlighten, second ed. Analytics Press.

Garvin, B.J., Cohen, M.B., Dwyer, M.B., 2011. Evaluating improvements to a meta-heuristic search for constrained interaction testing. Empir. Softw. Eng. 16 (1), 61–102. http://dx.doi.org/10.1007/s10664-010-9135-7.

Gonçales, L.J., Farias, K., da Silva, B.C., 2021. Measuring the cognitive load of software developers: An extended systematic mapping study. Inf. Softw. Technol. 136, 106563.

Gonçales, L., Farias, K., da Silva, B., Fessler, J., 2019. Measuring the cognitive load of software developers: a systematic mapping study. In: 2019 IEEE/ACM 27th International Conference on Program Comprehension. ICPC, IEEE, pp. 42–52.

Händel, M., Bukowski, A.-K., 2019. The gap between desired and expected performance as predictor for judgment confidence. J. Appl. Res. Mem. Cogn. Elsevier 8 (3), 347–354. http://dx.doi.org/10.1016/j.jarmac.2019.05.005.

Hart, S.G., Staveland, L.E., 1988. Development of NASA-TLX (task load index): Results of empirical and theoretical research. In: Hancock, P.A., Meshkati, N. (Eds.), Human Mental Workload. In: Advances in Psychology, vol. 52, North-Holland, pp. 139–183. http://dx.doi.org/10.1016/S0166-4115(08)62386-9, URL https://www.sciencedirect.com/science/article/pii/S0166411508623869.

Henard, C., Papadakis, M., Perrouin, G., Klein, J., Heymans, P., Traon, Y.L., 2014. Bypassing the combinatorial explosion: Using similarity to generate and prioritize T-wise test configurations for software product lines. IEEE Trans. Softw. Eng. 40 (7), 650–670.

Heradio, R., Perez-Morago, H., Fernández-Amorós, D., Cabrerizo, F.J., Herrera-Viedma, E., 2016. A bibliometric analysis of 20 years of research on software product lines. Inf. Softw. Technol. 72, 1–15. http://dx.doi.org/10.1016/j.infsof.2015.11.004.

Hervieu, A., Baudry, B., Gotlieb, A., 2011. PACOGEN: automatic generation of pairwise test configurations from feature models. In: ISSRE. pp. 120–129. http://dx.doi.org/10.1109/ISSRE.2011.31, URL http://doi.ieeecomputersociety.org/10.1109/ISSRE.2011.31.

Holmqvist, K., Andersson, R., 2017. Eye Tracking: a Comprehensive Guide to Methods, Paradigms, and Measures. CreateSpace.

Johansen, M.F., Haugen, Ø., Fleurey, F., 2012. An algorithm for generating t-wise covering arrays from large feature models. In: de Almeida, E.S., Schwanninger, C., Benavides, D. (Eds.), 16th International Software Product Line Conference, SPLC '12, Salvador, Brazil - September 2-7, 2012, Volume 1. ACM, pp. 46–55. http://dx.doi.org/10.1145/2362536.2362547.

Just, M.A., Carpenter, P.A., 1980. A theory of reading: from eye fixations to comprehension. Psychol. Rev. 87 (4), http://dx.doi.org/10.1037/0033-295X.87.4.329.

Kuhn, D.R., Kacker, R.N., Lei, Y., 2016. Introduction to Combinatorial Testing, First Chapman and Hall/CRC.

Lai, M., et al., 2013. A review of using eye-tracking technology in exploring learning from 2000 to 2012. Educ. Res. Rev. 10, 90–115.

Lankford, C., 2000. Gazetracker: software designed to facilitate eye movement analysis. In: Proceedings of the 2000 Symposium on Eye Tracking Research & Applications. pp. 51–55.

Lazar, J., Feng, J., Hochheiser, H., 2017. Research Methods in Human-Computer Interaction, second ed. Morgan Kaufmann, URL https://www.sciencedirect.com/science/book/9780128053904.

Lopez-Herrejon, R.E., Batory, D.S., 2001. A standard problem for evaluating product-line methodologies. In: Bosch, J. (Ed.), Generative and Component-Based Software Engineering, Third International Conference, GCSE 2001, Erfurt, Germany, September 9-13, 2001, Proceedings. In: Lecture Notes in Computer Science, vol. 2186, Springer, pp. 10–24. http://dx.doi.org/10.1007/3-540-44800-4_2.

Lopez-Herrejon, R.E., Chicano, J.F., Ferrer, J., Egyed, A., Alba, E., 2013. Multi-objective optimal test suite computation for software product line pairwise testing. In: ICSM. pp. 404–407.

Lopez-Herrejon, R.E., Ferrer, J., Chicano, F., Egyed, A., Alba, E., 2016. Evolutionary computation for software product line testing: An overview and open challenges. In: Pedrycz, W., Succi, G., Sillitti, A. (Eds.), Computational Intelligence and Quantitative Software Engineering. In: Studies in Computational Intelligence, vol. 617, Springer, pp. 59–87. http://dx.doi.org/10.1007/978-3-319-25964-2_4.

Lopez-Herrejon, R.E., Fischer, S., Ramler, R., Egyed, A., 2015. A first systematic mapping study on combinatorial interaction testing for software product lines. In: Eighth IEEE International Conference on Software Testing, Verification and Validation, ICST 2015 Workshops, Graz, Austria, April 13-17, 2015. IEEE Computer Society, pp. 1–10. http://dx.doi.org/10.1109/ICSTW.2015.7107435.

Lopez-Herrejon, R.E., Fischer, S., Ramler, R., Egyed, A., 2015. A first systematic mapping study on combinatorial interaction testing for software product lines. In: 2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops. ICSTW, IEEE, http://dx.doi.org/10.1109/icstw.2015.7107435.

Lopez-Herrejon, R.E., Illescas, S., Egyed, A., 2018. A systematic mapping study of information visualization for software product line engineering. J. Softw.: Evol. Process 30 (2), http://dx.doi.org/10.1002/smr.1912.

Lopez-Herrejon, R.E., Illescas, S., Egyed, A., 2018. A systematic mapping study of information visualization for software product line engineering. J. Softw.: Evol. Process 30 (2), e1912.

Lopez-Herrejon, R.E., Nezami Balouchi, K., 2024. JSS An Empirical Eye-Tracker Study of Visualization Techniques for Coverage of Combinatorial Interaction Testing in Software Product Lines. Borealis repository. http://dx.doi.org/10.5683/SP3/S7GJGW.

Medeiros, R., Martinez, J., Díaz, O., Falleri, J., 2023. Visualizations for the evolution of variant-rich systems: A systematic mapping study. Inf. Softw. Technol. 154, 107084. http://dx.doi.org/10.1016/j.infsof.2022.107084.

Merino, L., Ghafari, M., Anslow, C., Nierstrasz, O., 2018. A systematic literature review of software visualization evaluation. J. Syst. Softw. 144, 165–180.

Midway, S.R., 2020. Principles of effective data visualization. Patterns 1 (9), http://dx.doi.org/10.1016/j.patter.2020.100141, Publisher: Elsevier.

Obaidellah, U., Haek, M.A., Cheng, P.C., 2018. A survey on the usage of eye-tracking in computer programming. ACM Comput. Surv. 51 (1), 5:1–5:58. http://dx.doi.org/10.1145/3145904.

Peterson, M., Kramer, A., Irwin, D., 2004. Covert shifts of attention precede involuntary eye movements. Percept. Psychophys. 66 (3), 398–405.

Pleuss, A., Rabiser, R., Botterweck, G., 2011. Visualization techniques for application in interactive product configuration. In: Proceedings of the 15th International Software Product Line Conference, Volume 2. pp. 1–8.

Poole, A., Ball, L.J., 2004. Eye tracking in human-computer interaction and usability research : Current status and future prospects. URL https://api.semanticscholar.org/CorpusID:972615.

Posner, M.I., 1993. The Foundations of Cognitive Science. Bradford.

Raudenbush, S., Bryk, A., 2001. Hierarchical Linear Models: Applications and Data Analysis, second ed. Sage Publications.

Rayner, K., 1998. Eye movements in reading and information processing: 20 years of research. Psychol. Bull. 124 (3).

Rayner, K., 2009. The 35th sir frederick bartlett lecture: eye movements and attention in reading,scene perception, and visual search. Q. J. Exp. Psychol. 62 (8), 1457–1506.

Sepasi, E.R., Balouchi, K.N., Mercier, J., Lopez-Herrejon, R.E., 2022. Towards a cognitive model of feature model comprehension: an exploratory study using eye-tracking. In: Proceedings of the 26th ACM International Systems and Software Product Line Conference-Volume a. pp. 21–31.

Serra, M.J., Metcalfe, J., 2009. In: Hacker, D.J., Dunlosky, J., Graesser, A.C. (Eds.), Handbook of Metacognition in Education. Routledge/Taylor & Francis Group, pp. 278–298.

Sharafi, Z., Sharif, B., Guéhéneuc, Y., Begel, A., Bednarik, R., Crosby, M.E., 2020. A practical guide on conducting eye tracking studies in software engineering. Empir. Softw. Eng. 25 (5), 3128–3174. http://dx.doi.org/10.1007/s10664-020-09829-4.

Sharafi, Z., Soh, Z., Guéhéneuc, Y., 2015a. A systematic literature review on the usage of eye-tracking in software engineering. Inf. Softw. Technol. 67, 79–107. http://dx.doi.org/10.1016/j.infsof.2015.06.008.

Sharafi, Z., Soh, Z., Guéhéneuc, Y.-G., 2015b. A systematic literature review on the usage of eye-tracking in software engineering. Inf. Softw. Technol. 67, 79–107.

Spence, R., 2014. Information Visualization - An Introduction. Springer, http://dx.doi.org/10.1007/978-3-319-07341-5.

Telea, A.C., 2015. Data Visualization. Principles and Practice, second ed. CRC Press. Taylor & Francis Group.

Walny, J., Frisson, C., West, M., Kosminsky, D., Knudsen, S., Carpendale, S., Willett, W., 2020. Data changes everything: Challenges and opportunities in data visualization design handoff. IEEE Trans. Vis. Comput. Graphics 26 (1), 12–22. http://dx.doi.org/10.1109/TVCG.2019.2934538.

Walters, B., Falcone, M., Shibble, A., Sharif, B., 2013. Towards an eye-tracking enabled IDE for software traceability tasks. In: 2013 7th International Workshop on Traceability in Emerging Forms of Software Engineering. TEFSE, IEEE, pp. 51–54.

Ward, M., Grinstein, G., Keim, D., 2010. Interactive Data Visualization. Foundations, Techniques and Applications. A.K. Peters Ltd..

Ware, C., 2015. Information Visualization, fourth ed. Elsevier. Morgan Kaufman..

Williams, J.G., 1995. Visualization. Annu. Rev. Inf. Sci. Technol. (ARIST) 30, 161–207, URL https://www.learntechlib.org/p/80656.

Winne, P.H., 1996. A metacognitive view of individual differences in self-regulated learning. Learn. Indiv. Differ. 8 (4), 327–353. http://dx.doi.org/10.1016/S1041-6080(96)90022-9, URL https://www.sciencedirect.com/science/article/pii/S1041608096900229.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., 2012. Experimentation in Software Engineering. Springer, http://dx.doi.org/10.1007/978-3-642-29044-2.

Zagermann, J., Pfeil, U., Reiterer, H., 2016. Measuring cognitive load using eye tracking technology in visual computing. In: Sedlmair, M., Isenberg, P., Isenberg, T., Mahyar, N., Lam, H. (Eds.), Proceedings of the Sixth Workshop on beyond Time and Errors on Novel Evaluation Methods for Visualization, BELIV 2016, Baltimore, MD, USA, October 24, 2016. ACM, pp. 78–85. http://dx.doi.org/10.1145/2993901.2993908.

Kambiz Nezami Balouchi He earned in 2024 his Masters in Software Engineering from the École de technologie supérieure, University of Quebec in Montreal, Canada.

Julien Mercier Full professor at the Computer Science Department of the University of Quebec in Montreal. He is the director of NeuroLab, a laboratory for the study of human–computer interaction using psychophysiological data. His current research focuses on the online measurement of aspects of cognition and emotions on a moment-by-moment basis, during prolonged and authentic activities involving computer systems. His areas of interest are human–computer interaction, cognition, emotions, learning, performance, intelligent tutoring systems, and collaborative use of technologies. He obtained his Ph.D. in Applied Cognitive Science at McGill University.

Roberto Erick Lopez-Herrejon Associate Professor of Software Engineering and Information Technology of the École de technologie supérieure, University of Quebec in Montreal, Canada. Prior he was senior researcher at Johannes Kepler University in Linz, Austria. He was an Austrian Science Fund (FWF) Lise Meitner Fellow (2012–2014), External Lecturer at the Software Engineering at the University of Oxford (2008–2014), Intra-European Marie Curie Fellow (2010–2012), Career Development Fellow University of Oxford (2005–2008), Fulbright Fellow (1998–2001). He obtained his Ph.D. from the University of Texas at Austin in 2006. His main expertise is in software customization, software product lines, search-based software engineering, and empirical software engineering.