Movement Patterns and Trajectories in Three-Dimensional Software Visualization

Marcel Steinbeck
University of Bremen, Germany
marcel@informatik.uni-bremen.de

Rainer Koschke
University of Bremen, Germany
koschke@uni-bremen.de

Marc O. Rüdel
University of Bremen, Germany
mor@uni-bremen.de

***Abstract*—Software visualization is a growing field of research, in which developers are assisted in understanding and analyzing complex applications by mapping different aspects of a software system onto visual attributes. Under the assumption that virtual reality, due to the higher degree of immersion, may enhance user experience, researchers have begun to port existing visualization techniques to this environment. Oftentimes, layout algorithms and user interaction methods are more or less transferred one-to-one, though little is known about the effect of virtual reality in visual analytics and program comprehension. Moreover, little research on the behavior of developers in different three-dimensional visualization environments has been done yet.**

This paper extends the results of a previous controlled experiment, in which the EvoStreets visualization technique was compared in different two- and three-dimensional environments. In the original experiment, we could not find evidence that any of the environments, namely, 2D, 2.5D, and virtual reality, effects the time required to find an answer or the correctness of the given answer. However, we found indications that movement patterns differ between the 2.5D and the virtual reality environments. For this paper, we analyzed and refined the movement trajectories that have been recorded in the previous experiment. We found significant differences for some of the tasks that had to be solved by the participants. In particular, we found evidence that the path length, average speed, and occupied volume differ. Though we could find significant correlations between these metrics and correctness, we found indications that there is a correlation with time, which, in turn, differs significantly between the 2.5D and the VR environments for many tasks. These findings may have implications on the design of visualizations, interactions, and recommendation systems for these different environments.

I. Introduction

In the last decades, tools to analyze software systems regarding certain aspects, for instance, software clones, have been developed with great success. However, managing analysis results is still a challenging task due to the sheer amount of data. Hence, different visualization techniques, which are generally capable to synthesize a large amount of data, have been developed that are supposed to assist in assessing findings by aggregating results and, exploiting the human visual perception, simplifying the discovery of patterns in visual attributes. One of these visualizations is known as *EvoStreets*, a technique proposed by Steinbrückner and Lewerentz [1], [2]. Based on the software-as-a-city metaphor, originally introduced as *CodeCity* by Wettel and Lanza [3], [4], [5], EvoStreets are well suited to depict the evolution of arbitrary software elements (e.g. source files, classes, and methods). The hierarchical structure of these elements is depicted with nested streets, branching at each level change in the hierarchy. Leaves of the hierarchy, in turn, are shown as three-dimensional blocks, using the width, height, and breadth, as well as a color gradient as visual attributes to emphasize relevant metrics, such as lines of code, cyclomatic complexity, and change frequency. In comparison to the CodeCity approach, which visualizes the hierarchy by recursively subdividing a convex shape into subareas, the layout generated by EvoStreets is more stable to changes (with respect to the size and location of the leaves), allowing to track modifications more easily.

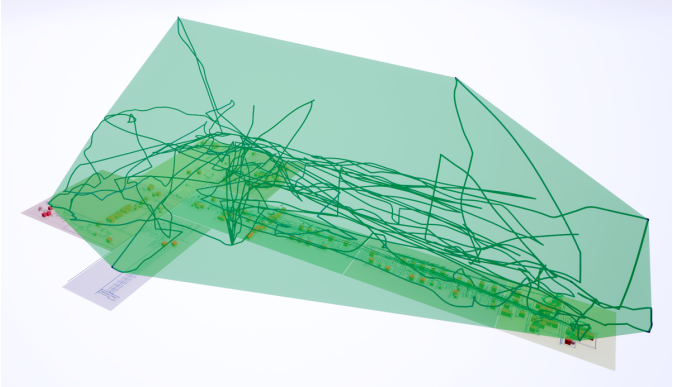


Fig. 1: Movement trajectories (and the convex hull spanned by the trajectories) of participants exploring software clones in EvoStreets.

First implementations of the CodeCity and EvoStreets visualizations used two- (known as 2D visualization) and three-dimensional (known as 2.5D visualization) rendering on regular two-dimensional displays. Lately, to enhance user experience and to take advantage of the better immersion, immersive virtual reality systems (IVRS) with head-mounted displays (HMD) and hand-held controllers are used to bring the city metaphor into the virtual reality (VR). Studies outside of software engineering have shown that HMDs may have a positive effect on the orientation in three-dimensional VR environments [6]. Researchers in software visualization have started to investigate whether these advantages observed in other fields also hold true in software engineering. In a previous controlled experiment, we compared the EvoStreets visualization technique in three different environments (2D, 2.5D, and VR) by forming three different tasks that had

to be solved by 34 participants—each task in one of these environments [7]. Contrary to the results by Chance et al. [6] outside of software engineering, our experiment did not find evidence that any of the environments effect the time to complete a task or the correctness of the given answers. However, indications had been found that movement patterns in the 2.5D and VR environments differ. These preliminary indications, however, were not analyzed in more depth.

Contributions. In this paper we examine the results of our previous experiment in more details with respect to differences in how the participants have moved within the city while solving a particular task.

In 2.5D, EvoStreets are rendered on a regular two-dimensional display, while in VR a head-mounted display presents a city in a stereoscopic manner. Furthermore, navigation differs in those environments. That is, in 2.5D a keyboard and mouse is used to move within EvoStreets whereas in VR participants can rotate their heads and bodies and use hand-held controllers to adjust the view. These two aspects may effect how humans move in EvoStreets, which leads us to our first research question.

RQ1 Do the two different environments 2.5D and VR effect movement patterns in EvoStreets?

If there are any effects (and we found initial evidence for that [7]), the next question becomes whether these differences have any impact on the correctness of the answers determined by human beholders. Our previous controlled experiment consisted of three different tasks, in which answers were either correct or wrong.

RQ2 Is there a correlation between movement patterns and correctness?

Along with correctness, the time required to find an answer for each task and environment has been recorded while running the experiment. Following the idea of RQ2, the question arises whether differences in movement patterns effect the time required to complete a task.

RQ3 Is there a correlation between movement patterns and the time required to complete a task?

The remainder of this paper is organized as follows. Section II presents related research. The design of the original experiment as well as our operational hypotheses are described in Section III. Results are presented and discussed in Section IV. Section V concludes.

II. Related Works

The focus of our research presented in this paper is on whether different kinds of visualization environments, namely, 2D and 2.5D desktop computers with monitor, keyboard, and mouse and virtual reality with head-mounted displays and hand-held computers, effect how humans move in EvoStreets. Such differences may be relevant, for instance, for recommendation systems tracking the movements of humans in the virtual worlds in order to learn from these and to make recommendations for the currently moving humans or for others who may follow them. If there are differences in the movement patterns, adjustments needs to be made in those recommender systems depending upon the type of environments developers move to comprehend a program.

In this section, we will describe related research on software visualization based on the city metaphor, empirical findings on differences between 2D and 2.5D desktop environments and virtual reality, and recommender systems.

A. Software Visualization Using the City Metaphor

Static as well as dynamic structures of software can be modeled by various kinds of graphs. We can use graphs, for instance, for the system decomposition along with dependencies, the class inheritance hierarchy, or for the static or dynamic calls between methods. These graphs are often the foundation for visualization [8]. There are many kinds of visualization for graphs, e.g., node-link diagrams (also known as network graphs), matrices [9], wheel views [10], or treemaps [11]. In this section, we will summarize the visualizations that come closest to EvoStreets, which is the one for which we analyzed experimental data. For a broader overview of software visualization, we refer the reader to more comprehensive surveys [12], [13], [14], [15], [8], [16], [17], [18], [19],

EvoStreets build on the idea of CodeCities which in turn are based on treemaps. A treemap presents hierarchical data (a tree) as a group of nested rectangles in 2D [11]. A rectangle captures each branch of the tree, which further contains smaller rectangles to denote sub-branches or leaves. The area size of an innermost rectangle is a visual encoding of a given property of the leaf element and chosen proportionally to that property. Often the color or texture of a rectangle are additional visual encodings of other properties. How rectangles are divided and ordered is determined by a tiling algorithm. There are different variants of these. One of the earliest and simplest ones is slice-and-dice tiling, which alternates the orientation of rectangles at each level of the hierarchy (vertically vs. horizontally). The popular squarified tiling instead attempts to keep each rectangle as square as possible, which is advantageous because humans have difficulties in comparing areas with different aspect ratios.

If one extends treemaps to the third dimension for yet another property to depict, the rectangles become blocks. Three-dimensional treemaps are often perceived as modern cities such as Manhattan. This impression is eponymous for the notion of *CodeCity*, originally introduced by Wettel and Lanza [3], [4], [5]. The tiling algorithms of treemaps and CodeCities make optimal use of the available screen estate, which is advantageous for large software systems. Yet, if not only a single version of a software but a sequence of multiple versions of an evolving program is to be visualized, the layout of treemaps and CodeCities is not flexible enough for accommodating changes in terms of new or deleted rectangles as well as the size of the rectangles. Their layout can change drastically from one version to another such that the mental map of a human beholder deteriorates [20]—although it must be mentioned that researchers have started to work on that

problem [21]. Another difficulty is that their compact layout offers very little distinct patterns that would help humans in recollecting visited places and their orientation.

Steinbrückner and Lewerentz [1], [2] have introduced EvoStreets as an alternative to CodeCities. The hierarchy is represented by road junctions rather than subdivided areas. Each level of the hierarchy is visualized by an individual line, representing a street within the city, whose width represents the nesting level: the lower the level, the thinner the street. Leaves of the hierarchy are represented as three-dimensional blocks as in CodeCities. EvoStreet require more space due their orthogonal street layout which causes large areas of empty space. Yet, that space allows them to grow or shrink to accommodate changes by preserving spatial relations of existing blocks, which helps in maintaining the mental map of a beholder.

Dependencies among the code entities depicted in treemaps, CodeCities, or EvoStreets can be visualized as edges. If edges are drawn straight, taking the shortest possible path between two connected elements, they easily create visual clutter by crossing nodes or other edges. If there are many such edges, the visualization becomes unreadable. Holten has proposed hierarchically bundled edges to reduce some of this visual clutter [10], [22]. Bundling means that edges from and to similar locations attract each other, analogously to a cable tie. Hierarchical bundling means that control points are added that act as the source of attracting forces for edges and the location of these control points is based on the hierarchical structure of the nodes. For treemaps, CodeCities, and EvoStreets the x and y co-ordinates of those control points are chosen as the center point of the area occupied by all elements of a subtree. In CodeCities and EvoStreets, which both offer a third dimension, the z co-ordinate of a control point is chosen proportionally to the closest common ancestor of the source and target of an edge: higher-level ancestors are farther away from the ground.

B. Virtual Reality Versus Desktop

3D and VR techniques have been explored for quite some time in software visualization. Knight and Munro gave an overview on VR for software visualization as early as 2000 [23]. Since then various other researchers have used VR and 3D for software visualization for static information [24], [25], [26], [27], [28], [29], [30], [31] or for dynamic data such as resource bottlenecks or memory leaks [32], [33], [34]. Elliott et al. discuss the affordances and challenges of VR in software engineering in general and present ideas on how it can be used for code reviews [35]. There is already also a great body of knowledge on VR and 3D visualization outside of computer science [36], [37], [38]. Studies in other disciplines have shown that head-mounted displays may enhance the orientation in three-dimensional environments as they allow to rotate and move more naturally as opposed to traditional approaches [6]. Sousa Santos et al., on the other hand, give an overview on virtual environments, discuss several papers that compared VR to desktop environments, and report on an experiment they conducted themselves [39]. In their experiment on navigation in a virtual maze, they found that—although users were generally satisfied with the VR system and found the VR interaction intuitive and natural—most performed actually better in the desktop environment. Ruddle et al. investigated the navigation in computer-simulated worlds with HMDs and with traditional desktop environments [40], [41]. In one of their experiments, participants had to navigate within large virtual buildings, consisting of one floor with nine rooms and one corridor [40]. The experiment was a repeated round tour through several rooms. In a second experiment, they looked into proprioceptive feedback and its influence on navigation within a virtual maze [41]. In the first experiment, they found that the HMD had an advantage to the speed of the participants [40], in contrast to their later experiment [41] and to Sousa Santos et al.'s experiment [39], which both took place in a maze. What exactly caused these conflicting results is a subject of further research.

The fact that working in different environments can cause different loads on human memory (and thus on a human's psyche) has long been known [42], [43] and is now referred to as *cognitive load* [44], [45], [46], [47], [48], [49]. Van der Land et al. [50] studied how 3D virtual environments affect group decision making in tasks with a strong visual component and found that the interaction and negotiation processes required for reaching a shared understanding in VR increase cognitive load and even make group processes inefficient. Gerry et al. [51] developed a VR system that keeps the cognitive load of its user in focus and tries to extend the performance during visual search tasks by adapting the visualizations.

Software-engineering researchers have already started to visualize EvoStreets in VR [52], [53], [7] to investigate whether that assists in solving software engineering tasks by offering more intuitive interactions. In a preliminary experiment, we investigated how successfully humans orient themselves in EvoStreets in 2.5D and VR, respectively [53]. The participants had to solve a homing task, that is, find their way back to the place where they started. Our primary hypothesis was that the HMD environment would make it easier to gain a spatial orientation inside of EvoStreets in comparison to a 2.5D environment but we found no statistically significant evidence for it in our controlled experiment with 20 participants. In a follow-up experiment, we investigated whether solving particular software-engineering tasks related to understand cloning in software is effected by 2D, 2.5D, and VR environments [7]. The details of that experiment will be presented in Section III. Similarly to Ruddle et al. [41] and to Sousa Santos et al. [39] for navigation tasks in computer-simulated mazes or interiors of buildings, we found no significant differences in solving tasks that required counting nodes and edges (clones) among 2D, 3D, and VR visualization for EvoStreets [7]. Yet, we found a significant difference in the way how participants moved between 3D desktop and VR environments. These indications form the starting point of our analysis described in Sections III and IV.

C. Recommendation Systems in Software Engineering

In the long run, we want to develop recommendation systems for developers moving in EvoStreets based on their previous history or of movements of other developers. Our current research presented in this paper is a foundational stepping stone towards this goal. In this section, we summarize related work on recommendation systems.

The development of ever faster and more powerful hardware and software has allowed the collection of data about the software and its development process. This essentially includes data on the structure and dynamics of the software and its parts, but also, for example, on the project team, the involvement of the team members and project progress. These data can be used for control, training, prediction, or auxiliary purposes [54]

The sheer amount of data, in turn, requires tools that make the desired informations in the data recognizable and understandable. Recommendation systems are tools that discover patterns in the data. They are often used in areas of analysis and diagnostics in general domains such as medical and vehicle diagnostics, or in commercial online shops where human decision-making is still an indispensable part of the process but should be supported by pre-filtering [55]. They can as well be used for software engineering tasks.

Developers leave their data traces in version control systems, issue trackers, or electronic conversations. Integrated development environments can be instrumented to collect additional data. The data can be leveraged by recommendation systems—if ethically used—to help them solving their complex tasks by learning from previous task solving by themselves or by others [56].

The data sources that can be of interest during the software development process are many and varied. Robillard and Walker [54] list seven typical additional data sources besides the actual, static source code, such as the project history, communication archives, dependent APIs, development environment, interaction traces, execution traces, and the web.

Many recommender systems are already implemented in tools. However, less attention in research has been paid to human aspects so far. The reason for this is probably the high cost that such user studies entail. As a result, little is known about the actual practical benefits of the recommendation systems [54]. Another reason is likely that development of software is much more complex than most other tasks currently supported by recommendation systems outside of software engineering, for instance, buying decisions in online shops. This makes it hard both to find a good solution algorithmically and to determine the right context first. Without the right context, it is not easy to make relevant recommendations.

D. Comparing EvoStreets in Different Environments

Merino compared CodeCities in 2.5D and augmented reality (AR) with respect to navigation, selection, occlusion, and text readability [57]. They found in a controlled experiment with nine participants and a subsequent user study that AR facilitates navigation and reduces occlusion, while performance in program-comprehension tasks is adequate, and developers obtain an outstanding experience. They identified selection and text readability as open issues.

This paper is based on the results of our previous controlled experiment [7], in which 34 participants, by means of the EvoStreets visualization technique, had to analyze existing Java systems regarding software clones. The subject systems are listed in Table I.

TABLE I: Java systems for the experiment.

Name	Lines of Code	Files	Clones
Jillion	75,520	929	66
JRuby	227,145	1360	110
Spring Boot	181,795	3042	228

The participants were recruited through convenience sampling. About half the participants (17) were undergraduate students attending a VR project on EvoStreets. The others were (graduate and advanced undergraduate) students of a software engineering course taught by our research group (5), regular students (3), researchers (7), and professional developers (2). The main objective of the experiment was to study the effect of three different environments, namely, orthographic projection with keyboard and mouse (2D), 2.5D projection with keyboard and mouse (2.5D), and virtual reality with head-mounted display and hand-held controllers (VR) on the time required to complete a task (no time limit was imposed to complete a task) and the correctness of the given answer. Though we could not find evidence that any of the environments has a significant effect, we found indications that movement patterns differ between the 2.5D and VR environments.

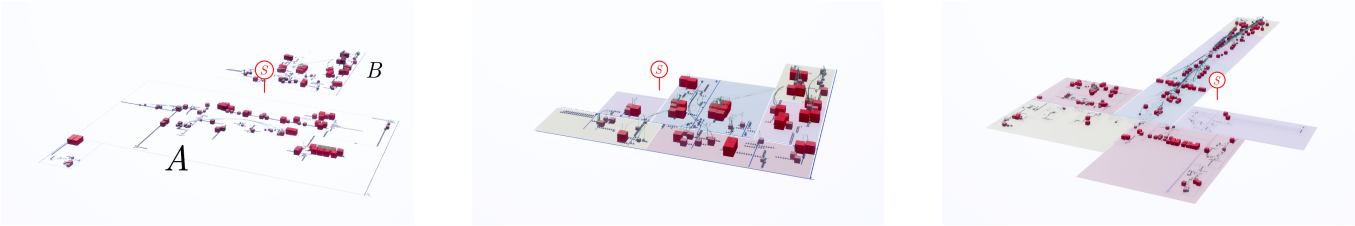
The three different tasks (depicted in Figure 2) were derived from a typical visual-analytics context. Each participant had to solve each task in one of the subject environments. To counter learning effects, the environments were presented to the participants in balanced latin square order. Before solving a task in a particular environment, the participant had the opportunity to learn the mechanics of the corresponding environment in a tutorial. Other details follow in the next section.

III. Experimental Setup and Hypotheses

As stated in Section II-D, the results of this paper are based on a previously conducted controlled experiment [7]. In the following, we first briefly describe the setup of this experiment. Then, we present our operational hypotheses which were derived from our research questions presented in Section I.

A. Technical Environments

The original experiment focused on the comparison of the three different environments 2D, 2.5D, and VR regarding their effects on visual clone analysis in EvoStreets. The main objective of this paper is to study the effect of the 2.5D and VR environments on the movement patterns of the participants. We focus on the three-dimensional environments only, because 2D lacks one dimension and is, hence, difficult to compare to



(a) Task 1 (count connections in different systems): Decide for the two systems A and B which one contains more fragments cloned in other source files.

(b) Task 2 (count connection in subsystems): Decide for the six subsystems which pair of subsystems shares the most connections.

(c) Task 3 (count blocks in subsystems): Decide for the five subsystems which contains the most source files with clones (red colored blocks).

Fig. 2: The three tasks of the original experiment. *S* marks the starting point.

the other two environments with more degrees of freedom to move.

2.5D environment (2.5D) In this environment, the three-dimensional model of an EvoStreets is rendered on a regular two-dimensional display. Using a keyboard and mouse, the participants were able to move within the EvoStreets and adjust the view as desired.

VR environment (VR) This environment uses the same visual attributes as the 2.5D environment, but renders the scene on a stereoscopic head-mounted display (HTC Vive). Using the full-room tracking mode, the participants were able to move in an area of around 2.5 $\times$ 2.5 meter. Along with the HMD, hand-held controllers of the HTC Vive could be used by the participants to navigate through the city.

B. Visualized Data

The shown data were Java projects where files are depicted as blocks and the streets represent the directory hierarchy. The height, width, breadth, and red color gradient of the buildings are a visual encoding of the clone rate of a file. The clone rate is the fraction of tokens contained in a clone out of all tokens of a file, no matter whether the tokens were clone internally or externally. All these visual attributes show the very same attribute so that the same information can be viewed from all angles within EvoStreets. Undirected edges connect files sharing cloned code.

C. Tasks

The tasks were designed based on three main user goals from Basit et al. [58], who list common tasks in the area of clone detection. Clone detection itself is a common task in analyzing and comprehending software. The tasks, which we introduce again in the following, are also concretizations of general visual graph analyzing and comprehension tasks.

Task 1 (Figure 2a): In the first task, the participants had to count and compare the number of non-recursive connections in two independent systems A and B (A and B where totally different projects). From the participants' point of view, this task consists of (a) counting connections between red colored blocks while (in a three-dimensional space) remembering which connections have already been analyzed, (b) remembering the result for each system, and (c) deciding which contains more connections.

Task 2 (Figure 2b): In the second task, the participants had to count and compare the connections between six different subsystems of the same system (the subsystems were depicted with colored underground areas). Similar to task 1, this task consists of (a) counting the number of connections between red colored blocks located in different subsystems, (b) remembering in three-dimensional space which connections have already been counted, and (c) deciding which two areas share the most connections.

Task 3 (Figure 2c): In the final task, the participants had to determine which of the five presented subsystems (again using colored underground areas to depict each subsystem) contains the majority of the red colored blocks. This task is characterized by the following aspects: (a) counting the number of red colored blocks in each designated subsystem while (b) remembering in a three-dimensional space which blocks have already been analyzed, and (c) deciding which area contains the most red colored elements.

The rate of correctness for the three different tasks obtained in the previous experiment shows that task 3 is by far the most difficult task and task 2 is clearly the most simplest one, while task 1 is in between the two. The vast majority of participants have not found correct answers for task 3, while most correct answers were found for task 2.

D. Hypotheses

Our first research question *RQ1* refers to whether movement patterns in EvoStreets differ between the two different environments 2.5D and VR. As described in Section III-A, EvoStreets in the 2.5D environment were rendered on a regular two-dimensional display where movement is provided by means of a keyboard (*W*: forward, *S*: backward, *A*: left, *D*: right, *Q*: up, *E*: down) for translation and mouse for rotation. The VR environment, on one hand, presents the city in a stereoscopic manner (using a head-mounted display) and allows to turn the view port by rotating one's head and body, as well as to move by using a hand-held controller. These interaction concepts are slightly different to that effect that movement in VR is closer to how humans move in natural. On the other hand, many developers have a background in computer gaming, thus being familiar with the WASDQE navigation of

the 2.5D environment. In order to compare movement patterns, we processed the location and trajectory data (that were logged while running the original experiment) for each participant. Contrary to common practice, we suspect that these differences will also affect user behavior in the two environments, and in particular, movements in the VR environment are likely to be easier and faster to execute. This would mean that the participants in the VR environment move much more than in the 2.5D environment.

From the previous experiment we got access to the movement data of the participants in the investigated environments. Of course, these movement data (referred to as trajectories in the following because the participants were able to fly) is very different for each participant, yet we used them to convey fundamental differences in the movements.

All participants started at the same location (denoted by *S* in Figure 2) for each task, so that the basic movement is structured as follows: (a) Moving from start position *S* to a point with good view, (b) after a while—in the following called the *residence time*—moving to another point with good view, (c) after a *residence time* moving to another point with good view, and so on. The length of a trajectory is thus determined by sum of the distances of all consecutive viewing positions. The larger a trajectory is, the farther apart the viewing positions are or the more sight positions a participant took. We suspect that in the different environments fundamentally different viewing positions and, thus, distances are necessary. Based on this assumption we formulate our first hypothesis:

H1.T The lengths of the trajectories of the 2.5D and VR environments differ.

The trajectory length is a well suited metric to gain insights into the distances covered by the participant while exploring a city, but does not show where the participants have been in the city. For instance, someone could run back and forth many times between two points. This person would have covered a long distance, but would not have seen much of the city. The extent of the movement is closely related to finding good viewing positions. Good viewing positions, in turn, depend on the technical conditions of the display devices. Although it is difficult to predict what effect the different fields of view and the different resolutions of the display devices will have, we suspect that the volumes of the convex hulls of the trajectories—in the following called the *volume*—of both environments will be different. With regard to the volume we formulate our next hypothesis:

H1.V The volumes of the 2.5D and VR environments differ.

H1.T and *H1.V* explicitly refer only to metrics related to distances and volumes. Both provide no indication of how long the subjects stayed at their viewing positions. For this we set the trajectory length in relation to the required time to calculate the *average speed* (movement speed was the same for both environments, thus average speeds can be compared with each other). The higher the average speed, the shorter the time spent at the individual viewing positions. At the same time this means that the individual visual positions were not well suited to solve the task. In comparison between the two environments, this shows differences in the quality of the display devices that influence the behavior of the subjects.

In the previous experiment, the average speed metric, which puts the length of a trajectory in relation to the time required to fly this trajectory, has also been gathered and with regard to this metric and the assumption of *H1.T*, we postulate the following hypothesis:

H1.S The average speed of the 2.5D and VR environments differ.

In research question *RQ2*, we focus on correlations between trajectories, in particular, the metrics introduces in *H1.T*, *H1.V*, and *H1.S*, and the correctness of the given answers, that is, the number of correct and incorrect answers. In the original experiment, we could not find evidence that correctness differs significantly between the environments for any of the tasks. However, the effectiveness and quality of visual analytics may be influenced by several, yet unknown factors. Knowing whether the length of a trajectory, the convex hull spanned by a trajectory, or the average speed correlates with correctness might be beneficial for recommendation systems that are supposed to assist humans in the use of EvoStreets (or similar visualization techniques). Obvious assumptions are that participants who fly another route whose movement covers a larger area or who fly faster also have a higher correctness. This would allow a prediction of correctness based on one of the three parameters. That being said, we formulate the following hypotheses (for the 2.5D and VR environments respectively):

H2.T There is a correlation between trajectory length and the correctness of answers.

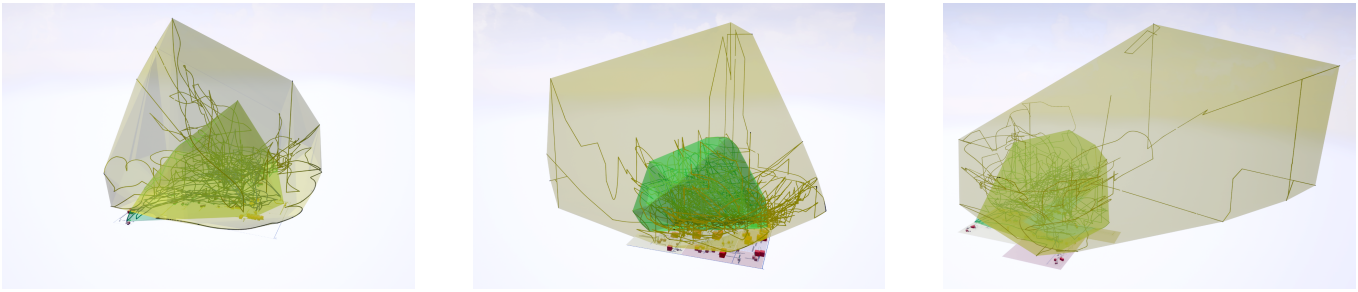
H2.V There is a correlation between the volume and the correctness of answers.

H2.S There is a correlation between the average speed and the correctness of answers.

As with the correctness of the given answers, we could not find evidence that the time required to find an answer differs significantly between the environments for any of the tasks. Nevertheless, similar to research question *RQ2*, correlations between the metrics described in *H1.T* and *H1.V* and the time required for task completion (*H1.S* already implicitly contains the time, thus a correlation is to be expected) could be a useful pointer to recommendation systems in software visualization. There is still the presumption of the hypotheses *H1.T* and *H1.V* that the participants only need a certain amount of time because they fly very far or cover a particularly large volume. In order to examine research question *RQ3*, we postulate the following hypotheses (for the 2.5D and VR environments respectively):

H3.T There is a correlation between the length of a trajectory and the time required for task completion.

H3.V There is a correlation between the volume and the time required for task completion.



(a) Volume and Trajectories of Task 1 (b) Volume and Trajectories of Task 2 (c) Volume and Trajectories of Task 3

Fig. 3: The trajectories and accumulated convex hull of each task (cf. Figure 2) and environment. The convex hull of the 2.5D environment is shown in gold and the VR environment in green.

IV. Results

While running the experiment, we gathered the location data of each participant and environment, and aggregated these data to movement trajectories. Further, we calculated the *average speed* (which puts the length of a trajectory in relation to the time required to "fly" along this trajectory) and found indications that these metrics differ between the 2.5D and VR environments. In addition to that, we calculated the convex hulls spanned by the trajectories (as well as their volumes) and investigated the results relating to trajectories in more detail for this paper. Pursuing our hypotheses from Section III-D, we present and discuss our findings below.

A. Trajectory Length (H1.T)

Figure 4 depicts the trajectory lengths for each task and environment as boxplots. While the median values of the 2.5D and VR environments do not differ much for task 1, a clear difference is visible for task 2 and 3. However, using the Mann-Whitney-Wilcoxon test to validate the results, only task 2 shows significance (p-values are: task 1: 0.3, task 2: 0.004, task 3: 0.1). Accordingly, we can accept *H1.T* only for task 2 and must reject it for task 1 and task 3.

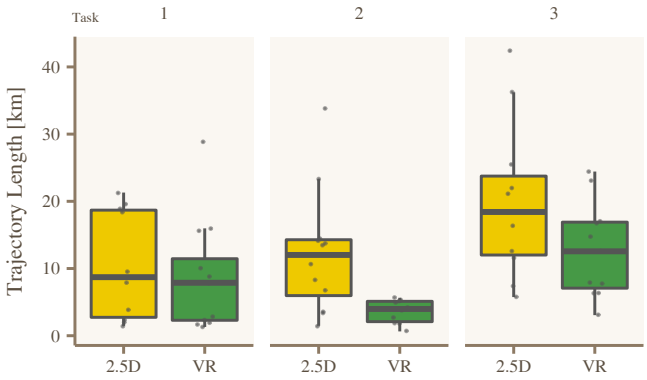


Fig. 4: Trajectory lengths for all tasks and environments.

Thus, trajectory lengths between the 2.5D and VR environments are significantly different only for task 2. Because we consider task 2 as the most simplest task in the experiment and obtained significant differences for it, one theory to explain this phenomenon is that VR leads to shorter trajectories only if a task falls below a certain level of difficulty.

If the trajectory lengths are not significantly different, the sum of the distances of all consecutive view points must be similar. Accordingly, participants in task 1 and 3 were able to cope with similar distances. Whether it is more a matter of many points of view with a small distance or a few points of view with a high distance, cannot be determined by the length of a trajectory.

B. Volume (H1.V)

The trajectory length is well suited to gain insights into the distance one moves within the EvoStreets, but does not indicate how much area was actually covered. In order to get more insights into movement patterns, we first determined the convex hull of each trajectory and calculated its volume afterwards (results are shown in Figure 5 and will be discussed below). To gain an impression of the overall area covered by the participants of the 2.5D and VR environments respectively, we also calculated the accumulated convex hull together with the trajectories of all participants and superimposed it onto the EvoStreets depicted in Figure 2. Figure 3 shows the results.

A visual pre-analysis leads to the assumption that the volumes differ significantly (as postulated in hypothesis *H1.V*). The actual values of the volumes are depicted by the boxplots in Figure 5. All three tasks show a noticeable outlier for the 2.5D environment (which are from different participants). To further investigate whether the volumes differ due to this single outlier, we reused the *Range of Movement* metric proposed in our original experiment. The metric is calculated for each participant by first determining the centroid (unweighted mean of the location points of a participant) and then calculating the maximum of the distances of all location points to its centroid. Technically, the *Range of Movement* metric indicates the maximum radius of the sphere in which a participant has moved. If the volume of a convex hull results primarily from single participants, then there must be a strong outlier in the *Range of Movement* metric, too. Otherwise, the convex hull is spanned by several participants.

According to the results in Figure 6, there is no such outlier for task 1 in the 2.5D environment, thus, we can conclude that the mean volume of task 1 depicted in Figure 5 does not result from the corresponding strong outlier alone. For task 2

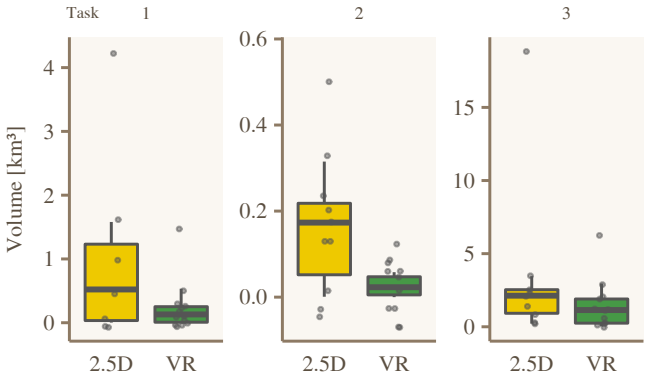


Fig. 5: Volume of convex hull.

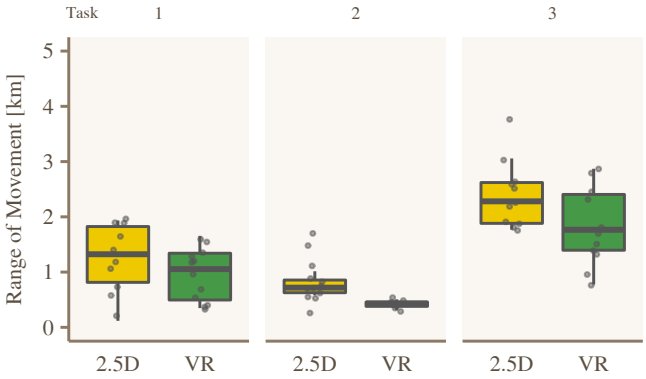


Fig. 6: The *Range of Movement* of each participant.

and task 3, two outliers for the 2.5D environment are present in Figure 6. However, (1) their deviation from the mean is not large enough to explain the single outlier in Figure 5 and (2) a single outlier in the volume of the convex hull cannot result from a single participant if two or more (relatively small) outliers are present in the *Range of Movement*. That being said, we can assume that the volumes shown in Figure 3 are spanned by several participants. We further validated the volumes with the Mann-Whitney-Wilcoxon test. The p-values are as follows: task 1: 0.07, task 2: 0.0002, task 3: 0.03. According to these results, we can accept *H1.V* for task 2 and task 3 if the level of significance is kept to 0.05. For task 1 the p-value is at least quite close to the significance threshold.

There may be different reasons why the significance level was not reached for task 1. On the one hand, task 1 was always the first task the participants had to solve. Therefore, learning effects can play a role despite the training level. On the other hand, we can not rule out that the result would not be significant by a larger number of participants. Especially in such borderline cases like this one, just one participant more or less can of course have an influence. Nevertheless, we can also ask ourselves whether processing task 1 has special requirements that were not necessary in tasks 2 and 3.

We assume that the participants try to reach good points of view as directly as possible. Therefore, the closer the points of view of a trajectory are to each other, the smaller will be the volume spanned by this trajectory. The volume of a trajectory will be considered by us as a simple measure of its compactness. Based on our results, this means that for task 2 and 3, the compactness of the trajectories in the environments are clearly differentiated. We attribute this difference to the characteristics of the output devices, for instance, the field of view, resolution, moving of the visible by turning the head, or other aspects. We can safely exclude that the viewport caused any difference because it was the same in all environments.

C. Average Speed (H1.S)

Figure 7 shows that the average speed in the 2.5D environment is generally higher than in the VR environment. An interesting observation is that the VR environment has many more outliers in all environments compared to the 2.5D environment. According to the Mann-Whitney-Wilcoxon test (p-values are as follows: T1: 0.2, T2: 0.001, T3: 0.01) the differences are significant for Task 2 and Task 3.

The trajectory length in conjunction with the time gives indirect information about how long the subjects stayed at certain points of view. All three tasks could not be solved, if you stand in only one place and look around—the participants had to move and, even if there was no time pressure, they will not delay task resolution consciously. In this respect, the path length indirectly also provides information about the number of changes of location, for example, in order to obtain a better or different view of the scene.

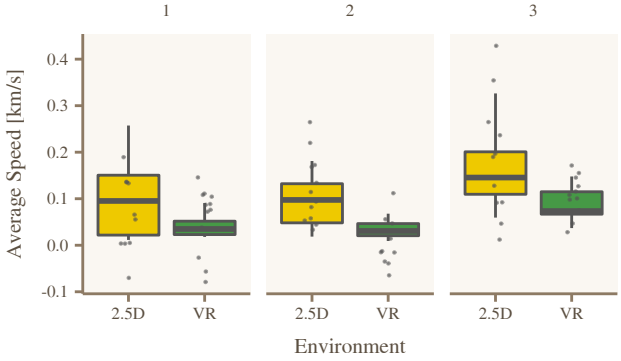


Fig. 7: Average speed.

The higher the speed, the shorter the residence time at the viewpoints. The residence times in tasks 2 and 3 differ significantly depending on the environment used. We suspect that the viewpoints selected by the participants in the two environments are of fundamentally different quality so that new ones have to be visited more frequently in one environment.

D. Correlation with Correctness (H2.T, H2.V, and H2.S)

We calculated three point-biserial correlation coefficients (Pearson, Spearman, and Kendall) for all tasks and environments , summarized in Tables II, III, and IV. Pearson's *cor* checks for a linear correlation, while Spearman's ρ and Kendall's τ both check for a monotonic correlation only.

Spearman's ρ tends to provide higher correlations values than Kendall's τ in general.

To our surprise, we have only one significant moderate correlation (Pearson) for the trajectory length and correctness of task 3 in the 2.5D environment (cor: 0.6516, p-value: 0.03), printed in bold in Table II. Moreover, it must be noted, too, that the significance test for Pearson's *cor* may not be trustworthy anyhow if the correlated variables are not bivariate normal. There are four more correlation values for the other two correlation measures at significance level $p = 0.1$ in Table II and three in Table III at least (shown in italic). Among the p-values, these are the ones that come closest to our significance level of 0.05. But even for all these cases, the correlation is only in between 0.4 and 0.5. Overall, there is only one significant difference with a moderate correlation. Hence, we fail to reject the null hypotheses for *H2.T*, *H2.V*, and *H2.S* and cannot claim that any of these hypotheses are true.

If there had been positive correlations between trajectory length or volume with correctness, there is a chance that a recommendation systems could detect that a human is potentially trying to give an answer too quickly by having observed his/her trajectory length or volume so far. Unfortunately, our data do not indicate that this can be done.

TABLE II: Correlations of trajectory length and correctness.

Task	Env.	Pearson		Kendall		Spearman	
		p	cor	p	τ	p	ρ
1	2.5D	0.6	0.1825	0.7	0.097 59	0.8	0.114
	VR	0.3	−0.325	0.4	−0.2289	0.4	−0.2693
2	2.5D	0.2	0.3709	*0.1*	*0.4082*	*0.1*	*0.4804*
	VR	0.9	0.035 69	1.0	0.0	1.0	0.0
3	2.5D	**0.03**	**0.6516**	*0.1*	*0.4264*	*0.1*	*0.5*
	VR	0.2	0.4391	0.2	0.3853	0.2	0.4518

TABLE III: Correlations of volume and correctness.

Task	Env.	Pearson		Kendall		Spearman	
		p	cor	p	τ	p	ρ
1	2.5D	0.6	0.225	0.6	0.1571	0.6	−0.024 48
	VR	1.0	−0.003 142	0.9	−0.020 81	0.9	−0.024 48
2	2.5D	0.2	0.4336	*0.1*	*0.4082*	*0.1*	*0.4804*
	VR	0.9	−0.039 17	0.7	−0.1217	0.7	−0.1421
3	2.5D	0.2	0.4091	0.2	0.3411	0.2	0.4
	VR	*0.1*	*0.4926*	0.2	0.3853	0.2	0.4518

E. Correlation with Time (H3.T, H3.V)

While we could not find evidence that there is a correlation between correctness and trajectory length, volume, and average speed, we found several significant moderate to strong correlations between time and trajectory length, as well as time and volume. The results for the environments are presented in Table V and Table VI (significant results at $p = 0.04$ are printed in bold). There are significant correlations for VR for tasks 1 and 3 for all correlation measures. There is a significant difference for montonic correlation of VR for trajectory length and time. There are significant differences for 2.5D for task 3 for both trajectory length and volume versus time. In addition, there is a significant linear correlation of 2.5D for task 2 with regard to trajectory length. All of these correlations are moderate to strong.

TABLE IV: Correlations of average speed and correctness.

Task	Env.	Pearson		Kendall		Spearman	
		p	cor	p	τ	p	ρ
1	2.5D	1.0	0.0	0.7	0.097 59	0.8	0.114
	VR	0.6	−0.1546	0.5	−0.1873	0.5	−0.2203
2	2.5D	0.3	0.2982	0.3	0.2598	0.3	0.3057
	VR	0.9	−0.0546	0.8	−0.060 86	0.8	−0.071 07
3	2.5D	0.5	−0.2095	0.8	−0.085 28	0.8	−0.1
	VR	1.0	−0.010 62	1.0	0.0	1.0	0.0

Task 3 is particularly interesting because there are significant correlations by all measures for both 2.5 and VR. It is also the most difficult task according to its low rate of correct answers where one would expect that the difficulty mirrors in longer and more wide-spread travel paths. As a reminder, task 3 required to count files with clones for seven subsystems.

The overall time to solve a task is the sum of the time needed to travel and the time needed to look at the data from a reached viewpoint. The significant correlations for trajectory length and volume with respect to time show that the time needed may indeed be at least partially be explained by the need to travel farther and to cover more wide-spread points of view. The relevance of this result for a recommendation system is that traveling takes a significant share of the time needed to solve a task and means should be provided to shorten that time so that a beholder can focus on the subtasks that can less easily be supported automatically. For instance, guidance could be offered where to look, a history of the travel so far could be recorded and made accessible, markers could be set as reminders and to quickly return to previous locations, general points of interest could be visualized where other people have spent time for similar tasks, and so on.

TABLE V: Correlations of trajectory length and time.

Task	Env.	Pearson		Kendall		Spearman	
		p	cor	p	τ	p	ρ
1	2.5D	0.8	−0.1111	0.9	0.066 67	0.8	0.078 79
	VR	**0.000 08**	**0.896**	**0.002**	**0.6667**	**0.001**	**0.8322**
2	2.5D	**0.02**	**0.6444**	0.2	0.303	0.2	0.3776
	VR	0.07	0.5869	**0.03**	**0.5394**	**0.04**	**0.6626**
3	2.5D	**0.008**	**0.7451**	**0.02**	**0.5636**	**0.02**	**0.6909**
	VR	**0.0001**	**0.9099**	**0.01**	**0.6**	**0.004**	**0.8182**

F. Threats to Validity

In the following, we want to point out threats that could affect validity. Some of those threats are inherited from the original experiments, others are specific to our analysis presented in this paper.

Internal validity: The ambition of an experimenter is that all dependent variables can be uniquely assigned to an independent variable. For that, the participants were randomly divided into six groups using a balanced latin-square order of the environments for these groups to ensure that demographic characteristics of the participants and the order of the tasks does not play a major role. Most of the participants already had previous experience with VR, so we can assume that the novelty effect for the VR environment should not have much impact. For minimize such effects, every participant got the chance to train the general task in a training level for both the 2.5D and the VR environment. There was no time

TABLE VI: Correlations of volume and time.

Task	Env.	Pearson		Kendall		Spearman	
		p	cor	p	τ	p	ρ
1	2.5D	0.4	−0.3153	0.8	−0.1111	0.7	−0.1667
	VR	**0.002**	**0.7922**	**0.005**	**0.6061**	**0.005**	**0.7692**
2	2.5D	0.2	0.3752	0.2	0.3333	0.2	0.4196
	VR	0.3	0.3516	0.2	0.3596	0.2	0.4438
3	2.5D	**0.04**	**0.6337**	**0.04**	**0.4909**	**0.04**	**0.6364**
	VR	**0.0009**	**0.8491**	**0.006**	**0.6364**	**0.005**	**0.8**

pressure—neither during the training nor the tasks itself. Of course VR is still closely associated with gamification and fun, so participants might spend more time in the VR environment than necessary.

By automatically recording quantitative data only by the experimental system during the experiment like the trajectories, the required time and the correctness, the influencing factor of the experimenter was reduced. Even if these data appear objective, the experimenter could still have influenced the test procedure by his or her personality, different speeches or fatigue. And even if there were no obvious dependencies between the experimenter and participants, hidden benefits or friendliness may have implicitly influenced the experiment. In addition, since all participants received the same tasks (using only different environments) and many of the participants knew each other, we cannot rule out that the participants informed each other about the test procedure and the desired results even though they were told not to do that.

The sample is relatively small and its composition is unknown, which does not allow the use of parametric statistical test. We have therefore used non-parametric tests such as the Wilcoxon test. In this way, we tried to eliminate the effects of any outliers in our results. At the same time, we are dealing with the disadvantage that these tests break down the concrete quantifiable values to rank numbers, so that the original differences of the values are ignored.

External validity: In addition to the question of the internal, the question of external validity is also of great importance: Are our results based on our sample generalizable to other contexts?

We based the experiment on clone detection issues, as it is well known that software has many clones. Accordingly, all visualizations that we have presented to participants is based on this clone information. Furthermore, we used the EvoStreets algorithm for the visualization and limited the visualization to the file level. The visualization created by the automatic algorithm is characteristic for the Java systems used, the metric displayed, the visualization level, and the color scheme used; If one of these factors is changed, the visualization also changes. Therefore, our results may refer to only the factors described—other factors might lead to different results.

Although our requirements for the participants regarding the use of the VR environment are just a few and most of the participants were somewhat familiar with VR environments, we cannot rule out that existing or missing previous experiences and the inherent game character of the VR environment influenced the results in one direction or the other. It is difficult to acquire non-university participants for experiments, especially in the field of basic research or when the personal or corporate benefit is not immediately apparent. We therefore had to resort to convenience sampling, which is often not a good profile of later users. In general, long-term on-site studies must complement controlled experiments, but this would involve more time, manpower, and financial effort for both the scientists and the companies involved.

Construct validity: We claim that there is a fundamental difference in the use of 2.5D and VR environments while solving visual comparison tasks such as visual clone detection, which cannot be explained by different fields of view or different input devices alone. Therefore, as far as possible, we have normalized the representation on the technical as well as on the visual side in order to achieve a high construct validity according to Cronbach and Meehl [59]. We assume that the time-dependent measurement of trajectories is an adequate measurement method.

V. Conclusion

We analyzed movements of participants in EvoStreets both in a 2.5D and VR environment gathered in a previous controlled experiment [7]. We introduced new measures to capture aspects of movements, namely, trajectory length, volume, and average speed, and compared these to various performance measures gathered in the experiment. We found a statistically significant difference of the trajectory length between 2.5D and VR only for one of the three tasks and a significant difference for two tasks in case of the volume (where the other task's p-value was 0.07, thus, relatively close to the our chosen threshold of significance of 0.05). Similarly, we found a significant difference for the average speed for two tasks. In addition, we searched for correlations of trajectory length, volume, and average speed with correctness. We could not find enough evidence for a correlation of these measures with correctness, but found many statistically significant moderate and strong correlations of trajectory length and volume with respect to time.

The relevance of our findings and the future directions that may be taken are as follows. Based on our results, it seems unlikely that recommendation systems could leverage trajectory length, volume, and average speed to detect whether a human is giving an answer too quickly to be correct. Yet, our data indicate that traveling in EvoStreets is a major contributor to the time needed to solve a task and looking into means to shorten this time through smart recommendations is worthwhile. More research is needed to look into differences in traveling in 2.5D and VR environments depending upon the kind of task. For some tasks, we could identify differences, for others we could not. If there are consistently no differences for a particular set of coherent tasks between 2.5D and VR, interactions and visualizations may be designed equally for both. If there is a difference, interaction and visualization design must pay attention to the peculiarities of those environments.

REFERENCES

[1] F. Steinbrückner and C. Lewerentz, "Representing development history in software cities," in *ACM International Symposium on Software Visualization*. ACM, 2010, pp. 193–202.

[2] F. Steinbrückner, "Consistent software cities: supporting comprehension of evolving software systems," Ph.D. dissertation, Brandenburgische Technische Universität Cottbus, 06 2013. [Online]. Available: https://opus4.kobv.de/opus4-btu/frontdoor/index/index/docId/1681

[3] R. Wettel and M. Lanza, "Visualizing software systems as cities," in *2007 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis*, June 2007, pp. 92–99.

[4] ——, "Codecity: 3d visualization of large-scale software," in *Companion of the 30th International Conference on Software Engineering*, ser. ICSE Companion '08. New York, NY, USA: ACM, 2008, pp. 921–922. [Online]. Available: http://doi.acm.org/10.1145/1370175.1370188

[5] ——, "Visual exploration of large-scale system evolution," in *2008 15th Working Conference on Reverse Engineering*, Oct 2008, pp. 219–228.

[6] S. S. Chance, F. Gaunet, A. C. Beall, and J. M. Loomis, "Locomotion mode affects the updating of objects encountered during travel: The contribution of vestibular and proprioceptive inputs to path integration," *Presence: Teleoper. Virtual Environ.*, vol. 7, no. 2, pp. 168–178, 1998.

[7] M. Steinbeck, R. Koschke, and M.-O. Rüdel, "Comparing the evostreet visualization technique in two- and three-dimensional environments—a controlled experiment," in *International Conference on Program Comprehension*. IEEE Computer Society Press, 2019, pp. 231–242.

[8] R. Koschke, "Software visualization in software maintenance, reverse engineering, and re-engineering: a research survey," *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 15, no. 2, pp. 87–109, 2003.

[9] R. Keller, C. M. Eckert, and P. J. Clarkson, "Matrices or node-link diagrams: which visual representation is better for visualising connectivity models?" *Information Visualization*, vol. 5, no. 1, pp. 62–76, 2006.

[10] D. H. R. Holten, "Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 741–748, Sep. 2006.

[11] B. Johnson and B. Shneiderman, "Tree-maps: A space-filling approach to the visualization of hierarchical information structures," in *Proceedings of the Conference on Visualization*. IEEE Computer Society Press, 1991, pp. 284–291. [Online]. Available: http://dl.acm.org/citation.cfm?id=949607.949654

[12] S. Carpendale and Y. Ghanam, "A survey paper on software architecture visualization," University of Calgary, Technical Report 2008-906-19, 2008.

[13] A. R. Teyseyre and M. R. Campo, "An overview of 3d software visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 1, pp. 87–105, 2009.

[14] S. Bassil and R. K. Keller, "A qualitative and quantitative evaluation of software visualization tools," in *Proceedings of the Workshop on Software Visualization*. IEEE, 2001, pp. 33–37.

[15] S. Bassi and R. K. Keller, "Software visualization tools: Survey and analysis," in *International Workshop on Program Comprehension*. IEEE, 2001, pp. 7–17.

[16] P. Caserta and O. Zendra, "Visualization of the static aspects of software: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 7, pp. 913–933, 2011.

[17] S. Diehl, Ed., *Software Visualization: International Seminar Dagstuhl Castle, Germany*, ser. Lecture Notes in Computer Science. Springer, 2001.

[18] S. Diehl, *Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software*. Springer, 2010.

[19] L. Merino, M. Ghafari, C. Anslow, and O. Nierstrasz, "A systematic literature review of software visualization evaluation," *Journal of Systems and Software*, vol. 144, pp. 165–180, 2018. [Online]. Available: https://doi.org/10.1016/j.jss.2018.06.027

[20] T. Bladh, D. A. Carr, and M. Kljun, "The effect of animated transitions on user navigation in 3d tree-maps," in *Ninth International Conference on Information Visualisation (IV'05)*, July 2005, pp. 297–305.

[21] W. Scheibel, C. Weyand, and J. Döllner, "Evocells - A treemap layout algorithm for evolving tree data," in *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2018, pp. 273–280.

[22] D. H. R. Holten, "Visualization of graphs and trees for software analysis," Ph.D. dissertation, Technical University of Delft, 2009.

[23] C. Knight and M. Munro, "Virtual but visible software," in *Information Visualization, 2000. Proceedings. IEEE International Conference on*. IEEE, 2000, pp. 198–205.

[24] N. Capece, U. Erra, S. Romano, and G. Scanniello, "Visualising a software system as a city through virtual reality," in *Augmented Reality, Virtual Reality, and Computer Graphics*, L. T. De Paolis, P. Bourdot, and A. Mongelli, Eds. Cham: Springer International Publishing, 2017, pp. 319–327.

[25] J. I. Maletic, J. Leigh, A. Marcus, and G. Dunlap, "Visualizing object-oriented software in virtual reality," in *International Workshop on Program Comprehension*, May 2001, pp. 26–35.

[26] T. Panas, R. Berrigan, and J. Grundy, "A 3d metaphor for software production visualization," in *International Conference Information Visualization*. IEEE, 2003, pp. 314–319.

[27] T. Panas, T. Epperly, D. Quinlan, A. Saebjornsen, and R. Vuduc, "Communicating software architecture using a unified single-view visualization," in *Engineering Complex Computer Systems, 2007. 12th IEEE International Conference on*. IEEE, 2007, pp. 217–228.

[28] F. Fittkau, A. Krause, and W. Hasselbring, "Exploring software cities in virtual reality," in *Working Conference on Software Visualization*. IEEE, 2015, pp. 130–134.

[29] P. Khaloo, M. Maghoumi, E. Taranta, D. Bettner, and J. Laviola, "Code park: A new 3d code visualization tool," in *Working Conference on Software Visualization*. IEEE, 2017, pp. 43–53.

[30] L. Merino, J. Fuchs, M. Blumenschein, C. Anslow, M. Ghafari, O. Nierstrasz, M. Behrisch, and D. A. Keim, "On the impact of the medium in the effectiveness of 3d software visualizations," in *Working Conference on Software Visualization*. IEEE, 2017, pp. 11–21.

[31] A. Schreiber and M. Brüggemann, "Interactive visualization of software components with virtual reality headsets," in *Working Conference on Software Visualization*. IEEE, 2017, pp. 119–123.

[32] K. Ogami, R. G. Kula, H. Hata, T. Ishio, and K. Matsumoto, "Using high-rising cities to visualize performance in real-time," in *Working Conference on Software Visualization*. IEEE, 2017, pp. 33–42.

[33] F. Fernandes, C. S. Rodrigues, and C. Werner, "Dynamic analysis of software systems through virtual reality," in *Symposium on Virtual and Augmented Reality*, Nov. 2017, pp. 331–340, in Spanish.

[34] L. Merino, M. Hess, A. Bergel, O. Nierstrasz, and D. Weiskopf, "Perfvis: Pervasive visualization in immersive augmented reality for performance awareness," in *ACM/SPEC International Conference on Performance Engineering*. ACM Press, 2019, pp. 13–16.

[35] A. Elliott, B. Peiris, and C. Parnin, "Virtual reality in software engineering: Affordances, applications, and challenges," in *IEEE/ACM International Conference on Software Engineering*, vol. 2, May 2015, pp. 547–550.

[36] D. A. Bowman, E. T. Davis, L. F. Hodges, and A. N. Badre, "Maintaining spatial orientation during travel in an immersive virtual environment," *Presence: Teleoper. Virtual Environ.*, vol. 8, no. 6, pp. 618–631, 1999. [Online]. Available: http://dx.doi.org/10.1162/105474699566521

[37] B. E. Riecke, D. W. Cunningham, and H. H. Bülthoff, "Spatial updating in virtual reality: the sufficiency of visual information," *Psychological Research*, vol. 71, no. 3, pp. 298–313, May 2007. [Online]. Available: https://doi.org/10.1007/s00426-006-0085-z

[38] J. W. Regian, W. L. Shebilske, and J. M. Monk, "Virtual reality: An instructional medium for visual-spatial tasks," *Journal of Communication*, vol. 42, no. 4, pp. 136–149, 1992. [Online]. Available: http://dx.doi.org/10.1111/j.1460-2466.1992.tb00815.x

[39] B. Sousa Santos, P. Dias, A. Pimentel, J.-W. Baggerman, C. Ferreira, S. Silva, and J. Madeira, "Head-mounted display versus desktop for 3d navigation in virtual reality: A user study," *Multimedia Tools and Applications*, vol. 41, no. 1, pp. 161–181, Jan. 2009. [Online]. Available: http://dx.doi.org/10.1007/s11042-008-0223-2

[40] R. A. Ruddle, S. J. Payne, and D. M. Jones, "Navigating large-scale virtual environments: what differences occur between helmet-mounted and desktop displays?" *Presence: Teleoperators & Virtual Environments*, vol. 8, no. 2, pp. 157–168, 1999.

[41] R. A. Ruddle and P. Pruch, "Effects of proprioceptive feedback and environmental characteristics on spatial learning in virtual environments," *International Journal of Human-Computer Studies*, vol. 60, no. 3, pp. 299 – 326, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1071581903001733

[42] J. Sweller, *Evolution of human cognitive architecture.* New York, NY, US: Elsevier Science, 2003, ch. Evolution of human cognitive

architecture., pp. 215–266.
[43] ——, "How the human cognitive system deals with complexity," *Handling complexity in learning environments: Theory and research*, pp. 13–25, 2006.
[44] ——, "Some cognitive processes and their consequences for the organisation and presentation of information," *Australian Journal of Psychology*, vol. 45, no. 1, pp. 1–8, 1993. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1080/00049539308259112
[45] J. L. Plass, R. Moreno, and R. Brünken, Eds., *Cognitive load theory*. Springer, New York, 2010, dFK_0237262. [Online]. Available: https://www.psychauthors.de/psychauthors/index.php?wahl=forschung&uwahl=psychauthors&uuwahl=p00830RB
[46] R. Moreno, "Cognitive load theory: more food for thought," *Instructional Science*, vol. 38, pp. 135–141, 2010.
[47] E. Kl and Z. Yildirim, "Evaluating working memory capacity and cognitive load in learning from goal based scenario centered 3d multimedia," *Procedia - Social and Behavioral Sciences*, vol. 2, pp. 4480–4486, 12 2010.
[48] A. Korbach, R. Brünken, and B. Park, "Measurement of cognitive load in multimedia learning: a comparison of different objective measures," *Instructional Science*, vol. 45, no. 4, pp. 515–536, Aug 2017. [Online]. Available: https://doi.org/10.1007/s11251-017-9413-5
[49] ——, "Differentiating different types of cognitive load: a comparison of different measures," *Educational Psychology Review*, vol. 30, no. 2, pp. 503–529, Jun 2018. [Online]. Available: https://doi.org/10.1007/s10648-017-9404-8
[50] S. van der Land, A. P. Schouten, F. Feldberg, B. van den Hooff, and M. Huysman, "Lost in space? cognitive fit and cognitive load in 3d virtual environments," *Computers in Human Behavior*, vol. 29, no. 3, pp. 1054 – 1064, 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0747563212002634
[51] L. Gerry, B. Ens, A. Drogemuller, B. Thomas, and M. Billinghurst, "Levity: A virtual reality system that responds to cognitive load," in *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems*, ser. CHI EA '18. New York, NY, USA: ACM, 2018, pp. LBW610:1–LBW610:6. [Online]. Available: http://doi.acm.org/10.1145/3170427.3188479
[52] J. Vincur, P. Navrat, and I. Polasek, "Vr city: Software analysis in virtual reality environment," in *2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, July 2017, pp. 509–516.
[53] M. Rüdel, J. Ganser, and R. Koschke, "A controlled experiment on spatial orientation in vr-based software cities," in *Working Conference on Software Visualization*, Sep. 2018, pp. 21–31.
[54] M. P. Robillard and R. J. Walker, *An Introduction to Recommendation Systems in Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 1–11. [Online]. Available: https://doi.org/10.1007/978-3-642-45135-5_1
[55] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender Systems: An Introduction*. Springer, 01 2010, vol. 24.
[56] *RecSys '07: Proceedings of the 2007 ACM Conference on Recommender Systems. Foreword*. New York, NY, USA: ACM, 2007, 609075.
[57] L. Merino, A. Bergel, and O. Nierstrasz, "Overcoming issues of 3d software visualization through immersive augmented reality1," in *Working Conference on Software Visualization*. IEEE Computer Society Press, 2018, pp. 54–64.
[58] H. A. Basit, M. Hammad, S. Jarzabek, and R. Koschke, "What do we need to know about clones? deriving information needs from user goals," in *International Workshop on Software Clones*. IEEE Computer Society Press, Mar. 2015, pp. 51–57.
[59] L. J. Cronbach and P. Meehl, "Construct validity in psychological tests," *Psychological Bulletin*, vol. 52, no. 4, pp. 281–302, 1955.