Scientific Visualization, 2021, volume 13, number 2, pages 50 - 66, DOI: 10.26583/sv.13.2.04

Visual Analytics of Gaze Tracks in Virtual Reality Environment

K.V. Ryabinin[1,A,B], K.I. Belousov[2,A,B]

[A] Saint Petersburg State University, Saint Petersburg, Russia
[B] Perm State University, Perm, Russia

[1] ORCID: 0000-0002-8353-7641, kostya.ryabinin@gmail.com
[2] ORCID: 0000-0003-4447-1288, belousovki@gmail.com

Abstract

The paper is devoted to the development of software tools to support eye-tracking-based research in an immersive virtual reality environment. Eye tracking is a popular technology for studying human behavior because it provides objective metrics to estimate human perception strategies. The corresponding hardware evolves rapidly, and nowadays its ergonomics and accessibility enable to use this hardware in a wide range of research. Recently, eye tracking devices are combined with head-mounted virtual reality displays, which allows the detecting of virtual objects the user is looking at. This, in turn, opens three new development roads. First, new interaction methods emerge, when the user can select objects with a gaze. Second, new ways of presenting virtual reality become possible, like, for example, foveated rendering (graphics rendering optimization that locates by eye tracker the zone the user is looking at, increases the image quality in that zone, and decreases the image quality in the peripheral vision). Third, new opportunities emerge to carry out the eye-tracking-based research of human behavior, wherein the spectrum of possible experiments increases dramatically compared to what is achievable in the real world. In this paper, we focus on the third road.

While there is a lot of mature software to support traditional eye-tracking-based experiments, virtual reality brings new challenges not yet tackled by the existing means. The main challenge is a seamless integration of eye tracking analytics tools with virtual reality engines. In the present work, we address this challenge by proposing a flexible data mining and visual analytics pipeline based on the ontology-driven platform SciVi that deeply integrates with the virtual scene rendered by Unreal Engine and displayed by HTC Vive Pro Eye head-mounted display.

We are interested in using eye tracking to study the reading process in immersive virtual reality. While the reading process in normal conditions is studied quite well, there is a lack of corresponding research related to the virtual reality environment. To the best of our knowledge, currently, just one attempt is reported in the literature, considering the reading of short phrases. In contrast, we plan to examine the reading of complete texts. The aim of the present work is to develop software tools needed to support the eye-tracking-based reading experiments in virtual reality and to obtain preliminary results.

To enable the visual mining of eye tracking data obtained in the reading experiments, we propose a new modification of a well-known radial transition graph that allows visually inspecting the scanpaths (sequences of eye fixations – moments when eyes are stationary – and interchanging saccades – moments when eyes rapidly move between viewing positions). Our modification is based on the SciVi::CGraph visualization module that performs well on handling large graphs and provides advanced search and filtering capabilities. The distinctiveness of our modification is the efficient tackling of the so-called “hairball problem” (problem of visual mess in the image that appears due to the big amount of data displayed at once) when inspecting fixations on the big number of interest areas (areas the gaze is tracked within). This tackling is leveraged by two main features. The first one is the concise yet comprehensive representation of interest areas as graph nodes color-coded according to the fixation count, while the dwell time is depicted by a radial histogram on top of the nodes. The second one is advanced filtering with the re-tracing function, which allows removing short intermediate fixations and merging corresponding saccades thereby enabling analysts to focus on the most significant parts of the scanpath studied.

The above features make it possible to study the reading process of the text on the word level (when each word is an individual area of interest). Currently, we implemented and tested the setup for the experiments. This setup includes an appropriate virtual reality scene and a particular visual analytics pipeline. Next, we plan to extend our pipeline with other eye tracking metrics and conduct the reading experiments.

Keywords: Visual Analytics, Data Mining, Eye Tracking, Virtual Reality, Circular Graph, Reading, Ontology Engineering, HTC Vive Pro Eye, Unreal Engine.

1. Introduction

Eye gaze tracking is a powerful technique to study human perception mechanisms [1], as well as to provide a new channel of human-machine interaction [2, 3]. Emerging technical solutions, including affordable hardware like the products of Tobii engineering company [4], and flexible software [5] bring eye tracking to a wide audience and leverage corresponding interdisciplinary research in many different application domains. This research focuses on three different aspects:

1. Study of human cognitive processes and ways humans perceive the information [1]. This research relates to Digital Humanities, involving knowledge of psychology, neurobiology, medicine, and, if it comes to study reading-based processes, of linguistics.
2. Study of ergonomics [6]. This research relies on the theory of design, involving marketing (when things like shopping or advertisements are studied) and artistic principles (when it comes to studying image-based content like, for example, graphical user interfaces).
3. Development of new human-machine interfaces [2, 3], in which eye gaze is involved as an essential interaction element (for example, as an alternative to traditional mouse pointer).

Each of the above branches of eye-tracking-based research requires adequate software tools to design and conduct appropriate experiments, collect the eye tracking data and analyze them. The eye-tracking-based research is based on analyzing specific metrics, which consider so-called fixations (moments when eyes are stationary), saccades (moments when eyes rapidly move between viewing positions), and scanpaths (sequences of eye gaze fixations and interconnecting saccades in chronological order) within areas of interest (AOIs, zones, in which the informant's gaze is analyzed) [6, 7].

While traditionally eye tracking is performed in a physical environment (when the informant looks at real objects or at the pictures displayed on the monitor), the technical progress of the last few years enables combining tracking hardware with head-mounted displays (HMDs) thus bringing eye tracking in virtual reality (VR) [8].

Eye tracking introduces new capabilities in VR technologies [9], and VR allows new ways of conducting eye-tracking-based research [10, 11]. In particular, VR enables a lot of flexibility for Digital Humanities research, since it allows to create different situations for informants, in which they should explore the space around, discover information, make decisions, perform specific actions, and thereby evince different socio-cognitive features [12].

However, there is a lack of flexible data mining (DM) software to be used for analyzing eye tracking data obtained from VR scenes. The goal of the present work is to adapt the DM platform SciVi [13] for performing visual analytics of gaze tracks collected in a VR environment. This platform proved its efficiency and flexibility in our previous research [14] focused on the Digital Humanities sphere. Currently, we are interested in studying the process of reading inside the immersive VR.

2. Key Contributions

We propose an extensible ontology-driven visual analytics pipeline to study gaze tracks of users in VR scenes. The key contributions of the research conducted are the following:

1. Seamless integration of and Unreal Engine-based VR scenes with ontology-driven DM platform SciVi to collect and analyze gaze tracks.
2. Visualization component based on circular graph for displaying and analyzing the scanpaths with multiple AOIs, supporting both real-time and recording-based operation modes.
3. Setup of the experiment to study the reading process in immersive VR.

3. Related Work

3.1. Eye Tracking Research Guidelines

Eye tracking is being used for research purposes for more than a hundred years, and in the past decade, the related hardware becomes much cheaper, easier to use, and therefore accessible for a wide number of research groups in different scientific domains [15]. Currently, results obtained across the scientific community allow to formulate methodological principles of eye-tracking-based research, building a set of tools and techniques on an appropriate theoretical basis. One of the most comprehensive guidebooks is written by K. Holmqvist et al. [15] covering theoretical aspects of eye tracking, its methodology, and corresponding measures.

Z. Sharafi et al. created a detailed guide on eye tracking metrics [7] and on conducting eye tracking studies in software engineering [16]. In these guides, the authors define key points of tracking the eye gaze, processing the collected data, and performing corresponding visual analytics.

T. Blascheck et al. made a vary elaborate review of visualization methods for eye tracking data [17], categorizing these methods by analytics purposes and providing references to the papers, which describe corresponding use cases.

Eye tracking is extensively used to study the reading process [18], as one of the most fundamental cognitive processes of perceiving structured information. But while the reading in normal circumstances is studied well [19], reading in a VR environment is covered quite poorly. Since the perception of VR often differs from the perception of the real world, one may expect some distinctiveness in the reading process when the text is presented in the immersive virtual environment. In the literature, there is a gap in measuring this distinctiveness [20]. In the present work, we are interested to prepare for filling this gap by creating a visual analytics pipeline to collect and study eye motions data in fully immersive VR.

Reading pattern is represented by scanpath – sequence of eye gaze fixations and interconnecting saccades in chronological order [6, 7]. Both Z. Sharafi et al. [16] and T. Blascheck et al. [17] indicate circular transition diagrams and radial transition graphs as adequate visualization techniques to analyze scanpaths. A detailed description of these kinds of visualization is given by T. Blascheck et al. in [21, 22]. These visualization methods were successfully used by T. Blascheck et al. to study the eye movements while reading natural text and source code snippets [22], as well as by C. S. Peterson et al. in exploring the patterns of program source code reading by novice and expert developer [23].

There are indeed much more measures, metrics, and corresponding visualization methods to consider when going deeper in studying the process of reading, but in the present work, we decided to start with scanpath analysis to obtain the preliminary results.

3.2. Eye Tracking in VR

Traditionally, eye tracking in VR is used for rendering optimizations (so-called foveated rendering – the technique of increasing image quality in the area the user is looking at, while descearing image quality in the peripheral vision) and interactions, but currently, it goes far beyond these scopes [9]. VR allows to create a "highly controlled environment [...] for a more in-depth amount of information to be gathered about the actions of a subject" [8]. This is why VR dramatically enlarges the spectrum of possible experiments related to human perception, and eye tracking brings precise measurements to these experiments allowing to use different metrics to interpret experiments' results. V. Clay et al. explored "the methods and tools which can be applied in the implementation of experiments using eye tracking in VR" and reported their results in a guide-like research paper [8]. Along with this paper, A. McNamara presented a course about eye tracking in VR within a remit of SIGGRAPH Asia 2019; the course notes are available online [24].

Eye tracking process in the VR environment requires special processing algorithms, which consider the informant's ability to freely move across the virtual scene. In contrast to the traditional setups with the informant's head fixed, movements in VR increase both degrees of freedom by exploring the virtual world, and the informant's comfort. J. Llanes-Jurado et al. proposed a robust algorithm to distinct fixations and saccades, taking into account the varying head position of the informant [25]. The reference implementation of this algorithm is freely available on GitHub as an OpenSource project, so we decided to use it in our visual analytics pipeline as one of the eye tracking data preprocessing stages.

In the last few years, research reports emerge on using VR as a versatile environment for conducting different experiments, which would be complicated (or even impossible) to conduct in the real world. For example, L. M. Zhang et al. benefited the VR to recreate different streets and study the characteristics of informants' street space perception [10]. D. Sonntag et al. introduced "a virtual reality environment that provides an immersive traffic simulation designed to observe behavior and monitor relevant skills and abilities of pedestrians who may be at risk, such as elderly persons with cognitive impairments" [11]. In this research, VR allows to study risks in safe conditions. A. Skulmowski et. al. conducted VR-based experiments related to moral and social judgments based on the well-known trolley dilemma [12]. These experiments would require a complex and very expensive setup involving human-like dolls and some technically complicated trolley park if conducted in reality, so this task can be considered impossible to fulfill outside of VR.

Regarding the study of reading in VR, to the best of our knowledge, the only research work is done by J. Mirault et al., but the aim of this work was to investigate the effects of transposed words in small sentences and not the reading of complete texts [20]. Thus we can argue, eye-tracking-based text reading study in immersive VR is a novel and significant task for Digital Humanities.

3.3. Eye Tracking Hardware

Nowadays, a wide variety of eye tracking hardware is accessible for research groups, including hardware that integrates eye tracker and HMD [26]. One of the most popular integrated devices is currently HTC Vive Pro Eye that was been released in 2019. According to the comparative study of N. Stein et al., its eye tracker is not the best one in terms of latency, being outperformed by Fove and Varjo [26]. The latency of HTC Vive Pro Eye is explained by the built-in low-pass filter applied to the eye tracking signal [26, 27], so this device cannot be used to study high-frequency saccade dynamics [27]. However, the spatial accuracy of HTC Vive Pro Eye eye tracker is quite reasonable for eye-tracking-based research [27], and the corresponding HMD possesses high display resolution (1440×1600 pixels per eye, 90 Hz refresh rate, 110° field of view), enables precise head positioning (using dual-camera outer tracking), and overall ergonomics of this device is rated high [28]. Moreover, HTC provides a reliable SDK and the HMD is supported by the most popular VR engines (Unreal Engine and Unity) out of the box. This is why we decided to use HTC Vive Pro Eye in our research.

3.4. Eye Tracking Software

There is a lot of mature software to support eye-tracking-based research; the most popular tools are reviewed by B. Farnsworth [5] and Z. Sharafi [16]. Typical functionality of these tools comprises collecting the eye tracking data from the appropriate hardware, classifying fixations and saccades, calculating statistical metrics of gaze tracks (like the average duration of fixations, the average frequency of saccades, etc.), measuring eye pupil dilation and constriction, as well as providing different visualization means to display gaze tracks and corresponding evaluated data [5, 16, 17].

However, when it comes to conducting eye-tracking-based research in VR, new challenges arise. To fully benefit from the opportunities of VR, eye tracking software has to consider deep integration with the virtual scene. For example, eye tracking data should be combined with head tracking data to allow the informant to freely navigate in the virtual world. Also, there should be mechanisms of retrieving individual virtual objects the informant is looking at, along with the hitpoints of eye gaze ray with these objects. This kind of data can significantly increase the number of observations during the experiments.

The traditional software provides no such functions yet, so the researchers have to build custom solutions out of the tools they can program themselves or find in the Internet. Most recent papers on the experiments involving eye tracking in VR provide descriptions of custom pipelines composed from loosely coupled heterogeneous third-party software tools with the data converted and transferred manually [20, 29]. Special IT skills are required to manage such pipelines, so higher-level software solutions are demanded to make VR-based eye tracking accessible for the wider scientific community.

One of the first attempts of creating this kind of solid software is proposed by J. Iacobi [30]. Iacobi's system is based on Unity graphics rendering engine and provides analytical functions tightly bound to the 3D content that can be created using the Unity level designer. Although this system is very promising, for now, it is rather a laboratory prototype. Moreover, it is tied to Unity and cannot be ported to another engine, because it is written in C#.

It can be concluded that the development of high-level flexible tools for designing and conducting eye-tracking-based experiments in immersive VR is an important and challenging task. Addressing this challenge, we propose an ontology-driven extensible software platform that enables seamless integration of DM tools with VR rendering engines and eye-tracking hardware. The distinctive features of our platform are high-level adaptation means based on ontologies and automation of data flow. This platform provides automated integration mechanisms to combine different software modules (self-written and third-party ones) into the solid DM pipeline.

4. Background

Previously, we proposed tackling configurability problems of visual analytics software by the methods and means of ontology engineering [13]. The idea is to describe the functionality of a visual analytics system by ontologies, which enable flexibility, extensibility, and semantic power. An ontology-driven visual analytics system can be adapted to solve new DM tasks by extending the system's ontological knowledge base, while the codebase of the system's core stays untouched. Built-in ontology reasoner traverses ontologies in the system's knowledge base and dynamically constructs a set of data processing and visualization operators described by these ontologies. Each operator's description contains a declaration of the operator's typed inputs, outputs, and settings, as well as a link to the corresponding implementation and information needed to execute the operator. Technically, operators play the role of micro-plugins for the visual analytics system, and the ontology acts as a semantic index for these plugins, defining the system's functionality.

We implemented the above principles in the ontology-driven client-server DM platform called SciVi (https://scivi.tools). The SciVi knowledge base contains ontologies of data types, filters, and visualization mechanisms, suitable to perform DM in the different application domains [31].

To enable efficient fine-tuning of SciVi for solving particular DM tasks, we propose describing concrete DM pipeline by data flow diagrams (DFDs) using a special high-level graphical editor. This approach proved its efficiency in different popular software like KNIME, Weka, and RapidMiner [32]. The distinctive feature of SciVi is that its operators' palette used to build DFDs is automatically constructed according to the ontologies and therefore is easily extendable.

Currently, the repository of SciVi plugins already contains a number of data processing filters and visual analytics tools. One of them is a circular graph (called SciVi::CGraph) suitable to analyze interconnected categorized data, which often arise in Digital Humanities research [14].

In the present work, we extended SciVi with the operators needed to retrieve and analyze the eye tracking data collected by the head-mounted VR display. We improved SciVi::CGraph to utilize it as a radial transition graph for scanpaths visualization. We also created appropriate SciVi plugins to communicate with Unreal Engine that renders VR scenes and to receive gaze ray hitpoints with the regions of interest within these scenes. We utilize Unreal Engine to build the VR scenes because we already have some experience in using this software in Digital Humanities research [33]. In the future, we plan to support Unity as well.

5. Proposed Solution

5.1. Architecture

The proposed software solution architecture is shown in Fig. 1. The central element of this architecture is a VR engine that performs rendering of a scene, composes stereo pair, passes it to the HMD, and collects gaze tracks from the eye tracking hardware. However, the key component is the SciVi platform that is responsible for preparing the scene data and analyzing the gaze tracks. First, it passes scene data to the VR engine and thereby controls, what the VR scene consists of. Second, it constantly receives data packages containing gaze ray hitpoints with the VR objects the informant is looking at. Each package has its timestamp provided by the eye tracker, which allows precise analysis of the gaze tracking data, even if the connection suffers from network lags.

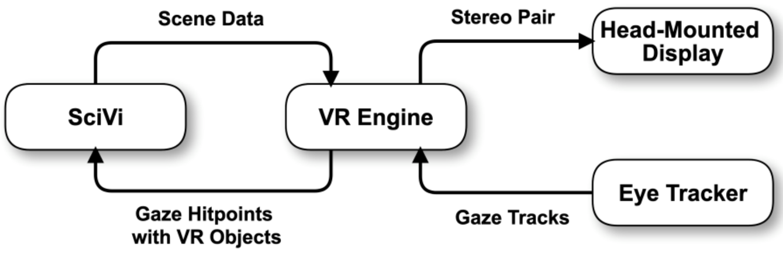


Fig. 1. The architecture of the software platform for eye-tracking-based research in immersive VR (arrows depict data links)

The communication of the VR engine with the VR hardware (eye tracker and HMD) is organized through the protocols defined by the VR engine: the modern VR engines (like Unreal Engine and Unity) are normally compatible with the modern VR headsets (like HTC Vive and Oculus Rift), so the communication relies on the drivers and SDKs provided by corresponding vendors. The communication of the SciVi platform with the VR engine is initiated and controlled by SciVi. For this, a special lightweight communication plugin should be installed in the VR engine. We propose using the WebSocket communication protocol because it enables low latency and fits well for bi-directional streaming of data.

5.2. General Setup

In our present research, we use Unreal Engine 4 to render VR scenes and HTC Vive Pro Eye to show the stereo-picture to the informants and simultaneously detect informants' eye movements. To communicate with the eye tracker, the SRanipal SDK plugin for Unreal Engine is used. The rendering is performed on the computer with the following characteristics. CPU: AMD Ryzen 9 3950X, RAM: DDR4 32Gb 3200 MHz, SSD: 512 Gb, GPU: NVidia Titan RTX, OS: Windows 10. In the future, other software and hardware can be used as well, but according to our observations, the current setup performs quite well for the needs of our research. Let us denote the machine with Unreal Engine running as the VR server.

VR server is placed in the laboratory room and connected to the laboratory local area network (LAN). HTC Vive Pro Eye is connected directly to the VR server.

The SciVi server may run anywhere in the same LAN as the VR server, but in the current setup we start it up directly on the VR server.

The experiments are directed through SciVi web interface from another computer connected to the same LAN. It may be any device capable of running an HTML5-compatible web browser. In the current setup, we use another laboratory desktop computer because it allows the operator to sit aside from the informant and thereby not distract him/her. For the informant, there is a reserved free space in the laboratory, where he/she can freely and safely move with the HMD on.

Before the start of the experiment, the informant signs a form of informed consent that states the data are collected anonymously and explains basic safety regulations of the VR immersion process. After that, the eye tracker is being calibrated for the informant. The calibration process is performed with the built-in HTC Vive Pro Eye software tools (using the standard 9-point pattern). Then, the informant has some free time in the VR scene to get used to the navigation and to check if he/she feels comfortable. Whenever ready, the informant clicks the button on the VR controller and the experiment begins. The total time of immersion should not exceed 15–20 minutes for one person to avoid fatigue.

5.3. Visual Analytics Tools

Beginning the eye-tracking-based experiments, we decided to start with the following two-staged data mining pipeline:

1. Detection of saccades and fixations in the raw data stream obtained from the eye tracker.
2. Visualizing the scanpath with the radial transition graph.

To perform the first step, we added to SciVi the detection algorithm proposed by J. Llanes-Jurado et al. [25]. While the reference implementation of J. Llanes-Jurado et al. is written in Python, to keep all the data mining process in the SciVi client, we implemented this algorithm in JavaScript.

The second step is achieved using the improved SciVi::CGraph [14] visualization module. According to Z. Sharafi et al. [16] and T. Blascheck et al. [17], circular transition diagrams and radial transition graphs suit well to visualize scanpaths.

A circular transition diagram in a form proposed by T. Blascheck et al. [21] is shown in Fig. 2a. In this diagram, AOIs are depicted as circle segments. Each segment is color-coded according to the fixation count inside the corresponding AOI, and the size of the segment indicates the total dwell time within that AOI. Transitions between AOIs are represented by arrows, with the thickness displaying the number of transitions [17]. When the number of AOIs is small, this diagram clearly shows the distribution of fixation count and fixation time, as well as interchanging saccades. However, with the increase of the AOIs number, this representation form appears messy.

Radial transition graphs in a form proposed by T. Blascheck et al. [22] are shown in Fig. 2b and Fig. 2c. The nodes placed in the circle represent color-coded AOIs, each one having incoming (white dot) and outgoing (black dot) points the arcs are connected to. Arcs represent saccades. In the graph in Fig. 2b, sector size represents total fixation duration for the corresponding AOI, so this graph variant has the same advantages and drawbacks as the circular transition diagram. In the graph in Fig. 2c, fixation duration is ignored and all the AOIs are drawn in the same size. This graph can show multiple AOIs at once, but the data about fixation time are lost.

We propose a SciVi::CGraph-based modification of the radial transition graph, which is shown in Fig. 2d. This modification aims to combine the advantages of different forms of radial transition graphs and circular transition diagrams, alleviating their drawbacks. In our graph, nodes represent AOIs, colors correspond to the count of fixation, and yellow boxes form a histogram that shows fixation durations. In this way, hundreds of AOIs can be presented, and no data about fixations are trimmed. Edges represent saccades, whereby the arrow thickness corresponds to the number of saccades. Each edge contains a timestamp, and a special filter is implemented that allows defining a time range to show saccades for.

Reusing SciVi::CGraph capabilities, edges and nodes can be filtered according to their weights (number of saccades and duration of fixations accordingly). To make sense of wight-based filtering, a re-tracing function is implemented, which recovers transitive paths after filtering: if the scanpath goes like $AOI_1 \rightarrow AOI_2 \rightarrow AOI_3$ and AOI_2 is filtered out, re-tracing adds a new edge $AOI_1 \rightarrow AOI_3$, and the weight of this edge is a sum of edges $AOI_1 \rightarrow AOI_2$ and $AOI_2 \rightarrow AOI_3$.

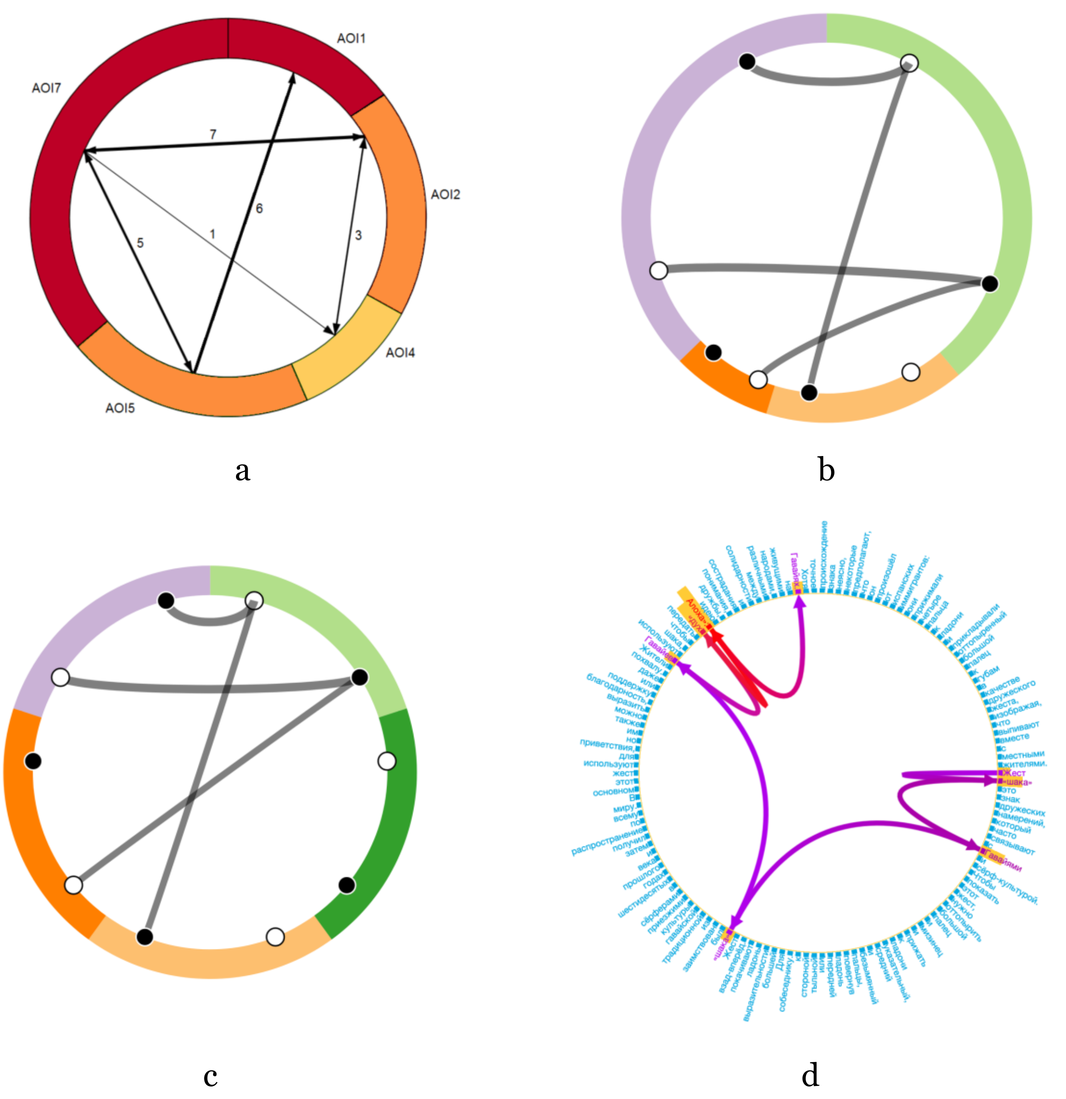


Fig. 2. Different visualizations of scanpath: circular transition diagram (a) (retrieved from [17]); radial transition graph depicting fixation durations (b) (retrieved from [23]); radial transition graph depicting no fixation durations (c) (retrieved from [23]); our modification of radial transition graph based on SciVi::CGraph rendered in SciVi (d)

Compared to the well-known form of radial transition graph, our modification allows analyzing a relatively large number of AOIs in a single view, which is important by studying the text reading on the word-scale level. It must be noted, that SciVi::CGraph already contains a lot of interactive functions (advanced search and filtering, highlighting on hover, zoom and pan, etc.) [34] to tackle the so-called "hairball problem" [35] (the problem of visual mess in the picture). These functions allow analyzing eye tracking data of reading relatively large texts (several sentences long, see Section 6.3 for details) on the relatively large timescales (working with the text for several minutes).

6. Setup to Study Reading in Virtual Reality

6.1. Virtual Reality Scene

To study reading in VR, we designed a simple VR scene in the game level editor of the Unreal Engine. In this scene, there is an open space and a whiteboard model in the middle. The whiteboard model was taken from the Blendswap 3D model storage (http://www.blendswap.com/blends/view/85304). The author of this model is the Blendswap user with the nickname *gadiskhatulistiwa*, who shared this model under the terms of the Creative Commons Attribution 3.0 license.

When the experiment begins, the informant's virtual avatar is placed in front of the whiteboard, and the whiteboard is empty. The informant can freely move in the scene's open space to get used to the VR navigation. Whenever ready to read, the informant clicks the button on the VR controller. The predefined text appears on the whiteboard and the recording of the informant's gaze direction starts.

The rendering result of the VR scene is shown in Fig. 3. The text is represented as a 2D texture generated by SciVi (see Section 6.3 for details) and mapped to the plain object with a 16:9 aspect ratio. The plain object is placed on top of the whiteboard. The texture is not mapped to the whiteboard itself to make it easier to hit-test the informant's gaze ray with the object of interest.

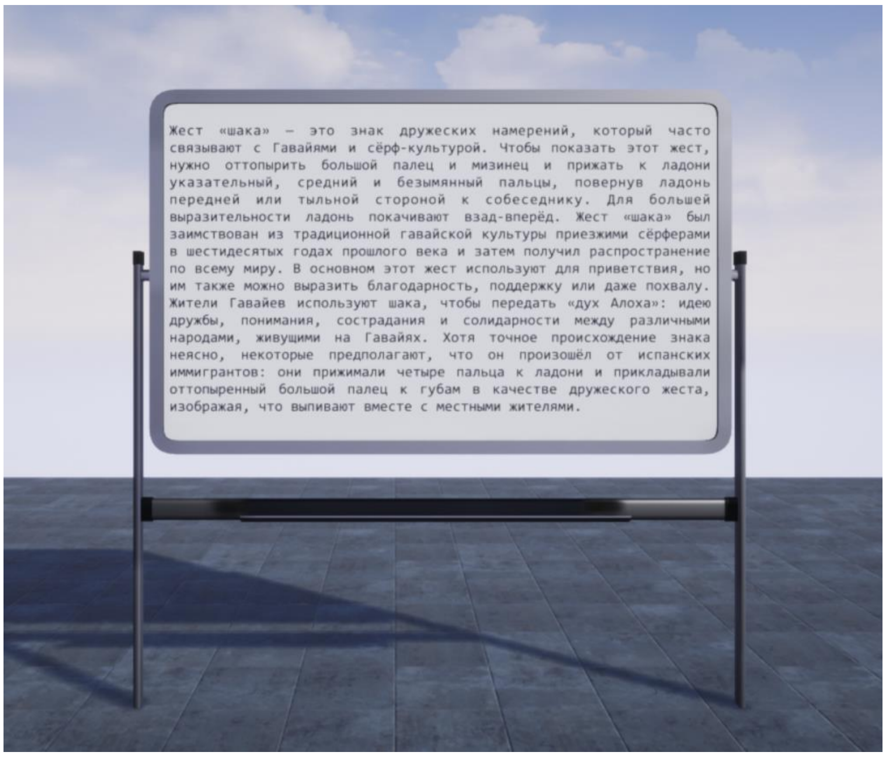

Жест «шака» – это знак дружеских намерений, который часто связывают с Гавайями и сёрф-культурой. Чтобы показать этот жест, нужно оттопырить большой палец и мизинец и прижать к ладони указательный, средний и безымянный пальцы, повернув ладонь передней или тыльной стороной к собеседнику. Для большей выразительности ладонь покачивают взад-вперёд. Жест «шака» был заимствован из традиционной гавайской культуры приезжими сёрферами в шестидесятых годах прошлого века и затем получил распространение по всему миру. В основном этот жест используют для приветствия, но им также можно выразить благодарность, поддержку или даже похвалу. Жители Гавайев используют шака, чтобы передать «дух Алоха»: идею дружбы, понимания, сострадания и солидарности между различными народами, живущими на Гавайях. Хотя точное происхождение знака неясно, некоторые предполагают, что он произошёл от испанских иммигрантов: они прижимали четыре пальца к ладони и прикладывали оттопыренный большой палец к губам в качестве дружеского жеста, изображая, что выпивают вместе с местными жителями.

Fig. 3. VR scene rendered by Unreal Engine

6.2. Visual Analytics Pipeline

The visual analytics pipeline composed as a DFD in the SciVi environment is shown in Fig. 4.

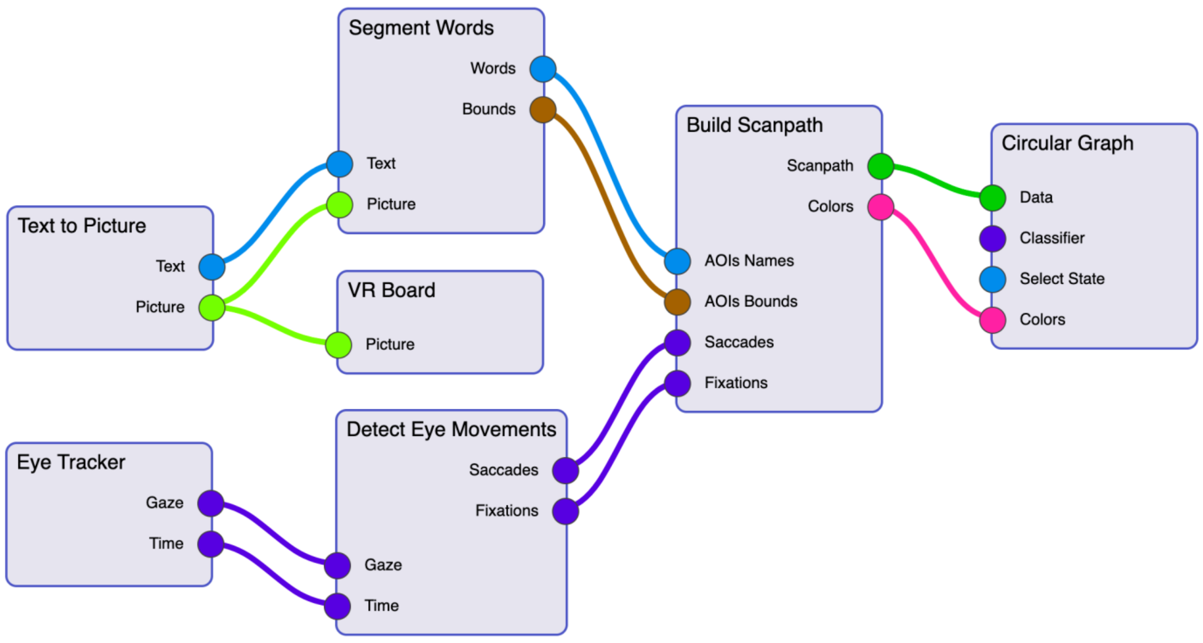


Fig. 4. SciVi DFD representing the visual analytics pipeline of eye tracking data

Each DFD node represents an individual DM operator and has its ontological description stored in the SciVi knowledge base. This description contains a list of the operator's inputs, outputs, and settings along with the link to the function or library implementing this operator. The details on how ontologies are utilized to drive the data processing and visualization can be found in our previous reports [13, 14, 31].

"Text to Picture" operator creates a raster image according to the text provided in the settings (the settings are not displayed on the DFD nodes because there is individual settings panel in the SciVi graphical user interface). "VR Board" operator transmits the input picture to the VR scene as a texture, where it is mapped to the plain object on top of the whiteboard. "Segment Words" operator utilizes a computer vision approach to find bounding rects for the words the input text consists of (see Section 6.3 for details). "Eye Tracker" operator receives gaze direction data and corresponding timestamps from the VR scene. "Detect Eye Movements" operator classifies eye tracking data to saccades and fixations (utilizing the algorithm described in [25], see Section 6.3 for details). "Build Scanpath" operator combines the eye tracking data with the AOIs data to compose a scanpath. This scanpath is then visualized using a "Circular Graph".

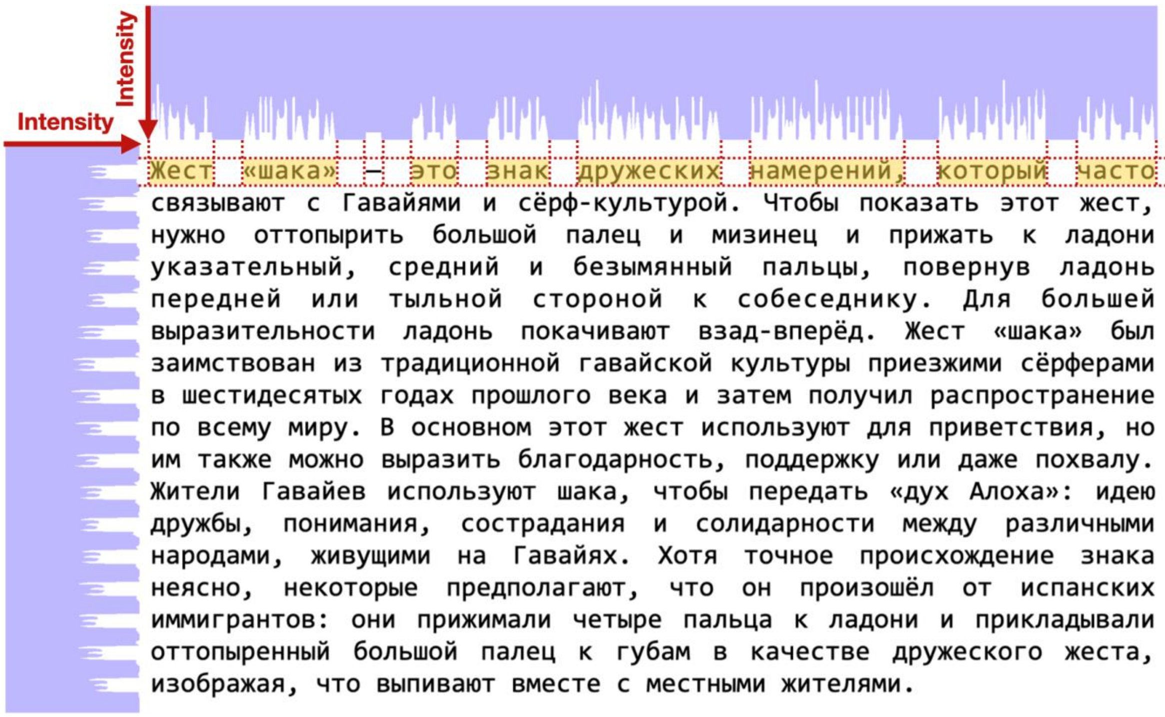
It must be noted, that the DFD shown in Fig. 4 defines real-time data processing and displaying, but the "Circular Graph" visualizer allows to save the data being collected and reload them afterwards for offline analysis.

The proposed approach of building analytics pipelines is flexible enough to handle different eye tracking DM cases. If needed, new operators can be easily added by extending SciVi ontologies, introducing new functionality to solve specific visual analytics tasks.

6.3. Text Preparation

To study reading in VR, we take texts containing several sentences, with a total length of no more than 200 words. Currently, we consider texts in Russian, and the informants are native Russian speakers. The texts contain neutral encyclopedic information about different phenomena. The example of considered texts is given in Fig. 3. In this example, a short description of the “shaka sign” (gesture of friendly intent) is given.

The texts are rasterized using HTML5 canvas API to the image of size 1920×1080, with Consolas font and justified alignment. On the one hand, this image is transmitted to the VR scene and used as a texture. On the other hand, this image is segmented to extract the precise bounding rects for individual words. The segmentation is based on the horizontal and vertical intensity histograms as proposed in [36]. The horizontal intensity histogram allows to find the bounds of lines, and then, for each line, the vertical intensity histogram allows to find borders of words. The segmentation of the first line of the text about the shaka sign is shown in Fig. 5. The words’ bounding rects are highlighted yellow; the dash is excluded because it is not a word.



Жест «шака» — это знак дружеских намерений, который часто
связывают с Гавайями и сёрф-культурой. Чтобы показать этот жест,
нужно оттопырить большой палец и мизинец и прижать к ладони
указательный, средний и безымянный пальцы, повернув ладонь
передней или тыльной стороной к собеседнику. Для большей
выразительности ладонь покачивают взад-вперёд. Жест «шака» был
заимствован из традиционной гавайской культуры приезжими сёрферами
в шестидесятых годах прошлого века и затем получил распространение
по всему миру. В основном этот жест используют для приветствия, но
им также можно выразить благодарность, поддержку или даже похвалу.
Жители Гавайев используют шака, чтобы передать «дух Алоха»: идею
дружбы, понимания, сострадания и солидарности между различными
народами, живущими на Гавайях. Хотя точное происхождение знака
неясно, некоторые предполагают, что он произошёл от испанских
иммигрантов: они прижимали четыре пальца к ладони и прикладывали
оттопыренный большой палец к губам в качестве дружеского жеста,
изображая, что выпивают вместе с местными жителями.

Fig. 5. Text segmentation based on the horizontal and vertical intensity histograms

The gaze ray obtained from the eye tracker is hit-tested with the plain object rendered with the texture containing the text. SRanipal SDK plugin provides a gaze ray hitpoint with the given object in the global scene coordinates. These data are transmitted to SciVi via WebSocket and received with the “Eye Tracker” operator. In the “Build Scanpath” SciVi operator, this point is then mapped to the texture space of the plain object and hit-tested with the word rects to find, which word the informant is looking at. The hit-testing results are assembled into the scanpath and visualized with the “Circular Graph” renderer in SciVi.

7. Conclusion

In the present work, we propose the visual analytics pipeline to perform a DM of eye tracking data in a VR environment. In particular, we discuss the setup to study the reading process of small texts (up to 200 words) in VR. To the best of our knowledge, this is a second attempt to apply eye tracking technique for studying the reading in VR. The first one has been taken by J. Mirault et al. as reported in [20], but in that work, small sentences are considered. In contrast, we are focusing on the complete texts.

We propose using the following hardware and software in the eye-tracking-based experiments:

1. HTC Vive Pro Eye VR HMD with the built-in eye tracker to present the VR scene to the informant and simultaneously capture the informant's gaze direction.
2. Unreal Engine to render the immersive VR environment. This engine supports HTC Vive HMD out of the box; to communicate with the eye tracker, the SRanipal SDK plugin is used. In the future, we plan to consider integration with the Unity engine as well.
3. SciVi DM platform to preprocess, analyze and store the data obtained from the eye tracker.

We propose to visualize the scanpaths using a circular graph leveraged by the SciVi::CGraph module. The general idea is similar to the one proposed by T. Blascheck et al. in [21, 22]. The distinctive features of our visualization tool are the following:

1. AOIs are displayed as small nodes on the circle, color-coded according to the fixation count, and the total fixation time per AOI is displayed as a radial histogram on top of the nodes.
2. The graph is supplemented with the advanced search and filtering capabilities, as well as the re-tracing functionality, which in combination allow to focus on the most significant parts of the gaze tracks being studied.

Both of the above features tackle a hairball problem when a lot of AOIs are being displayed at the same time. This enables to study the reading process of the text on the word level (when each word is an individual AOI).

Currently, we use the circular graph to visually analyze the eye tracking data. This graph allows us to estimate scanpaths in the text, as well as numbers of fixations and dwell time on each word in the text. In the future, we plan to adopt more different metrics, similar to the ones used in [37, 38].

Although the actual eye-tracking-based experiments conducted are rather preliminary, the main result of the reported work is the flexible setup that involves ontology-driven DM tools of the SciVi platform for processing and analyzing the eye tracking data collected in VR. The SciVi DM platform and all its plugins described in the paper are OpenSource projects available on GitHub: https://github.com/scivi-tools/. In particular, the ontologies (stored in the ONTOLIS ONT format [39]) describing the eye-tracking-related operators and renderers can be found under https://github.com/scivi-tools/scivi.web/tree/master/kb/eye.

Acknowledgments

This study is supported by the research grant No. ID75288744 from Saint Petersburg State University.

References

1. Rayner, K., Chace, K.H., Slattery, T.J., Ashby J. Eye Movements as Reflections of Comprehension Processes in Reading // Scientific Studies of Reading. – 2006. – Vol. 10. – PP. 241–255. DOI: 10.1207/s1532799xssr1003_3.

2. Piumsomboon, T., Lee, G., Lindeman, R.W., Billinghurst, M. Exploring Natural Eye-Gaze-Based Interaction for Immersive Virtual Reality // 2017 IEEE Symposium on 3D User Interfaces (3DUI). – 2017. – PP. 36–39. DOI: 10.1109/3DUI.2017.7893315.

3. Sidorakis, N., Koulieris, G.A., Mania, K. Binocular eye-tracking for the control of a 3D immersive multimedia user interface // 2015 IEEE 1st Workshop on Everyday Virtual Reality (WEVR). – 2015. – PP. 15–18. DOI: 10.1109/WEVR.2015.7151689.

4. The Tobii Group [Electronic Resource]. URL: https://www.tobii.com (last accessed: 07.05.2021).

5. Farnsworth, B. 10 Free Eye Tracking Software Programs [Pros and Cons] [Electronic Resource] // iMotions – 2021. URL: https://imotions.com/blog/free-eye-tracking-software/ (last accessed: 07.05.2021).

6. Poole, A., Ball, L.J. Eye Tracking in HCI and Usability Research // Encyclopedia of Human Computer Interaction. – 2006. – 9 p. DOI: 10.4018/978-1-59140-562-7.ch034.

7. Sharafi, Z., Shaffer, T., Sharif, B., Guéhéneuc, Y.-G. Eye-Tracking Metrics in Software Engineering // 2015 Asia-Pacific Software Engineering Conference (APSEC). – 2015. – PP. 96–103. DOI: 10.1109/APSEC.2015.53.

8. Clay, V., König, P., König, S.U. Eye Tracking in Virtual Reality // Journal of Eye Movement Research. – 2019. – Vol. 12, No. 1. DOI: 10.16910/jemr.12.1.3.

9. Lang, B. Eye-tracking is a Game Changer for VR That Goes Far Beyond Foveated Rendering [Electronic Resource]. – 2018. URL: https://www.roadtovr.com/why-eye-tracking-is-a-game-changer-for-vr-headsets-virtual-reality/ (last accessed 07.05.2021).

10. Zhang, L.M., Jeng, T., Zhang, R.X. Integration of Virtual Reality, 3-D Eye-Tracking, and Protocol Analysis for Re-designing Street Space // CAADRIA 2018 - 23rd International Conference on Computer-Aided Architectural Design Research in Asia. – 2018. – Vol. 1. – PP. 431–440.

11. Sonntag, D., Orlosky, J., Weber, M., Gu, Y., Sosnovsky, S., Toyama, T., Toosi, E.N. Cognitive Monitoring via Eye Tracking in Virtual Reality Pedestrian Environments // Proceedings of the 4th International Symposium on Pervasive Displays. – 2015. – PP. 269–270. DOI: 10.1145/2757710.2776816.

12. Skulmowski, A., Bunge, A., Kaspar, K., Pipa, G. Forced-Choice Decision-Making in Modified Trolley Dilemma Situations: a Virtual Reality and Eye Tracking Study // Frontiers in Behavioral Neuroscience. – 2014. – Vol. 8. DOI: 10.3389/fnbeh.2014.00426.

13. Ryabinin, K., Chuprina, S. Development of Ontology-Based Multiplatform Adaptive Scientific Visualization System // Journal of Computational Science. – 2015. – Vol. 10. – PP. 370–381. DOI: 10.1016/j.jocs.2015.03.003.

14. Ryabinin, K.V., Belousov, K.I., Chuprina, S.I. Novel Circular Graph Capabilities for Comprehensive Visual Analytics of Interconnected Data in Digital Humanities // Scientific Visualization. – 2020. – Vol. 12, No. 4. – PP. 56–70. DOI: 10.26583/sv.12.4.06.

15. Holmqvist, K., Nyström, M., Andersson, R., Dewhurst, R., Jarodzka, H., van de Weijer, J. Eye Tracking: A Comprehensive Guide to Methods and Measures. – 2011. – 560 p.

16. Sharafi, Z., Sharif, B., Guéhéneuc, Y.-G., Begel, A., Bednarik, R., Crosby, M. A Practical Guide on Conducting Eye Tracking Studies in Software Engineering // Empirical Software Engineering. – 2020. – Vol. 25. – PP. 3128–3174. DOI: 10.1007/s10664-020-09829-4.

17. Blascheck, T., Kurzhals, K., Raschke, M., Burch, M., Weiskopf, D., Ertl, T. State-of-the-Art of Visualization for Eye Tracking Data // EuroVis – STARs. – 2014. – PP. 63–82. DOI: 10.2312/eurovisstar.20141173.

18. Farnsworth, B. How We Read – What Eye Tracking Can Tell Us [Electronic Resource] // iMotions. – 2018. URL: https://imotions.com/blog/reading-eye-tracking/ (last accessed 07.05.2021).

19. Rayner, K. Eye Movements in Reading and Information Processing: 20 Years of Research. // Psychological Bulletin. – 1998. – Vol. 124, No. 3. – PP. 372–422. DOI: 10.1037/0033-2909.124.3.372.

20. Mirault, J., Guerre-Genton, A., Dufau, S., Grainger, J. Using Virtual Reality to Study Reading: An Eye-Tracking Investigation of Transposed-Word Effects // Methods in Psychology. – 2020. – Vol. 3. DOI: 10.1016/j.metip.2020.100029.

21. Blascheck, T., Raschke, M., Ertl, T. Circular Heat Map Transition Diagram // Proceedings of the 2013 Conference on Eye Tracking South Africa. – 2013. – PP. 58–61. DOI: 10.1145/2509315.2509326.

22. Blascheck, T., Sharif, B. Visually Analyzing Eye Movements on Natural Language Texts and Source Code Snippets // Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications. – 2019. - PP. 1–9. DOI: 10.1145/3314111.3319917.

23. Peterson, C.S., Saddler, J.A., Blascheck, T., Sharif, B. Visually Analyzing Students' Gaze on C++ Code Snippets // 2019 IEEE/ACM 6th International Workshop on Eye Movements in Programming (EMIP). – 2019. – PP. 18–25. DOI: 10.1109/EMIP.2019.00011.

24. McNamara, A., Jain, E. Eye Tracking and Virtual Reality // SIGGRAPH Asia 2019 Courses. – 2019. – PP. 1–33. DOI: 10.1145/3355047.3359424.

25. Llanes-Jurado, J., Marín-Morales, J., Guixeres, J., Alcañiz, M. Development and Calibration of an Eye-Tracking Fixation Identification Algorithm for Immersive Virtual Reality // Sensors. – 2020. – Vol. 20, No. 17. DOI: 10.3390/s20174956.

26. Stein, N., Niehorster, D.C., Watson, T., Steinicke, F., Rifai, K., Wahl, S., Lappe, M. A Comparison of Eye Tracking Latencies among Several Commercial Head-Mounted Displays // i-Perception. – 2021. – Vol. 12, No. 1. – PP. 1–16. DOI: 10.1177/2041669520983338.

27. Lohr, D.J., Friedman, L., Komogortsev, O.V. Evaluating the Data Quality of Eye Tracking Signals from a Virtual Reality System: Case Study using SMI's Eye-Tracking HTC Vive [Electronic Resource] // arXiv. – 2019. URL: https://arxiv.org/abs/1912.02083 (last accessed 07.05.2021).

28. Imaoka, Y., Flury, A., de Bruin, E.D. Assessing Saccadic Eye Movements With Head-Mounted Display Virtual Reality Technology // Frontiers in Psychiatry. – 2020. – Vol. 11. DOI: 10.3389/fpsyt.2020.572938.

29. Reichenberger, J., Pfaller, M., Mühlberger, A. Gaze Behavior in Social Fear Conditioning: An Eye-Tracking Study in Virtual Reality // Frontiers in Psychology. - 2020. – Vol. 11. DOI: 10.3389/fpsyg.2020.00035.

30. Iacobi, J. Software for Analyzing User Experiences in Virtual Reality using Eye Tracking [Electronic Resource] // KTH Royal Institute of Technology. – 2018. URL: https://kth.diva-portal.org/smash/get/diva2:1231972/FULLTEXT01.pdf (last accessed 07.05.2021).

31. Ryabinin, K., Chuprina, S. High-Level Toolset For Comprehensive Visual Data Analysis and Model Validation // Procedia Computer Science. – 2017. – Vol. 108. – PP. 2090–2099. DOI: 10.1016/j.procs.2017.05.050.

32. Naik, A., Samant, L. Correlation Review of Classification Algorithm Using Data Mining Tool: WEKA, Rapidminer, Tanagra, Orange and Knime // Procedia Computer Science. – 2016. – Vol. 85. – PP. 662–668. DOI: 10.1016/j.procs.2016.05.251.

33. Taleski, A. Speaker's Behavior in Virtual Reality (Methodology of the Experiment and Description of Preliminary Results) // Perm University Herald. Russian and Foreign Philology. – 2020. – Vol. 12, Iss. 4. – PP. 54–67. DOI: 10.17072/2073-6681-2020-4-54-67.

34. Ryabinin, K.V., Belousov, K.I., Chuprina, S.I., Shchebetenko, S.A., Permyakov, S.S. Visual Analytics Tools for Systematic Exploration of Multi-Parameter Data of Social Web-Based Service Users // Scientific Visualization. – 2018. – Vol. 10, No. 4. – PP. 82–99. DOI: 10.26583/sv.10.4.07.

35. Williams, D. Graph visualization: fixing data hairballs [Electronic Resource] // Cambridge Intelligence. – 2019. URL: https://cambridge-intelligence.com/how-to-fix-hairballs/ (last accessed 07.05.2021).

36. Druki, A.A. Application of Convolutional Neural Networks for Extraction and Recognition of Car Number Plates on Images with Complex Background // Bulletin of the Tomsk Polytechnic University. – 2014. – Vol. 324, No. 5. – PP. 85–92.

37. Petrova, T.E., Riekhakaynen, E.I., Bratash, V.S. An Eye-Tracking Study of Sketch Processing: Evidence From Russian // Frontiers in Psychology. – 2020. – Vol. 11. DOI: 10.3389/fpsyg.2020.00297.

38. Zubov, V.I., Petrova, T.E. Lexically or Grammatically Adapted Texts: What Is Easier to Process for Secondary School Children? // Procedia Computer Science. – 2020. – Vol. 176. - PP. 2117–2124. DOI: 10.1016/j.procs.2020.09.248.

39. Chuprina, S., Nasraoui, O. Using Ontology-based Adaptable Scientific Visualization and Cognitive Graphics Tools to Transform Traditional Information Systems into Intelligent Systems // Scientific Visualization. – 2016. – Vol. 8, No. 3. – PP. 23–44.