UNIFORM SMARTPHONE CONTROLLER FOR WEB-BASED VIRTUAL REALITY PURPOSES

Marián HUDÁK, Martin SIVÝ, Branislav SOBOTA
Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics,
Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic, tel. +421 55 602 2550,
E-mails: marian.hudak.2@tuke.sk, martin.sivy@tuke.sk, branislav.sobota@tuke.sk

ABSTRACT

This work introduces a uniform smartphone controller interface integrated into LIRKIS G-CVE web-based global collaborative virtual environments. In general, VR controllers provide various kinds of interaction techniques to manipulate virtual objects. Mostly, those aim focus on controlling the virtual context and the interaction with 3D GUI integrated in the virtual environment. With respect to web-based virtual reality, the progress in development of uniform interfaces is raising thanks to emerging web technologies and frameworks with cross-platform support. Although there are many manufacturers of VR controllers, their usage is often limited only for specified display device. Our intention is to cover multiple devices through only one simple controller interface, that is capable to provide a variety of interactions for web-based VR. In this study we proposed Enhanced Smart Client Interface designed for providing fully immersive interaction through smartphones. We performed several experiments focused on user experience and usability under two cloud platforms. Results obtained from experiments performed in our study confirm that utilization of our interface is mostly affected by the server response time. Based on the results this solution is suitable for further development and improvements.

Keywords: *virtual reality, web-based, smartphone controller, virtual environment, cross-platform.*

1. INTRODUCTION

Over the last few years, a lot of research focused on using smartphones for virtual reality purposes, mostly as screens for virtual reality helmets. Some of them were used even as input devices using a built-in gyroscope and other sensors. All these solutions use some sort of native mobile application, which needs to be installed, and controlling was performed mostly by communication over Bluetooth. This hinders the easy and fast use of smartphone as remote VR controller. In this article, we propose our solution ESCI, which is web-based, does not need to be installed, and is cross-platformed. Cross-platformed means that this approach can be used on any device which can run a web browser which provides access to device orientation and device motion data and is connected to the server. The device sends the data to the server, the server then interprets achieved data. The server can be global or local based. We prototype 3 types of control for demonstration purposes:

- Raycaster cursor interaction,
- VR Manipulator - used to simulate 3D wheel or 3D joystick for gaming purposes,
- Touch VR Joystick - 2D joystick to control movement of avatar in virtual environment.

Our solution supports multitouch, based on the device, up to 5-point multitouch. As a feedback vibration or sound is used based on the usage context. The rest of the paper is structured as follows. Firstly, we introduce the relevant state of the art. Secondly, we briefly describe the LIRKIS G-CVE and used methodology and techniques used in the implementation. Thirdly, we described the whole user testing process and experiments on two cloud platforms. In conclusion, we summarize the results and obtained experiences.

2. RELATED WORK

When using virtual reality handheld devices with indirect remote control are becoming an inseparable part of it. Using smartphones as a controller for virtual reality purposes has been a topic of several research works. Work [1] deal with using smartphones as controllers for gaming. They used smartphone sensors to track the 3D rotation, with a combination of using a touchscreen, they made it possible to control the position of a cursor in 3D space. The similarity with our work is in the focus on manipulating objects in 3D space using smartphone gyroscope and touchscreen. They also did user testing on a small group of users focused on speed, accuracy, and fatigue by performing tasks. The study [1] differs from our work in the approach, their application needs to be installed and is not web-based. We implemented 3 more sophisticated separate types of controllers and our focus was also on the virtual reality environment.

The article [2] presents a system for collaborative 3D object manipulation in a web browser, also user study on groups of users working simultaneously on 3D cooperative tasks was made. The event-driven approach was used. It is similar to our work in a web-based approach and in the experiments from a user study. The article differs from our work in approach and communication architecture. Their solution was only made in the browser and did not use some sort of custom controller as our solution with a smartphone as a dedicated input device. Their focus was made more on the collaborative skills of participants.

Another work, Lipari [3] try to integrate touch menus into smartphone-based virtual reality controllers. The screen of the smartphone offers new interaction styles and can help with virtual reality interaction when tracking is absent. In Handymenu, the smartphone screen is divided into two areas – menu interaction and other spatial interactions. Also, they tested their solution on users, which perform nested, repeated selections. Their solution needs some extra hardware such as OptiTrack and markers, which are not affordable by many people. Their solution focuses only on one type of controller and on the ration between menu areas of Handymenu. We did three separate types of controlling virtual reality objects.

The work [4] states that there is already big hardware diversity among virtual reality hardware controllers and that there is a need to develop VR applications that can be cross-platform. They wanted to create an application to maintain consistent user experience across all VR devices. Their application works on CAVETM, Oculus Rift HMD, and a mobile HMD. This work is similar to our in cross-platform control, but this work is restricted only to three specific devices and limited operating systems. Our work is truly cross-platform as long as a web browser can be used. The article [5] deals with opportunities and drawbacks of the realization of web-based smartphone remote controls. They introduce an open-source framework named ATREUS for advanced browser-based remote controls interactive applications. They present four demonstrators for remote controls. This study is similar to ours in fact that the whole solution is web-based and therefore it can be cross-platform. This study differs from ours in demonstrated controllers and in testing. They focused on direct tester's feedback and feelings rather than measured data. Questionnaires are not relevant to evaluate the systems performance. In this paper we measure time of completed tasks and server response time under two cloud platforms Glitch and Heroku based on results from testing 20 participants.

3. LIRKIS G-CVE

In 2019, we developed the LIRKIS G-CVE (Global Collaborative Virtual Environments) [6] as a web-based VR system accessible through multiple platforms and devices. Unlike other VR systems, the LIRKIS G-CVE supports real-time sharing collaborative virtual environments (CVEs) accessible through web browsers. The whole system utilizes client-server architecture with the Remote Web Server and Web-client interface.

The Remote Web Server is responsible for sharing the client's data and mediating the real-time interaction in CVEs. As described in the [6] , the server uses three JavaScript frameworks namely Node.js, Express.js, and Networked-Aframe (NAF). Their main purpose is to provide full backend services and asynchronous client-server data stream.

The Web-client interface renders the whole CVE 3D content with using the A-Frame with respect to its cross-platform support and rapid development of web-based VR. The A-Frame is based on top of HTML with utilizing Entity-Component Architecture that provide unlimited access to DOM, JavaScript, Three.js and WebGL. Each user can access the web-client interface by using web browser which supports WebGL 3D API Standard. Another feature of the Web-client interface is to provide the client´s inputs to control user movement and interaction. This functionality is available under different platforms such as Desktop Computers, VR and MR Headsets.

4. MULTIPURPOSE CONTROLLER INTERFACE

The utilization of handheld controllers is becoming popular for a variety of VR purposes. Unlike standard inputs, the handheld controllers are more convincing in immersing users through natural haptic interaction and feedback. They also offer an intuitive interface for object manipulation. However, most of VR and MR controllers (HTC Vive, Oculus, MS HoloLens) are not uninformed for use with other VR/MR devices. Similarly, the range of their functionality offers only default forms of interaction.

In spite of LIRKIS G-CVE being multiplatform, there hasn't been any uniformed controller interface for natural interaction. Our inspiration arose from mobile devices like smartphones, which are common in everyday usage and are equipped with a variety of sensors and inputs.

The first uniform smartphone controller extension for web-based VR purposes in the LIRKIS G-CVE was a *Smart-client Interface* (SCI) [7] developed in 2019. The SCI provided a multipurpose VR interface that eliminates the number of VR inputs by employing smart devices. However, this interface was focused only on user-object interaction by using the raycaster mechanism without any additional features and feedback actions. The interaction data sent by the smartphone contained a single device orientation and trigger button event. Successively, the SCI brought a standard for the design of the next interface in LIRKIS G-CVE development.

Considering this issue, we decided to design *Extended Smart-client Interface* (ESCI) shown in Fig.2 with a variety of features providing haptic and touch commands and natural feedback.

4.1. Extended Smart-client interface

The use of a smartphone as a multipurpose VR controller is advantageous because of the touchscreen presence, whose GUI can be customized according to the purpose of control. Another important factor is the employment of smartphone sensors such as accelerometer, gyroscope, or magnetometer. Our intention was to enhance the original interface of the SCI in a manner that would allow using multiple means of interaction. Henceforth, we decided to design a user-friendly GUI which would enable smart-phone users to interact intuitively. To design Extended Smart-client Interface (ESCI) in the LIRKIS G-CVE, we have selected a web-platform to relieve the user of the installation of supplementary software. The multipurpose interface ESCI designed for smartphones carries three basic techniques of interaction: Raycaster Cursor, VR Manipulator, and Touch VR Joystick.

4.1.1. Raycaster Cursor

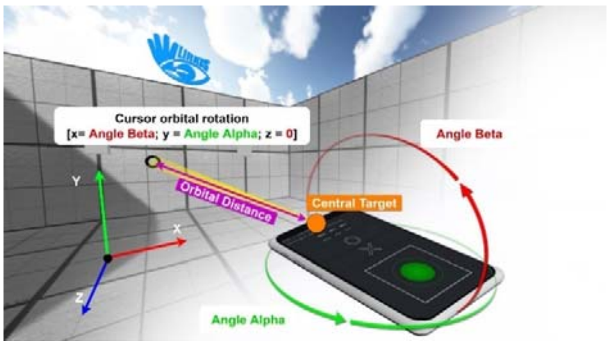
One of the primary means of interaction techniques is a virtual cursor. Using the cursor, a user can click and move virtual objects or mark various sections of the virtual environment. In general, gaze-based interaction employs this technique when the cursor centres to the user's view. In the ESCI interface, we have implemented a component known as Raycaster Cursor (Fig. 1. a), which the smartphone dynamically commands in the field of the user's view while being fully movable. As a result, the user does not have to centre their view to the object which they intend to control, rather they use smart-phone orientation to move the cursor. To allow the user to use click-events for interaction with the virtual objects, two sorts of button widgets are present in the GUI - one for confirming while the other for declining actions in the virtual environment. In the virtual environment, the cursor is placed on the orbit from the user. The orbital distance parameter is variable; however it is set to 2m distance from the central target represented by the user's right hand by default. When turning the smartphone, the cursor changes its rotation which the user can perceive as a change in the cursor position. There are two smartphone rotation angles used for cursor movement control, that is alpha and beta. Alpha affects horizontal cursor position when its translation is being changed in the x axis direction. Beta is responsible for control of vertical cursor movement in the y axis direction.

4.1.2. VR Manipulator

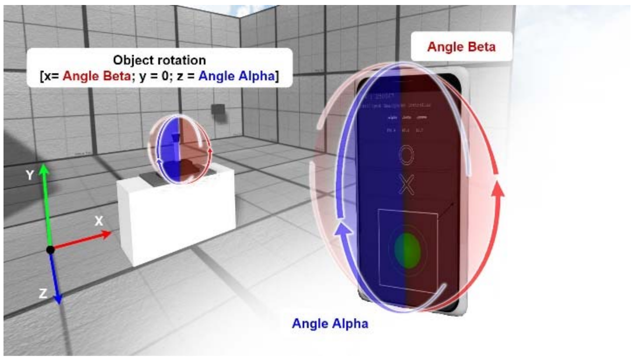
Another technique of interaction has been developed by the addition of a VR Manipulator component. This mean of interaction allows the user to manipulate surrounding objects much more precisely. It's another feature is the emulation of game devices such as 3D Joystick, Steering Wheel, or a GamePad (Fig. 1. b). Using device orientation events, it is possible to emulate object behaviour such steering, or the standard rotation and position. To work with a virtual object, two smartphone orientation angles are employed – alpha and beta. Both angles are responsible for horizontal object rotation. Those differ in axes – where alpha turns the object around the Z axis, while beta does so around the X axis.

4.1.3. Touch VR Joystick

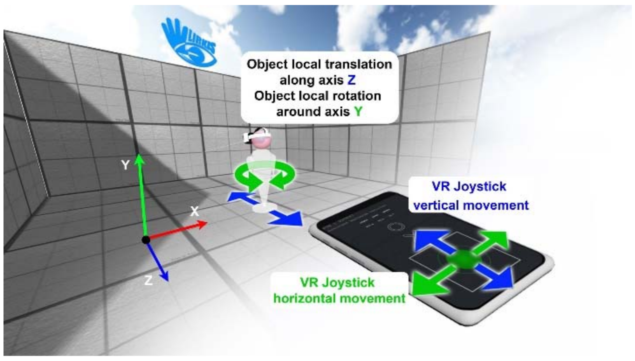
The touch VR Joystick (Fig. 1. c) presents the third way of interaction. This component allows users to interact by touching the smartphone's screen. The virtual joystick allows controlling the movement and avatar rotation in the virtual environment. Followingly, its utilization in control of some of the virtual objects is feasible. The VR Joystick has four functions to control the movement and the rotation: moving forward, moving backward, rotation to the left, and rotation to the right. Vertical movement of the Touch VR joystick changes user's position, while horizontal movement changes user's rotation to the sides.



a)



b)



c)

Fig. 1 ESCI interaction techniques:
a) Raycaster Cursor,
b) VR Manipulator,
c) Touch VR Joystick

5. FEATURES IMPLEMENTATION

The ESCI interface implementation for the LIRKIS G-CVE took place in three stages (Fig.2), where it was vital to develop smart-phone interaction, extend the Remote Web Server and Web-client components. The primary aim of the component integration was the Input Commands transmission from the ESCI to the Web-Client and vice versa gaining the feedback for the smartphone. As a consequence of the LIRKIS G-CVE support for the virtual collaboration of multiple users, it proved relevant to ensuring data transmission in such manner, that the data remained consistent and secured.

5.1. ESCI Components

In the first implementation stage, the HTML and JavaScript-based components to the ESCI were created, which allowed interaction techniques (Input Commands) as described in Section 4.1. Further components integrated were Output Feedback retrieval of the feedback such as Visual, Haptic, and Sound Feedback. This feedback allows generating output signals immersing the user during the interaction with a virtual environment.

Visual feedback guides users to handle the interface properly. Its primary task is to notify the user while working with virtual objects.

Haptic Feedback uses vibration signals in the case of object collision occurrence of the Web-Client in the virtual environment, which continuously reminds the user during the interaction.

Sound feedback notifies the user if the controller connection to the server is lost, or if the sensory data from the smartphone are unavailable.

The primary component of the ESCI is the Device PIN Generator employed in the identifier generation of the controller. This identifier is aimed at pairing the ESCI with the Web-Client for which the controller is used. After connecting the smart-phone to the ESCI, a PIN is generated by the Device PIN Generator which is consequently displayed on the smart-phone screen. Then, the user is prompted to enter the PIN code to the Web-Client interface. Web-socket communication is used to send the data from the ESCI to the Remote Web Server and then to the Web-client. The data from ESCI are encapsulated into the json objects that contain PIN, sensorial values about the device orientation, and input commands from the smartphone GUI.

5.2. Remote Web Server Extension

The second implementation stage is aimed at server functionality extension to allow it to connect ESCI with the Web-Client interface. There has been an implemented component named *Extended Smart-client Input-Output Access Control* on the server level. The primary task of this component is to transfer data bi-directionally under the PIN identifier in-between the controller and the Web-Client. To allow the transfer of these data, we have developed two sub-components: *Extended Smart-client PIN* and *Extended Smart-client data.*

The first one - *Extended Smart-client PIN* mediates PIN identifiers which provide secure connection of ESCI with the Web-Client. In the case of multiple ESCI connections, a unique PIN saved on the server-side is generated for each smartphone. The server monitors PINs that are active during the interaction. If one of the smartphones generates the PIN that is currently in use, the server notifies it and requests to generate other PIN again.

The second component - *Extended Smart-client data* ensures data transfer under the PIN of the ESCI client. Using this identifier, data received and send directly to the Web-client interface in which those are processed for needs of interaction with the virtual environment.

5.3. Web-client

The third stage showed vital to integrate components of interactions on the Web-Client side. Unlike ESCI, the Web-Client runs on an end VR/MR device which visualizes a virtual environment and acquires networked data directly from the Remote Web Server. Those data contain all information about objects and entities of users including Avatar Coordination as well as information about interaction in the virtual environment. To secure ESCI connection with Web-client interface the pairing mechanism was applied. For this reason, Device PIN verification service has been implemented on the Web-Client under which the entry data from ESCI can reach the right Web-Client, whereas the output data are transferred back to ESCI. To allow ESCI access to the virtual scene, we have implemented three Web-Client Interaction Components. Those were implemented under the A-Frame web framework as a part of its architecture Entity-Component System.

The first component is called *3D Cursor Interaction* and supports virtual object control using 3D cursor. Its primary functions are clicking and object marking in the scene by using a smartphone. From the ESCI interface, entry commands from the *Raycaster Cursor* component are gathered.

The second Web-client interface extension is represented by *3D Object Manipulation*. Its main function is position and rotation control for the object in the scene as well as 3D input control such as 3D Joystick, Steering Wheel, or GamePad. This interaction utilizes the *VR Manipulator* component on the ESCI side.

The third component – *User Movement* has been created to support user motion in the virtual environment. This interaction used *Touch VR Joystick* included in ESCI.

Besides input commands to control the virtual environment, all the components can send the events to the ESCI including visual, sound, and haptic feedback.

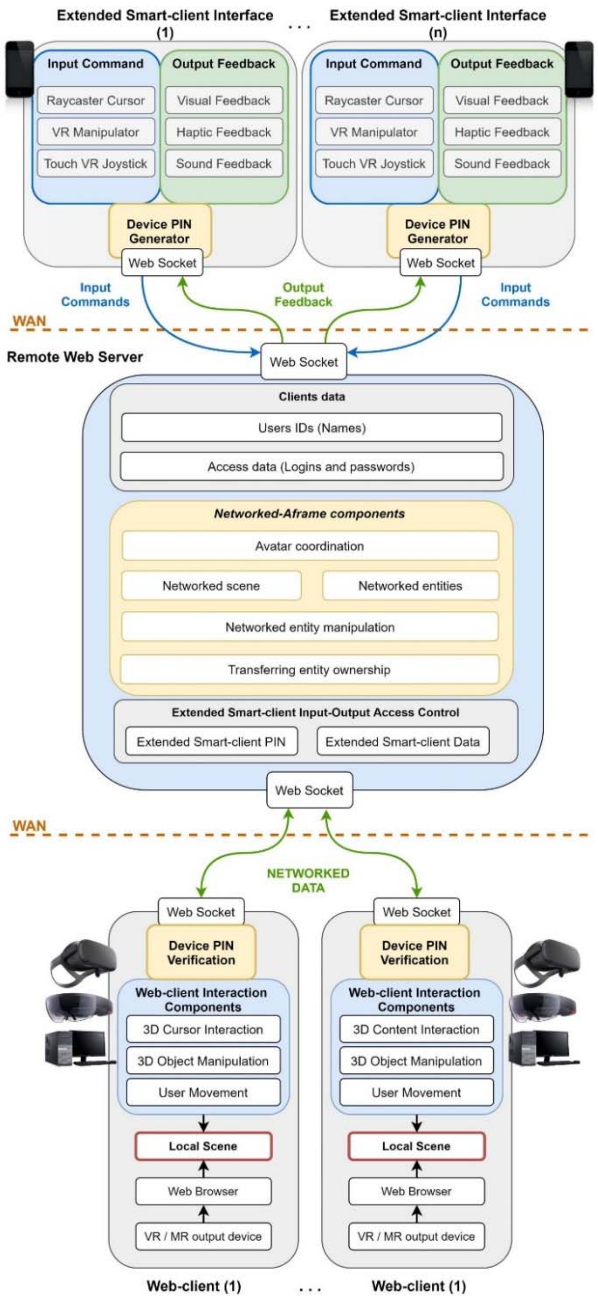


Fig. 2 The LIRKIS G-CVE architecture extension with Extended Smart-client Interface

6. EXPERIMENTS

After finalizing the implementation of ESCI in LIRKIS G-CVE, we performed several experiments focused on acquiring and processing the data from ESCI to Web-client. Because the LIRKIS G-CVE is intended for global use, all experiments were carried out under two cloud platforms, Heroku and Glitch. Both utilize the same Amazon Web Services (AWS) EC2 server instances with customizable performance and geographical location. In our experimental setup, the server has been equipped with Intel Xeon E5-2686 v4 CPU, 2.00GB RAM, and Amazon Elastic Block Store (EBS). Servers were located in Virginia, USA since that is the only location available for Glitch.

6.1. Evaluation on rendering response time under different platforms

Our intention was to compare the rendering response time of data processing between the ESCI and the Web-Client interface. The main purpose of this experiment was to evaluate time between interface input action and Web-client's visual output reaction. The testing was held with three VR/MR devices with running Web-client interface: ASUS FX504 SERIES notebook (VR), Microsoft HoloLens (1st generation) (MR) and Oculus Quest (VR). To test the ESCI the Xiaomi Redmi Note 7 smartphone was used.
The experiment was aimed to utilize two novel components: The *Raycaster Cursor* (Fig.1. a) on the ESCI side and *3D Cursor Interaction* component running on the Web-client´s device (Fig. 3). Those components were chosen to evaluate data transmission containing smartphone orientation and touch inputs over the network.

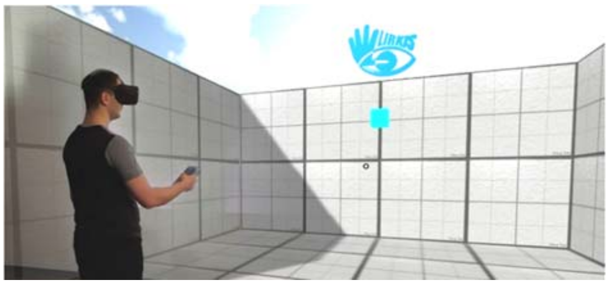
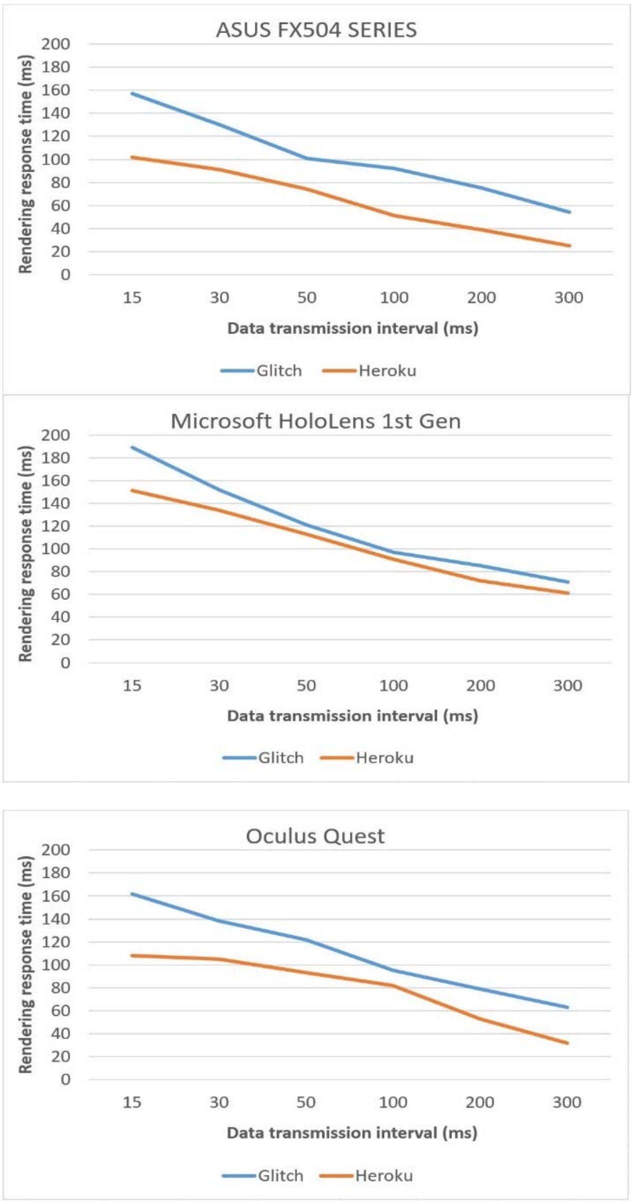


Fig. 3 Utilization of smartphone with ESCI Raycaster Cursor component and Oculus Quest with Web-client´s Interaction Component called 3D Cursor Interaction during testing

The measurements were carried out in the following steps: First, the ESCI input action was performed and the time *ti* of the input action was recorded. In the second step, the time corresponding *to* the rendering of visual response on Web-client's device was recorded. Afterward, the response rendering time has been computed as *to* - *ti*.

Considering that ESCI sends the data in periodic intervals, we decided to transmit data in 15, 30, 50, 100, 200, and 300 millisecond intervals from ESCI to Web-Client. Each measurement of rendering response time was replicated in 1000 trials under all devices and both cloud platforms. The averages of resulted response times are shown in Fig. 4.



c)

Fig. 4 The LIRKIS G-CVE Extended Smart-client Interface with comparison of rendering response time under
a) ASUS FX504,
b) Microsoft HoloLens and
c) Oculus Quest device

Based on the evaluation, we noticed that the measurements performed on both cloud platforms were significantly different. The Web-client response time under Heroku has reached shorter times compared to Glitch in each case of testing (Fig. 4.). Considering the data transmission intervals, the ASUS FX504 SERIES notebook reached excellent overall performance with the lowest rendering response time under both cloud platforms (Fig.4.a). However, during the utilization of the Microsoft HoloLens (Fig. 4. b), the highest response time in each trial was observed. Given the various interval of data transmission, we conclude the following findings: The highest rendering response time has resulted when used data transmission interval with 15 milliseconds periods. On the other hand, decreasing response times were observed with increasing data transmission interval. This observation was quintessential for the verification of the impact the response times could have on the efficiency and quality of the user interaction.

6.2. Experimental evaluation of user testing

Further testing was aimed on task completion time by a group of users with utilizing ESCI. Due to the pandemic restrictions, we were not able to perform face-to-face testing with utilizing LIRKIS Laboratory equipment such as MS HoloLens and Oculus Quest. Therefore, we decided to test user interaction remotely through their equipment. The study employed a total of 20 participants, 10 women and 10 men aged from 20 to 25. In order to keep the same geographical distance to the server, all participants were connected from the same city region in Kosice, Slovakia. All participants were connected under the same network provider. The geographical location of servers was the same as in the previous experiments.

6.2.1. User equipment

Because of the remote nature of the testing, each participant was equipped with optimal required hardware and software. The experiment setup required the following system specifications: To use the ESCI controller, the Smartphone with CPU of 1.6 GHz, 2 GB RAM, and Android operating system later than 7.0 was required. Another mandatory equipment was a gyroscope, accelerometer, and multi-touch display. To run the Web-client interface the notebook with a CPU of 1.8 GHz, 4 GB RAM, and Windows 10 operating system was expected. In order for interaction measurements to be relevant to displaying the virtual environment, participants had to use a 15-inch display with a resolution of 1920x1080. To maintain consistency of users' access through the same web browser, the Google Chrome 79.0 was required for smartphones and notebooks as necessary. All participants in this experiment met the requirements for software and hardware. Therefore, their data were considered relevant. Data transmission interval between ESCI a Web-client interface was set to transfer data each 50ms. The study [8] considers this interval borderline, however still viable for human immersion in VR controllers use.

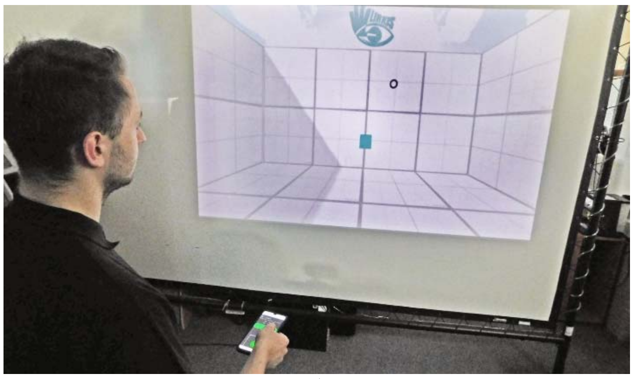
6.2.2. Testing procedures and tasks

The testing was composed of three scenarios. Each of those has validated a different type of user experience, separately for Glitch and Heroku. In each scenario, we measured task completion time to compare how the server response affects user interaction. We consider these findings to be necessary because during the virtual collaboration is expected that sharing user interactions is based on time synchronization. All scenarios were concerned to test LIRKIS G-CVE extensions of ESCI and Web-client´s Interaction components.

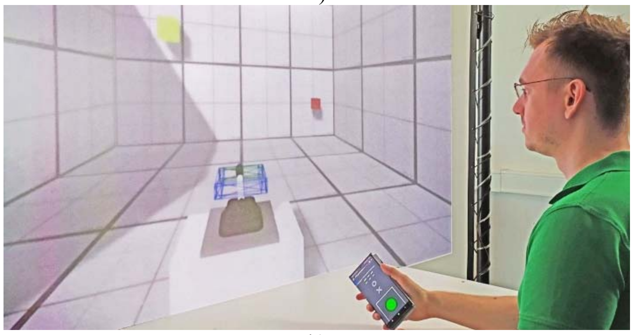
The first scenario shown in (Fig.5.a) was focused on the 3D cursor interaction. In the virtual environment, the four virtual targets (cyan boxes) were implemented. These objects were displayed in the same sequence for each user. During each step, only one target was visible, on which the user pointed the 3D cursor and clicked. After the click, the next target was displayed on which the user has pointed the cursor and clicked. This continued until the user clicked all 4 objects. The scenario was completed successfully only if the user followed all steps correctly.

In the second scenario (Fig. 5. b), a smartphone was used to manipulate a 3D object. In the virtual environment, a 3D joystick object was implemented, which was used to control the movement of the yellow box. The 3D joystick was controlled by the phone's gyroscope. The user's task was to move the yellow box to the red area until an intersection occurred. If the user moved the box correctly, the task was evaluated to be successful.

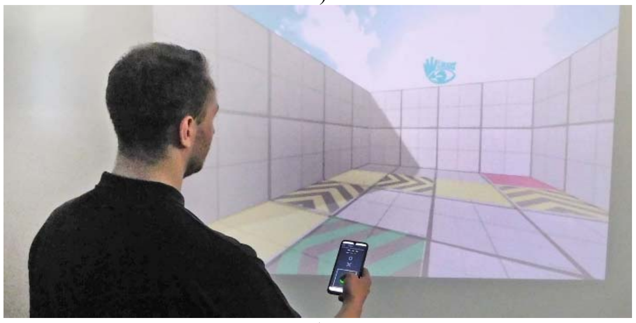
The third scenario (Fig. 5. c) was based on testing the user's movement in a virtual environment. The task for each participant was to use the VR Touch Joystick to move from the initialization point (green tile) to the destination point (red tile). The speed of movement and rotation of the user was constant. The route of movement was also preserved. Tiles are installed in the virtual scene, along which the user had to move. During the task, the time the user moved between the initialization and destination point on the scene was measured.



a)



b)



c)

Fig. 5 The experimental scenarios demonstration with utilizing ESCI:
a) 3D Cursor Interaction,
b) 3D Object Manipulation,
c) User Movement

6.2.3. Testing results

In each experiment, participants were instructed to connect a phone to the server's URL and open the ESCI interface at first. Secondly, the users opened the Web-client interface on the computer and waited for the smartphone to connect to the virtual environment. Afterwards, the users were asked to run each of the three scenarios. In each scenario, the task completion time was measured in the following steps: First, the Web-client interface asked the user to start performing the task. Then, the task start time *ts* was recorded. When the user completed the task, the task end time *te* was recorded. The final task completion time has been computed as *te* - *ts*. Both recordings were performed by the Web-client interface. After completing all tasks, the times were collected from each user separately. Finally, a comparison of average task completion times between cloud platforms was performed per scenario.

When using the Glitch and Heroku cloud platforms, it was confirmed that the server's response time significantly affected the time required to complete the task. Participants on the Heroku platform had shorter task completion time than using Glitch. As shown in Fig.6, the averages of users' task completion times were different for each scenarios and platforms. Averages include the results of all 20 users. The best average task completion time was achieved on Scenario One testing - The 3D Cursor interaction with average values of: Glitch 17,4 seconds and 16,7 seconds for Heroku. In the second scenario – 3D Object Manipulation average times were measured as: Glitch 29,1 seconds and Heroku 27,8 seconds. The third scenario - User movement showed to have the longest average task completion time, which is for Glitch 36,1 seconds and 35,5 seconds for Heroku.
For interaction speed comparison, we have considered time differences. The smallest time difference was observed in Scenario 3, where it was 0,6 second between the task performance on Glitch and later on the Heroku platform. In contrast, the longest time difference was measured in scenario 2 where it was estimated to be 1,3 second.

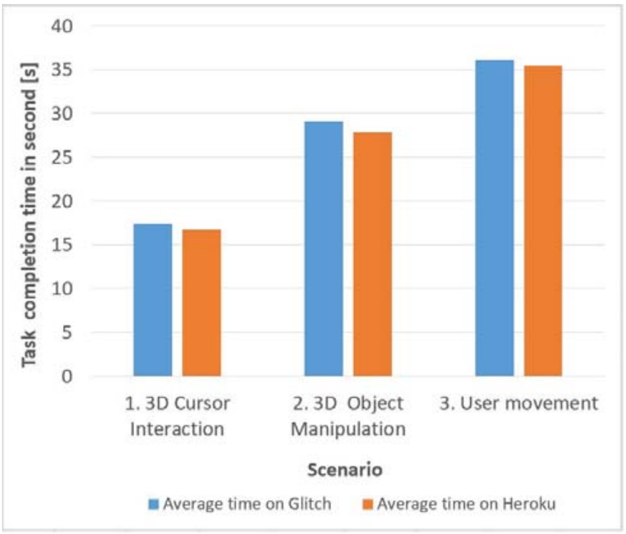


Fig. 6 The task completion time evaluation under Glitch and Heroku cloud platforms

Measurement results confirm that experiment execution on Glitch took longer to complete than on the Heroku platform. Even though with both platforms, we have used the same server performance settings as well as a client performance setting, measured differences were significantly high. Based on the result, we have concluded that use of smartphones as VR controllers is possible only if the server response time is not greater than 100ms. In the opposite case the interaction is not immersive and is confusing for the user. Since web-based VR is still in progress, we expect those experiments to be important for use of various remote controllers and smartphone applications. Different factors impacting user experience will be part of future study

7. CONCLUSION

In this work, we introduced the ESCI a uniform smartphone controller interface that enhance user interaction and access for web VR purposes. We demonstrated three typical scenarios with different cases of ESCI´s usage containing a variety of interactions. Then, we tested the ESCI usability with focusing on task completion time taken by users to perform required activity in scenarios. All participant used hardware such as smartphones and notebooks with similar specifications under the same server conditions. We verified two cloud platforms with the use of these experiments, and we found out that fluent interaction with using ESCI is possible only when the server response time is equal or less than 100ms. The server parameters of our experimental setup provided sufficient results to meet the response requirement of 100ms. We would like to test our solution with multiple-user real-time interaction in the future. We also found out that even though both cloud solutions had the same parameters and server location, the Glitch server is slower, and it had a negative effect on user interaction. In further research, we consider the deployment of ESCI in the process of motor rehabilitation training. Based on results from our study we assume that ESCI can be integrated to remote training over long distances among patients and therapists. This can be more helpful especially in cases where it is not possible to perform face-to-face training.

ACKNOWLEDGMENTS

This work has been supported by the APVV grant no. APVV-16-0202 "Enhancing cognition and motor rehabilitation using mixed reality" and by the KEGA grant No. 035TUKE-4/2019: „Virtual-reality technologies and handicapped people education “.

REFERENCES

[1] KATZAKIS, N. – HORI, M. – KIYOKAWA, K. – TAKEMURA, H.: Smart-phone game controller. In Proceedings of the 74th HIS SigVR Workshop (2011), Citeseer.

[2] DESPRAT, C. – CAUDESAYGUES, B. – LUGA, H. – JESSEL, J.-P.: Loosely coupled approach for web-based collaborative 3d design: Doctoral symposium. In Proceedings of the 11th ACM International

Conference on Distributed and Event-based Systems (2017), pp. 370–373.

[3] LIPARI, N. G. – BORST, C. W.: Handymenu: Integrating menu se-lection into a multifunction smartphone-based VR controller. In2015 IEEE Symposium on 3D User Interfaces (3DUI) (2015), IEEE, pp. 129–132.

[4] SCHLUETER, J. – BAIOTTO, H. – HOOVER, M. – KALIVARAPU, V. – EVANS, G. – WINER, E.: Best practices for cross-platform virtual reality development. In Degraded Environments: Sensing, Processing, and Display2017(2017), vol. 10197, International Society for Optics and Photonics, p.1019709.

[5] BALDAUF, M. – ADEGEYE, F. – ALT, F. – HARMS, J.: Your browser is the controller: advanced web-based smartphone remote controls for public screens. In Proceedings of the 5th ACM International Symposium on Pervasive Displays (2016), pp. 175–181.

[6] HUDÁK, M. – SIVÝ, M.: Web-based collaborative virtual environments to support cross-platform access. In: Poster 2019 International student scientific conference (2019), 178–182.

[7] KOREČKO, S. – SOBOTA, B. – HUDÁK, M.: Enhancing team interaction and cross-platform access in web-based collaborative virtual environments. In: Proceedings of 2019 IEEE 15th International Scientific Conference on Informatics (2019), 160–164.

[8] RAAEN, K. – KJELLMO, I.: Measuring latency in virtual reality systems. In: *International Conference on Entertainment Computing*. Springer, Cham, 2015. p. 457-462.

Received November 2, 2020 , accepted January 27, 2021

BIOGRAPHIES

Marián Hudák was born in 1992 in Košice, Slovakia. In 2017 he graduated (MSc) at the Department of Computers and Informatics of the Faculty of Electrical Engineering and Informatics at Technical University of Košice. He defended his master's thesis in the field of Informatics. Currently, he is a PhD student in the same department. His research is focused on web-based virtual reality interfaces, cross-platform XR development, and collaborative immersive technologies.

Martin Sivý was born in 1993 in Humenné, Slovakia. In 2017 he graduated (MSc) at the Department of Computers and Informatics of the Faculty of Electrical Engineering and Informatics at Technical University of Košice. He defended his master's thesis in the field of Informatics. Currently, he is a PhD student in the same department. His research is focused on virtual reality and smart user interfaces.

Branislav Sobota was born in 1967. In 1990 he graduated (MSc.) with honors at the Department of Computers and Informatics of the FEEI at Technical University in Košice. He obtained his PhD in 1999 and habilitation thesis in 2008 in the area of virtual reality and computer graphics. He is currently working as an associate professor at the Department of Computers and Informatics at the Technical University of Košice, Slovakia. His scientific research is focusing on computer graphics, parallel computing and especially virtual reality and related technologies.