applied sciences

MDPI

*Article*

# Experimental Performance Evaluation of Enhanced User Interaction Components for Web-Based Collaborative Extended Reality

**Štefan Korečko [1,*,†,‡], Marián Hudák [1,‡], Branislav Sobota [1,‡], Martin Sivý [1,‡], Matúš Pleva [2,*,‡] and William Steingartner [1,‡]**

[1] Department of Computers and Informatics, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, Slovakia; marian.hudak.2@tuke.sk (M.H.); branislav.sobota@tuke.sk (B.S.); martin.sivy@tuke.sk (M.S.); william.steingartner@tuke.sk (W.S.)

[2] Department of Electronics and Multimedia Communications, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Němcovej 32, 042 00 Košice, Slovakia

* Correspondence: stefan.korecko@tuke.sk (Š.K.); matus.pleva@tuke.sk (M.P.); Tel.: +421-55-602-4313 (Š.K.)

† Current address: Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, Slovakia.

‡ These authors contributed to this work, which is an extended version of the paper presented on CogInfoCom 2020 conference.

**Abstract:** COVID-19-related quarantine measures resulted in a significant increase of interest in online collaboration tools. This includes virtual reality (VR) or, in more general term, extended reality (XR) solutions. Shared XR allows for activities such as presentations, training of personnel or therapy to take place in a virtual space instead of a real one. To make online XR as accessible as possible, a significant effort has been put into the development of solutions that can run directly in web browsers. One of the most recognized solutions is the A-Frame software framework, created by Mozilla VR team and supporting most of the contemporary XR hardware. In addition, an extension called Networked-Aframe allows multiple users to share virtual environments, created using A-Frame, in real time. In this article, we introduce and experimentally evaluate three components that extend the functionality of A-Frame and Networked-Aframe. The first one extends Networked-Aframe with the ability to monitor and control users in a shared virtual scene. The second one implements six degrees of freedom motion tracking for smartphone-based VR headsets. The third one brings hand gesture support to the Microsoft HoloLens holographic computer. The evaluation was performed in a dedicated local network environment with 5, 10, 15 and 20 client computers. Each computer represented one user in a shared virtual scene. Since the experiments were carried out with and without the introduced components, the results presented here can also be regarded as a performance evaluation of A-Frame and Networked-Aframe themselves.

**Keywords:** extended reality; WebXR; performance evaluation; A-Frame; virtual collaboration



**Citation:** Korečko, Š.; Hudák, M.; Sobota, B.; Sivý, M.; Pleva, M.; Steingartner, W. Experimental Performance Evaluation of Enhanced User Interaction Components for Web-Based Collaborative Extended Reality. *Appl. Sci.* **2021**, *11*, 3811. https://doi.org/10.3390/app11093811

Academic Editor: Jozsef Katona

Received: 19 March 2021
Accepted: 20 April 2021
Published: 23 April 2021



## 1. Introduction

While online collaboration tools are available for years, their utilization has been rising significantly because of the COVID-19-related quarantine measures. This is probably most evident in education, where, according to Burgess and Sievertsen [1], teaching is moving online on an untested and unprecedented scale. When training activities are needed, the educators opt for virtual reality (VR) solutions more often than ever. For example, VR simulators are used for surgical training in ophthalmology courses [2] and Mohan et al. [3] envisioned a place for similar solutions in plastic surgeons training. Wijkmark et al. [4] used virtual simulation for firefighter training and Luimula et al. [5] indicated an immediate need for virtual hands-on training solutions in the construction industry. Similar to other application areas, extended reality applications are migrating from the desktop

environment to web browsers. Extended reality (XR) is an umbrella term, covering virtual, augmented and mixed reality. Web-based extended reality simplifies users' access through a variety of devices under different operating systems [6]. The usage of web-based XR has been increasing since 2017 when Augmented Reality (AR) and Mixed Reality (MR) were united in the same development process with VR. This resulted in WebXR Device API (https://www.w3.org/TR/webxr/, accessed on 22 April 2021) [7], a W3C specification defining access to both VR and AR devices.

One of the most recognized software frameworks for developing XR web applications is A-Frame (https://aframe.io/, accessed on 22 April 2021). It is an open-source platform, which supports most of the contemporary XR hardware. Initially created by the Mozilla VR team in 2015, its development is now led by experts from Supermedium and Google. A valuable extension of the framework is Networked-Aframe (NAF) (https://github.com/networked-aframe, accessed on 22 April 2021), which allows multiple users to share virtual environments, created using A-Frame, in real time. To extend the functionality of A-Frame and NAF and address some of their limitations, new components were proposed and evaluated by the authors. The components, together with a selection of virtual environments (scenes), constitute a platform called LIRKIS Global—Collaborative Virtual Environments (G-CVE) [8–10]. An instance of LIRKIS G-CVE, with all the components described and evaluated here, is available at https://lirkisgcve.herokuapp.com/ (accessed on 22 April 2021).

The original contribution of this paper dwells in a performance evaluation of three of the proposed components under different loads. The first proposed component, called Enhanced Client Access (ECA) [10], extends NAF with the ability to monitor and control client apps sharing a virtual scene. The second one is 6-DoF SMC (Smartphone Motion Capture) [9], which addresses the lack of Six Degrees of Freedom (6-DoF) motion tracking for smartphones. The 6-DoF motion capture is supported "out of the box" for VR headsets, such as Oculus Touch, Oculus Quest, HTC Focus HTC Vive or Windows MR-compatible devices [11]. However, it is not supported for the stereoscopic mode in smartphones, which is a significant disadvantage as interaction with less than six DoF rules out all-natural movement in XR. Smartphones offer 3-DoF capture with the rotational motion tracking but not the translational one. Thus, the users may turn but not move from place to place in a virtual scene. They need additional devices to move, such as gamepads or joysticks, which are less natural for such interaction. The third proposed component, named HoloGOM [9], deals with Microsoft (MS) HoloLens, where available web frameworks do not include access to gesture processing. Therefore, the users are limited to interact only by the clicker device and object manipulation by hand tracking is not available. HoloGOM solves the issue by supporting free-hand manipulation and tap gestures. While developed for A-Frame, the second and third component can be used for virtually any application, implementing WebXR.

All three components are incorporated into the client–server architecture of A-Frame and NAF (Figure 1) to offer a more natural user experience in shared virtual environments (scenes). A-Frame and NAF provide the virtual scene and means to share it between multiple users: a server machine (SRV in Figure 1) runs a web server application (app) that hosts all the resources for client apps and runs the server-side modules of NAF. Each user device (PC, PH or HL in Figure 1) runs an instance of a web browser with a client app that renders the shared virtual scene using A-Frame modules and client-side modules of NAF. NAF keeps track on networked entities, which represent users and shared objects in the scene. Thanks to this, a user can see avatars of other users and shared objects in the scene. ECA runs inside both the server and the client apps and allows to rise mutual awareness of users sharing the scene by providing access to their positions and other properties. 6-DoF SMC and HoloGOM run inside the client apps on the corresponding devices to translate the user movement in a real environment to the virtual one.

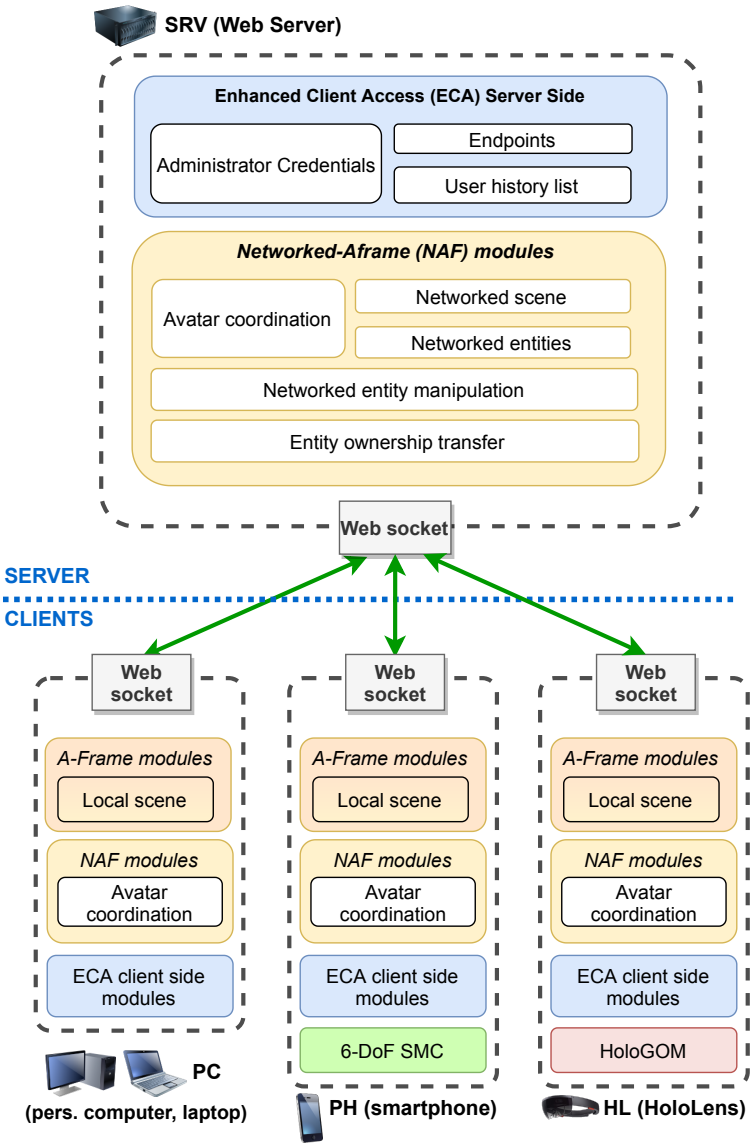



**Figure 1.** A-Frame, NAF and the proposed components in a typical configuration.

The aim of the performance evaluation of the proposed components was to find answers to the following research questions:

- RQ.1: How does the added functionality, implemented in the proposed components, affect the A-Frame and NAF performance under different loads?
- RQ.2: How do A-Frame and NAF perform under different loads on mid-range hardware and MS HoloLens?

The evaluation was carried out in the form of three sets of experiments, each focusing on one component. To eliminate the interference from internet traffic, the experiments took place in a dedicated local network environment, but with Internet connectivity available for all devices. In principle, the network environment resembled the one shown in Figure 1. Personal computers (PC), representing users in a shared virtual scene, were connected, via a switch and to a server (SRV) running the web server app. MS HoloLens (HL) and a smartphone (PH) were accessed through a WiFi router, connected to the same switch. By the different loads we mean different number of active users, sharing the scene. Each set, except of the one for the MS HoloLens component, consisted of experiments with 1, 5, 10, 15 and 20 users. Only 1, 5 and 10 users were used for MS HoloLens as the performance dropped to unacceptable levels with a higher number of users. Two performance measures

were recorded. The first one was the frame rate of the client application. The second one was the network response time between the client app and a server, hosting the virtual environment and running NAF. The mid-range hardware used were computers with Intel Core i5 CPU, 4GB RAM and integrated graphics and a Xiaomi Redmi Note 7 smartphone with Android 10. The experiments used A-Frame version 1.0.4 on personal computers and MS HoloLens and 0.9.2 on the smartphone. The NAF version was 0.6.1 on all devices.

The rest of the article is organized as follows. Section 2 lists other works related to the evaluated components and the performance evaluation. It also links LIRKIS G-CVE to the cognitive infocommunications research area. Section 3 overviews the proposed components and explains their connection to A-Frame and NAF in more detail. Experimental evaluation setup and procedures are described in Section 4. Section 5 presents results of the experiments and discussion. Findings regarding reliability of the proposed components, revealed during the experiments, are presented in Section 6. Section 6 also overviews additional experiments, dealing with the usability of the proposed components and utilizing the standardized System Usability Scale (SUS) questionnaires. The article concludes with a summary of the achieved results and plans for future research and development in Section 7.

## 2. Related Work

The topicality of the proposed components is evident from recent works that deal with similar functionality, described in Sections 2.1–2.3. What distinguishes our components from other solutions is the implementation of features that aim at different goals or the utilization of web technologies, which allow easy access to shared virtual environments.

Regarding the performance evaluation of shared XR solutions, the work presented here is, to our knowledge, unique in its scope and complexity. Thus far, this work seems to be the only study focusing on the impact of an increasing number of homogeneous clients in a shared virtual scene, implemented using WebXR. There also seem to be no other studies measuring the A-Frame performance on MS HoloLens or an impact of the ARCore-based motion tracking on the performance. The most related works with respect to the performance evaluation are listed in Section 2.4.

### *2.1. User Management in Web-Based Shared Virtual Environments*

Considering the user management in shared virtual scenes, both A-Frame and NAF were used by Scavarelli et al. [12] for their virtual learning environment platform, called Circles. Circles provides a prototypical user interface for managing virtual rooms and invitations for other users. In Circles, a user can take a role of learner, instructor or developer. However, it is not clear whether Circles extends NAF in any way, as our ECA component, or uses its default features for the desired functionality.

Another solution based on A-Frame is the work of Gunkel et al. [13,14], which focuses on enhancing the possibilities of collaboration in virtual space. Our collaboration-related functionality utilizes NAF and the ECA component, where ECA extends NAF by providing more information about connected users and by allowing to remove the users from a shared scene. On the other hand, the authors of [13,14] focused on videoconferencing. They provided photo-realistic representations of the users by means of 2D video streams [13] or 3D scans [14].

Huh et al. [15] proposed an architecture for online collaboration in XR, based on decentralized web. The communication between users is provided by WebRTC through WebSockets. As in our case, they utilized A-Frame for virtual scenes, but they did not use NAF. For scene sharing, they implemented a solution that utilizes peer-to-peer connection. Each user has a local database of objects that is shared with others. As our solution is based on NAF, it uses a client–server architecture without local databases.

### 2.2. 6-DoF Motion Tracking for Smartphones

The proposed 6-DoF SMC component, addressing the lack of 6-DoF motion tracking in smartphones, is related to the work [16] in visual-inertial fusion mechanism. The experimental platform, introduced in [16], contains a smartphone utilizing external monocular camera and IMU (Inertial Measurement Unit). The motion detection is based on two concurrent data streams: visual frames captured from camera and IMU linear acceleration data with angular velocity. In comparison to Fang et al. [16], the 6-DoF SMC component relies on a built-in smartphone camera. While the contribution of Fang et al. [16] is presented as a standard for smartphone AR/VR applications, our solution offers a component that can be deployed on any platform implementing WebXR.

Smartphone-based headsets have the potential to be closer to the standard VR, with 6-DoF motion tracking and environmental understanding [17]. This idea was implemented by Mendez [18], in a smartphone VR application to track the user's motion and transform it into a movement inside VR. The application can process data obtained from a camera and IMU sensors, while the user interacts through a smartphone-based headset. The motion tracking is based on the Google ARCore (https://developers.google.com/ar/, accessed on 22 April 2021) services and inside-out tracking. The 6-DoF SMC component also utilizes the same ARCore technology as Mendez [18]. Our solution is implemented in HTML and JavaScript under the WebXR Device API. Our interface is fully accessible through a web-browser on a variety of platforms while the application of Mendez (2018) [18] is limited by the underlying platform.

### 2.3. Gesture Recognition on MS HoloLens

Regarding the gesture recognition, ExplorViz [19,20] offers a similar approach to obtain gesture events as our HoloGOM component. ExplorViz represents a web-based tool [19] intended for a real-time exploration of so-called virtual cities and landscapes. From the technological point of view, it uses JavaScript access to the experimental WebVR API, a predecessor of WebXR. WebVR is used to obtain data from the MS Kinect device. The HoloGOM component works with MS HoloLens, however the sensors of HoloLens and Kinect are similar.

The contribution of Mohr et al. [21] introduces TrackCap, which allows using smartphones as intelligent 6-DoF controllers with a variety of VR headsets. The positional tracking in TrackCap is provided by a smartphone, held in the user's hand. The smartphone senses a marker placed on the front side of the headset. In relation to the marker, the smartphone calculates its relative position and rotation around the headset. The TrackCap is also suitable for MS HoloLens, where smartphones can be used as virtual hands. The communication with the headset is provided via Wifi or Bluetooth. Unlike the HoloGOM component, the work of Mohr et al. [21] does not support free-hand gesture inputs. HoloGOM utilizes the JavaScript Gamepad API (https://developer.mozilla.org/en-US/docs/Web/API/Gamepad_API, accessed on 22 April 2021) that obtains sensed gestures directly from MS HoloLens sensors in real-time. Therefore, the user can interact more intuitively and without external equipment.

### 2.4. Performance Evaluation of Shared and Web XR

A performance evaluation, related to the one presented in this article, was carried out by Toasa et al. [22]. They developed a custom web application and used it to evaluate web-based VR performance on various devices. The devices were a gaming PC, a smartphone and three types of VR headsets. Because of the nature of our evaluation, we use a PC, a smartphone and MS HoloLens but no VR headsets. Toasa et al. [22] focused on WebGL and WebVR performance and their application is a single-user one, developed in the Unity game engine. Our evaluation deals with original components for A-Frame and NAF, which utilize WebXR.

In [23], Parthasarathy et al. presented an evaluation focused on the quality of service-related network parameters. They prepared a multi-user VR application in the Unity

engine and measured network parameters such as throughput and round-trip time to understand how many users can be simultaneously connected while maintaining high image quality. Three experiments were conducted by Parthasarathy et al. [23], with 2, 5 and 10 users. A design element, similar to our study, is that they used one human user and the other users were simulated by a computer. The human user used the Oculus Rift headset in [23], while in our work it was the smartphone or MS HoloLens. In our study, each simulated user ran on a dedicated machine, while, in[23], it was one PC for all of them. The work of Parthasarathy et al. [23] also differs in the implementation platform and the set of evaluated parameters.

Letić et al. [24] presented an implementation of a real-time web-based solution that projects pre-rendered map tiles onto a VR plane. As in our case, the implementation was carried out in A-Frame and its performance was measured. The performance measures in [24] were frame rate and JavaScript heap size. Similarly to us, Letić et al. [24] considered 30 fps as a frame rate acceptable for human eyes. Due to the nature of their solution, the measurements in [24] were performed with one client application only.

### *2.5. Relation to Cognitive Infocommunications Area*

Human–computer interaction within XR is a recognized area in emerging trends of cross-domain cognitive infocommunications, where the cognitive capabilities of the modern information and communication technologies and human users are considered and evaluated [25]. Probably the most related development is the MaxWhere (https://www.maxwhere.com/, accessed on 22 April 2021) platform, which evolved from the VirCA [26] collaborative VR platform. MaxWhere has also been used for several interesting tasks, such as an evaluation of a 2D advertising in 3D virtual space [27], an assessment of the role of VR in communication and memory management [28], a virtual laboratory system [29] or to increase digital workflow effectiveness [30]. The experiments performed by Lampert et al. [31] and Horváth and Sudár [32] demonstrated that MaxWhere is an effective platform for collaborative information and workflow sharing. The key difference between MaxWhere and LIRKIS G-CVE is in the way they see the role of VR and web technologies: In MaxWhere, users share resources via 3D virtual scenes and it requires dedicated software to be installed. The collaboration in MaxWhere is possible thanks to multiple web browser panels, included in the scene. The panels allow accessing the resources directly or by running corresponding web applications. In LIRKIS G-CVE, the virtual scenes run within a browser, therefore no installation is needed. The users see avatars of other users in the scene and interaction with other users or the scene can be supported. As LIRKIS G-CVE scenes are written using the standard HTML5 Stack, it should be possible to include existing web applications for sharing and collaborating on a desired content. However, such functionality is not yet available in the underlying A-Frame framework. It should also be noted that MaxWhere is a solution already used in practice, while LIRKIS G-CVE is primarily an experimental platform.

## 3. Enhanced User Interaction Components

The proposed components, described and evaluated in this paper, can be divided into two groups. The sole member of the first one is the Enhanced Client Access (ECA) component for NAF. ECA provides remote client administration and full access to clients' entity information. The second group includes 6-DoF Smartphone Motion Capture (6-DoF SMC) and MS HoloLens Gesture-based Object Manipulation (HoloGOM). They add new features to A-Frame but can do the same for any application supporting WebXR. The 6-DoF SMC component brings 6-DoF motion tracking to smartphone-based VR headsets and HoloGOM provides gesture recognition for XR web applications on MS HoloLens.

### 3.1. ECA: Enhanced Client Access

While A-Frame with NAF allows multiple users to simultaneously access a shared virtual scene just by typing the same URL to a browser, it is not so easy for the client app to get information about other users that share the scene.

To achieve the scene sharing, NAF uses both server and client-side modules (Figure 1), which communicate via web sockets and RTC data channels, utilizing the UDP protocol. When a new client app connects to the shared scene, NAF creates a separate data channel for it. The client app uses the channel to send data about its user avatar to the server and to receive data about avatars of other users connected to the scene. The server modules are responsible for the replication of avatar data to all connected client apps. The frequency of the avatar data update is 15 times per second. This is the default value, set by NAF to prevent server lags when a high number of clients is connected. The avatar data contain the position and rotation of the avatar in the scene and allow client apps to render avatars of all users sharing the scene. The problem with NAF is that the data are encapsulated and the application programming interface (API) of NAF only allows accessing user identifiers and events caused by the users.

This problem is the primary one addressed by the ECA component. ECA implements an API, which allows accessing the data channels with each avatar data update. To achieve this, ECA utilizes and modifies corresponding NAF modules, namely Utils, NetworkConnection and NetworkedEntities. ECA also extents the avatar data by new items, for the purpose of avatar customization. In total, five data items can be extracted from the data channel of each user avatar by the ECA API:

1. avatar position in the scene (x,y,z coordinates);
2. avatar rotation in the scene (yaw, pitch, roll);
3. user name;
4. avatar head color; and
5. position of avatar hands in the scene (x,y,z coordinates).

The first two items are the ones originating from NAF, while Items 3–5 have been added by ECA. The functionality ECA provides for a NAF client can be divided into five client-side modules:

**User identification** allows a user to enter his or her name when connecting to a shared virtual scene (VS).

**Avatar data API** gives access to the five data items described above.

**User history** provides a list of users that have visited the shared VS, together with the time when they entered and left it.

**User administration** allows control over users in the same shared VS. In the current version, its functionality is limited to removing users from the VS. After a user is removed, his or her client app still renders the scene, but without the avatars of other users.

**Dashboard** provides an administrative GUI (Figure 2) for the previous modules, where their functionality can be accessed in a user-friendly way. The data from the shared users dataset and user history are displayed in tables and the possibility of removing the clients is also available. The module is password-protected.

The server-side of ECA (Figure 1) stores administrator credentials for accessing the dashboard module and a list of user connections for the history module. It also includes three endpoints: the first one authenticates dashboard users against the stored credentials; the second one sends commands from the client administration to other client apps, representing users in the same VS; and the third one is used for the user history list maintenance.

ECA is built on top of the version 0.6.1 of Networked-Aframe and requires A-Frame of the version 0.8.1 or later. Earlier versions of A-Frame cannot be used as they do not support so-called dynamic scene graphs: Dynamic scene graphs allow adding or removing scene entities without the need of the corresponding web page refresh. Avatars of connected

users belong to the scene entities, too. It is true that more recent versions of NAF (0.7.0 and 0.7.1) have been issued in 2020. However, because of some unfixed issues, it was decided to finish the development of ECA and perform the experiments (in mid to late 2020) with the stable version 0.6.1. Meanwhile, the compatibility of ECA with NAF 0.7.1 was tested, with success (A LIRKIS G-CVE instance, demonstrating ECA utilization with NAF 0.7.1, is available at https://lgcve-cbot.herokuapp.com/, accessed on 22 April 2021).

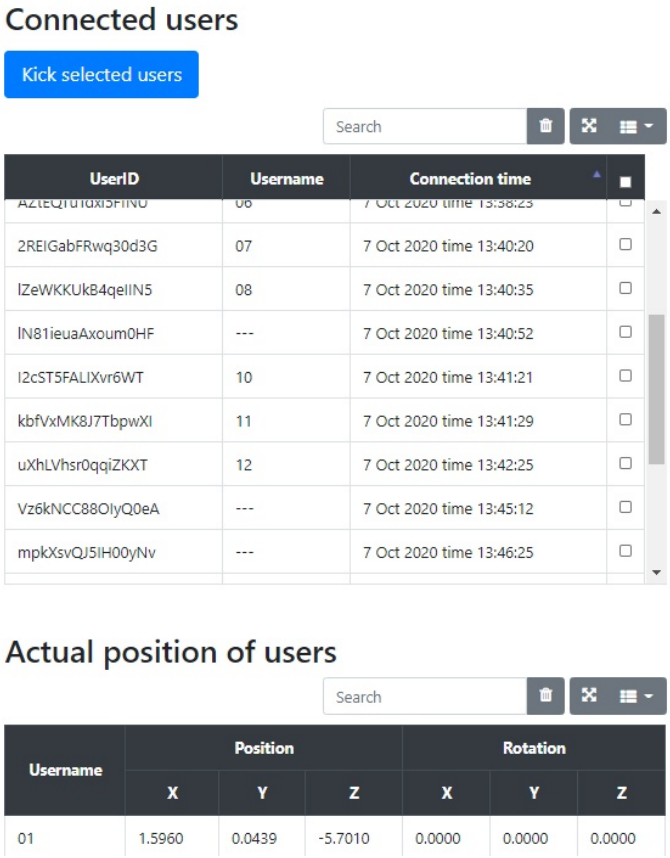



**Figure 2.** Part of the ECA dashboard during the performance evaluation.

### *3.2. 6-DoF SMC: 6 Degrees of Freedom Smartphone Motion Capture*

A-Frame did not include components to translate smartphone motion into movement in virtual environments. A similar limitation was also present in the whole web-based VR. The situation changed in 2017, when Google introduced its ARCore library, which brings 6-DoF motion tracking to smartphones. In web browsers, ARCore is available via the WebXR Device API, which has certain safety restrictions and is still considered experimental in some browsers (A web page using it should be provided via the HTTPS protocol. For browser support and special settings required, one can search for "WebXR Device API" at https://caniuse.com/, accessed on 22 April 2021). A-Frame integrated the ARCore 6-DoF motion tracking functionality to its version 1.0.3, released in December 2019. While it is possible to turn on the 6-DoF motion tracking in A-Frame 1.0.3 or later, it is not available in the stereoscopic rendering mode, which is essential for the smartphone-based VR headsets, such as Google Cardboard (https://arvr.google.com/cardboard/, accessed on 22 April 2021). This issue was solved by the 6-DoF SMC component.

The 6-DoF SMC component (Figure 3) has been designed for mobile devices with the operating system Android 8.0 or later. We intended to utilize the Google ARCore library only, but for web-based purposes it was necessary to employ WebXR, too. In general, the ARCore obtains IMU and camera data from the mobile device to detect its position and rotation, relatively to the surrounding space. The motion tracking is based on features detection from the image stream, captured by a camera. This mechanism detects significant

points that contrast in the image stream. The ARCore is able to track how those points are moving over time, and, through this, it can process the device motion. After that, the ARCore data are directed to the WebXR module, which controls the whole web-based backend. To achieve cross-compatibility between ARCore and WebXR, the web browser Chrome Canary version 70 was used. In our solution, the WebXR controls A-Frame camera position and rotation, relatively to the virtual scene. When the user moves in the real world, the motion is detected and translated into the virtual environment. 6-DoF SMC has been tested with A-Frame version 1.0.4; however, significant performance issues have been detected. This is most probably caused by the 6-DoF SMC component running in parallel with the 6-DoF motion capture functionality of A-Frame 1.0.4, which, unfortunately, lacks the stereoscopy support. Therefore, A-Frame version 0.9.2 was used on a smartphone during the experiments presented in this article.

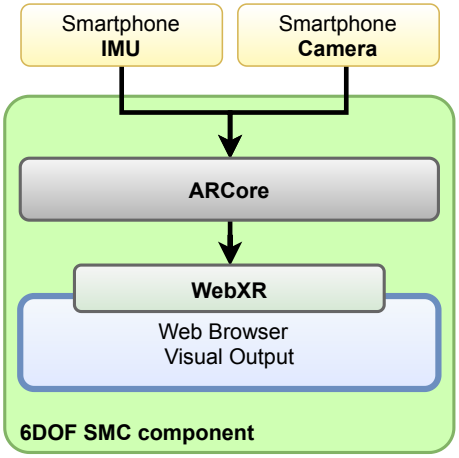



**Figure 3.** 6-DoF SMC component position tracking mechanism.

### *3.3. HoloGOM: MS HoloLens Gesture-Based Object Manipulation*

One of the characteristic features of MS HoloLens is the support of interaction by hand gestures. The HoloGOM component (Figure 4) can detect gesture inputs from MS HoloLens and transform them into the virtual environment. During the HoloGOM implementation, we found that the current version of Gamepad API provides new events for reading gesture inputs as controller buttons. By using a set of standard HoloLens gestures (tap, hold and drag), we tested the API outputs. Then, the gesture commands were implemented. The HoloGOM component supports two types of object manipulation.

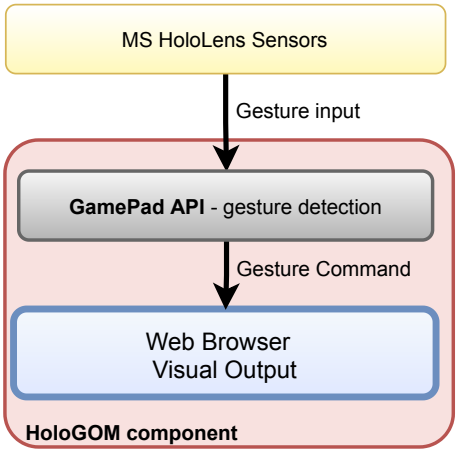



**Figure 4.** HoloGOM component for obtaining MS HoloLens gestures.

The first one, called click-and-push, offers a static manipulation, where a virtual object is "teleported" from one position to another without rendering its movement between the

positions. The user can select an object and then choose the destination point, where the object will be moved. This manipulation is event-driven and proceeds as follows (Figure 5a): First, the user selects the virtual object to move using the tap gesture. Next, the user moves his or her hand to the destination point, while showing the ready gesture. When on the destination point, the user shows the tap gesture again and the object disappears from its original position and appears at the destination point. The manipulation can be used even when the user moves (walks) in the real environment. This is especially useful when moving virtual objects over greater distances. The user has to keep the hand performing the gestures in the field of view of the MS HoloLens sensors during the whole manipulation.

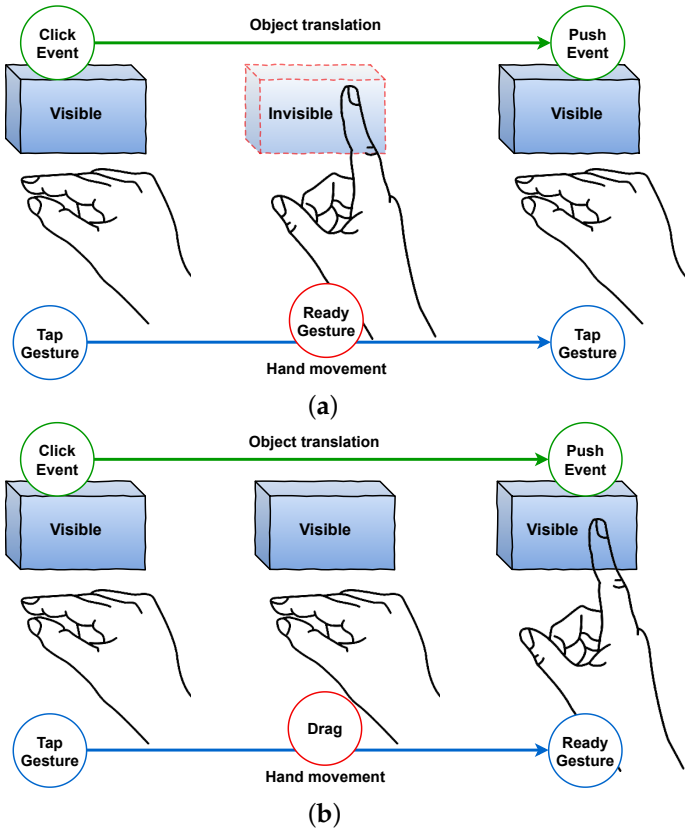



**Figure 5.** Object manipulation with HoloGOM: (**a**) static gesture interaction; and (**b**) dynamic gesture interaction.

The second manipulation, called tap-and-drag, provides dynamic object manipulation (Figure 5b). It consists of an activity in which the user taps on the object and moves it continuously to its destination, i.e., the classical drag-and-drop. This is more immersive as the click-and-push, but the user cannot walk when performing it. As shown in Figure 5b, the user first selects a virtual object using the tap gesture. Then, he or she maintains the tap gesture during the movement. At the destination point, the user releases the object by showing the ready gesture.

Both types of manipulation can be performed by one hand only. The HoloGOM component, as well as the whole A-Frame, work on MS HoloLens via its version of the Microsoft Edge browser. For the full functionality of the component, it is necessary to enable experimental JavaScript features and WebVR in the browser.

## 4. Experimental Setup and Procedures

The performance evaluation of the proposed components consisted of three sets of experiments, E.ECA, E.6DF and E.HL. The E.ECA experiments measured the performance of the ECA component, E.6DF of the 6-DoF SMC component and E.HL of the HoloGOM component.

### *4.1. Experimental Setup*

To eliminate the influence of the Internet traffic and ensure replicability, all experiments took place in a dedicated local network environment at the home institution of the authors.

The topology of the network is shown in Figure 6 and details about the used devices can be found in Table 1. The shortcuts used for the devices include those introduced in Section 1 and Figure 1 (SRV, PC, PH and HL).

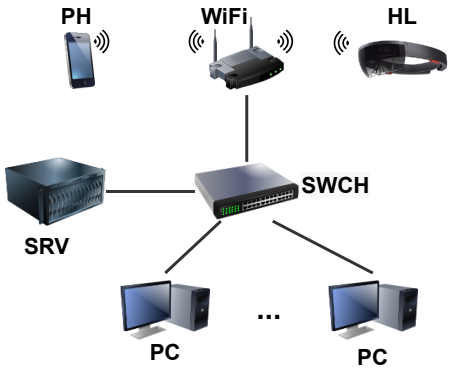



**Figure 6.** Local network topology used in the performance evaluation experiments.

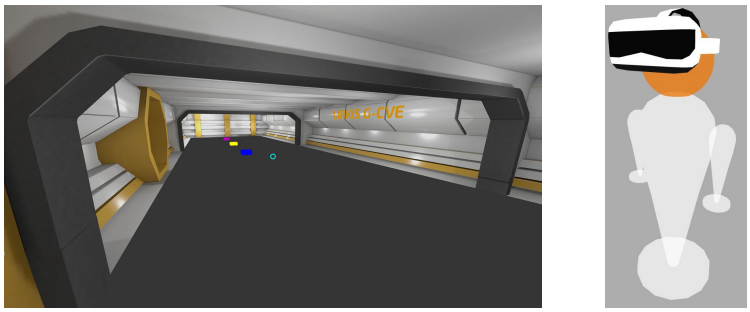
**Table 1.** Devices used in the performance evaluation experiments.

| Label | Type | Model or Configuration |
|---|---|---|
| WiFi | Wireless router | Mercusys MW 325R |
| SWCH | Ethernet switch | Cisco Catalyst 2960-X series |
| SRV | Server computer | Intel® Core™ i7-9700F 3GHz CPU;<br>16GB RAM; Crucial BX500 HDD;<br>NVIDIA GeForce GTX 1650 GPU; Full HD display;<br>Windows Server 2019 Standard OS;<br>Node.js ver.14.10.1; Express.js ver.4.17.1;<br>Socket.IO 2.3.0; |
| PC | Personal computer | Intel® Core™ i5-4590 3.3GHz CPU;<br>Intel HD Graphics 4600 GPU; Full HD display;<br>4GB RAM; Seagate Barracuda 7200.12 HDD;<br>Windows 10 education OS |
| PH | Smartphone | Xiaomi Redmi Note 7;<br>4GB RAM; 64GB internal storage;<br>Android 10 OS |
| HL | MS HoloLens | Microsoft HoloLens 1 |

All experiments used up to 20 PC, one SRV server and one SWCH switch. The experiments from the E.6DF set also used one PH and the ones from the E.HL set one HL. Of course, the wireless router (WiFi) was needed for E.6DF and E.HL experiments, too. Each client device, that is PC, PH and HL, ran one instance of a web browser with one instance of a virtual scene (VS), used for the experiments. This means that each client device represented one user in the shared VS. There were no other software processes running on the client devices, except for the ones belonging to the operating system. The PC devices used Google Chrome browser version 85 and the smartphone (PH) used Google Chrome Canary version 70. The MS HoloLens device HL used its built-in version of the MS Edge browser. The server computer SRV ran one instance of a web server, providing the shared VS and NAF, and one instance of the Google Chrome browser version 86, with the ECA dashboard page. On each PC, the virtual scene was rendered in the windowed

mode with 1920 × 970 pixels (px) resolution. Full-screen stereoscopic modes were used on PH and HL. The resolution was 1170 × 1080 px per eye on PH and 1280 × 720 px per eye on HL.

The same virtual scene was used for all experiments. The scene had a form of a sci-fi-esque hall interior (Figure 7a). The model of the hall had 8550 polygons and was textured by a baked texture. The texture resolution was 2048 × 2048 px with 24 bit color depth. The scene was illuminated by a white ambient light, without dynamic shadows. Each user, connected to the scene, was represented by a 3D avatar (Figure 7b). The avatar model had 1370 polygons and did not use textures.



(**a**) (**b**)

**Figure 7.** 3D models used in the experiments: virtual scene (**a**); and user avatar (**b**).

### *4.2. Experimental Procedure*

In total, 13 experiments were carried out, each using the experimental setup in a slightly different way.

The subject of each experiment, i.e., the evaluated component, and devices involved are listed in Table 2. Each experiment was carried out with and without the corresponding component. The experiments E.6DF.1 and E.HL.1 were the only ones that did not use NAF. As the MS HoloLens performance was dropping rapidly with the rising number of users, we decided not to perform the E.HL experiments with 15 and 20 client devices (users). The procedure, used in every experiment, consisted of the following steps:

1. If not already turned on, turn on the server computer SRV.
2. If not already running, launch the web server on SRV, with or without the ECA component. For E.6DF.1 and E.HL.1, do not launch NAF at all.
3. If ECA is used, launch one instance of the web browser on SRV and open the ECA dashboard page in it.
4. If not already turned on, turn on the desired number of the personal computers PC.
5. On each PC used, launch one instance of the web browser and open the client app with the shared virtual scene in it.
6. If the experiment belongs to the E.6DF set, turn on the smartphone PH (if not already turned on), launch one instance of the web browser there and open the client app with the shared virtual scene in it.
7. If the experiment belongs to the E.HL set, turn on MS HoloLens device HL (if not already turned on), launch one instance of the web browser there and open the client app with the shared virtual scene in it.
8. Let the client app with the virtual scene run on each client device for a desired period of time. In the case of PH and HL, walk with the device to emulate user activity.
9. After the desired period, close the browsers on the client devices (PC, HL and PH).
10. If a re-configuration is needed, close the web server on SRV.

Two performance measures were recorded during the Step 8 of the experiments: frame rate and response time. The measures were recorded on one of the client devices with the constant sampling period of 10 s. For the E.ECA experiments, the recordings took place on one of the personal computers, the E.HL ones on MS HoloLens and E.6DF

on the smartphone. The frame rate was measured in the scene, using the A-Frame API. The response time was measured in the following way:

1. At the time $t_s$, the client app on the corresponding device (PC, HL or PH) sends a ping packet to the node.js server, running on the SRV.
2. The node.js server responds with a pong packet (32 bytes).
3. The client app receives the pong packet at the time $t_r$.
4. The response time is computed as $t_r - t_s$.

The ping.js library (https://github.com/alfg/ping.js/, accessed on 22 April 2021) was used to implement the measurement. The node.js server is the one that hosts the client app and runs the server side of NAF. In the case of experiments E.6DF.1 and E.HL.1, only the frame rate was measured.

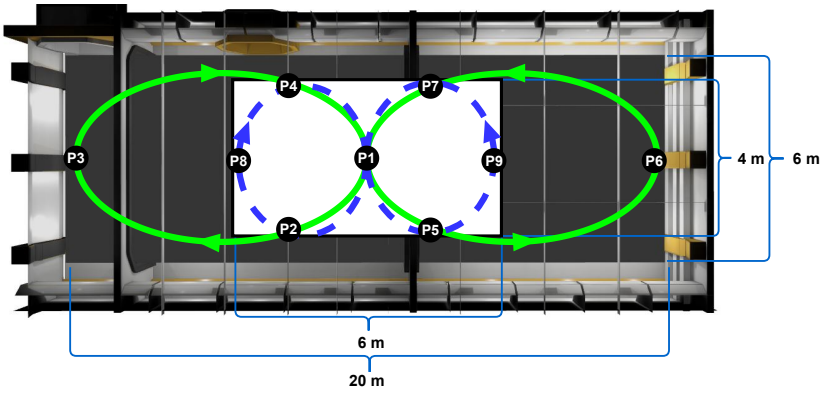
**Table 2.** Subjects of individual experiments and devices used.

| Experiment | Subject (Component) | Number of Client Devices | | |
|---|---|---|---|---|
| | | PC | PH | HL |
| E.ECA.1 | ECA | 1 | 0 | 0 |
| E.ECA.5 | ECA | 5 | 0 | 0 |
| E.ECA.10 | ECA | 10 | 0 | 0 |
| E.ECA.15 | ECA | 15 | 0 | 0 |
| E.ECA.20 | ECA | 20 | 0 | 0 |
| E.6DF.1 | 6-DoF SMC | 0 | 1 | 0 |
| E.6DF.5 | 6-DoF SMC | 4 | 1 | 0 |
| E.6DF.10 | 6-DoF SMC | 9 | 1 | 0 |
| E.6DF.15 | 6-DoF SMC | 14 | 1 | 0 |
| E.6DF.20 | 6-DoF SMC | 19 | 1 | 0 |
| E.HL.1 | HoloGOM | 0 | 0 | 1 |
| E.HL.5 | HoloGOM | 4 | 0 | 1 |
| E.HL.10 | HoloGOM | 9 | 0 | 1 |

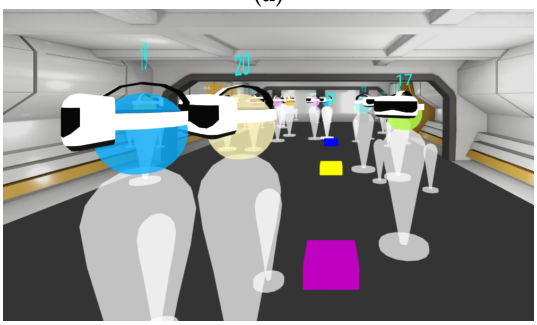
In each experiment, the procedure was repeated twice, first with the evaluated component turned on and second with the component turned off. The time period was set to 60 min for the E.ECA experiments and to 15 min for the E.6DF and E.HL experiments. The reason for the shorter period was that the E.6DF and E.HL experiments required human activity and the fact that the shortening had no significant influence on the measured data, thanks to the stable performance of the devices used.

As we expect some activity from users in a shared virtual scene, it was necessary to emulate such activity for each client device during the experiments. The activity was represented by a user avatar movement through the scene. In the case of the client apps on PC, the movement was carried out automatically, following the trajectory drawn in green in Figure 8a. The avatars on PC moved in the direction of the green arrows, repeatedly through the points P1, P2, P3, P4, P1, P5, P6 and P7. Random excursions were implemented to have some variety in the trajectories. How the scene looked with 20 moving users, represented by client apps on PC, can be seen in Figure 8b. For the smartphone and MS HoloLens, the movement was performed by one of the experimentalists, holding or wearing the device. The experimentalist followed a similar trajectory, but on a smaller area as the avatars of the PC client apps. The area is depicted as the white rectangle in Figure 8a, and the experimentalist walked repeatedly through the points P1, P2, P8, P4, P1, P5, P9 and P7 (the blue dashed line).

To ensure that the recorded performance measures were not affected by the server machine (SRV) using all of its resources, its performance was observed during the experiments. The corresponding measurements included CPU and memory (RAM) usage and were performed on the operating system level at SRV. In all experiments, both the CPU and RAM usage was below 60%.




(**a**)



(**b**)

**Figure 8.** User trajectories in the virtual scene (**a**); and the virtual scene populated by user avatars (**b**) in the E.ECA.20 experiment .

As NAF and ECA use UDP for the client–server communication, an additional monitoring procedure was implemented to the node.js server. It monitored whether all NAF messages from all clients were received correctly. It repeatedly counted all messages received within 1 min and checked whether the count equals (1).

$$60 \times 15 \times n \tag{1}$$

If not, an inconsistency was logged on the server. No inconsistencies were observed during the experiments. In (1), $n$ is the number of connected clients.

## 5. Experiment Results and Discussion

In this section, we first look on the experiment results related to the frame rate (Section 5.1) and then on the ones related to the response time (Section 5.2). The results are provided as statistics and comparisons, in the form of min–max–average charts (Figures 9 and 10) and tables (Tables 3–6). An interested reader can find detailed results, presented in the form of scatter plots, online, using the links provided in the captions of Figures 9 and 10. In Section 5.3, we formulate and discuss answers to the research questions RQ.1 and RQ.2. When referring to individual experiments and devices, the labels introduced in Section 4 (Tables 1 and 2) are used.

### *5.1. Frame Rate*

How the components and increasing load influence the frame rate can be seen in Figure 9. Here, the gray dots and lines represent data measured without the components and the black ones are for data measured with the proposed components. Only the points indicating maximum and minimum values are black in both cases. The reference frame rate value of 30 frames per second (fps) is highlighted by a thicker gray line. Individual frame rate values, recorded during the experiments, can be seen in a document, linked to Figure 9, where each point represents one recorded value.

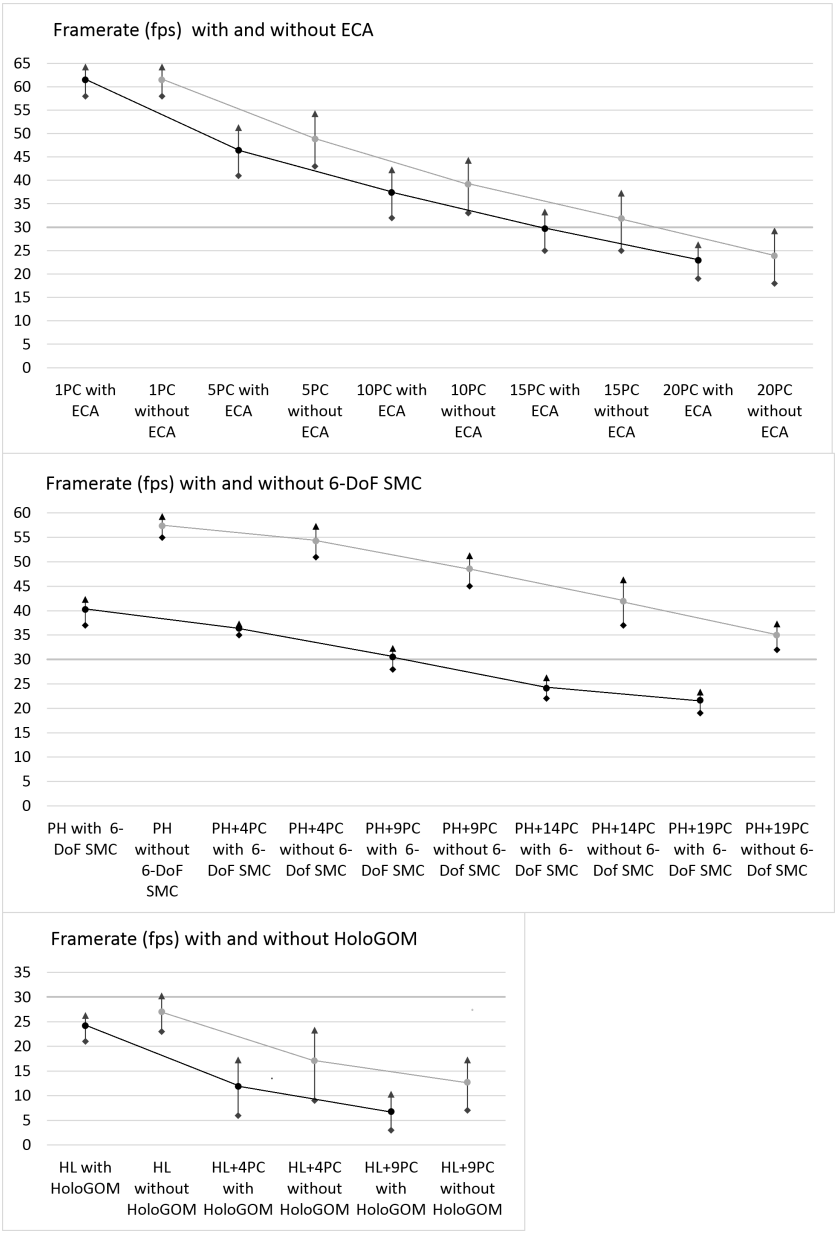



**Figure 9.** Min–max–average charts comparing the frame rate in frames per second with and without the ECA (**top**), 6-DoF SMC (**middle**) and HoloGOM (**bottom**) component under various loads. The complete data can be seen in http://lirkis.kpi.fei.tuke.sk/wp-content/collVREvalScPlFps.png (accessed on 22 April 2021).

The numerical values of the statistics in Figure 9 (minimum, average (mean) and maximum) can be found in Tables 3 and 4. Tables 3 and 4 also provide the sample standard deviation (std) for the collected data sets. Each set of statistics is given for the dataset collected with the corresponding component first, and then for the set collected without the component. By the corresponding component, we mean the component that was the subject of the experiment. The last row (FR.drop) tells us how the frame rate decreased after the corresponding component had been employed. It is computed as the percentage of the average frame rate without the component, according to (2).

$$FR.drop\ (\%) = 100\% * (avg_{fn} - avg_{fy})/avg_{fn} \quad (2)$$

In (2), $avg_{fn}$ is the average frame rate without the proposed component and $avg_{fy}$ is the average frame rate with the component. The most significant frame rate drop was

observed with the 6-DoF SMC component (36.07% on average), followed by HoloGOM (29.1%). On the other hand, the ECA component had negligible influence (4.03%).

**Table 3.** Frame rate summary statistics of the data collected in the E.ECA experiments and frame rate drop computed from the averages. The statistics are given in *frames per second* and the frame rate drop as a percentage. All values are rounded to two decimal places. "FR.drop" means Frame rate drop, "With" means with the proposed component and "Without" without the component.

| | | E.ECA.1 | E.ECA.5 | E.ECA.10 | E.ECA.15 | E.ECA.20 |
|---|---|---|---|---|---|---|
| **With** | **max** | 65.00 | 52.00 | 43.00 | 34.00 | 27.00 |
| | **avg** | 61.44 | 46.40 | 37.44 | 29.66 | 22.93 |
| | **min** | 58.00 | 41.00 | 32.00 | 25.00 | 19.00 |
| | **std** | 2.33 | 3.46 | 3.56 | 2.86 | 2.65 |
| **Without** | **max** | 65.00 | 55.00 | 45.00 | 38.00 | 30.00 |
| | **avg** | 61.42 | 48.82 | 39.12 | 31.85 | 23.88 |
| | **min** | 58.00 | 43.00 | 33.00 | 25.00 | 18.00 |
| | **std** | 2.24 | 3.81 | 3.72 | 4.00 | 3.52 |
| **FR.drop (%)** | | −0.03 | 4.94 | 4.30 | 6.88 | 3.98 |

**Table 4.** Frame rate summary statistics of the *frames per second* collected in the E.6DF and E.HL experiments and frame rate drop in percentage computed from the averages. The table is organized in the same way as Table 3.

| | | E.6DF. 1 | E.6DF. 5 | E.6DF. 10 | E.6DF. 15 | E.6DF. 20 | E.HL. 1 | E.HL. 5 | E.HL. 10 |
|---|---|---|---|---|---|---|---|---|---|
| **With** | **max** | 43.00 | 38.00 | 33.00 | 27.00 | 24.00 | 27.00 | 18.00 | 11.00 |
| | **avg** | 40.26 | 36.42 | 30.51 | 24.12 | 21.65 | 24.23 | 11.87 | 6.75 |
| | **min** | 37.00 | 35.00 | 28.00 | 22.00 | 19.00 | 21.00 | 6.00 | 3.00 |
| | **std** | 1.97 | 1.10 | 1.67 | 1.81 | 1.72 | 1.98 | 3.95 | 2.71 |
| **Without** | **max** | 60.00 | 58.00 | 52.00 | 47.00 | 38.00 | 31.00 | 24.00 | 18.00 |
| | **avg** | 57.35 | 54.26 | 48.54 | 41.91 | 34.97 | 26.97 | 17.02 | 12.70 |
| | **min** | 55.00 | 51.00 | 45.00 | 37.00 | 32.00 | 23.00 | 9.00 | 7.00 |
| | **std** | 1.68 | 2.20 | 2.12 | 3.14 | 2.08 | 2.85 | 4.58 | 3.47 |
| **FR.drop (%)** | | 29.79 | 32.89 | 37.15 | 42.45 | 38.09 | 10.15 | 30.28 | 46.89 |

### *5.2. Response Time*

The results for the response time are presented in the same way as for the frame rate. The summary statistics and comparisons can be found in Figure 10 and Tables 5 and 6. The complete datasets are available in the form of a set of scatted plots, linked in Figure 10. Again, the gray color is used for data measured without the proposed components and the black color for data measured with the components.

The response time increase in Tables 5 and 6 is computed as the percentage of the average response time without the component, according to (3).

$$RT.inc\ (\%) = 100\% * (avg_{ry} - avg_{rn})/avg_{rn} \quad (3)$$

In (3), $avg_{rn}$ is the average response time without the component and $avg_{ry}$ is the average response time with the component. The most significant response time increase was observed with the 6-DoF SMC component (34.15% on average). The response time increase with the ECA component was similar to the frame rate drop (6.08%). Surprisingly, HoloGOM had negligible influence (2.27%). However, the response time on MS HoloLens was significantly higher than on other devices, in all cases.

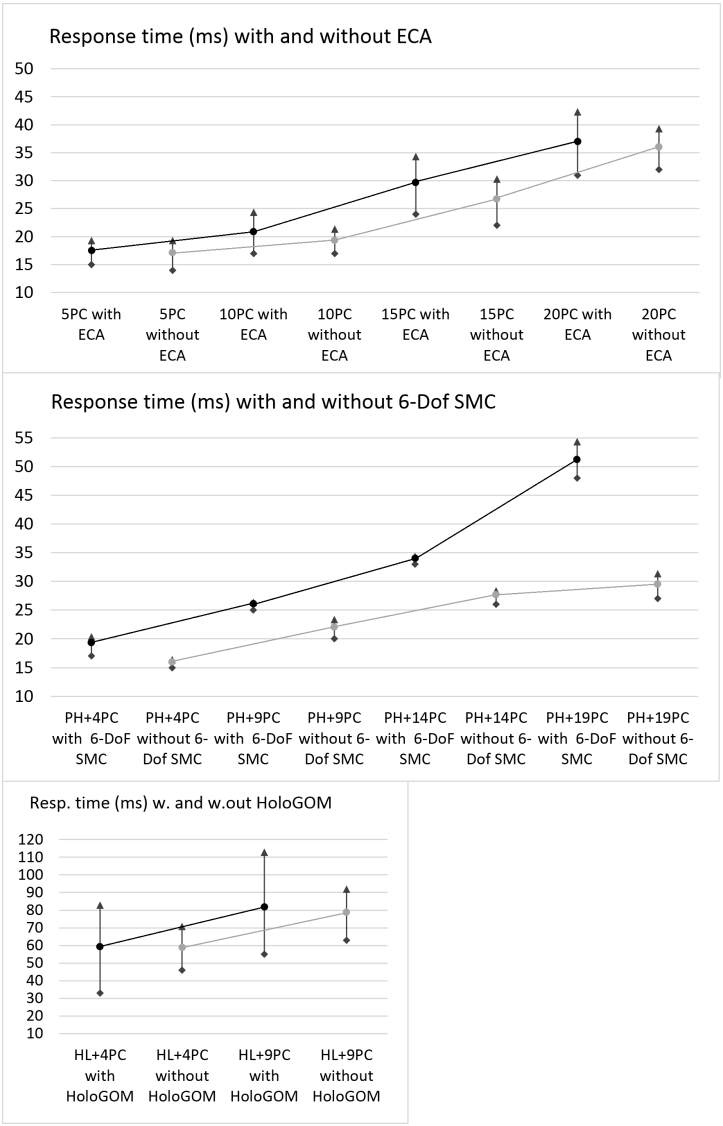



**Figure 10.** Min–max–average charts comparing the response time in miliseconds (ms) with and without the ECA (**top**), 6-DoF SMC (**middle**) and HoloGOM (**bottom**) component under various loads. The complete data can be seen in http://lirkis.kpi.fei.tuke.sk/wp-content/collVREvalScPlResponse.png (accessed on 22 April 2021).

**Table 5.** Response time summary statistics of the data collected in the E.ECA experiments and response time increase computed from the averages. The statistics are given in *milliseconds* and the response time increase as a percentage. All values are rounded to two decimal places. "With" means with the component, "Without" without the component and "RT.inc" means response time increase.

| | | E.ECA.5 | E.ECA.10 | E.ECA.15 | E.ECA.20 |
|---|---|---|---|---|---|
| **With** | **max** | 20 | 25 | 35 | 43 |
| | **avg** | 17.50 | 20.89 | 29.65 | 37.00 |
| | **min** | 15 | 17 | 24 | 31 |
| | **std** | 1.72 | 2.67 | 3.50 | 3.83 |
| **Without** | **max** | 20 | 22 | 31 | 40 |
| | **avg** | 17.11 | 19.32 | 26.70 | 36.00 |
| | **min** | 14 | 17 | 22 | 32 |
| | **std** | 1.98 | 1.67 | 2.88 | 2.54 |
| **RT.inc (%)** | | 2.30 | 8.17 | 11.07 | 2.78 |

**Table 6.** Response time summary statistics (*ms*) of the data collected in the E.6DF and E.HL experiments and response time increase computed from the averages. The table is organized in the same way as Table 5.

| | | E.6DF.5 | E.6DF.10 | E.6DF.15 | E.6DF.20 | E.HL.5 | E.HL.10 |
|---|---|---|---|---|---|---|---|
| **With** | **max** | 21 | 27 | 35 | 55 | 85 | 115 |
| | **avg** | 19.35 | 26.14 | 34.04 | 51.29 | 59.22 | 81.87 |
| | **min** | 17 | 25 | 33 | 48 | 33 | 55 |
| | **std** | 1.36 | 0.82 | 0.76 | 2.20 | 14.91 | 18.11 |
| **Without** | **max** | 17 | 24 | 29 | 32 | 73 | 94 |
| | **avg** | 16.00 | 22.02 | 27.65 | 29.51 | 58.85 | 78.79 |
| | **min** | 15 | 20 | 26 | 27 | 46 | 63 |
| | **std** | 0.80 | 1.37 | 1.10 | 1.77 | 8.18 | 9.02 |
| **RT.inc (%)** | | 20.95 | 18.71 | 23.13 | 73.82 | 0.63 | 3.91 |

### *5.3. Discussion*

Regarding the first research question (RQ.1), we may conclude that each component has certain negative impact on the performance, in terms of both the frame rate and the client–server response time. The only exception was the E.ECA.1 experiment, where the ECA component has virtually zero influence on the frame rate. As shown in Tables 4 and 6 and scatter plots, linked in Figures 9 and 10, the performance was stable both with and without the corresponding components. In all cases but one, the values measured with the corresponding component followed the same trend as the values without it. The only exception was the response time in the E.6DF.20 experiment (PH+19PC) with the significant rise of the response time when 6-DoF SMC was involved.

The ECA component has the lowest overall impact (4.03% on the frame rate and 6.08% on the response time on average). Based on this and its stable performance, we may conclude that ECA was implemented correctly and can be used without the risk of a significant performance loss. The 6-DoF SMC component was the most influential one. The primary reason for this is the resource-hungry motion tracking process, where the input from the phone primary camera has to be analyzed and the user motion calculated. This conclusion is based on two facts. First, the impact on the frame rate (36.07%) was almost the same as the impact on the response time (34.15%). Second, all the computations of the 6-DoF SMC happen in the phone and network communication is not involved in any way. The influence may be noticeably lower on a device better optimized for the motion tracking.

On the mid-range device used (Redmi Note 7 with Android 10), the performance was still acceptable with 10 users sharing the scene (the E.6DF.10 experiment). In E.6DF.10, the minimal frame rate was 28 fps and the maximal response time was 27 ms. This means that the user of the phone still perceives the movements in the virtual scene (VS) as fluent and the information about the avatars of other users is up to date in each frame. The HoloGOM component has almost no impact on the response time (2.27%), while the frame rate drop is significant (29.1%). However, the performance on MS HoloLens is not good overall, with or without the component. Even with only one user in the scene, the average frame rate with the component was only 24.23 fps. With 5 and 10 users, the video output turned into a slideshow experience (11.87 fps for 5 and 6.75 fps for 10 users on average). Because of the performance drop being so significant, we decided not to carry out the experiments with 15 and 20 users on MS HoloLens.

When interpreting the results with respect to the second research question (RQ.2), we have to be more careful. In the case of RQ.1, some generalizations have been possible as the experiments compared the performance with and without the components under the same conditions. The answer to RQ.2 is valid only for the used experimental setup, i.e. the virtual scene, the personal computers (PC) and the smartphone. According to their features and performance (see Table 1 in Section 4), the devices used belong to the middle

range and that is why we limit RQ.2 to them and MS HoloLens. Considering a relatively simple virtual scene with one ambient light source and no dynamic shadows, the average frame rate on PC dropped below 30 fps only when 20 users shared the scene (to 23.88 fps). The response time on PC was always high enough to keep the scene up to date in every frame. With respect to the frame rate, the Redmi Note 7 smartphone is the most successful device. Even with 20 users in the scene, the frame rate was always higher than 30 fps (without 6-DoF motion tracking). The response time was high enough to update the scene at least 36 times per second. MS HoloLens was the worst performer in this evaluation set, and even without HoloGOM the experience was acceptable only without any other users sharing the scene.

## 6. Components Reliability and Usability

While not focused on such goal, the experiments described in the previous sections also tested the reliability of the proposed components, ECA and 6-DoF SMC in particular, on the user experience level. During the E.ECA experiments with ECA turned on, one experimentalist watched the ECA dashboard on the SRV. As the dashboard was displaying the table with the actual positions of the avatars and the virtual scene, he was able to compare the position coordinates, captured by ECA, with the positions of avatars in the scene. No visible differences were observed. The 6-DoF SMC was permanently employed during the E.6DF experiments with the component turned on. The component was performing without problems and translated the experimentalist movement in the real world to the avatar movement in the virtual scene reliably. During the E.HL experiments, the experimentalist walking with MS HoloLens occasionally performed hand movements in the MS HoloLens field of view to stimulate the HoloGOM component. However, to make the user activity with MS HoloLens similar to the activities of users with other devices (PH and PC), we did not include any gesture-involving tasks. The reliability of the gesture recognition was tested separately, in a modified version of the scene from Figure 7. The scene included two movable blocks: a blue block, movable by the dynamic gesture (tap-and-drag), and a brown block, movable by the click-and-push gesture. An experimentalist was moving the blocks using the gestures. Each gesture was successfully performed 60 times. The modified scene is available at https://lirkisgcve.herokuapp.com/hololens.html (accessed on 22 April 2021). Because it is adjusted to MS HoloLens, it is not possible to move in it or manipulate with the blocks when opened on other devices.

The same scene was used for user experience testing of HoloGOM with eight participants. The MS HoloLens device with the scene loaded was provided to them, and they were asked to move the blocks using the corresponding gestures. After performing the task, they filled in the standard System Usability Scale (SUS) questionnaire [33]. During the testing, one of the authors was available to explain the task to the participants and assist them. The participants were aged from 24 to 54 years, and there were six men and two women. The resulting SUS score was 66.56, which is slightly below the average (68, according to Brooke [34]). When asked, all the users selected the dynamic gesture (tap-and-drag) as the more suitable one. This was because the block was visible during the movement, so the users were sure that they were performing the movement correctly. Some of them suggested implementing visual feedback for the tap gesture to improve the situation.

One of the reasons behind our decision to focus on web-based XR was the easiness of making shared virtual environments available to potential users. Just by providing a link to the environment and a set of instructions, arbitrary users can utilize the environment in a proper way. A promising scenario, outlined in [10], is a testing of virtual prototypes of autonomous devices. In such case, a shared scene is used as a testing area, where one client app takes the role of the tested prototype and real persons, using their client apps, act as testers. Thanks to the ECA component, the client app representing the prototype can be aware of the testers and react to their actions. To test usability of such scenario when implemented with A-Frame, NAF and ECA, we prepared a shared virtual scene with an autonomous robot, which cleans positions marked with red tiles. The robot reacts to other

users by avoiding them when moving or by interrupting the cleaning process. The whole application with the scene, including the server side (NAF and ECA modules), is hosted on the Heroku cloud application platform (https://lgcve-cbot.herokuapp.com, accessed on 22 April 2021), so the usability test was performed under real network conditions. Volunteers were asked to visit the page and perform a simple task in the scene, according to written instructions. No live assistance was provided for the volunteers and they did not have any previous experience with the application. After the task, they filled in the standard SUS questionnaire with additional questions to express their opinion about the application and suggest improvements. We also asked them whether they had seen the robot in the scene. The task was very simple: after entering the scene, a participant had to visit some positions to get in the way of the robot. Within two days dedicated to the testing, there were 183 participants, 157 men and 26 women. The age range was from 18 to 54 years, with 83.9% from 20 to 22 years. The resulting SUS score was 77.48, an above-average result. Overall, 22.9% (42) of the participants would like to have improved movement controls; most of them did not like the need to hold the right mouse button down when looking around. Only 3.8% (7) of the participants reported performance issues, namely low frame rates or long loading periods. Fifteen participants (8.2%) did not see the cleaning robot in the scene. Finally, 7.1% (13) of the participants explicitly mentioned that everything was fine.

## 7. Conclusions

Software components, introduced and evaluated in this paper, solved certain limitations of contemporary A-Frame and Networked-Aframe-based web XR applications.

The proposed 6-DoF SMC and HoloGOM components are more universal ones and, after some modifications, can be used without A-Frame. The 6-DoF SMC component provides six degrees of freedom (DoF) inside-out motion tracking for smartphones and smartphone-based VR headsets. The HoloGOM component allows web XR users to manipulate objects in virtual scenes using hand gestures. To our knowledge, there are no other implementations providing corresponding functionality for A-Frame, and this functionality is also lacking on other platforms, for example the Mozilla Hubs. Of course, as the evaluation (RQ.1) revealed, the components are not flawless: 6-DoF SMC requires a special version of web browser (Chrome Canary) and has a significant impact on both the frame rate and network response time. This performance decrease can be attributed to the motion tracking, which has to analyze the phone camera input. It seems that HoloGOM suffers from overall performance of WebXR on MS HoloLens in our experiments. Both components rely on features considered experimental. These issues are expected to be solved in future versions of the underlying software platforms and corresponding devices. The proposed ECA component is closely tied to A-Frame as it enriches the functionality of its Networked-Aframe (NAF) extension. Thanks to ECA, user positions in a shared virtual scene can be easily accessed. It also allows removing users from the scene. The evaluation confirmed minimal impact of the ECA (RQ.1) on the performance (about 5%). The only disadvantage is that ECA needs an upgrade to work with the newest version of NAF. Fortunately, the upgrade has already been developed.

An interesting result of the experimental evaluation is an answer to the question how the shared XR, utilizing A-Frame and NAF, performs on MS HoloLens and mid-range hardware (RQ.2). The mid-range hardware representatives were the devices used in the experiments, namely PC with Intel Core i5 CPU, 4GB RAM and integrated graphics and a Xiaomi Redmi Note 7 smartphone. The answer can be formulated as follows: when using relatively simple virtual scene and avatar models, one can enjoy shared XR with A-Frame and NAF even on mid-range hardware, except MS HoloLens. To be more precise, the smartphone handled the shared scene with 20 users without any problems. PC handled the scene with 15 users equally well, while the average frame rate dropped to 23.88 fps with 20 users. MS HoloLens was able to handle the scene with no other users connected. With five users in the scene, the average frame rate dropped below 20 fps.

The components and scenes, developed as parts of the LIRKIS G-CVE platform, offer promising application and experimentation areas for future research and development. For example, virtual industrial installations, such as the one presented in [35], and distant education, in particular aiming at the current efforts to develop so-called digital intelligence [36] or supporting people with special needs, such as autism spectrum disorder [37]. It is also planned to utilize LIRKIS G-CVE for future versions of the experiments described in [38,39], which focus on an assessment of VR influence on visuospatial cognitive functions [40]. Other applications, already under development, are training of civil engineering personnel, safety training and visualization of data from physical experiments in virtual environments. Such applications can also be extended by undergoing developments regarding authentication utilizing keyboard dynamics [41], additional EMG sensing and face recognition, together with emotion recognition [42] and EEG sensing. The experimental setup, introduced in this article, will be reused for other evaluations. For now, the evaluation of 6-DoF SMC with multiple mobile devices is planned. It would also be interesting to compare its performance with VR headsets, such as Oculus Quest, where the 6-DoF motion tracking is supported out of the box.

**Author Contributions:** Conceptualization, Š.K. and B.S.; methodology, Š.K. and M.H.; software, Š.K. and M.H.; validation, Š.K., B.S., M.S., W.S. and M.P.; formal analysis, Š.K.; resources, B.S.; data curation, Š.K., B.S., M.S. and W.S.; writing—original draft preparation, Š.K. and M.H.; writing—review and editing, Š.K. and M.P.; visualization, Š.K., M.H. and M.P.; supervision, B.S.; project administration, B.S.; and funding acquisition, B.S. and M.P. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Slovak Research and Development Agency (Agentúra na podporu výskumu a vývoja) grant no. APVV-16-0202 "Enhancing cognition and motor rehabilitation using mixed reality" and the APC was funded by the Cultural and Educational Grant Agency (Kultúrna a edukačná grantová agentúra MŠVVaŠ SR) grant Nos. KEGA 009TUKE-4/2019 "Content innovation and lecture textbooks for Biometric Safety Systems" and KEGA 035TUKE-4/2019 "Virtual-reality technologies and handicapped people education".

**Institutional Review Board Statement:** Ethical review and approval were waived for this study, because the research presents no more than minimal risk of harm to subjects and involves no procedures for which written consent is normally required outside the research context.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data presented in this study are available on request from the corresponding authors.

**Acknowledgments:** The authors would like to thank our alumni students Michal Ivan, Tomáš Balluch, Kristína Matiková and Dávid Gula for their contributions to the implementation of the proposed components and corresponding virtual scenes.

**Conflicts of Interest:** The authors declare no conflict of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| 3D | three-dimensional |
| 6-DoF | Six Degrees of Freedom |
| API | Application programming interface |
| AR | Augmented Reality |
| ARCore | Google Play Services for AR |
| CoVAR | collaborative virtual and augmented reality system |
| EasyRTC | open source WebRTC toolkit |
| ECA | Enhanced Client Access |
| EEG | Electroencephalogram |
| EMG | Electromyography |

| | |
|---|---|
| G-CVE | Global—Collaborative Virtual Environments |
| GOM | Gesture-Based Object Manipulation |
| GUI | Graphical User Interface |
| fps | frames per second |
| HL | MS HoloLens |
| HMD | Head Mounted Display |
| HTC | High Tech Computer Corporation |
| HTML | HyperText Markup Language |
| HoloGOM | MS HoloLens Gesture-Based Object Manipulation |
| HUB (Mozilla) | Hub is a VR chatroom designed for headset and browser |
| IMU | Inertial Measurement Unit |
| LIRKIS | Laboratórium Inteligentných Rozhraní Komunikačných a Informačných Systémov |
| LIRKIS | En: Laboratory of Intelligent inteRfaces of Communication and Information Systems |
| MR | Mixed Reality |
| MS | Microsoft corporation |
| NAF | Networked-Aframe |
| PC | Personal computer |
| PH | Smart phone |
| px | pixels |
| RQ | research question |
| RTC | Real Time Communication |
| RT | Response Time |
| SMC | Smartphone Motion Capture |
| SRV | server |
| SU | shared users |
| SUS | System Usability Scale |
| SWCH | Ethernet Switch |
| Three.js | cross-browser 3D JavaScript library |
| TrackCap | Smart phone 6-DoF motion Tracking and Capturing library |
| VirCA | Virtual Collaboration Arena |
| VE | Virtual Environment |
| VR | Virtual reality |
| VS | Virtual Scene |
| W3C | World Wide Web Consortium |
| WebGL | JavaScript API for rendering interactive 2D and 3D graphics |
| WebRTC | Web Real-Time Communication |
| WebVR | Web Virtual reality devices support API |
| WebXR | Web Extended reality devices (VR, AR) support API component |
| XR | eXtended reality |

## References

1. Burgess, S.; Sievertsen, H.H. Schools, Skills, and Learning: The Impact of COVID-19 on Education. Available online: https://voxeu.org/article/impact-covid-19-education (accessed on 11 April 2021).
2. Mishra, K.; Boland, M.V.; Woreta, F.A. Incorporating a virtual curriculum into ophthalmology education in the coronavirus disease-2019 era. *Curr. Opin. Ophthalmol.* **2020**, *31*, 380–385. [CrossRef] [PubMed]
3. Mohan, A.T.; Vyas, K.S.; Asaad, M.; Khajuria, A. Plastic Surgery Lockdown Learning during Coronavirus Disease 2019: Are Adaptations in Education Here to Stay? *Plast. Reconstr. Surg. Glob. Open* **2020**, *8*. [CrossRef] [PubMed]
4. Wijkmark, C.H.; Fankvist, S.; Heldal, I.; Metallinou, M.M. Remote Virtual Simulation for Incident Commanders: Opportunities and Possibilities. In Proceedings of the 2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Mariehamn, Finland, 23–25 September 2020; pp. 445–452.

5. Luimula, M.; Linder, M.; Pieskä, S.; Laimio, E.; Lähde, T.; Porramo, P. Unlimited Safety Productivity—A Finnish Perspective Using Virtual Learning Methods to Improve Quality and Productivity in the Construction Industry. In Proceedings of the 2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Mariehamn, Finland, 23–25 September 2020; pp. 263–266.
6. Qiao, X.; Ren, P.; Dustdar, S.; Liu, L.; Ma, H.; Chen, J. Web AR: A promising future for mobile augmented reality—State of the art, challenges, and insights. *Proc. IEEE* **2019**, *107*, 651–666. [CrossRef]
7. Maclntyre, B.; Smith, T.F. Thoughts on the Future of WebXR and the Immersive Web. In Proceedings of the 2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), Munich, Germany, 16–20 October 2018; pp. 338–342.
8. Hudák, M.; Korečko, Š.; Sobota, B. Enhancing Team Interaction and Cross-platform Access in Web-based Collaborative Virtual Environments. In Proceedings of the 2019 IEEE 15th International Scientific Conference on Informatics, Poprad, Slovakia, 20–22 November 2019; pp. 160–164.
9. Hudák, M.; Korečko, Š.; Sobota, B. Advanced User Interaction for Web-based Collaborative Virtual Reality. In Proceedings of the 2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Mariehamn, Finland, 23–25 September 2020; pp. 343–348.
10. Hudák, M.; Korečko, Š.; Sobota, B. LIRKIS Global Collaborative Virtual Environments: Current State and Utilization Perspective. *Open Comput. Sci.* **2021**, *11*, 99–106. [CrossRef]
11. Butcher, P.W.S.; John, N.W.; Ritsos, P.D. VRIA: A Web-based Framework for Creating Immersive Analytics Experiences. *IEEE Trans. Vis. Comput. Graph.* **2020**. [CrossRef] [PubMed]
12. Scavarelli, A.; Arya, A.; Teather, R.J. Circles: Exploring multi-platform accessible, socially scalable VR in the classroom. In Proceedings of the 2019 IEEE Games, Entertainment, Media Conference (GEM), New Haven, CT, USA, 18–21 June 2019; pp. 1–4.
13. Gunkel, S.; Prins, M.; Stokking, H.; Niamut, O. WebVR meets WebRTC: Towards 360-degree social VR experiences. In Proceedings of the 2017 IEEE Virtual Reality (VR), Los Angeles, CA, USA, 18–22 March 2017; pp. 457–458. [CrossRef]
14. Gunkel, S.N.; Stokking, H.M.; Prins, M.J.; van der Stap, N.; Haar, F.B.t.; Niamut, O.A. Virtual Reality Conferencing: Multi-user immersive VR experiences on the web. In Proceedings of the 9th ACM Multimedia Systems Conference (MMSys '18), Amsterdam, The Netherlands, 12–15 June 2018; Association for Computing Machinery: New York, NY, USA, 2018; pp. 498–501.
15. Huh, S.; Muralidharan, S.; Ko, H.; Yoo, B. XR Collaboration Architecture Based on Decentralized Web. In Proceedings of the 24th International Conference on 3D Web Technology (Web3D '19), Los Angeles, CA, USA, 26–18 July 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 1–9. [CrossRef]
16. Fang, W.; Zheng, L.; Deng, H.; Zhang, H. Real-time motion tracking for mobile augmented/virtual reality using adaptive visual-inertial fusion. *Sensors* **2017**, *17*, 1037. [CrossRef] [PubMed]
17. Zikas, P.; Bachlitzanakis, V.; Papaefthymiou, M.; Papagiannakis, G. A mobile, AR inside-out positional tracking algorithm,(mariopot), suitable for modern, affordable cardboard-style VR HMDS. In Proceedings of the Euro-Mediterranean Conference, Nicosia, Cyprus, 31 October–5 November 2016; Springer: Cham, Switzerland, 2016; pp. 257–268.
18. Mendez, R.L. Mobile inside-out VR tracking, now available on your phone. In *ACM SIGGRAPH 2018 Appy Hour*; Association for Computing Machinery: New York, NY, USA; Vancouver, BC, Canada, 2018; pp. 1–2.
19. Fittkau, F.; Krause, A.; Hasselbring, W. Software landscape and application visualization for system comprehension with ExplorViz. *Inf. Softw. Technol.* **2017**, *87*, 259–277. [CrossRef]
20. Hasselbring, W.; Krause, A.; Zirkelbach, C. ExplorViz: Research on software visualization, comprehension and collaboration. *Softw. Impacts* **2020**, *6*, 100034. [CrossRef]
21. Mohr, P.; Tatzgern, M.; Langlotz, T.; Lang, A.; Schmalstieg, D.; Kalkofen, D. TrackCap: Enabling smartphones for 3D interaction on mobile head-mounted displays. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, Glasgow, UK, 4–9 May 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 1–11.
22. Toasa, R.M.; Egas, P.F.B.; Saltos, M.A.G.; Perre no, M.A.; Quevedo, W.X. Performance Evaluation of WebGL and WebVR Apps in VR Environments. In Proceedings of the International Symposium on Visual Computing, Lake Tahoe, NV, USA, 7–9 October 2019; Springer: Cham, Switzerland, 2019; pp. 564–575.
23. Parthasarathy, V.; Simiscuka, A.A.; O'Connor, N.; Muntean, G.M. Performance evaluation of a multi-user virtual reality platform. In Proceedings of the 2020 International Wireless Communications and Mobile Computing (IWCMC), Limassol, Cyprus, 15–19 June 2020; pp. 934–939.
24. Letić, M.; Nenadić, K.; Nikolić, L. Real-time map projection in virtual reality using WebVR. In Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 21–25 May 2018; pp. 1439–1443.
25. Katona, J. A Review of Human–Computer Interaction and Virtual Reality Research Fields in Cognitive InfoCommunications. *Appl. Sci.* **2021**, *11*, 2646. [CrossRef]
26. Vincze, D.; Kovács, S.; Gácsi, M.; Korondi, P.; Miklósi, Á.; Baranyi, P. A novel application of the 3D virca environment: Modeling a standard ethological test of dog-human interactions. *Acta Polytech. Hung.* **2012**, *9*, 107–120.
27. Berki, B. 2D advertising in 3D virtual spaces. *Acta Polytech. Hung.* **2018**, *15*, 175–190.
28. Csapó, Á.B.; Horvath, I.; Galambos, P.; Baranyi, P. VR as a medium of communication: from memory palaces to comprehensive memory management. In Proceedings of the 2018 9th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Budapest, Hungary, 22–24 August 2018; pp. 389–394.

29. Budai, T.; Kuczmann, M. Towards a modern, integrated virtual laboratory system. *Acta Polytech. Hung.* **2018**, *15*, 191–204.
30. Horváth, I. MaxWhere 3D capabilities contributing to the Enhanced Efficiency of the Trello 2D management software. *Acta Polytech. Hung.* **2019**, *16*, 55–71.
31. Lampert, B.; Pongracz, A.; Sipos, J.; Vehrer, A.; Horvath, I. MaxWhere VR-learning improves effectiveness over clasiccal tools of e-learning. *Acta Polytech. Hung.* **2018**, *15*, 125–147.
32. Horvath, I.; Sudar, A. Factors contributing to the enhanced performance of the maxwhere 3d vr platform in the distribution of digital information. *Acta Polytech. Hung.* **2018**, *15*, 149–173.
33. Brooke, J. SUS: A 'quick and dirty' usability scale. In *Usability Evaluation in Industry*; Jordan, P.W., Thomas, B., McClelland, I.L., Weerdmeester, B., Eds.; Taylor and Francis Group, CRC Press: Cleveland, OH, USA, 1996; Chapter 21, pp. 189–194.
34. Brooke, J. SUS: A retrospective. *J. Usability Stud.* **2013**, *8*, 29–40.
35. Bellalouna, F. Virtual-Reality-based Approach for Cognitive Design-Review and FMEA in the Industrial and Manufacturing Engineering. In Proceedings of the 2019 10th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Naples, Italy, 23–25 October 2019; pp. 41–46.
36. Dostál, J.; Wang, X.; Steingartner, W.; Nuangchalerm, P. Digital Intelligence-New Concept in Context of Future School of Education. In Proceedings of the ICERI2017 Conference, Seville, Spain, 16–18 November 2017.
37. Costescu, C.; Rosan, A.; Brigitta, N.; Hathazi, A.; Kovari, A.; Katona, J.; Heldal, I.; Helgesen, C.; Thill, S.; Demeter, R. Emotion recognition in typical and atypical development—A technology-based paradigm. In Proceedings of the 2019 10th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Naples, Italy, 23–25 October 2019; pp. 349–352.
38. Korečko, Š.; Hudák, M.; Sobota, B.; Marko, M.; Cimrová, B.; Farkaš, I.; Rosipal, R. Assessment and training of visuospatial cognitive functions in virtual reality: Proposal and perspective. In Proceedings of the 2018 9th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), Budapest, Hungary, 22–24 August 2018; pp. 39–44.
39. Korečko, Š.; Sobota, B.; Hudák, M.; Farkaš, I.; Cimrová, B.; Vasiľ, P.; Trojčák, D. Experimental Procedure for Evaluation of Visuospatial Cognitive Functions Training in Virtual Reality. In Proceedings of the International Conference on Advanced Intelligent Systems and Informatics, Cairo, Egypt, 26–28 October 2019; Springer: Cham, Switzerland, 2019; pp. 643–652.
40. Aldridge, A.; Barnes, E.; Bethel, C.L.; Carruth, D.W.; Kocturova, M.; Pleva, M.; Juhár, J. Accessible electroencephalograms (EEGs): A comparative review with openbci's ultracortex mark IV headset. In Proceedings of the 2019 29th International Conference Radioelektronika, Pardubice, Czech Republic, 16–18 April 2019; pp. 1–6.
41. Pleva, M.; Bours, P.; Ondáš, S.; Juhár, J. Improving static audio keystroke analysis by score fusion of acoustic and timing data. *Multimed. Tools Appl.* **2017**, *76*, 25749–25766. [CrossRef]
42. Turabzadeh, S.; Meng, H.; Swash, R.M.; Pleva, M.; Juhár, J. Facial expression emotion detection for real-time embedded systems. *Technologies* **2018**, *6*, 17. [CrossRef]