WebTransceiVR

Asymmetrical Communication Between Multiple VR and Non-VR Users Online

Haohua Lyu*
University of California, Berkeley
Berkeley, CA, USA
haohua@berkeley.edu

Cyrus Vachha*
University of California, Berkeley
Berkeley, CA, USA
cvachha@berkeley.edu

Qianyi Chen*
University of California, Berkeley
Berkeley, CA, USA
qianyi.chen@berkeley.edu

Odysseus Pyrinis
University of California, Berkeley
Berkeley, CA, USA
odysseus.pyrinis@berkeley.edu

Avery Liou
University of California, Berkeley
Berkeley, CA, USA
liou.avery@berkeley.edu

Balasaravanan Thoravi Kumaravel
University of California, Berkeley
Berkeley, CA, USA
bala@eecs.berkeley.edu

Björn Hartmann
University of California, Berkeley
Berkeley, CA, USA
bjoern@eecs.berkeley.edu

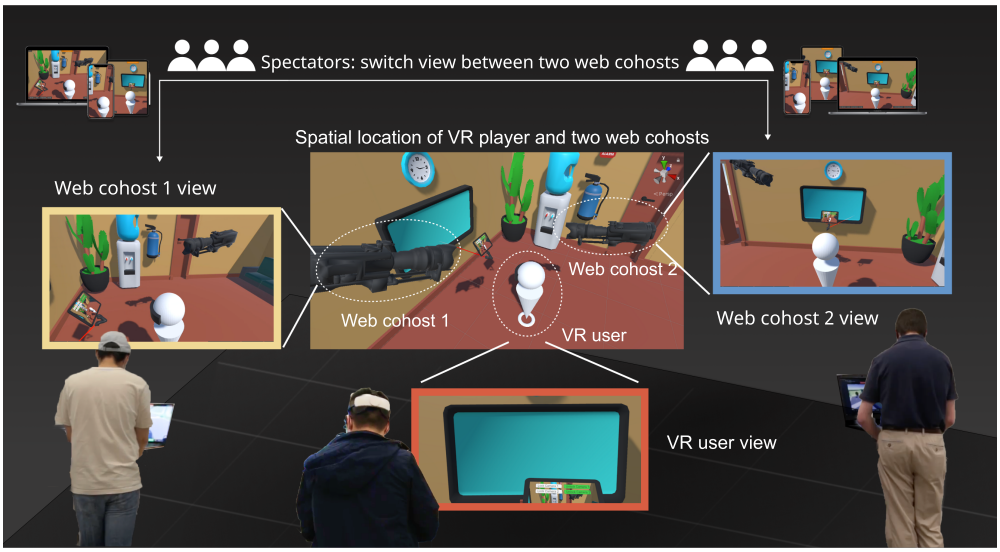

Figure 1: Asymmetric remote interaction using WebTransceiVR; two web co-hosts (left and right) are observing the VR user (middle) in the virtual environment through web browsers on laptops and mobiles. Spectators can view from the cameras of the two web co-hosts.

ABSTRACT

Increasing adoption of Virtual Reality (VR) systems in various fields has created the need for collaborative work and communication. Today, asymmetric communication between a VR user and other non-VR users remains a challenge. The VR user cannot see the external non-VR users, and the non-VR users are restricted to the VR user's first-person view. To address this, we propose WebTransceiVR, an asymmetric collaboration toolkit which when integrated into a VR application, allows multiple non-VR users to share the virtual space of the VR user. It allows external users to enter and be part of the VR application's space through standard web browsers on mobile and computers. These external users can explore and interact with the other users, the VR scene as well as the VR user. WebTransceiVR also includes a cloud-based streaming solution that enables many passive spectators to view the scene through any of the active cameras. We conduct informal user testing to gain additional insights for future work.

*The authors contributed equally to this research.

CHI '22 Extended Abstracts, April 29-May 5, 2022, New Orleans, LA, USA

ACM ISBN 978-1-4503-9156-6/22/04.
https://doi.org/10.1145/3491101.3519816

CCS CONCEPTS

• **Human-centered computing** → **Virtual reality**; **Collaborative interaction**.

KEYWORDS

Virtual Reality, Asymmetric Communication

ACM Reference Format:

Haohua Lyu, Cyrus Vachha, Qianyi Chen, Odysseus Pyrinis, Avery Liou, Balasaravanan Thoravi Kumaravel, and Björn Hartmann. 2022. WebTransceiVR: Asymmetrical Communication Between Multiple VR and Non-VR Users Online. In *CHI Conference on Human Factors in Computing Systems Extended Abstracts (CHI '22 Extended Abstracts), April 29-May 5, 2022, New Orleans, LA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3491101.3519816

1 INTRODUCTION

Virtual reality (VR) is an increasingly popular medium that is being used for different applications ranging from art, design, medicine, entertainment, amongst others. Like with other software, a user in VR may have to interact and collaborate with others while they perform a task. However, these users may not be using a VR headset. There is a growing interest for such an asymmetric interaction between users who operate across VR and non-VR platforms [14, 17, 28]. These interactions occur in numerous domains. In fact, today during the development of a VR application, it is common to show and get feedback from remotely located collaborators who may not have access to VR headsets. A powerful use case of VR is education. It may be enticing for an instructor to use VR to teach concepts that involve complex 3D information. However, many students lack access to VR devices, especially if the instruction is remote. Non-VR students have limited ways to interact with the instructor due to the asymmetry of the display and input abilities between them. This introduces challenges for effective communication. Similar issues occur in VR eSports, where event organizers face difficulty in making the non-VR spectator experience more engaging. In contrast to streaming, one way to solve these issues is to gravitate towards applications that have a counterpart application capable of multi-player style non-VR spectator support. This would require each spectator to download a separate app that contains all the original graphical assets. Such an application can be heavy and may pose specific demands to be met by the spectator's computers.

We seek inspiration from prior work such as TransceiVR [28], ShareVR [7], and XRStudio [17]. Both ShareVR and TransceiVR support asymmetric communication to allow a VR user and a non-VR user to collaborate, either in the same physical space [7] or using the same computer [28]. However, these tools were designed primarily for dyadic interactions and did not scale with viewers. In situations like remote learning, VR live streaming, the ability to support more than a couple of views as well as a large number of passive spectators is key. XRStudio [17] achieves the remote multi-user sharing of VR educational content. Vreal [32] allows multiple spectators to be within the scene of a VR application. However, these approaches still require non-VR users to have dedicated applications installed as well as access to a VR/AR headset.

To address these concerns, we propose WebTransceiVR, an asymmetric collaboration toolkit that brings the non-VR users into the shared virtual space of the VR user. It allows non-VR users to interact with the VR environment through multiple perspectives. On the non-VR user's end, WebTransceiVR runs on modern mobile and desktop web browsers, without the need for installation of any additional software. On the VR user's end, showcasing and sharing a VR application experience requires them to simply import a Unity package. We introduce a three-tiered system: one VR user, one or multiple non-VR "co-hosts" each with asymmetric collaboration tools individually interacting with the scene, and an arbitrary number of non-VR "spectators" each of whom can switch between the different camera perspectives of connected co-hosts. Our goal is to use WebTransceiVR to enable and study asymmetric communication at scale. Specifically, we want to (i) enable multiple remote external users to take different perspectives of a VR user's immersive environment and (ii) allow a subset of them ("co-hosts") to provide input through a set of interactive tools.

2 RELATED WORK

There are three domains of related research to WebTransceiVR: Sharing of VR experience in asymmetric settings; Use of WebRTC for VR networking and consideration of content streaming in XR.

2.1 Sharing VR Experience in Asymmetric Communication

Asymmetric communication between VR and non-VR users has been previously studied in different contexts. One category is collaborative games between VR and PC players who work together to achieve a shared goal [9, 20, 25, 31]. The asymmetry is derived from the game design, which provides different visual representations and perspectives in order to create motivations for users to collaborate [25]. Another aspect of asymmetry comes from the difference in interactions possible in these mediums [27].

Other research explores how to best share the view of a VR user with the external user. Ishii et al. [10] developed a CAVE-based visualization method that shares the VR experience with bystanders using translucent screens. MagicTorch [11] is another system that connects a VR user, an AR user, a tablet user, and the physical space. Users can have verbal communication in the co-located mode and gestures in the online mode. In OmniGlobeVR [12], the first-person view of the VR user is projected on a globe, in order for external collaborators to view the 360-degree video. Similarly, ShareVR [7] projects the VR scene elements on the ground, allowing non-HMD users to interact with the VR user in the same physical space. In contrast, TransceiVR [28] takes an application-agnostic approach, utilizing VR platform APIs to share the VR scene and provide asymmetric collaborative features without source code access. DreamStream [29] also takes a similar approach where it allows a spectator to view a 3D reconstruction of a VR scene. All these works require additional custom hardware devices or software on the viewer's end. However, WebTransceiVR uses web-based interfaces to make communication cross-platform and scalable, requiring minimal setup and no additional software for access. Our work also contributes to the study of sharing VR experience to the outside user by giving the external user control over the camera.

In addition, with the development of remote work during the pandemic, apps that enable multi-platform collaboration emerged in the commercial field, including Horizon Workrooms [4], Mozilla Hubs [15], and Spatial [24]. All of them support users to join via VR or web browser, which provides an environment for asymmetric VR interaction in groups. These social web-based applications are

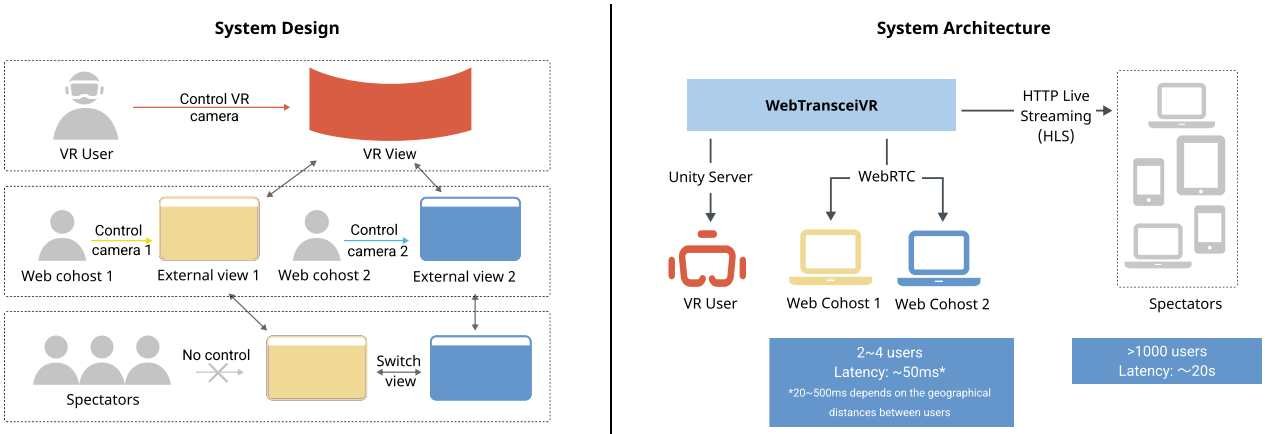

Figure 2: Three-tiered interaction system with VR host, non-VR co-hosts, and spectators

primarily for users meet in and explore virtual areas. However, our system is not focused on a particular type of application and can be applied to any application (education, industrial, or training) through the form of a toolkit. Compared to other platforms, the uniqueness of our work is that we compile it as a Unity package so that it can be integrated with any Unity-based application. This allows the content creator to use this system in different contexts and have more freedom in world-building. Unity is commonly used in VR development, and having our toolkit support Unity will allow for our system to be used widely. While our toolkit is written for Unity, we note that, with additional engineering, asymmetric interactions proposed in our work can be extended to other technology stacks. We use a browser to make it easier for external users to interact with the application without having to download anything, but the view for the spectator is rendered within the the VR user's program. This system that can be incorporated into existing VR applications to facilitate communication with asymmetric users and spectators without the constraints of web-based rendering solutions. Prior works have highlighted the advantages of such a *streaming approach* [29].

2.2 WebRTC and VR

A cross-platform, robust networking solution is required to achieve the synchronous interactions between VR and non-VR users. WebRTC (Web Real-Time Communication) [6] is a real-time peer-to-peer connection framework that has been explored in many video conferencing and educational scenarios. For example, USE Together [13] illustrates how WebRTC can be used with a star topology to allow multi-user collaboration on non-VR environments, where a central callee shares captured media to multiple callers, potentially from different platforms. Gunkel et al.'s 360-degree social VR experience [8] looks into how two VR users can share the same WebVR environment over WebRTC, and XRDirector [16] explores how a VR "actor" and an AR "camera" can interact with each other for film production through WebRTC. More recently, XRStudio [17] combines multi-user media sharing with VR/AR support, creating an educational environment where students can join a VR teacher using extended reality (XR) devices.

WebTransceiVR adopts a similar networking design in these prior works, but changes the implementation and delivery of the VR environment. In the examples above, most rely on the WebVR framework *A-Frame* to create the VR world, and the sharing of VR items is done through *A-Frame* network solutions, separating from the WebRTC data channel. This approach potentially limits the VR performance as WebVR is less capable in terms of graphics and interactions; it also introduces possible time lag between contents shared by WebRTC and WebVR. We use Unity and its Render Streaming functionality to avoid these problems, using Unity's full-fledged engine and integrated WebRTC support for better synchronous collaboration. Unity also allows us to increase rendering rate more easily compared to WebVR solutions; as pointed out by Vikberg, increasing rendering rate could efficiently reduce server-side latency for a WebRTC-based cloud XR system [30].

2.3 Network Streaming Considerations and Limitations

Optimizing the client/server communication channel as well as minimizing latency where possible are important goals for WebTransceiVR. Beginning with the limitations to connected clients accessing a single Unity server, Parthasarathy et al. [21] utilized a Unity server with which they connected a number of both physical and virtual clients that would move about the scene. Unsurprisingly, as the number of client connections increased, network throughput decreased and round-trip time (RTT) increased. Likewise, Pathak et al. [22] also observed in their WebXR application that frame rates decreased and latency increased significantly when only six users connected to the same server. These observations, though unsurprising, frame the goal of designing WebTransceiVR with optimizations that would maximize the number of connected clients while minimizing frame rate loss and latency spikes.

One such optimization is suggested by Pathak et al. [22] where the streamed video resolution from the server to the clients changes depending on network congestion, a process the authors call Adaptive Resolution Scheme. WebTransceiVR benefits from this adaptive resolution scheme using WebRTC's built-in congestion control,

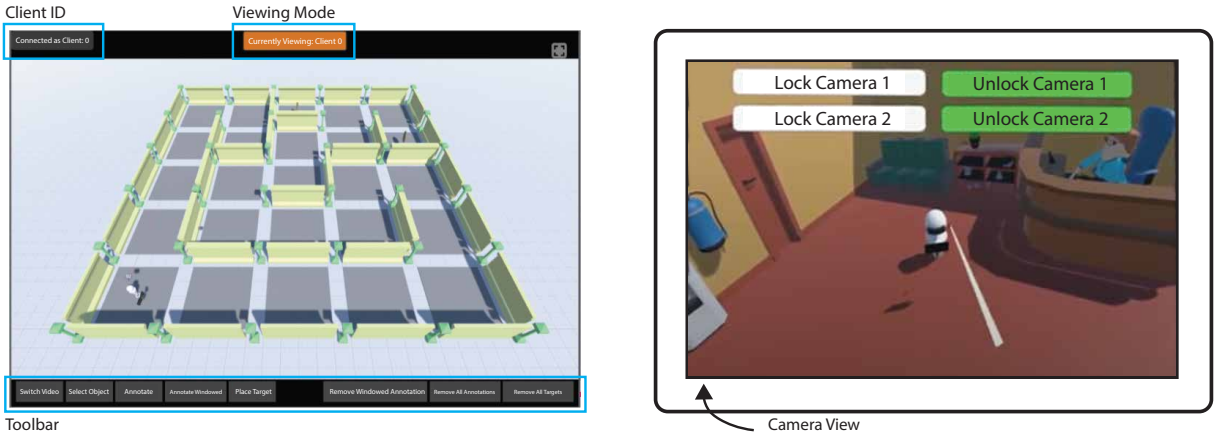

Figure 3: User Interface for the web co-host and for the VR tablet

which reduces the encoded video bitrate from the server when available network bandwidth decreases [30]. Additionally, monitoring the server load and adapting the video resolution to all connected clients when the server is overly taxed further optimizes frame rates and reduces latency for all client streams.

3 DESIGN AND IMPLEMENTATION

Our prototype aims to facilitate asymmetric collaboration and communication between a VR user and multiple remotely located non-VR users. We introduce a three-tiered system shown in Figure 2. WebTransceiVR allows devices to connect via two different operating modes over the internet to a device running a Unity scene along with the WebTransceiVR asset. The two modes are (1) connecting as a "co-host", which provides connected devices with their own scene camera which they can control along with scene interaction features, and (2) connecting as a spectator, which utilizes HTTP Live Streaming and Amazon Web Services (AWS) [2] backend services for easy scalability (to possibly over million viewers by leveraging AWS services). Spectator devices can only view the video streams rendered from scene cameras, and they have no control over the cameras, nor are they provided any scene interaction features.

This two-way approach balances between the scale of participants and latency; we provide real-time, highly interactive participation for a limited number of users, as well as a more traditional live-streaming experience that can be accessed by potentially thousands of spectators. From the VR user's perspective, such a design also enables better control over the influx of inputs from other users, by limiting their interaction to a few "co-hosts".

3.1 Interaction Design through Video Player

We designed and implemented several interaction features available to co-hosts (Fig. 3) in order to facilitate asymmetric communication within the virtual scene as seen in Fig. 4.

3.1.1 Select Object. (Fig. 4.d) This feature allows the co-host to select any object that is in the Unity scene utilizing raycast functionality. Clicking an object places an outline over the 3D mesh of the object in the scene, so that the non-VR user can draw the attention of a particular object to the VR user. Co-hosts can select and deselect multiple objects within the scene.

3.1.2 Place Target. (Fig. 4.c) Co-hosts can place a blue target object in the scene which is visible to all other spectators and the VR user. The co-host clicks in the scene viewport to place the target normally to the surface selected. This is helpful to guide the VR user to a particular location in the scene.

3.1.3 Scene Annotation. (Fig. 4.a) A scene annotation function allows for the non-VR co-host to draw an annotation over any part of the scene or objects, which is adjusted to be placed at the appropriate depth. The co-host sends the mouse coordinates to Unity which maps them to 3D coordinates based on the depth of the objects in the scene raycasted from the camera.

3.1.4 Windowed Annotation. (Fig. 4.b) A second type of annotation function is a windowed annotation. Instead of rendering an annotation within the scene, the windowed annotation feature allows the co-host to draw an annotation rendered on top of a snapshot of the non-VR user's view of the scene that is only displayed on the VR user's tablet, which is permanently attached to the user's right controller.

The above interactions inside browser require a tracking method for the mouse position over the video feed in order to navigate, properly select objects, place targets, and annotate the Unity scene. This tracking, and the processing for normalizing pixel coordinates over the video stream, are performed on the co-host client device prior to sending the coordinates to the custom interaction server running on the VR user's machine which executes the action. Non-VR co-hosts can remotely navigate the Unity scene with a free view camera controllable by mouse and keyboard.

3.2 Interaction Design through VR View

To allow better management of the virtual scene, we design additional features for the VR user to control the positions of the cameras representing the connected co-hosts. The VR user can grab the scene cameras and move or rotate them to any position they desire, hence change the co-hosts' viewpoints. This could be a very handy feature for tutorial purposes, where the VR user wants to

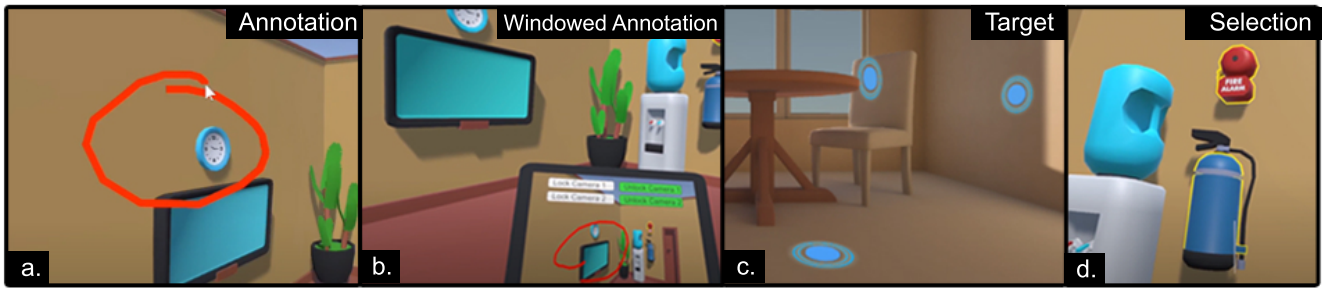

Figure 4: Interactive features for the co-host

display static angles or perspectives of the actions they are performing. We also provide the VR user with access to a virtual tablet (Fig. 3) that allows them to examine the cameras' views. Using this tablet view, the VR user can also lock the positions of the scene cameras and prevent non-VR co-hosts from moving their cameras around, providing better control over the virtual scene.

3.3 Network Architecture

We design a WebRTC-based network architecture to support communication between the Unity VR application and the web front, as shown in Fig. 5.

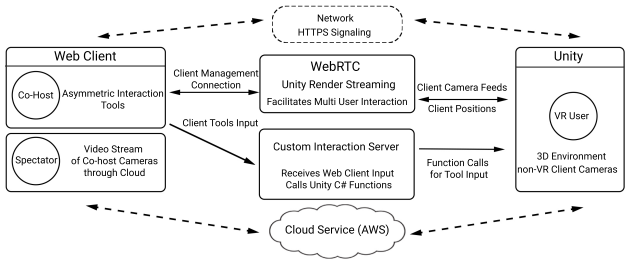

Figure 5: The underlying network architecture of WebTransceiVR (STUN and TURN servers not shown)

3.3.1 WebRTC and Unity Render Streaming. Devices connecting as co-hosts to the Unity server on the VR user's computer establishes a WebRTC peer-to-peer connection with the computer through the Unity Render Streaming package. The peer connection is established with help from three servers: a signaling server, a public STUN (Session Traversal Utilities for NAT) server that discovers IP addresses for devices, and a TURN (Traversal Using Relays around NAT) server that relays traffic when direct connection fails. A signaling server, deployed on AWS, serves as the central relay server that handles the initiation of the peer connection. The signaling server will relay offers from the server to the co-host client and vice versa; once offers and answers are exchanged, two multimedia channels will be created for the two cameras, streaming the rendered video from the VR program to the web frontend. A TURN server we deployed on AWS will automatically relay the streaming traffic if the direct peer connection becomes unstable.

3.3.2 Custom HTTP Interaction Server and WebSocket Relay Service. Co-hosts are provided with an array of interaction tools with which they can interact or mark-up objects in the Unity scene. The actions performed by the co-host client using these interactive features are relayed to the Unity machine by sending JavaScript GET/POST requests to a custom interaction server running inside the Unity application. Such a server implements a REST API, parses the requests from the client which includes identifying information such as the client's ID along with the operation that should be performed in the Unity scene and any appropriate parameters necessary to perform the action (such as mouse cursor position). The server is also responsible for keeping track of scene cameras' availability at a given time, appropriately handling clients that signal for disconnection from the server along with newly connecting clients requesting to be assigned a client ID and scene camera to control. To help with the complications of NAT and firewalls during connection, we implemented a relay service on the central signaling server utilizing the WebSocket protocol. This allows us to open up bidirectional communication through TCP connections, relaying requests and responses between the web client and the Unity application.

3.3.3 Spectator Connection. Spectators connect to WebTransceiVR using a different URL than devices wishing to connect as co-hosts. Spectators can view the video feed being rendered from the scene cameras, being able to freely switch between all available cameras. The rendered output streams of these cameras are the first broadcast from the VR user's machine to the AWS Elemental Live service using FFmpeg [3, 5]. The service then buffers and transcodes the video streams into multiple video resolutions. Connecting spectators can then view those video streams from the AWS storage using HTTP Live Streaming (HLS), and switching video sources in the browser allows users to view from different cameras. Building the spectator component on the cloud allows the VR user to scale the service on demand, allowing a very large group of spectators.

4 PILOT TESTING

We carried out pilot testing using demo scenes to get early feedback on the interface and interaction in WebTransceiVR. We recruited 4 pairs of users (VR user-External user) who used two Unity VR scenes: a maze scene and an office scene (Fig. 6). In the maze scene, the system generates a coin at a random position within it. The external user needs to guide the VR user to the coin, using features like annotation or place target. The office scene is more complex and

has various 3D models for the users to interact with. In this scene, the VR user and web co-host don't have a specific goal, but are encouraged to explore the environment together and communicate using features including annotation, select objects, etc. We used the two scenes to test the usability and the impact of WebTransceiVR on asymmetric collaboration between VR and external users.

Compiling the responses from users testing the WebTransceiVR interface, we were able to formulate the following key points of feedback with respect to the user interface. From the web co-host side, there seemed to be a learning curve for the users to learn the UI for controlling the camera. One co-host user suggested us to include a simple diagram showing how to control the camera in the scene on the co-host side. From the VR side, a user who had basic VR experience reported that the difficulty for completing the maze task is low, requiring low mental and physical demand to complete. Two of the four VR users reported that features like annotation and windowed annotations provided rich visual cues to navigate through the maze. We also observed that in the maze task, co-host users preferred to use the windowed annotation feature. Since this feature takes a snapshot of the non-VR user's view, it allows them to move above the maze, annotate a path from a birds-eye view, and share to the VR user a mini-map of the maze.

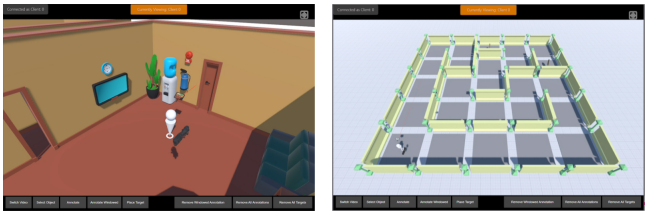

Figure 6: Unity office scene (left); Unity maze scene with coins to collect (right).

5 DISCUSSION AND LIMITATION

WebTransceiVR shows strong potential in addressing needs of asymmetric remote collaboration by enabling a scalable, cross-platform communication interaction. So far, our prototype (i) enables asymmetric interactions between one VR user and multiple non-VR users, (ii) supports a large number of spectators (possibly up to million spectators by leveraging AWS services) observing such interactions, and (iii) allows the above functions to operate remotely on standard web browser, with no additional software on the external user side. As mentioned earlier, this toolkit can facilitate typical asymmetric interaction scenarios. For instance, in a learning scenario, the teacher, teaching assistants, and students can be the VR user, the web co-hosts, and the spectators respectively. Another is with eSports: streamers, commentators, and spectators in a VR eSport game streaming session. In this setting, co-host cameras can be controlled by commentators from a freely moving perspective or an over-the-shoulder shot; thousands of spectators can view the eSport session from these perspectives.

A major limitation to our prototype is the number of co-hosts; currently, only two to three co-hosts are allowed. This is because of GPU accelerated video encoding, such as Nvidia NVENC is limited to only two concurrent hardware encoded streams[19] from the VR user's PC. While co-hosts have low-latency of interaction (~50ms), the spectator mode has a much higher latency (~20 seconds). This is the current industry standard for scalable streaming, and is due to real-time video buffering and transcoding amongst other process that happens on AWS servers. On the interaction side, the VR user currently has little control over the annotations and targets placed by the co-hosts; this can lead to spamming and poor navigation inside the VR scene.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose WebTransceiVR to address issues with asymmetric communication between a VR user and remotely located non-VR users. We explored a multi-user avenue of solving this problem and implemented a three-tiered system of one VR host, two non-VR co-hosts, and spectators. We believe that this system can help with VR collaboration in a variety of applications such as remote learning, live streaming, and VR conferencing.

Our pilot testing points to various avenues for future work. In the future, we plan to conduct user studies and understand the effectiveness of our current features in facilitating asymmetrical collaboration and communication. This includes testing the task completion time between pairs using WebTransceiVR and pairs under other conditions such as mirrored VR view.

The future design iterations of WebTransceiVR include enhancing co-hosts' awareness of each other and the VR user's awareness of different non-VR users. In our current version, the position of two co-hosts are both visualized with a camera model, and the annotations of co-hosts are shown in the same color. This means there are no visual cues for the VR user to distinguish between the two co-hosts. Features like different colors of annotations or viewing only one co-host's annotation or targets can be helpful in filtering the information. We will iterate on the current design of the tablet held by the VR user to address this issue. Besides, since each co-host can not access the view of another co-host, there can be features to further support the communication between co-hosts, such as utilizing spatial audio to enhance the co-presence and spatial awareness.

At this stage, the spectators are only passive viewers, but in the future we can design more features to encourage spectator participation. Meanwhile, it is possible for co-hosts to visualize and moderate interactions by a large number of spectators. For instance, one can have spatial polls where spectators can select parts of a scene where an action is to be executed by the VR user. Such interactions are useful and remain unexplored in prior literature. We hope to study these using WebTransceiVR in the future.

ACKNOWLEDGMENTS

We thank pilot test participants for their time and feedback. The office model used in Fig. 1, 3, 4, 6 is sourced from "Low Poly Office Pack: Characters & Props" by Polygonal Mind [23] (Standard Unity Asset Store EULA license). The camera model used in Fig. 1 is sourced from "Movie Camera" by Adobe Inc. [1] (Adobe Education License). The maze model used in Fig. 3, 6 is sourced from "Maze Generator" by styanton [26] (Standard Unity Asset Store EULA license). The outline model used in Fig. 4 is sourced from "Quick

Outline" by Chris Nolet [18] (Standard Unity Asset Store EULA license).

REFERENCES

[1] Adobe Inc. 2022. Movie Camera. https://substance3d.adobe.com/assets/allassets/eb9281d40787949b272ee449014edba1481a7c59

[2] Amazon Web Services. 2022. Amazon Web Services. https://aws.amazon.com/

[3] Amazon Web Services. 2022. AWS Elemental Live. https://aws.amazon.com/elemental-live/

[4] Facebook Technologies. 2021. Horizon Workrooms. https://www.oculus.com/workrooms/

[5] FFmpeg Developers. 2022. FFmpeg. https://www.ffmpeg.org/

[6] Google Developers. 2022. WebRTC. https://webrtc.org/

[7] Jan Gugenheimer, Evgeny Stemasov, Julian Frommel, and Enrico Rukzio. 2017. ShareVR: Enabling Co-Located Experiences for Virtual Reality between HMD and Non-HMD Users. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 4021–4033. https://doi.org/10.1145/3025453.3025683

[8] Simon Gunkel, Martin Prins, Hans Stokking, and Omar Niamut. 2017. WebVR meets WebRTC: Towards 360-degree social VR experiences. In *2017 IEEE Virtual Reality (VR)*. Institute of Electrical and Electronics Engineers, Los Angeles, CA, USA, 457–458. https://doi.org/10.1109/VR.2017.7892377

[9] Ionized Studios. 2017. IronWolf VR. https://www.ionized.tech/

[10] Akira Ishii, Masaya Tsuruta, Ippei Suzuki, Shuta Nakamae, Junichi Suzuki, and Yoichi Ochiai. 2019. Let Your World Open: CAVE-Based Visualization Methods of Public Virtual Reality towards a Shareable VR Experience. In *Proceedings of the 10th Augmented Human International Conference 2019* (Reims, France) *(AH2019)*. Association for Computing Machinery, New York, NY, USA, Article 33, 8 pages. https://doi.org/10.1145/3311823.3311860

[11] Jiabao Li, Honghao Deng, and Panagiotis Michalatos. 2017. MagicTorch: A Context-Aware Projection System for Asymmetrical VR Games. In *Extended Abstracts Publication of the Annual Symposium on Computer-Human Interaction in Play* (Amsterdam, The Netherlands) *(CHI PLAY '17 Extended Abstracts)*. Association for Computing Machinery, New York, NY, USA, 431–436. https://doi.org/10.1145/3130859.3131341

[12] Zhengqing Li, Theophilus Teo, Liwei Chan, Gun Lee, Matt Adcock, Mark Billinghurst, and Hideki Koike. 2020. OmniGlobeVR: A Collaborative 360-Degree Communication System for VR. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference* (Eindhoven, Netherlands) *(DIS '20)*. Association for Computing Machinery, New York, NY, USA, 615–625. https://doi.org/10.1145/3357236.3395429

[13] Laurent Lucas, Hervé Deleau, Benjamin Battin, and Julien Lehuraux. 2017. USE Together, a WebRTC-Based Solution for Multi-user Presence Desktop. In *Cooperative Design, Visualization, and Engineering*, Yuhua Luo (Ed.). Springer International Publishing, Cham, 228–235.

[14] Stefan Marks, David White, and Manpreet Singh. 2017. Getting up Your Nose: A Virtual Reality Education Tool for Nasal Cavity Anatomy. In *SIGGRAPH Asia 2017 Symposium on Education* (Bangkok, Thailand) *(SA '17)*. Association for Computing Machinery, New York, NY, USA, Article 1, 7 pages. https://doi.org/10.1145/3134368.3139218

[15] Mozilla. 2021. Mozilla Hubs. https://hubs.mozilla.com/

[16] Michael Nebeling, Katy Lewis, Yu-Cheng Chang, Lihan Zhu, Michelle Chung, Piaoyang Wang, and Janet Nebeling. 2020. XRDirector: A Role-Based Collaborative Immersive Authoring System. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3313831.3376637

[17] Michael Nebeling, Shwetha Rajaram, Liwei Wu, Yifei Cheng, and Jaylin Herskovitz. 2021. XRStudio: A Virtual Production and Live Streaming System for Immersive Instructional Experiences. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 107, 12 pages. https://doi.org/10.1145/3411764.3445323

[18] Chris Nolet. 2018. Quick Outline. https://assetstore.unity.com/packages/tools/particles-effects/quick-outline-115488

[19] Nvidia Developers. 2022. Video Encode and Decode GPU Support Matrix. https://developer.nvidia.com/video-encode-and-decode-gpu-support-matrix-new

[20] Odd Raven Studios. 2018. Carly and the Reaperman - Escape from the Underworld. https://www.oddravenstudios.com/

[21] Venkatakrishnan Parthasarathy, Anderson Augusto Simiscuka, Noel O'Connor, and Gabriel-Miro Muntean. 2020. Performance Evaluation of a Multi-User Virtual Reality Platform. In *2020 International Wireless Communications and Mobile Computing (IWCMC)*. Institute of Electrical and Electronics Engineers, St. Raphael Resort, Limassol, Cyprus, 934–939. https://doi.org/10.1109/IWCMC48107.2020.9148390

[22] Rishabh Pathak, Anderson Augusto Simiscuka, and Gabriel-Miro Muntean. 2021. An Adaptive Resolution Scheme for Performance Enhancement of a Web-based Multi-User VR Application. In *2021 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*. Institute of Electrical and Electronics Engineers, Chengdu, China, 1–6. https://doi.org/10.1109/BMSB53066.2021.9547069

[23] Polygonal Mind. 2018. Low Poly Office Pack: Characters & Props. https://assetstore.unity.com/packages/3d/characters/low-poly-office-pack-characters-props-119386

[24] Spatial Systems. 2016. Spatial. https://spatial.io/

[25] Steel Crate Games. 2015. Keep Talking and Nobody Explodes. https://keeptalkinggame.com/

[26] styanton. 2015. Maze Generator. https://assetstore.unity.com/packages/tools/modeling/maze-generator-38689

[27] Balasaravanan Thoravi Kumaravel and Bjoern Hartmann. 2022. Interactive Mixed-Dimensional Media for Cross-Dimensional Collaboration in Mixed Reality Environments. *Frontiers in Virtual Reality* 3 (2022). https://doi.org/10.3389/frvir.2022.766336

[28] Balasaravanan Thoravi Kumaravel, Cuong Nguyen, Stephen DiVerdi, and Bjoern Hartmann. 2020. TransceiVR: Bridging Asymmetrical Communication Between VR Users and External Collaborators. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 182–195. https://doi.org/10.1145/3379337.3415827

[29] Balasaravanan Thoravi Kumaravel and Andrew D Wilson. 2022. DreamStream: Immersive and Interactive Spectating for VR. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3491102.3517508

[30] Esa Vikberg. 2021. *Optimizing WebRTC for Cloud Streaming of XR*. Master's thesis. Aalto University. School of Science. http://urn.fi/URN:NBN:fi:aalto-202108298611

[31] Vinlia Games. 2017. Eye In The Sky. https://www.vinliagames.com/

[32] Vreal Inc. 2022. Next generation streaming technology for XR. https://vreal.net/