Open-Sourcing VR2Gather: A Collaborative Social VR System for Adaptive Multi-Party Real Time Communication

Jack Jansen[*]
Thomas Röggla[*]
jack@cwi.nl,t.roggla@cwi.nl
Centrum Wiskunde & Informatica
The Netherlands

Silvia Rossi
Irene Viola
s.rossi@cwi.nl,i.viola@cwi.nl
Centrum Wiskunde & Informatica
The Netherlands

Pablo Cesar
p.s.cesar@cwi.nl
Centrum Wiskunde & Informatica
TU Delft
The Netherlands

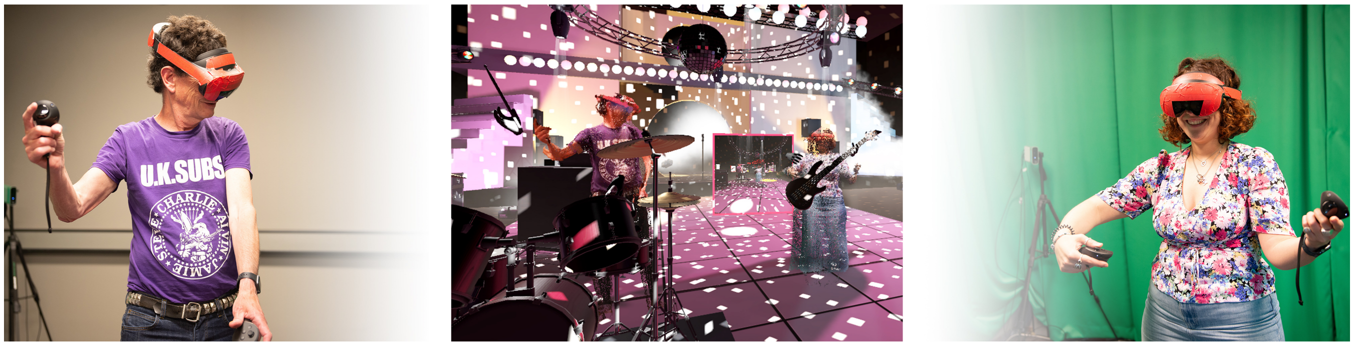
Figure 1: Playing instruments together virtually with VR2Gather: in the center, the virtual experience of two users playing drums and guitar while on the sides their physical setup.

Abstract

Social Virtual Reality is envisioned to transform how individuals communicate remotely, offering a sense of immersion and co-presence within a virtual space. Current platforms enabling remote social interactions rely on synthetic user representations. We address this limitation by enabling realistic human representation through volumetric content capture, encoding and transmission. Specifically, we present an extended version of VR2Gather, now a fully open source Unity package, available at https://github.com/cwi-dis/VR2Gather-acmmm-oss. Our platform is a customisable system to transmit volumetric content in a multi-party real-time environment, easy to integrate into existing applications.

CCS Concepts

• **Computing methodologies → Virtual reality**; • **Information systems → Multimedia streaming**; • **Human-centered computing →** *Collaborative interaction.*

Keywords

Open Source Software, Social Virtual Reality, Communication System, Real-time Volumetric Capture, Realistic Point Clouds

[*]Both authors contributed equally to this research.

This work is licensed under a Creative Commons Attribution International 4.0 License.

MM '24, October 28-November 1, 2024, Melbourne, VIC, Australia
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0686-8/24/10
https://doi.org/10.1145/3664647.3685515

ACM Reference Format:
Jack Jansen, Thomas Röggla, Silvia Rossi, Irene Viola, and Pablo Cesar. 2024. Open-Sourcing VR2Gather: A Collaborative Social VR System for Adaptive Multi-Party Real Time Communication. In *Proceedings of the 32nd ACM International Conference on Multimedia (MM '24), October 28-November 1, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3664647.3685515

1 Introduction

Social Virtual Reality (VR) enhances immersive experiences by transforming how individuals interact with media content, surpassing traditional video technology. These technologies are envisioned to lead the next generation of virtual worlds [4]. VR is already changing traditional ways to communicate remotely, placing users at the center and offering a sense of immersion and novel interaction possibilities. Social VR goes a step further by enabling the virtual co-presence of multiple users within the same virtual space and allowing for interactions with others or virtual objects.

Social VR platforms such as *Roblox*[1], *Mozilla Hubs*[2] and *Ubiq* [2] are already available to general audiences, providing immersive and remote social interactions. For instance, *Roblox* supports in-game content creation and distribution while *Mozilla Hubs* offers easy design and deployment, giving users the possibility to create and customize their own social VR spaces. *Ubiq* is a research-oriented Unity framework that includes standard features of social VR platforms such as avatar personalisation and shared objects to prototype immersive experiences. Some solutions are also extensible to enable large-scale experimentation [13] and are either open-source or offer free solutions for small projects [1]. However, all these platforms only support synthetic avatar representations,

[1]Roblox: https://www.roblox.com/
[2]Mozilla Hubs: https://hubs.mozilla.com

compromising both physical and emotional presence in immersive experiences [3, 9, 10]. To overcome this, *VR2Gather* is a social VR multi-party real-time communication system which enables realistic digital human representation based on point clouds which capture live participants [5, 15].

Figure 1 shows an experience running on the VR2Gather platform: two remote participants, physically separated and equipped with a headset and controllers (left and right side of Figure 1), can communicate and play various virtual instruments together on the same stage, viewing realistic representations of themselves (middle part of Figure 1). The instruments (e.g. guitar and drums) are virtual objects, shared and synchronized across all participants. An initial version of VR2Gather was presented in [15] as a monolithic Unity application, making integration into third party applications more difficult. Moreover, not all components of that system were open source: the orchestrator responsible for facilitating communication between participants was closed-source and a few internal dependencies were only available under a commercial license.

In this paper, we present an extended and reworked version of VR2Gather, which is now a fully open source Unity package, provided under permissive licenses (MIT and BSD 2-Clause), available at https://github.com/cwi-dis/VR2Gather-acmmm-oss. Being a Unity package makes the integration into existing social VR applications easier. All previously closed source components of the system have either been replaced by open source equivalents or were re-implemented from the ground up. VR2Gather experiences do not rely on a central cloud-based game engine. Instead, each participant runs a local copy of the whole application while communication and synchronization are handled through a central, experience-agnostic cloud component that manages forwarding of control messages, point cloud streams and conversational audio between participants. Finally, our updated framework includes a complete code overhaul, enabling the customization of assemblies[3] to better meet requirements of different applications. Specifically, VR2Gather includes an assembly to process point clouds for acquisition, compression and rendering; a data transport assembly with support for different transmission protocols (i.e. SocketIO, TCP and DASH are available by default, with WebRTC under development) for data transfer over the Internet; and a receiver assembly that takes the transmitted packets and renders them in the virtual scene. Our platform is already being used by an international community in various use cases across different domains such as cultural heritage, healthcare and entertainment [15]. This shows its versatility and adaptability in addressing diverse application needs also in terms of system requirements. By offering an easier-to-integrate and fully open-source version, VR2Gather can become an invaluable tool for researchers and practitioners across different fields, facilitating experimentation at large scale and fostering collaboration and innovation in the realm of immersive technologies.

2 System Architecture

Figure 2 outlines the general architecture of VR2Gather. Two or more geographically separated users, equipped with HMDs and controllers, are captured by one or more RGBD cameras. Thanks to VR2Gather, participants can experience the same virtual scene in which they can communicate and interact with objects. VR2Gather is comprised of a local application running at the client side, which allows users to share a virtual scene with a representation of themselves and shared objects; communication and synchronization are handled instead through a cloud component of the system. Specifically, our VR2Gather platform is a Unity package comprised of a series of assemblies, each one dedicated to a specific purpose such as: *user representation*, *shared objects and actions*, *orchestration and session management* and *live media transport*. Each assembly also exposes a well-defined abstract interface, enabling reuse and replacement of individual components with custom implementations as needed. In the following, we provide more in depth detail of the principal parts of our VR2Gather system.

In terms of *user representations*, different methods are supported by VR2Gather, such as synthetic avatars, point clouds or invisible spectators. For example Figure 1 is a screenshot taken by a third user who did not have a representation but merely acted as a spectator. This also implies that our platform can support different capture systems composed by a variable number of cameras, as described in [11]. All point cloud handling, including capturing, tiling, compression, decompression, synchronization and rendering are implemented in a separate Unity package, *cwipc_unity*[4] and the underlying native package *cwipc*[5] which is based on the MPEG Anchor codec presented in [8]. The package allows the creation of pipelines for point cloud streams (represented by the thick lines in Figure 2) with minimal copying of the actual point cloud data, even across language boundaries (C to Python or C#). The point cloud pipeline architecture is described in more detail in [15], while additional information on tiling and multi-quality techniques are given in [14]. Note that *cwipc_unity* is distributed as a separate standalone package because it has a wider application area than only social VR: it can also be used to render pre-recorded point cloud streams, or capture and display point clouds for other types of Unity applications.

Each user runs their own, independent instance of the VR2Gather Unity application, which is responsible for rendering the virtual scene and managing the *shared objects* and *actions taken* within it. Shared objects in this context refer to virtual objects within the scene that users can interact with. For instance, in Figure 1, the bass guitar is a shared object that users can pick up and play by hitting the strings to produce sound. This behavior and these actions are synchronised across all users by a central session manager.

The *Orchestrator* is the central cloud component of VR2Gather that is responsible for *orchestration and session management*. It consists of the Orchestrator itself and optionally an external Selective Forwarding Unit (SFU), as shown in Figure 2. This cloud component is a server application deployed on either one of the clients or an external machine reachable by all participating clients. It can be deployed on a headless server running any operating system, with the only requirement being a stable and fast internet connection. In more detail, the orchestrator is responsible for keeping track of users and their associated data, creation and destruction of sessions and their data as well as distributing updates on positions

[3]The term *assembly* in the context of Unity refers to a unit of modularisation, bundling together code, materials, assets and/or prefabs and allowing their reuse.

[4]https://github.com/cwi-dis/cwipc_unity
[5]https://github.com/cwi-dis/cwipc

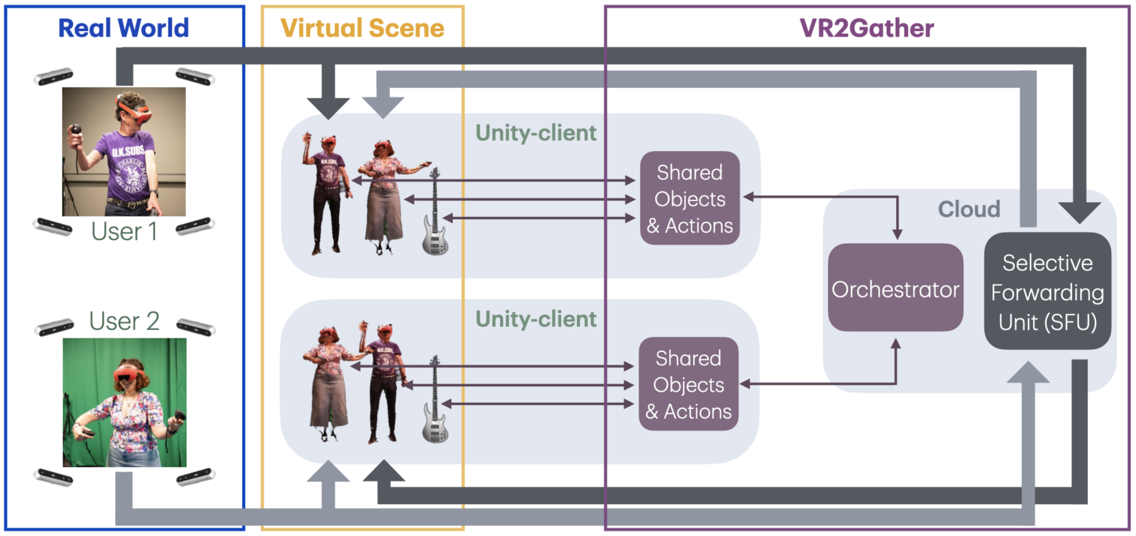

Figure 2: VR2Gather architecture: two users, physically separated, are captured by depth cameras and their representations are broadcast over the Internet. A local copy of VR2Gather on the client side allows sharing a virtual scene; communication and synchronization are handled by cloud components. Thick and thin arrows represent media streams and control messages.

and movements of users and interactive objects to all users within a scene. The SFU component is used for exchanging binary data between users, more specifically, it handles transmission of point cloud data and audio between users within a session over an existing Internet connection. In a default setup, the orchestrator also handles the functions of the SFU via its Socket.IO connection, but external SFUs providing support for transmission over raw TCP or DASH can be configured. Support for media transmission using the WebRTC protocol is currently under development. More details on the provided transport protocols are given in [15].

This separation of control and live media messaging allows each component to be replaced independently. Moreover, all VR2Gather-specific business logic is contained entirely within the clients, meaning there is no strict requirement for clients to be implemented in Unity.

3 Implementation Details

VR2Gather is based on the Unity platform and is implemented in C#, making it usable on all major operating systems. It ships as a package with the intention that end users can integrate the functionality into their own applications. Therefore, the package is divided into a series of assemblies, each capturing a distinct functional area. The package also ships with a series of prefabs[6] that make it easy to get started quickly by already providing scenes that take care of user settings, login and user representation setup.

The assemblies managing user representation offer different ways a user can be represented in the virtual space from digital avatars to fully-formed point clouds, taking advantage of functionality offered by `cwipc_unity`. The package *cwipc_unity* is a wrapper around the `cwipc` open-source point cloud library. The *cwipc* native package core is implemented in C++, with the API also available in C, C#, Python and Jupyter. It is available for Windows, Mac and Linux, and to a limited extent for Android. Binary installers are available. Similarly, the *cwipc_unity* package is available for Windows, Mac and Linux. An Android version is also planned.

Other key assemblies of the VR2Gather package are the ones responsible for session and user management, broadcasting of object positions and custom events as well as media transport. By default, VR2Gather ships with code to interact with the orchestrator for session management and management of custom events triggered within the virtual scene. However, the system is designed in such a way, that end-users can provide their own session management solution and write code to interact with VR2Gather by implementing the corresponding abstract interfaces. The orchestrator itself is implemented in Typescript, using Socket.IO in combination with its underlying transport protocol Engine.IO. This allows VR2Gather clients to communicate even when behind NAT firewalls, such as in a home setup. Socket.IO provides a wrapper around WebSocket connections with optional HTTP long polling fallback. Its architecture is designed around an event system, where clients send named events with optional data payloads to a central server and the server updates its internal state accordingly. New sessions are registered with a unique name and an associated *scenario* which designates the virtual scene rendered within the session. Once this data is registered to the orchestrator’s internal data tree, participating users can join the session. The orchestrator takes care of continuously broadcasting updates on positions and movements of users and interactive objects to all users within a session. Events triggered within a session, e.g. the press of a button, are relayed from the originating user through the orchestrator to the session creator, which then triggers the broadcast of the event to all other users, again, through the orchestrator.

Transmission of point cloud and audio streams can also be handled by the orchestrator. Each user, upon joining a session, registers a number of streams that they are able to provide with the orchestrator. Other users can then subscribe to these streams to receive the data they are interested in, thus effectively establishing a publish/subscribe mechanism. The orchestrator will then handle

[6]Unity term for prefabricated scenes and associated behaviors.

routing of the binary data from senders to interested parties. By default, this data transmission is handled through the Socket.IO protocol over its underlying WebSocket connection. It is also possible, to hand off this task to external SFUs, taking advantage of different protocols, such as WebRTC, DASH or raw TCP. The transmission protocol to use can be specified during session creation.

4 Use Cases

As described in [15], VR2Gather has been employed in several use cases proving the versatility and adaptability of our platform to different system requirements (e.g. local or public networks, single- or multi-camera setups) and applications across several domains (e.g. healthcare, cultural heritage or entertainment).
New possibilities in the healthcare sector have been demonstrated with VR2Gather through remote consultation between a doctor and an (acting) injured patient on the street, both equipped with mainstream mobile devices and relying on a public 5G network [15]. Museum experiences can also be enriched thanks to our platform, as shown in Mediascape XR [12]. A historical costume of a Dutch pop singer, too fragile to be exhibited in public, has been made accessible digitally to visitors who were able to wear and experience it virtually in a historical concert setting. In this case, the immersive experience was set up on-premise in a museum, with direct network connections and a multi-camera setup to ensure the best visual quality and lowest latency. In terms of entertainment, our VR2Gather platform offers numerous possibilities for connecting with others and engaging in shared virtual experiences. For example, relying on VR2Gather, two to five users with different devices (i.e. HMDs and computers) were captured from multiple angles to ensure a photorealistic representation and were able to experience a 3D immersive mystery murder movie [7]. The users were co-present in the scene with the movie characters and helped in solving the crime together. Recently, VR2Gather was also used to evaluate the effect of experiencing a VR theater lobby in a social setting, as opposed to experiencing it alone [6]. Participants were traversing different virtual rooms, captured in real-time by depth cameras and represented in the virtual world as point clouds.

5 Conclusion

In this paper, we proposed an easier-to-integrate and fully open-source version of VR2Gather, originally presented in [15]. Our platform is now a Unity package that allows the creation of immersive social VR applications. Participants in a VR2Gather-based experience can be represented as live volumetric video and see themselves as they are captured live, thereby allowing more realistic social interaction than in avatar-only based systems. We presented its key customizable assemblies and showcased several use cases where the proposed system has already been applied. Currently, we are distributing a single Unity package of our VR2Gather platform, but we are planning to turn it in a set of packages, one per each main assembly, to allow end users to only use components which they are interested in. Moreover, since Unity has gained a large foothold in the world of independent game developers and user experience researchers, we will further extend our software to support freely available network libraries, such as Photon[7], which provides solutions to handle complex multiplayer functionalities such as client-server communication and synchronization.

[7]Photon: www.photonengine.com/pun

Acknowledgements

This work was supported by the European Union as part of the Horizon Europe Framework Program under grant agreement No 101070109 (*TRANSMIXR*), No 101135556 (*INDUX-R*) and the European Union Horizon 2020 research and innovation programme under grant agreement No 870610 (*TRACTION*). The authors would like to thank all the colleagues who contributed to this project.

References

[1] A. Abilkaiyrkyzy, A. Elhagry, F. Laamarti, and A. Elsaddik. 2023. Metaverse key requirements and platforms survey. *IEEE Access* (2023).

[2] S. J. Friston, B. J. Congdon, D. Swapp, L. Izzouzi, K. Brandstätter, D. Archer, O. Olkkonen, F. J. Thiel, and A. Steed. 2021. Ubiq: A system to build flexible social virtual reality experiences. In *Proceedings of the 27th ACM symposium on virtual reality software and technology*. 1–11.

[3] S. Gunkel, R. Hindriks, K. M. El Assal, H. M. Stokking, S. Dijkstra-Soudarissanane, F. Haar, and O Niamut. 2021. VRComm: an end-to-end web system for real-time photorealistic social VR communication. In *Proceedings of the 12th ACM multimedia systems conference*. 65–79.

[4] Hupont Torres Isabelle, Charisi Vasiliki, De Prato Giuditta, Pogorzelska Katarzyna, Schade Sven, Kotsev Alexander, Sobolewski Maciej, Duch Brown Nestor, Calza Elisa, Dunker Cesare, and Others. 2023. *Next Generation Virtual Worlds: Societal, Technological, Economic and Policy Challenges for the EU.* Technical Report. Joint Research Centre.

[5] J. Jansen, S. Subramanyam, R. Bouqueau, G. Cernigliaro, M. : Cabré, F. Pérez, and P. Cesar. 2020. A pipeline for multiparty volumetric video conferencing: transmission of point clouds over low latency DASH. In *Proceedings of the 11th ACM Multimedia Systems Conference*. 341–344.

[6] S. Lee, I. Viola, S. Rossi, Z. Guo, I. Reimat, K. Ławicka, A. Striner, and P. Cesar. 2024. Designing and Evaluating a VR Lobby for a Socially Enriching Remote Opera Watching Experience. *IEEE Transactions on Visualization and Computer Graphics* (2024).

[7] J. Li, S. Subramanyam, J. Jansen, Y. Mei, I. Reimat, K. Ławicka, and P. Cesar. 2021. Evaluating the user experience of a photorealistic social VR movie. In *2021 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 284–293.

[8] Rufael Mekuria, Kees Blom, and Pablo Cesar. 2016. Design, implementation, and evaluation of a point cloud codec for tele-immersive video. *IEEE Transactions on Circuits and Systems for Video Technology* 27, 4 (2016), 828–842.

[9] R. Mekuria, P. Cesar, I. Doumanis, and A. Frisiello. 2015. Objective and subjective quality assessment of geometry compression of reconstructed 3D humans in a 3D virtual room. In *Applications of Digital Image Processing XXXVIII*, Vol. 9599. SPIE, 537–549.

[10] Mario Montagud, Jie Li, Gianluca Cernigliaro, Abdallah El Ali, Sergi Fernández, and Pablo Cesar. 2022. Towards socialVR: evaluating a novel technology for watching videos together. *Virtual Reality* 26, 4 (2022), 1593–1613.

[11] I. Reimat, E. Alexiou, J. Jansen, I. Viola, S. Subramanyam, and P. Cesar. 2021. CWIPC-SXR: Point cloud dynamic human dataset for social XR. In *Proceedings of the 12th ACM Multimedia Systems Conference*. 300–306.

[12] I. Reimat, Y. Mei, E. Alexiou, J. Jansen, J. Li, S. Subramanyam, I. Viola, J. Oomen, and P. Cesar. 2022. Mediascape XR: A Cultural Heritage Experience in Social VR. In *Proceedings of the 30th ACM International Conference on Multimedia*. 6955–6957.

[13] Thomas Röggla, David A Shamma, Julie R Williamson, Irene Viola, Silvia Rossi, and Pablo Cesar. 2024. A Platform for Collecting User Behaviour Data during Social VR Experiments Using Mozilla Hubs. In *Proceedings of the 16th International Workshop on Immersive Mixed and Virtual Environment Systems*. 41–44.

[14] S. Subramanyam, I. Viola, J. Jansen, E. Alexiou, A. Hanjalic, and P. Cesar. 2022. Evaluating the impact of tiled user-adaptive real-time point cloud streaming on VR remote communication. In *Proceedings of the 30th ACM International Conference on Multimedia*. 3094–3103.

[15] I. Viola, J. Jansen, S. Subramanyam, I. Reimat, and P. Cesar. 2023. VR2Gather: A collaborative social VR system for adaptive multi-party real-time communication. *IEEE MultiMedia* (2023).