Multi-Sensor Capture and Network Processing for Virtual Reality Conferencing

Sylvie Dijkstra-Soudarissanane, Karim El Assal, Simon Gunkel, Frank ter Haar, Rick Hindriks,
Jan Willem Kleinrouweler, Omar Niamut
TNO
Den Haag, the Netherlands
firstname.lastname@tno.nl

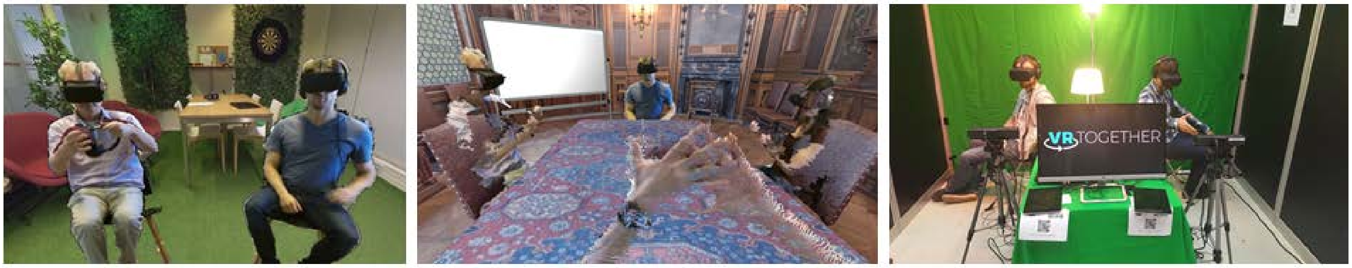

Figure 1: Examples of shared VR environments.

ABSTRACT

Recent developments in key technologies like 5G, Augmented and Virtual Reality (AR/VR) and Tactile Internet result in new possibilities for communication. Particularly, these key digital technologies can enable remote communication and collaboration in remote experiences. In this demo, we work towards 6-degrees of freedom (DoF) photo-realistic shared experiences by introducing a multi-view multi-sensor capture end-to-end system. Our system acts as a baseline end-to-end system for capture, transmission and rendering of volumetric video of user representations. To handle multi-view video processing in a scalable way, we introduce a Multi-point Control Unit (MCU) to shift processing from end devices into the cloud. MCUs are commonly used to bridge videoconferencing connections, and we design and deploy a VR-ready MCU to reduce both upload bandwidth and end-device processing requirements. In our demo, we focus on a remote meeting use case where multiple people can sit around a table to communicate in a shared VR environment.

CCS CONCEPTS

• **Information systems** → **Multimedia information systems**; • **Human-centered computing** → **Virtual reality**; • **Networks** → *Cloud computing*.

KEYWORDS

telepresence, VR, social VR, videoconferencing

MMSys '19, June 18–21, 2019, Amherst, MA, USA
© 2019 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-6297-9/19/06.
https://doi.org/10.1145/3304109.3323838

ACM Reference Format:
Sylvie Dijkstra-Soudarissanane, Karim El Assal, Simon Gunkel, Frank ter Haar, Rick Hindriks, Jan Willem Kleinrouweler, Omar Niamut. 2019. Multi-Sensor Capture and Network Processing for Virtual Reality Conferencing. In *10th ACM Multimedia Systems Conference (MMSys '19), June 18–21, 2019, Amherst, MA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3304109.3323838

1 INTRODUCTION

As traveling comes with costs both in money and time, as well as a drastic impact on our ecological footprint, there is a strong need to make communication and remote collaboration as transparent and easy as possible. Current means of remote communication (e.g. Skype and FaceTime) have limitations because communication is more than an exchange of words; it forms the basis for sharing knowledge and experiences between people.

New possibilities for communication are brought about by recent developments in 5G, Augmented and Virtual Reality (AR and VR) and Tactile Internet. While remote communication systems may be improved by increasing the quality of auditory and visual media, decreasing network transmission delays, and adding multiple sensory modalities like tactile and haptics, it is still unclear if VR can be of benefit; in an analysis of whether and how immersive VR can enhance our lives [8], Slater and Sanchez-Vives point out that when it comes to remote collaboration, although we assume that travelling to meet a person is still the best choice to achieve high-quality conversations, "probably some readers of the article would have experienced the situation of several hours of travel to attend or speak at a 1-h meeting and then to travel home shortly afterward – sometimes wondering what the point of it all might have been".

Current technologies for remote immersive communication and participation face limitations with respect to capture, processing, transmission and rendering of multimodal media over mobile networks. In particular, creating high-quality and immersive shared VR experiences between remote participants puts a significant demand on the communication infrastructure. The TogetherVR platform infrastructure presented in this demonstrator provides multi-sensor capture and in-network orchestration and processing, to resolve three major technical challenges; i) optimize the user capture to allow a variety of end devices with different constraints to participate in, and fully benefit from, shared VR experiences; ii) control (e.g. synchronize, transmit, process) current and emerging immersive media in a shared VR system to allow large numbers of users (>100) in one communication session; and iii) optimize the composition of different media objects in the client device, user representation and VR environments in order to reduce the system complexity. For our demo, we focus on a remote meeting use case, where multiple people (up to four) can sit around a table to communicate in a shared VR environment.

2 RELATED WORK

Remote collaboration has been an extensive topic of research and VR-based collaboration is addressed in [2, 5]. However, understanding how to build robust end-to-end systems that can support multiple user scenarios as well as cater for different limitations in end devices has not been thoroughly addressed. With respect to capture, we are particularly interested in high-quality low-cost capture solutions that provide sparse views, e.g. as few as two or three commodity RGB-D sensors. For example, [1] proposes an integrated approach for the calibration and registration of colour and depth (RGB-D) sensors into a joint coordinate system for 3D telepresence applications. The method employs a tracked checker-board to establish a number of correspondences between positions in colour and depth camera space and in world space. While this approach reduces reconstruction latency by omitting image rectification processes during runtime, the setup and calibration phase still requires users to have sufficient technical knowledge to install sensors with the correct spatial alignment and to run the calibration process. And while [10] proposes a simplified calibration phase, we consider the 4-sensor setup still too complex for our system.

For addressing scalability in network and end-device, we look towards multipoint control units (MCU) [11], i.e. conference servers that support multi-party multimedia conferences and coordinate the distribution of audio, video, and data streams amongst the multiple participants in a video conference. An MCU can alleviate bottlenecks in bandwidth and performance, e.g. by reducing the CPU load on client devices [9]. While [4] proposes a novel telepresence platform for immersive video conferencing based on a distributed architecture with a stream forwarding approach, the usage of an MCU for shared VR has not yet been explored.

In [6] we introduced TogetherVR as a modular platform based on web technologies, that allows both to easily create VR experiences that are social and to consume them with off-the-shelf hardware. The platform included browser screen share functionality to provide flexibility in the type of shared application within the VR room. In [3] we scaled up communication between participants to three persons and explored integration of new media formats to represent users as 3D point clouds. Compared to our earlier work, this demo incorporates new volumetric video formats through multi-sensor capture and addresses scalability through design and deployment of a VR MCU.

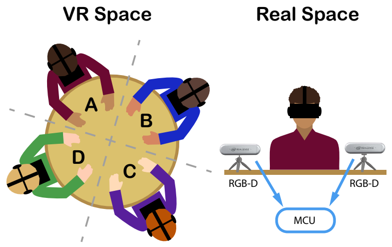

Figure 2: A schematic view of a four users setup in a virtual meeting (left) and the capture with two RGB-D sensors per user in the real space (right)

3 MULTI-SENSOR CAPTURING AND RENDERING THE 3D ENVIRONMENT

Our aim is to create a shared VR environment (see Figure 1), where participants get the feeling of being in the presence of, and interacting with, other persons at a remote location. That is, we want to provide true shared and collaborative 6 Degrees Of Freedom (6-DoF) experiences, using photo-realistic and volumetric human representations in a format that can be easily captured, compressed and transported to current and upcoming VR devices.

Point clouds offer a natural representation of a scene as volumetric media. A static point cloud is represented as a set of 3D points in Euclidean space, where each point reflects the position of a surface. A dynamic point cloud is a sequence of static point clouds, which can be considered a 3D video of volumetric data. Such 3D media have emerged in the past decade as the most prominent representation for immersive communication. However, due to the complexity of the data and its significant size, the direct usage of 3D data becomes difficult in a VR communication system that needs to comply to stringent requirements such as high throughput, low latency and reliable communication. Within MPEG standards, [7] presents an efficient and low complexity 2D video based compression of 3D volumetric media. In this way, volumetric captures of the 3D environment can be streamed as 2D frames, and unpacked back as 3D data at the renderer/client. An easy way to obtain a near real-time 3D representation of a participant is to place two depth cameras (e.g. Intel RealSense D415) that are aimed at the user from two different angles. This particular set-up of capturing participants located close to the capturing device enables us to make use of low-end high-resolution depth cameras, which limitations often lie in the range of capture and the noisy output.

The registration of these two captures enables an 180° 3D representation of a participant. In particular, this is important in a close-range VR setup in which participants have to turn their head up to 45° to face each other. Knowing that each RGB-D capture results in a partial 3D representation of the user, the two captures are registered and aligned using a system calibration phase. The calibration parameters (i.e. rigid body transformation parameters) are sent with the visual data streams as metadata. The resulting stream is a 3D point cloud that can be transmitted, for instance, as a 2D video frame following [7].

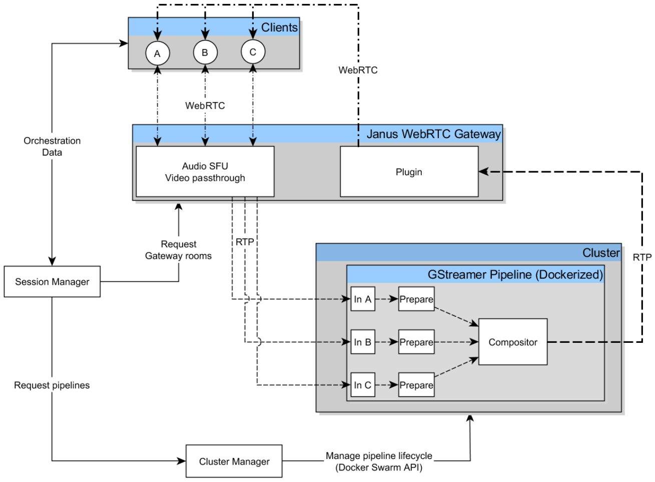

Figure 3: High-level MCU architecture

The system presented in this paper enables four people to interact both auditory and visually. In the VR environment, the participants are situated in a square setup (see Figure 2). The capture module of the TogetherVR framework is extended with two RGB-D capture devices. The captured participant image is also rendered in his/hers VR environment directly, for instant selfview. For this, a Foreground/Background (FGBG) removal function is used prior transmitting the stream data to the MCU. This technique allows to extract the data representing the participant from the RGB-D capture, and only transmit what is necessary. In this way, a significant gain in bandwidth is achieved.

4 MCU FOR SCALABLE VR CONFERENCING

A second focus of our system is on scalability, with respect to computation and bandwidth. Our aim is to provide a large number of participants (>100) with the ability to enter the shared environment. Furthermore, we want them to use their low-cost equipment such as mobile head-mounted displays and common off-the-shelf capture hardware. Our framework employs WebRTC for browser-based real-time communication. For our current system, a clear disadvantage of its peer-to-peer nature is the fully connected mesh network of live streams being transmitted when scaling up to more than a few peers. Each peer transmits its stream $n - 1$ times and receives $n - 1$ streams, where n is the total amount of peers. This results in $n(n - 1)$ streams being transmitted over the network, requiring considerable bandwidth. In addition, in contrast to single video stream processing, multiple video streams can currently not benefit from hardware acceleration. As a consequence, locally encoding and decoding all these streams requires a significant portion of the available performance at each peer, even with current-generation high-performance equipment.

A centralized MCU-based solution mitigates both of these problems. With the support of an MCU, multiple audio and video streams are mixed into one single stream. Each participant would therefore only need to upload one stream and download one stream. The MCU handles the mixing of different streams, and the output of that stream is delivered in a "tailor made" format, that fulfils the requirements of each participant device. That is, for each participant the RGB-D video streams that are captured by the two depth sensors are sent via WebRTC to an MCU. There, the streams from different users are combined into a single stream which is sent to each individual user. At the user side, the multi-user stream is unpacked into individual streams, which are in turn converted to 3D renderings, one for every participant.

As explained in Section 3, user stream data is expected to consist of two 2D representations of 3D volumetric data. For instance, combining 2 streams (multi-view) with a resolution of $1080x960$ pixels of a particular participant would result in Full HD ($1080x1920$ pixels) content. Current browser-based solutions can typically handle a maximum video resolution corresponding to AVC level 5.2 (i.e. slightly more than 4K), thus limiting the amount of participants based on the actual resolution of the users’ streams. In practice, up to four simultaneous user streams can now be processed.

5 MCU ARCHITECTURE AND PERFORMANCE

Figure 3 shows the transmission of streams between clients and the MCU, composed of the WebRTC Gateway and Processing Components. The dashed lines represent video, the dotted lines audio and the dash-dotted lines both video and audio. Clients transmit their own stream via WebRTC to the WebRTC Gateway interface of the MCU. This separates the video and audio tracks and sends them via RTP to the video Mosaic and Audio forwarding components, respectively. Each video stream is decoded and forwarded to any optional processing components (although not shown in figure 3, which could be placed between the ’In A/B/C’ and ’Mosaic Generator’ blocks). The decoded and possibly processed streams are sent to the Mosaic Generator which composits all incoming streams into a single stream (a mosaic). This stream is sent back to the clients via the WebRTC Gateway.

Several design decisions were made for the MCU in our system. The ’In A/B/C’ processes ensure a consistent frame rate for the streams sent to the Mosaic Generator. Frames are dropped or duplicated if a stream has a higher or lower rate than the target rate, respectively. Some streams might encounter increased latency. In the current design, streams are processed as fast as possible such that the mosaic stream adds the least amount of latency. An alternative approach would be to synchronize the incoming streams by applying buffering to the faster streams and ’waiting’ for the slower one. This approach is not chosen due to the increased overall latency that is undesirable in conferencing applications.

Considering the performance, there are three main aspects to take into account: CPU usage, GPU usage, and bandwidth. Table 1 shows the differences between the scenario of using regular peer-to-peer transmission and the scenario using an MCU, assuming ten users, and a required bandwidth of $2mbps$ per stream. Receiving one large stream from the MCU results in less decoding processes, as only a single hardware decoder on the GPU is used, hence the 90% reduction in CPU load. The resulting increased GPU usage is desired: in contrast to the CPU, the GPU is designed to handle the video streams. The required upload bandwidth for one peer in the peer-to-peer scenario is in theory about $18mbps$: each peer uploads his stream to nine other peers. In the MCU scenario this is reduced to uploading his stream just once to the MCU. Because the MCU ensures each peer downloads one large stream instead of nine smaller streams, the overhead and in turn the required download bandwidth slightly decreases. Overall, this is a significant performance improvement and indicates the MCU is a valid and necessary solution for the goal of unloading the network and the client's hardware, especially when scaling up the amount of peers.

	P2P	MCU
CPU	50%	5%
GPU	5%	20%
BW_{up}[1]	$9 \times 2mbps$	$1 \times 2mbps$
BW_{down}	$9 \times 2mbps$	$< 18mbps$

Table 1: Performance differences on the CPU, GPU and bandwidth in the ten-person peer-to-peer and MCU scenarios on the same client and with the same streams

6 DEMONSTRATOR

With the proposed demonstration, we show the concept of VR communication, where four participants can share an experience. The usage scenario considers remote conferencing and collaboration as part of a business meeting and will allow multiple users to discuss in a shared environment supported by a virtual whiteboard where pointing and gazing actions of the participants and their point in space are aligned with the virtual environment. Several aspects can be identified when collaborating and working together at a distance that can make interaction and cooperation a challenge. When collaborating at a distance, the collaborators do not only have a common ground regarding cues from the environment, but also the social context, such as voice volume or facial expressions. Depending on the medium that is chosen for collaboration (e.g., video conferencing, mail, phone), some aspects are present, other are not. However, none of the current media support a feeling of immersion and presence in a shared environment. In this demonstrator, first steps are made towards a shared common ground regarding environmental cues to work towards a shared context and the experience of presence.

With this goal in mind, we propose a demonstrator setup in which each user is recorded with two RGB-D capture devices following the description of section 3. The complete setup can be used by two to four persons at the same time, has a user friendly and intuitive setup, and fits a $3x3\ m^2$ area. In this setup of multiple users, with multiple sensors, each user requires a laptop with a VR head-set and two capture devices, a shared VR environment with four locations around a table to render three participants and selfview, and a network to support the data transfer between users. For the demo, it is foreseen that the MCU runs on an server-PC that is physically present at the demo site. Also, to accommodate transport feasibility and space restrictions, we target two live users and two pre-recorded users.

7 CONCLUSIONS

In this paper, we present the demonstration of our TogetherVR platform infrastructure, which has been extended with multi-sensor capture and in-network based media processing using an MCU. Multi-sensor capture allows us to create realistic volumetric representations of remote participants, so one can see other participants from front and side views. By introducing a VR-enabled MCU, we create an efficient multi-user VR conferencing platform that allows us to increase the number of participants while reducing the load on the client CPUs.

Our approach of moving towards network-based processing aligns well with the current advances in mobile network technologies that enable high throughput at a low delay. In future work, we will study how we can employ 5G-enabled edge computing capabilities to further offload the media processing from clients towards the network. For instance, the background removal process and encoding can be handled by an edge computing node.

We foresee that the shift towards network-based processing will even further increase the clients flexibility (e.g. smaller devices with less computing power) and mobility, eventually finding its way into 5G-enabled HMDs and capture devices.

ACKNOWLEDGMENTS

The authors would like to thank their colleague Lucia D'Acunto for her review of the paper. This paper was partly funded by the European Commission as part of the H2020 program, under the grant agreement 762111 (VRTogether, http://vrtogether.eu/), and was partly funded by the TNO research programme on Social eXtended Reality.

REFERENCES

[1] S. Beck and B. Froehlich. 2015. Volumetric calibration and registration of multiple RGBD-sensors into a joint coordinate system. In *Proc. 2015 IEEE Symposium on 3D User Interfaces.* 89–96. https://doi.org/10.1109/3DUI.2015.7131731

[2] Sam Ekong, Christoph W. Borst, Jason Woodworth, and Terrence L. Chambers. 2016. Teacher-Student VR Telepresence with Networked Depth Camera Mesh and Heterogeneous Displays. In *Proc. 2016 International Symposium on Visual Computing*, Vol. 10073. 246–258. https://doi.org/10.1007/978-3-319-50832-0_24

[3] Simon N. B. Gunkel, Hans M. Stokking, Martin J. Prins, Nanda van der Stap, Frank B. ter Haar, and Omar Aziz Niamut. 2018. Virtual reality conferencing: multi-user immersive VR experiences on the web. In *Proc. 9th ACM Multimedia Systems Conference.* 498–501. https://doi.org/10.1145/3204949.3208115

[4] D. Y. Kim, M. S. Lee, S. H. Choi, K.-J. Koo, I. Hwang, and Y. J. Kim. 2013. An immersive telepresence platform based on distributed architecture. In *Proc. 2013 International Conference on ICT Convergence.* 465–467. https://doi.org/10.1109/ICTC.2013.6675397

[5] A. Maimone, X. Yang, N. Dierk, A. State, M. Dou, and H. Fuchs. 2013. General-purpose telepresence with head-worn optical see-through displays and projector-based lighting. In *Proc. 2013 IEEE Virtual Reality (VR) Conference.* 23–26. https://doi.org/10.1109/VR.2013.6549352

[6] M. J. Prins, S. N. B. Gunkel, H. M. Stokking, and O. A. Niamut. 2018. TogetherVR: A Framework for Photorealistic Shared Media Experiences in 360-Degree VR. *SMPTE Motion Imaging Journal* 127, 7 (Aug. 2018), 39–44. https://doi.org/10.5594/JMI.2018.2840618

[7] Sebastian Schwarz, Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo Cesar, Philip Chou, Robert Cohen, Maja KrivokuÄĞa, Sebastien Lasserre, Zhu Li, Joan Llach, Khaled Mammou, Rufael Mekuria, Ohji Nakagami, Ernestasia Siahaan, Ali Tabatabai, Alexis M. Tourapis, and Vladyslav Zakharchenko. 2018. Emerging MPEG Standards for Point Cloud Compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* PP (12 2018), 1–1. https://doi.org/10.1109/JETCAS.2018.2885981

[8] Mel Slater and Maria V. Sanchez-Vives. 2016. Enhancing Our Lives with Immersive Virtual Reality. *Frontiers in Robotics and AI* 3 (2016), 74. https://doi.org/10.3389/frobt.2016.000745

[9] R. Sorokin, J. Rougier, R. Pastrana-Vidal, and N. Tranquart. 2017. Impact of CPU load on video conferencing quality. In *Proc. 2017 Conference on Principles, Systems and Applications of IP Telecommunications.* 1–5. https://doi.org/10.1109/IPTCOMM.2017.8169753

[10] V. Sterzentsenko, A. Karakottas, A. Papachristou, N. Zioulis, A. Doumanoglou, D. Zarpalas, and P. Daras. 2018. A low-cost, flexible and portable volumetric capturing system. In *Proc. 14th International Conference on Signal Image Technology & Internet based Systems.* 89–96. https://doi.org/10.1109/3DUI.2015.7131731

[11] M. H. Willebeek-LeMair, D. D. Kandlur, and Z. . Shae. 1994. On multipoint control units for videoconferencing. In *Proc. 19th Conference on Local Computer Networks.* 356–364. https://doi.org/10.1109/LCN.1994.386585

[1] These values are theoretical and based on our current knowledge