Extended Reality Multipoint Control Unit – XR-MCU
Enabling Multi-user Holoconferencing via Distributed Processing

G. Cernigliaro, A. Ansari, M. Martos, M. Montagud, S. Fernandez
{gianluca.cernigliaro, amir.ansari, marc.martos, mario.montagud, sergi.fernandez}
@i2cat.net
i2CAT Foundation (Spain)

ABSTRACT

Connecting remote people in a hyper-realistic and immersive manner is a major challenge and requirement, recently magnified by the lockdown and the forced social distancing measures the population has had to deal with. This paper presents a key technological component to efficiently enable multi-user holoconferencing systems, where remote participants can virtually meet represented as 3D volumetric Point Clouds. The contribution resides in bringing the traditional MCU concept broadly used in 2D videoconferencing services to emerging 3D holoconferencing scenarios, with the development of a virtualized cloud-based Extended Reality Multipoint Control Unit (XR-MCU). The XR-MCU aims at reducing the requirements in terms of computational resources and bandwidth consumption at the client side, thanks to the development of a set of novel features, like: fusion of volumetric videos from different users, adjustment of Level of Detail (LoD), and removal of non-visible data. The results from an experimental test confirm the benefits of the XR-MCU when compared to a baseline scenario without its usage. These promising results, together with further planned optimizations, open to door to new technological solutions to enable scalable and adaptive 3D holoconferencing services using lightweight devices. The contribution of this paper can therefore provide relevant societal and economic benefits to our society, by enabling hyper-realistic virtual meetings using inexpensive hardware, while overcoming spatial barriers and travel requirements, and minimizing the environmental burden.

1. INTRODUCTION

Distributed media services, like videoconferencing, have become fundamental for connecting remote people in real-time. Platforms like Skype and Google Hangouts are widely used in a variety of private and professional settings, exchanging large volumes of time-sensitive data with demanding needs in terms of processing and scalability. Due to the common limited computational resources at the client side, Multi-point Control Units (MCUs) [1] rapidly became core components in video communication systems, managing sessions and communications, and performing additional advanced features like layout and quality adaptability. The demand for such kind of services, and thus also for scalable solutions, has been magnified with the recent outbreak of the Covid-19 pandemic, with an estimation of around one-third of the world's population experiencing some kind of lockdown or quarantine [2].

In this research field, significant efforts are additionally devoted to providing higher feeling of realism and immersion in video-oriented services. Video technologies are indeed witnessing an evolution in terms of relevant aspects, like 3D video paradigms, and improved quality and representation formats. Particular attention is being given to the specification of novel

solutions to capture, compress, transmit and represent 3D volumetric video, especially for natural content and real-time scenarios. As proof of evidence, the very first Point Clouds compression standardization process has been initiated within Moving Pictures Experts Group (MPEG) [3]. All the aforementioned advances have jointly opened the door to the holoportation concept, enabling a real-time rendering of volumetric videos captured from remote locations in a shared space, either virtual or real. **Figure 1** shows one of the pioneering holoportation systems developed by Microsoft [4].

Figure 1: Microsoft Real-time holoportation system [5].

However, the quality of 3D natural video for volumetric holoconferencing is not yet comparable to the one of 2D video in traditional conferencing systems. The technology behind holoportation systems is still at an early stage, and given the high volumes of data involved in the representation of Point Clouds, off-the-shelf clients are not yet ready for their real-time processing. These issues have been raised in the Network Based Media Processing (NBMP) task force within MPEG [5], by suggesting to move the overloading parts of the processing of immersive content services to the cloud.

This paper presents a new technological enabler to address key challenges in real-time multi-user 3D holoconferencing systems. In particular, the widespread MCU concept for multi-user 2D videoconferencing is brought to 3D volumetric scenarios with end-users represented as Point Clouds (PC). The presented XR-MCU is a virtualized cloud-based component applicable to any distributed multi-user Virtual/Augmented/Mixed/eXtended Reality (VR/AR/MR/XR) service, which aims at alleviating the demands at the client side, by minimizing the computational resources and bandwidth consumption. A set of innovative and coordinated features have been proposed and implemented to achieve this:

- **Multiple Volumetric Video de/coding**, compatible with most common Point Cloud compression strategies.
- **Reception and delivery** of multiple MPEG Dynamic Adaptive Streaming over HTTP (DASH) streams.
- **Level of Detail (LOD) adjustment:** the incoming Point Cloud representations can be down-sampled, providing the most appropriate resolution based on the users' relative position, activity and underlying context.
- **Removal of non-visible volumetric video:** the non-visible parts of the volumetric environment can be removed from the specific stream delivered to each user.
- **Fusion of volumetric videos:** the incoming volumetric videos are decoded, appropriately processed and fused as a single volumetric video for the scene, which can be delivered as a single (personalized) stream to the client devices.

This set of features allows us to provide an optimized stream for each involved user, depending on their position, viewpoint and available resources, thus alleviating the requirements at the client side, and ensuring a smooth experience.

The XR-MCU and its features have been evaluated in a realistic scenario with two remote virtual users in order to get initial evidence of its potential benefits, when compared to the same scenario using a peer-to-peer communication, as baseline. The obtained results show

a significant reduction in terms of computational resources (RAM, CPU, GPU) and bandwidth consumption, thus proving its benefits and encouraging further research on this area.

The contributions of this paper can provide relevant societal and economic benefits to our society, by enabling hyper-realistic virtual meetings using inexpensive hardware, while overcoming spatial barriers and travel requirements, and minimizing the environmental burden. Section 2 highlights use cases and scenarios in which the XR-MCU can be applicable. Section 3 briefly reviews the state-of-the-art in this area. Then, Section 4 presents the XR-MCU and its key features. Section 5 describes the evaluation setup, and provides the obtained results. Finally, Section 6 brings forward some future research plans, and provides our conclusions.

2. USE CASES AND SCENARIOS

Multi-party videoconferencing services are currently used for a wide variety of use cases and scenarios, including: remote gatherings between family members and relatives, tele-work, e-learning, and other professional settings, like virtual meetings and tele-consultation. Their usage has been recently multiplied by the current social distancing measures due to the COVID-19 outbreak. This situation not only imposes further scalability requirements, but also higher reliability and quality requirements: the adoption of such tools is no more an alternative, but the solution.

3D volumetric holoconferencing services provide the opportunity of significantly enhancing these remotely shared experiences, in terms of realism, immersion and quality of interaction (as they enable a full body representation and 3D space reconstruction). The XR-MCU is a key component to bring real-time holoconferencing services into reality in home-based scenarios using off-the-shelf hardware devices, especially for multi-party Social VR, or more generally Social XR. Beyond those scenarios cited for video conferencing, the XR-MCU can act as an enabler for the next Social XR use cases, among others:

- Shared video watching (e.g. [6]) in entertainment, educational and professional environments.
- Virtual conferences and meetings (e.g. [7])
- Culture and virtual tours (e.g. [8]).
- Massive events, like concerts and sports (e.g. [9]).
- Gaming, where real representations of gamers can replace avatars.

3. RELATED WORK

The strategy of reducing the computational load of videoconferencing client machines, by moving the heaviest operations to the cloud, has been widely addressed for communications based on traditional 2D video. The MCU concept introduced in the ’90s has been indeed deeply analyzed and considered in standardization works, like the ITU H.323 recommendation that defines the protocols to provide audio-visual communication sessions [10]. The community has also devoted efforts on adapting the MCU concept to newly developed video distributions systems (e.g. [11], [12]), and converting it into an essential component of virtualization architectures, like the widely known Cisco Unified Computing System (UCS) [13]. Likewise, the idea of virtualizing part of the processing load for immersive applications has been suggested by the Global System for Mobile Communications Association (GSMA) [14], providing a high level architecture with shared computational resources in the cloud for VR/AR services.

The research contributions oriented to the optimization of volumetric video transmission systems not only can exploit the previous work on traditional 2D video, but also the new strategies generically defined for other immersive formats. A relevant example is given by viewport-aware adaptive streaming strategies for 360 degree videos, like the one by Ozcinar et al. [15], which consists of dividing the content into tiles, and providing higher quality tiles to the region associated to the users’ viewport. The adaptation of such strategy has been

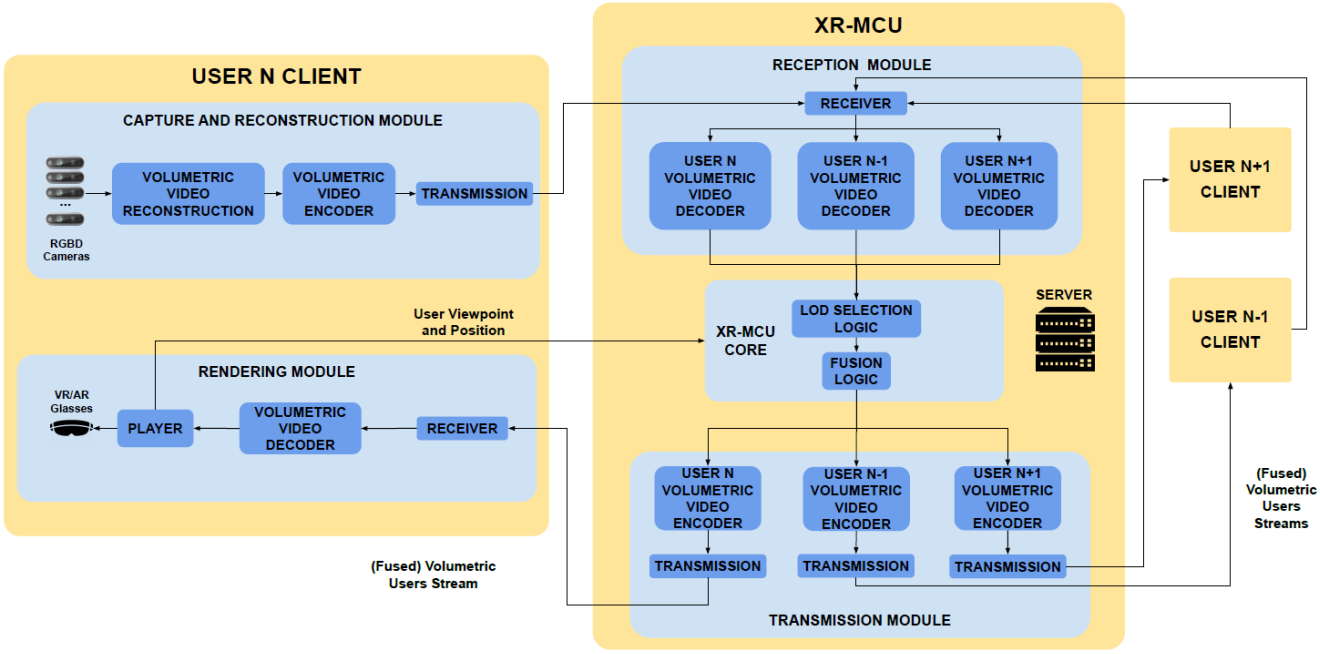


Figure 2: XR-MCU Overall Architecture

explored by Park et al. [16], by proposing an adaptive streaming strategy for point cloud based volumetric video. Instead of a 2D segmentation of the 360-degree videos, they utilized a volumetric approach for tiling in which the three-dimensional environment is divided into cubic sections. Such streaming strategy considers both bit-rate and user's viewpoint to define which tiles and quality to provide. Other peer-to-peer oriented Point Cloud streaming strategies have been proposed by Hosseini and Timmerer [17] including a resolution scalability strategy to meet a, trade-off between quality and bandwidth. Qian et al. [18] proposed a volumetric video streaming system for commodity mobile phones, where heuristics are used to decide on representations and to use edge computing features to reduce the computational load at the client side. The usage of an MCU for videoconferencing in three-dimensional environments has been explored by Dijkstra et al. [19] for Social VR applications as the one presented by Gunkel et al. [6]. In particular, that work applies the MCU features to scenarios in which users are represented as multi-view plus depth video. This way, it is possible to take advantage of the bi-dimensional nature of such content to minimize the load at the client side.

In summary, the presented XR-MCU provides the next key advantages and/or novel aspects compared to the existing state-of-the-art solutions:

First time to the MCU concept has been applied to fully three-dimensional, volumetric video represented as Point Clouds.

Tasks such as resolution selection and viewport aware streaming are now performed in a cloud-based management system for an efficient distribution of volumetric video.

The availability of a virtualized cloud-based XR-MCU and its features, allows every user to send a full resolution of their volumetric representations.

The next section describes the architecture, components and aforementioned features of the proposed XR-MCU, which aims at being a standard-compliant enabler for multi-user volumetric conferencing services.

4. EXTENDED REALITY MULTIPOINT CONTROL UNIT (XR-MCU)

This section presents the design of the proposed XR-MCU. Section 4.1 describes how the XR-MCU component is integrated in a multi-user system, including the interaction with other components, and the specified inputs and outputs. Then, the specific features provided by the XR-MCU, and their effects on the holoconferencing experience, are described in Section 4.2.

4.1. XR-MCU architecture and sub-components

We consider a holoconferencing network where a number of N users are connected to the XR-MCU. A graphical representation of the overall architecture of the system is shown in **Figure 2**. Each user's client includes: i) a *Capture and Reconstruction module*, in charge of creating, compressing and transmitting the volumetric video, represented as Point Clouds, and captured by a number of RGBD sensors (e.g., 1-4); and ii) a *Rendering module*, which is able to receive, decode and represent the volumetric videos of the other users in a VR/AR/XR display. Apart from the volumetric video, the user's client is in charge of transmitting the Viewpoint and Position of the user. This will allow the XR-MCU handling such information to provide an optimized content stream for each user. The XR-MCU interfaces to the user's clients are the Reception and Transmission modules. The first one is in charge of receiving and decoding the videos from the users, before providing them to the Core Sub-system. The second one compresses and transmits the specific content to each user. The Core Sub-system includes the main features available in the XR-MCU, explained next.

4.2 XR-MCU Features

The XR-MCU Core Sub-system implements key features that triggered in a coordinated and event-driven manner, allow a significant reduction of the client computational load and bandwidth consumption, providing an optimized stream to each user. These features are explained in the following sub-sections.

Volumetric Video De/Coding

The Reception Module receives one Point Cloud per user and provides the incoming streams to a number of decoder instances equal to the number of users. Once uncompressed, the data are provided to the Core Sub-system which performs a series of optimization task (explained next) to provide a tailored stream for each client. Then, the Core Sub-system provides the streams to the Transmission Module, with includes a dynamically instantiated encoding instance for each client connected to its corresponding transmission component via MPEG DASH. The XR-MCU has been conceived with the goal of being compatible with the most popular volumetric video compression methods that can perform a real time, low latency, encoding and decoding of volumetric content. The version presented in this paper uses the MPEG anchor implementation developed by Mekuria *et al.* [20] using only Intra frames. The aforementioned details are provided for completeness. However, the specific encoding and adaptive transmission strategies for the DASH streams are out-of-scope of this paper.

Level of Detail (LoD) Selection

Traditional MCUs for videoconferencing are capable of controlling the resolution of the incoming videos, adapting it according to the bandwidth capacity and to the client device characteristics. In a volumetric video, the resolution can be adjusted by tuning a parameter called Level of Detail (LoD) [21] which controls the geometrical characteristics of the video, usually represented as vertexes of polygons or simple three-dimensional points, in Cartesian (x, y, z) coordinates. The adjustment and selection of appropriate LoDs is then a powerful tool to save bandwidth and computational requirements, exploiting the knowledge about the relative distance and positions of the elements in the 3D environment. When a user is observing the 3D scene from a certain viewpoint, the other users will be perceived closer or further depending on their relative positions. When users are close, dense point cloud representations are needed, as having a high resolution becomes key. When users are further away, the resolution can be lowered by downgrading the LoD. This will reduce the amount of data to process, without potentially having a negative impact on the perceived quality of the representation. The XR-MCU Core Sub-system includes a module called LoD Selection Logic that after uncompressing the volumetric video, will apply a specific LoD downgrading level depending on the users' positions.

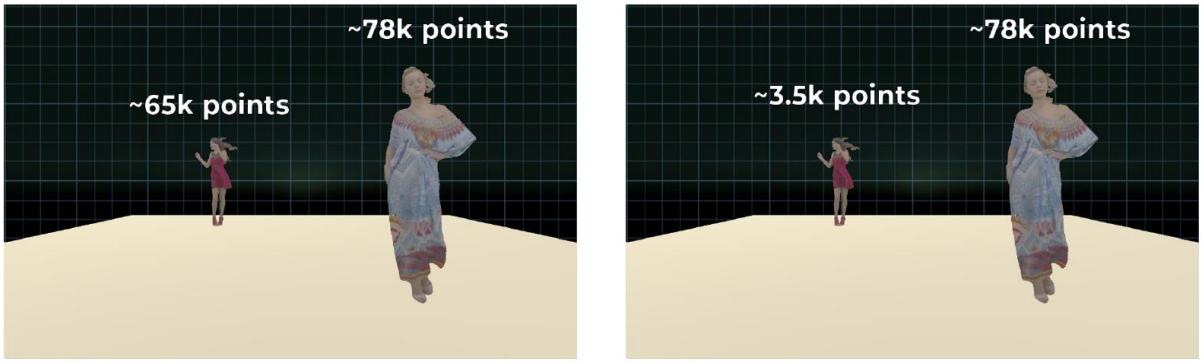


Figure 3: Example of LoD selection operated by the XR-MCU. Close User: High Representation in kept by the XR-MCU. Far User: The XR-MCU can apply LoD downgrading

Figure 3 shows an example of the XR-MCU LoD Selection Logic, where a user is receiving two Point Clouds placed in different relative positions. On the left we show how, without the XR-MCU, the two received streams have similar resolutions. On the right, see how the XR-MCU reduces the resolution of the further Point Clouds, by analyzing the relative positions of the volumetric elements. **Figure 4** shows a visual comparison of the two resolutions applied to the Point Cloud when affected or not by the LOD Selection Logic. Section 5 confirms how the application of such strategy results in a reduced consumption of bandwidth and computational resources.

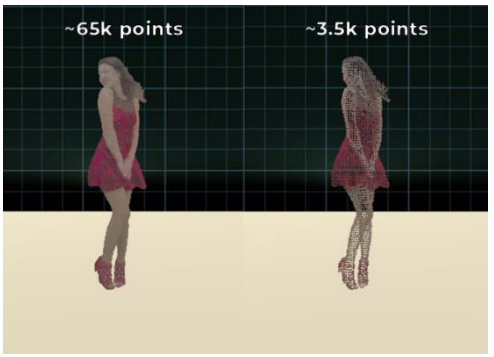


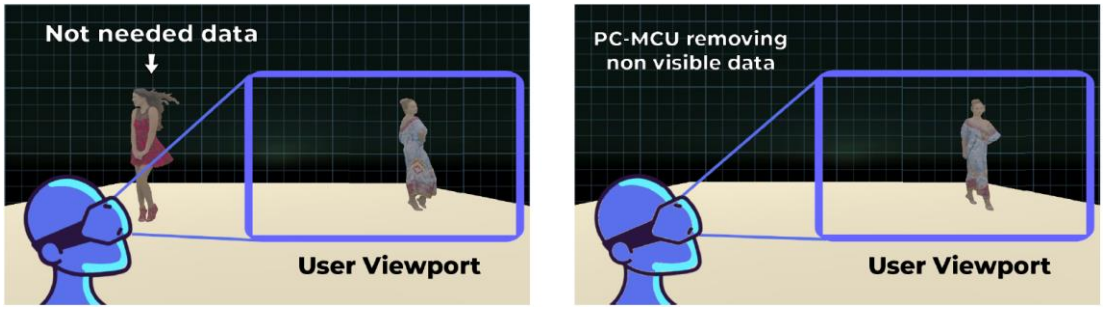
Figure 4: Comparison of two resolutions applied to the Point Cloud affected by the LOD Selection

Removal of non-visible volumetric video

In 2D videoconferencing systems, all pixels from the videos are needed because they are fully visible to all active participants. In 3D systems, users can freely navigate around the environment and look around. Therefore, not all the information may be needed by all the users, or not for the whole duration of the session. In rendered 3D environments, the viewport represents the portion of the virtual environment that is visible from the viewpoint of a specific user. Therefore, at a specific time, users are able to visualize only certain elements, but will not be able to see parts of the scene located outside the viewport. The delivery of the whole data to all the participants would then result in an inefficient solution, having a significant impact on the consumption of bandwidth and computational resources and bandwidth consumption without enhancing the Quality of Experience (QoE). Therefore, the availability of some logic, able to select the right information to be delivered at the right time becomes fundamental for volumetric holoconferencing systems. The presented XR-MCU implements a first version of this feature, by directly removing the data outside the user's viewport from the delivered stream for that specific user. The removal is performed by relying on the viewport information reported by each client (i.e. the xyz position and viewing angle). Other advanced (adaptive and hybrid) versions of this feature, like the ones introduced in Section 2, will be analyzed in future work. **Figure 5** shows an example of how the XR-MCU considers the users viewport to remove the part of the video that would not be visualized. Section 5 proves the benefits of this feature.

Fusion

Traditional MCUs are typically in charge of merging pixels of two or more videos from several users in a videoconferencing session. If that is the case, every participant will receive and visualize a merged and composed version of the content as a single 2D video. In volumetric holoconferencing systems, the same layout and merging strategies (e.g., side-by-side) are non-applicable, because each volumetric video is an independent structure with a geometry that has to be placed in the 3D space, and will also depend on the user's position.



(a) Presence of data not visible by the end-user *(b) Non-visible data removed by the XR-MCU*

Figure 5: Example of non-visible data removal operated by the XR-MCU

However, the geometry of two or more volumetric videos can be fused considering a single coordinate system. After selecting the appropriate LoD and removing the non-visible information, the XR-MCU Core Sub-system performs, for each user, a fusion of the representations of all visible participants and elements. This feature provides key benefits, as each user client will just receive a single stream, without needing to execute N of the Point Cloud receiver and decoder modules.

5. EXPERIMENTAL RESULTS

The experimental assessment presented in this paper is based on the use of a first implementation of the proposed XR-MCU, comparing it to a system in which the volumetric videos are delivered in a peer-to-peer fashion. The goal is to assess the benefits of using the XR-MCU in a scenario with two volumetric videos compared to a baseline scenario without the use of the XR-MCU. This section starts by describing the experimental setup in Section 5, then continues by detailing the evaluation methodology and metrics in Section 5.2, and it concludes by presenting the obtained results in Section 5.3.

5.1. Experimental Setup

The setup used in this experiment considers a set of 10 simulated holoconferencing sessions where one of three end-users receives the Point Clouds of two other users remotely connected (although the networking aspects are out-of-scope of the paper). The sequences considered in the tests are two among the ones available at the 8i Voxelized Full Bodies database [22]: *Red and Black* and *Longdress*. In order to facilitate a real time processing of the whole pipeline, the resolution of the two sequences have been downsampled to 65k points and 78k points, respectively. In order to recreate a typical holoconferencing system, a set of actions are forced to activate and assess the benefits of the XR-MCU features (Section 4). The duration of each single session was 34 seconds. The forced actions in the holoconferencing scenario are the following:

- Step 1: initial position purposely defined to receive the two Point Clouds at their maximum resolution (65k and 78k points) within the viewport.
- Step 2: viewpoint panning, excluding *Red and Black* from the viewport to evaluate the computational load reduction when the *Non Visible Area Removal* feature is active.
- Step 3: viewpoint panning, excluding *Longdress* from the Viewport to evaluate the computational load reduction when the *Non Visible Area Removal* function is active for this second simulated user.
- Step 4: user's viewpoint back to the initial position.
- Step 5: simulation of user moving away from both Point Clouds to evaluate the benefits when the *LoD Selection* feature is active.

The XR-MCU Fusion function, in charge of merging the sequences of several users into one, is always active. For the comparison, the same sequence of actions is simulated also when the volumetric videos are delivered in a peer-to-peer fashion, without the XR-MCU. All the features presented in Section 4 have been implemented according to a CPU oriented

sequential programming model. At this stage, the implementation follows a sequential calls composition without any parallelization technique, nor GPU implementation. The GPU involved in this study is the one used at the end-user client machine, for the volumetric video rendering. The specifications of the machine used as the client are the following:

- CPU: Intel Core i7-6700 @ 3.40GHz
- RAM: 16 GB @ 2133MHz
- GPU: NVIDIA Quadro K4200 4GB GDDR5
- NET: Realtek PCIe GbE Family Controller

5.2. Evaluation Methodology and Metrics

The tests consisted of 10 iterations with the XR-MCU plus 10 iterations without the XR-MCU. In each iteration samples of the following set of metrics were collected by using a tool from Montagud *et al.* [23]:

- CPU usage at the client machine (in %).
- GPU usage at the client machine (in %).
- Memory usage at the client machine (in MBs).
- Bandwidth consumption (in Mbps).

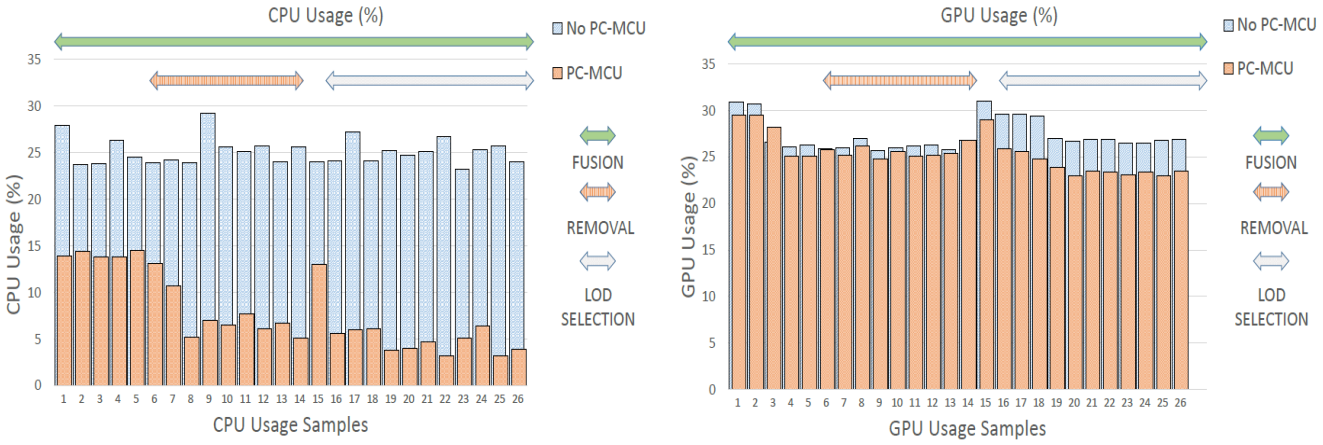
5.3. Results

In order to provide a thorough comparison, we have performed a simulation of 10 sessions without the XR-MCU and 10 sessions using the XR-MCU. During each session, the mentioned metrics have been sampled along the whole experience. The final results show the average of each sample over the 10 iterations, showing the different benefits of the XR-MCU. **Figure 6.a** shows the reduction of exploiting the XR-MCU on CPU usage when the XR-MCU is included in the holoconferencing session, when compared to the baseline condition (without the XR-MCU). The intervals in which the XR-MCU features are active are specified on the top part of the figure. When the XR-MCU is not used, the percentage of CPU usage does not suffer strong fluctuations along the duration of the 10 sessions, due to the constant incoming stream from the two Point Clouds. When the XR-MCU is used, it is possible to notice that, initially, when only the Fusion feature is active, there is already a considerable gain in terms of CPU usage. The gain is considerable when, afterwards, the XR-MCU performs the Non Visible Areas Removal actions, first excluding the Longdress sequence, and then excluding Red and Black. The CPU usage increases again when the 2 Point Clouds are newly available (see sample 15 in **Figure 6.a**) and then starts being further reduced when the LoD Selection function is active. **Figure 6.b** shows the evolution for the percentage of GPU usage. In this case, the benefits of the Non Visible Areas Removal function are less noteworthy than for the CPU usage, because the GPU is in charge of the rendering of the visible part of the 3D scenario; however, when the LoD Selection function is active, it is possible to notice a gain. The GPU load is indeed reduced thanks to the lower number of voxels needed to represent the Point Clouds. The overall average results for the 10 iterations are summarized in **Table 1**, by also including additional ones regarding the RAM and bandwidth consumption. It is possible to observe how the introduction of the XR-MCU resulted in a reduction of 69% of the CPU usage, 7% of the GPU usage, 18% of memory usage and of 84% of bandwidth consumption. For completeness, the additional latency introduced by the XR-MCU has also been evaluated. In average, the XR-MCU adds a latency of 89 ms when the Point Clouds are both at their lowest resolution and an average of 160 ms when they are both visible at their maximum resolution. For the intermediate cases the latency is kept in between

-	No XR-MCU	XR-MCU	Δ Reduction
CPU (%)	25	7.8	68.7%
GPU (%)	27.2	25.4	6.6%
RAM (MBs)	445.5	366.3	17.85%
BW (Mbps)	92.3	14.9	83.9%

Table 1: Average results in terms of resources consumption

those values. A demo showing the virtual scenarios, the set of simulated actions, and the registered metrics can be watched here: https://youtu.be/qEENaFVeLrk.



(a) CPU usage *(b) GPU usage*

Figure 6: Percentage of CPU and GPU usage: comparison between sessions with and without the XR-MCU.

6. DISCUSSION AND FUTURE WORK

The presented results are obtained with the XR-MCU based on a sequential implementation. Next releases of the XR-MCU will include a parallelization of CPU or GPU cores to improve the performance in terms of scalability, delays and quality

6.1. Parallelization

Serving a volumetric video stream for multiple users involves a huge processing burden. Besides, a user may join or leave the session while other users are using XR-MCU, and this should have no adverse effect on the users who are currently being served by XR-MCU. In a sequential implementation, the CPU will serve the users one by one. Additionally, the required tasks for every single user are performed one by one. Provided that the CPU can meet the computational requirements for Point Clouds processing and streaming, the sequential approach is a valid solution. However, as the number of the users or the density of the Point Clouds rise, the CPU may spend more and more time on serving the data from all users. This implies that the data from every user has to wait for all the data from other users to be served before being served by XR-MCU. To tackle this potential hardship, a future implementation of XR-MCU will adopt a parallel approach. In a parallel implementation, the data from all the users will be served immediately upon the reception of a new incoming frame. As soon as a new frame is received from any user, an available core will undertake the processing of the new frame, regardless of the situation of the data from any other user. Not only the frames of all the users will be processed in parallel simultaneously, but also the required tasks of every single user will also be conducted in parallel. That is to say, while e.g. the current frame is being encoded and a new frame just comes in, the new frame will instantly be captured and fed to the decoder along with the ongoing encoder tasks. At the same moment, the data from all the other users are also being served by XR-MCU. This way, the waiting times for XR-MCU to process the frames from other users will be significantly reduced, which will also reduce the total latency of XR-MCU, and thus the end-to-end latency of the service.

6.2. GPU Implementation

All the aforementioned equations are derived with the assumption of available hardware resources to catch the incoming Point Clouds immediately upon reception. Experimentally, this assumption can be very challenging as the number of available CPU cores are limited. So, if all the CPU cores are serving other tasks and a new frame is just received, then the performance of the parallel XR-MCU will be downgraded. The GPU is capable of running billions of threads in parallel and will considerably increase the XR-MCU processing capabilities. The numerous GPU cores can run plenty of tasks in parallel and will definitely

improve the XR-MCU performance. However, special care should be paid to the time required for CPU-GPU data exchange and the lower amount of available memory in GPU.

7. CONCLUSIONS

The holoportation concept is attracting the attention of the research and industry community. Volumetric video systems involve high requirements in terms of data exchange and processing, challenging the typically limited resources at the client side. In addition, in 3D holoconferencing scenarios, not all the information needs to be delivered to the users, and not always is the highest resolution necessary. To avoid unnecessary operations and optimize the heaviest ones, the proposed XR-MCU includes a set of features providing most adequate stream to each user. This paper has proved that the use of the XR-MCU provides significant savings in terms of bandwidth, CPU, GPU, and RAM consumption. These benefits are achieved at the cost of a reasonable extra latency, mainly due to the sequential CPU implementation of the XR-MCU. Future work will be targeted at using GPU based programming models and parallelization techniques to improve the performance in terms of latency, scalability and quality (i.e. by optimizing the media processing tasks). In addition, further research will be devoted at determining the most appropriate strategies for LoD selection and handling the non-visible elements based on different aspects, like distance, navigation patterns, number of volumetric media elements, etc. Finally, scalability and QoE tests will be conducted for each one of the provided features by the XR-MCU.

8. REFERENCES

[1] MH Willebeek-LeMair, Dilip D Kandlur, and Z-Y Shae. 1994. On multipoint control units for videoconferencing. In Proceed. of 19th conf. on local comp. networks. IEEE, 356–364.

[2] Online, Available: https://www.sciencealert.com/one-third-of-the-world-s-population-are-now-restricted-in-where-they-can-go

[3] Sebastian Schwarz, Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo Cesar, Philip A Chou, Robert A Cohen, Maja Krivokuća, Sebastien Lasserre, Zhu Li, et al. 2018. Emerging MPEG standards for point cloud compression. IEEE Journal on Emerging and Selected Topics in Circuits and Systems 9, 1 (2018), 133–148.

[4] Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L Davidson, Sameh Khamis, Mingsong Dou, et al. 2016. Holoportation: Virtual 3d teleportation in real-time. In Proceedings of the 29th Annual Symposium on User Interface Software and Technology. 741–754.

[5] Kyungmo Park. [n.d.]. Network Based Media Processing. ISO/IEC JTC1/SC29/WG11 N18270, January, 2019.

[6] Gunkel, Simon NB, Hans Stokking, Tom De Koninck, and Omar Niamut. 2019. Everyday photo-realistic Social VR: communicate and collaborate with an enhanced co-presence and immersion. IBC 2019, Amsterdam (The Netherlands), October 2019.

[7] Duc Anh Le, Blair MacIntyre, Jessica Outlaw, Enhancing the Experience of Virtual Conferences in Social Virtual Environments, Virtual Conferencing Workshop (VR in VR), IEEE VR 2020, Atlanta (US), March 2020

[8] Redouane Kachach, Pablo Perez, Alvaro Villegas, Ester Gonzalez-Sosa, Virtual Tour: An Immersive Low Cost Telepresence System, Virtual Conferencing Workshop (VR in VR), IEEE VR 2020, Atlanta (US), March 2020

[9] Hayashi, K., Saito, H., Synthesizing free-viewpoint images from multiple view videos in soccer stadium. In: International Conference on Computer Graphics, Imaging and Visualisation (CGIV'06), pp. 220{225. IEEE (2006)

[10] Gary A Thom. 1996. H. 323: the multimedia communications standard for local area networks. IEEE communications Magazine 34, 12 (1996), 52–56.

[11] M Reha Civanlar, Oznur Ozkasap, and Tahir Çelebi. 2005. Peer-to-peer multipoint videoconferencing on the Internet. Signal Processing: Image Communication 20, 8 (2005), 743–754.

[12] Junlin Li, Li-wei He, and Dinei FlorΩncio. 2007. Multi-party audio conferencing based on a simpler MCU and client-side echo cancellation. In 2007 IEEE International Conference on Multimedia and Expo. IEEE, 84–87.

[13] Silvano Gai, Tommi Salli, and Roger Andersson. 2010. Cisco Unified Computing System (UCS)(Data Center): A Complete Reference Guide to the Cisco Data Center Virtualization Server Architecture. Pearson Education.

[14] Global System for Mobile Communications Association. Cloud AR/VR Whitepaper. www.gsma.com/futurenetworks/wiki/cloud-ar-vrwhitepaper. Accessed:2020-03-09.

[15] Cagri Ozcinar, Ana De Abreu, and Aljosa Smolic. 2017. Viewport-aware adaptive 360 video streaming using tiles for virtual reality. In 2017 IEEE International Conference on Image Processing (ICIP). IEEE, 2174–2178.

[16] Jounsup Park, Philip A Chou, and Jenq-Neng Hwang. 2019. Rate-utility optimized streaming of volumetric media for augmented reality. IEEE Journal on Emerging and Selected Topics in Circuits and Systems 9, 1 (2019), 149–162.

[17] Mohammad Hosseini and Christian Timmerer. 2018. Dynamic adaptive point cloud streaming. In Proceedings of the 23rd Packet Video Workshop. 25–30.

[18] Feng Qian, Bo Han, Jarrell Pair, and Vijay Gopalakrishnan. 2019. Toward practical volumetric video streaming on commodity smartphones. In Proceedings of the 20th International Workshop on Mobile Computing Systems and Applications. 135–140

[19] Sylvie Dijkstra-Soudarissanane, Karim El Assal, Simon Gunkel, Frank ter Haar, Rick Hindriks, Jan Willem Kleinrouweler, and Omar Niamut. 2019. Multi-sensor capture and network processing for virtual reality conferencing. In Proceedings of the 10th ACM Multimedia Systems Conference. 316–319.

[20] 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.

[21] David Luebke, Martin Reddy, Jonathan D Cohen, Amitabh Varshney, Benjamin Watson, and Robert Huebner. 2003. Level of detail for 3D graphics. Morgan Kaufmann.

[22] Eugene d'Eon, Bob Harrison, Taos Myers, and Phil A. Chou. 2017. 8i voxelized full bodies-a voxelized point cloud dataset. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document WG11M40059/WG1M74006 (2017).

[23] M. Montagud, J.A. De Rus, R. Fayos-Jordan, M. Garcia-Pineda, and J. Segura- Garcia. 2020. Open-Source Software Tools for Measuring Resources Consumption and DASH Metrics. In Proceedings of the 11th ACM Multimedia System Conference.