Collaborative Cooking in VR: Effects of Network Distortion in Multi-User Virtual Environments

Javad Sameri[1], Sam Van Damme [1,2], Susanna Schwarzmann[3], Qing Wei[3], Riccardo Trivisonno[3], Filip De Turck[1], and Maria Torres Vega[1,2]

[1] IDLab, Department of Information Technology (INTEC), Ghent University - imec, Ghent, Belgium
[2] eMedia Research Lab, Department of Electrical Engineering (ESAT), KU Leuven, Leuven, Belgium
[3] Huawei Technologies, Munich, Germany
Javad.Sameri@UGent.be

ABSTRACT

The future of human interaction is virtual. Thus it will require effective collaboration on tasks among users in remote settings. eXtended Reality (XR) is playing a leading role in this transition, offering a realm where virtual collaboration becomes not just possible but essential in situations where physical presence is limited by risk, cost, or complexity. However, while networks are continuously evolving, they can still introduce unexpected impairments that potentially degrade the user perception, i.e., the Quality-of-Experience (QoE), of such Collaborative Virtual Reality (CVR) scenarios. In response to this challenge, this paper presents a demonstrator designed to explicitly showcase the effects of network conditions on CVR. Our platform, centered around a pizza-making game, allows for exploration of the real-time impact of different network parameters, such as packet delay, loss, and throttling on the user engagement and perception in CVR. The framework employs a combination of subjective, objective, and physiological assessments, including the capture of heart rate and skin conductivity, to gain comprehensive insights into user experiences. Our platform not only allows users to directly experience the impact of network impairments on CVR interactions but also provides initial evidence of how such distortions affect both subjective perceptions and objective performance metrics.

CCS CONCEPTS

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

KEYWORDS

Collaborative Virtual Reality, human-computer interaction, Network Impairments, Quality of Experience

ACM Reference Format:
Javad Sameri[1], Sam Van Damme [1,2], Susanna Schwarzmann[3], Qing Wei[3], Riccardo Trivisonno[3], Filip De Turck[1], and Maria Torres Vega[1,2]. 2024. Collaborative Cooking in VR: Effects of Network Distortion in Multi-User Virtual Environments. In *ACM Multimedia Systems Conference 2024 (MMSys '24), April 15–18, 2024, Bari, Italy*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3625468.3652201

MMSys '24, April 15–18, 2024, Bari, Italy

ACM ISBN 979-8-4007-0412-3/24/04
https://doi.org/10.1145/3625468.3652201

1 INTRODUCTION

eXtended Reality (XR) is meant to shape the future human-computer and human-human interaction. As a result, it is gaining attention in multiple fields of society such as gaming and entertainment, industry [4], mental healthcare [7], and Virtual Reality (VR) training [11]. Within these fields, XR facilitates a spectrum of interaction including collaborative, competitive, and observational experiences. Collaborative environments, in particular, emerges as a pivotal area, enabling shared virtual experiences that foster teamwork, communication, and shared goal achievement. These VR experiences are typically applied in sectors where the real-life alternative may present challenges due to safety concerns, financial constraints, time limitations, or complexity. These include hazardous industrial environments, costly training simulations, time-intensive processes, and complex therapeutic interventions.

Often, these alternatives require real-time collaboration among multiple users remotely located in virtual environments, leading to the emergence of Collaborative Virtual Reality (CVR). The immersive and interactive potential of such an environment makes it particularly suitable for such collaborative remote training tasks, where one can get the necessary training in a safe, low-cost, and engaging environment.

However, factors such as the fluctuations in the network conditions (e.g., the available network bandwidth or increased latency), end-user device characteristics, or content can have unexpected impacts on the user perception of CVR. These effects, however, remain largely unexplored [9]. In fact, the network can heavily influence CVR experiences. Ensuring a good Quality-of-Experience (QoE) in CVR requires some stringent requirements that the networks can in some case not handle. This can result in distortions such as delay actions, stalling in images, visual artifacts, and desynchronizations, which may affect the end-user's experience, interactivity, and performance.

In this demonstrator, we introduce a platform designed to enable a comprehensive evaluation of the impact of network distortions on the user perception of remote CVR. This platform is uniquely equipped to conduct subjective, objective assessments, offering an opportunity to comprehensively understand the full extent of the user experience under varying network conditions. We selected a collaborative pizza-making use case. In it, two users connect from remotely in a virtual kitchen to prepare a pizza. This choice was motivated by a desire to maintain a level of gamification and enjoyment in the envisioned use case. Given that exploring the effect of network impairment might require multiple playthroughs, the gamification element helps prevent subjects from becoming biased due to fatigue and boredom. This platform lays the groundwork for developing CVR environments. Thus, it allows for the integration other types of environments where collaboration is necessary.

From an objective standpoint, our platform allows for granular measurements of the time required to complete individual subtasks within the VR environment. This capability is crucial for performing detailed performance analyses. Furthermore, we employ Wireshark software to collect and analyze network and USB traffic data, enabling us to understand the system's performance under various conditions.

On the subjective front, Our platform is designed to capture implicit subjective data, specifically by measuring physiological signals, thereby enriching our monitoring effectiveness. This feature allows for the collection and analysis of key physiological data such as heart rate and skin conductivity, providing insights into the users' nervous system responses during the experience. Additionally, the platform supports explicit subjective assessments by enabling the collection of responses to pre-configured questionnaires. This approach allows us to gather direct user feedback and perceptions, offering a comprehensive view of the subjective user experience. Furthermore, we have open-sourced our platform's implementation, making it accessible to other researchers and developers [1].

The remainder of the paper is structured as follows. In Section 2, we provide an overview of the related works in the literature. Section 3 presents an in-depth explanation of the platform, including system architecture, its use case, and monitoring tools. Additionally, the results of a preliminary set of experiment using this platform is included. In section 4, we outline a demonstration session showcasing the capabilities of our proposed system. Lastly, in Section 5, we summarize the key findings and insights derived from this study.

2 RELATED WORK

Recent advancements in networked CVR have significantly expanded the scope and utility of this technology. In a comprehensive survey, Jinjia et al. [6] explore holistic solutions for networked VR that surpass traditional networking limitations. They highlight the integration of VR data capture, encoding, network, and user navigation, emphasizing how mass usage and data management critically influence the user experience. This study also delves into aspects such as wireless operation and ultra-low latency, offering foresight into the future of immersive experience networks and unified data set measurement in VR video transmission, particularly in the realm of Six-Degrees-of-Freedom (6DoF) VR.

Building on these network challenges, Concannon et al. [3] address the practical implications of VR technology in enhancing remote operation and human-machine interactions. Their work introduces a unique QoE evaluation system for VR telepresence, incorporating tools like the HTC Vive Head-Mounted Display (HMD) and Unity-based environments. This system, adept at tracking user interactions and physiological responses, offers valuable insights into the user's perception in VR settings, particularly under varying network conditions. This research not only contributes to understanding VR's network demands but also paves the way for optimizing user experience in virtual environments.

Transitioning from network-centric to application-oriented advancements, Antonelli et al. [1] present the Typhis project, an educational applications in VR. This project is an Open Networked Platform for Industry 4.0 training, enabling users to design and simulate industrial processes in VR. The project stands out for its modular approach in model structuring, facilitating integration with existing systems and future adaptability. This initiative underscores the potential of collaborative VR in revolutionizing educational paradigms, particularly in dynamic learning environments.

In a similar vein of technological innovation, Guo et al. [5] introduce an adaptive VR framework designed for high-quality wireless VR in future mmWave-enabled wireless networks. Their approach is notable for its use of Mobile Edge Computing (MEC) to offload real-time VR rendering tasks, thereby enhancing performance. Incorporating a distributed learning algorithm that synergizes Deep Reinforcement Learning (DRL) with game theory illustrates a significant stride in optimizing user QoE. This research not only showcases the potential of advanced network technologies in VR but also sets a benchmark for future developments in the field.

Complementing these technological strides, practical applications of networked CVR are present in the work of Triandafilou et al. [8] and Chheang et al. [2]. Triandafilou et al. developed the VERGE system, a multi-user VR platform for home therapy, enabling remote interaction among stroke survivors and therapists. This system is particularly noteworthy for its user-friendly design, employing Kinect devices for avatar control, and has demonstrated effectiveness in promoting repetitive arm movement practice. Meanwhile, Chheang et al. introduced CollaVRLap, a CVR system for laparoscopic liver surgical planning and simulation. This system's utilization of medical imaging and laparoscopic joysticks for VR surgery illustrates the potential of VR in enhancing surgical training and planning, further validated by positive feedback from surgeons.

In summary, these studies paint a picture of the current landscape in networked CVR. From network enhancements and user experience optimization to innovative educational and medical applications, each contribution highlights the various nature and growing potential of VR technology. As this field continues to evolve, these advancements underscore the transformative impact of VR across diverse domains. Despite these significant advancements, the complex relationship between network impairments and user experience in CVR remain unexplored. This gap necessitates an in-depth investigation into how network impairment influences QoE in XR. Our research aims to lay the groundwork for future research to bridge this gap.

3 CVR PLATFORM

This Section aims to provide a comprehensive overview of our CVR platform, including the system architecture, necessary hardware to run this platform, and the development technologies. We then detail the platform's use case and discuss the software and tools used for monitoring user behavior and network performance. The

[1] https://github.com/mj-sam/CVR_cooking.git

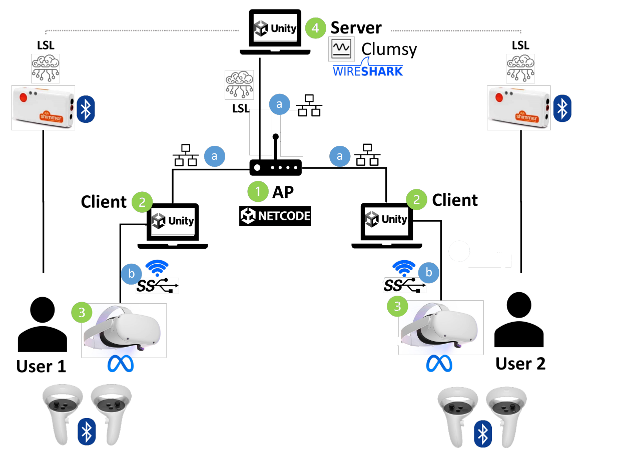


Figure 1: Schematic representation of the testbed architecture, depicting the interconnections and spatial arrangement of its elements

Section finalizes with a set of preliminary results captured in an initial experimental session.

3.1 System Architecture

Figure 1 presents the testbed architecture. It consists of two gaming laptops **(2)** and a server (4) hosting a Unity environment in version 2021.3.17f1. These laptops are *HP ZBook Studio 16 inch G9 Mobile Workstation PCs* with 32 GB of RAM, a *12th Gen Intel(R) Core(TM) i7-12800H@2.4Ghz CPU* and an *NVIDIA GeForce RTX 3070 Ti@1.48GHz Laptop GPU* with 8 GB GDDR6 memory. The latter, i.e., an NVIDIA GPU, is required to enable communication and rendering to the *Meta Quest 2 VR Headset* **(3)**. The server **(4)** is a *Corsair Graphite 380T Black Portable Mini ITX* with with 16 GB of RAM, a *4th Gen Intel(R) Core(TM) i7-4790 @3.6Ghz CPU* and an *EVGA GeForce GTX 980 Ti* with 6 GB GDDR5 memory. The content inside the two Unity environments is synchronized, i.e., kept identical, via a LAN over Ethernet **(a)** through a local access point **(1)**.

Each of the end-users is provided with a Meta Quest 2 VR HMD **(3)** to portray the virtual environment to the users. This HMD, produced by Meta, runs the Android OS, and provides 6DoF tracking with an LCD resolution of 1832 x 1920 pixels per eye and a refresh rate up to 120Hz. It has 6GB of RAM, a Qualcomm Snapdragon XR2 processor consisting of 4 Kyro 585 Silver@1.8GHz, 3 Kyro 585 Gold@2.42GHz and 1 Kyro 585 Prime@3.2GHz CPU cores and an Adreno 650@0.67GHz GPU (1.2 TFLOPS). Its movement tracking is provided by a combination of sensors and cameras on the outside of the HMD. The integrated sensors are Inertial Measurement Units (IMUs) that use an accelerometer, a gyroscope, and a magnetometer to track the position, velocity, and rotation of the HMD, resulting in full 6DoF tracking. As solely relying on this IMU would result in drifting overtime, Simultaneous Localization And Mapping (SLAM) and Light Detection and Ranging (LiDAR) are used on top of this as additional methods for tracking. Combining these methods results in a reliable 6DoF tracking of the HMD.

The Quest 2 is connected to the gaming laptop **(2)** by means of a USB 3.0 connection **(b)** provided by a specific Oculus Quest Link cable to maximize the available throughput (5 Gbps). This connection is used to stream the rendered Unity viewport to the headset and to send user movement (controllers and 6DoF movement of the HMD) back to the engine for updating the virtual environment. Furthermore, The HMD could also provide a Meta Quest AirLink connection that operates over Wi-Fi, providing more user movement but with less stability and lower throughput compared to Quest Link cable.

3.2 Implementation Details

In the design and implementation of our VR platform, the Unity engine[2] was selected for its comprehensive capabilities, crucial to the development of an immersive virtual experience. Unity's ability in physics simulation, such as the accurate simulation of gravity effects and collision dynamics, is integral to creating an engaging virtual experience.

Regarding the networking component of our VR platform, the Netcode for GameObjects (NGO)[3], which is specifically tailored for Unity, was utilized. NGO excels in transforming traditional *GameObject* and *MonoBehaviour* workflows into network-ready formats such as *NetworkObject* and *NetworkBehaviour*. This transformation is fundamental in enabling interoperability with a diverse range of low-level transport protocols. The significance of NGO lies in its efficiency in managing data exchange and synchronization within multiplayer environments. It adeptly oversees state synchronization, Remote Procedure Calls (RPCs), and network authority control, which are fundamental in ensuring a robust and responsive experience for multiple users interacting within the same VR environment.

Since the game consists of a series of consecutive tasks to be done, we also incorporated a mechanism for tracking the task or game objective. The tracking tasks within our game are architected around the Server Authoritative Mode. This design paradigm necessitates that any alteration within the game's environment must first be authenticated and approved by the server following a client's request. Post-approval, the server is responsible for updating all clients about the change, ensuring consistency across the user base. Furthermore, the server plays a crucial role in broadcasting its state to the clients. By accessing the server's state information, each client stays informed about the game's current state.

In contrast, to make a game more robust to network impairment, the movement of entities within the VR environment is governed by the Client Authoritative Mode. This mode allows clients to initiate and effectuate environmental changes, which are then communicated to the server. The server, in turn, updates other clients about these changes, ensuring a synchronized and cohesive multiplayer experience. It is important to note that clients can only modify entities for which they have ownership, thus preventing desynchronization between the users. This dual-mode approach of server and client authoritative systems provides a balanced and effective framework for managing interactions and states within our VR platform.

[2]https://unity.com/products/unity-engine
[3]https://unity.com/products/netcode

Figure 2: A scene of the cooperative virtual kitchen layout.

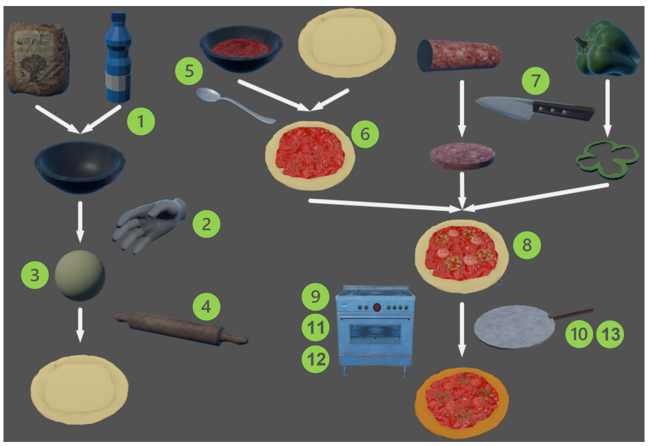


Figure 3: Illustration of the sequential steps involved in the collaborative pizza-making task within the virtual kitchen

Finally, to introduce network impairments, we integrated Clumsy[4], a tool that allows network traffic control. Clumsy offers the functionality to introduce various network conditions such as artificial lag, packet loss, throttling, and duplication, as well as out-of-order delivery and tampering. This enables us to simulate a range of network environments for thorough testing and optimization of our VR platform.

3.3 Use Case

In this research, we have developed a novel, CVR task as a proof of concept for such a system. The task entails a two-user scenario where participants collaboratively engage in the virtual preparation of a pizza within a carefully designed kitchen environment. Figure 2 illustrates the kitchen's layout, revealing an "H" shaped arrangement of tables. This specific configuration strategically divides the users, encouraging interactive and cooperative behavior.

A critical aspect of the virtual environment's design is the deliberate placement of utensils and ingredients, ensuring that neither participant has solo access to all the necessary items. This design choice necessitates the passing of objects between users, underscoring the essence of cooperation and teamwork for the successful completion of the task. The environment, thus, not only simulates a cooking activity but also serves as a testing ground for studying interaction dynamics and collaborative strategies in a VR setting.

Another feature of this virtual kitchen is an oven conveniently placed within reach of one user. This user is responsible for baking the pizza and then transferring it to the other participant via a pizza shovel, who completes the task by serving it on a plate.

A blackboard displaying the next steps is prominently placed on one of the virtual walls to guide the users through the pizza-making process, which inherently follows a sequential method. This aids in task organization and collaboration between the users. The cooking process itself is represented in Figure 3 and consists of the following steps.

(1) Combine water and flour in a bowl, necessitating User 1 to hand over water and flour to User 2.
(2) Knead the mixture until a ball of dough appears (User 2).
(3) Place the ball of dough on the shown indicator (User 2).
(4) User 1 has to Pick up the rolling pin with two hands and spread out the dough.
(5) User 1 passes the spoon to User 2. Afterward, User 2 dips the spoon into the bowl filled with tomato sauce.
(6) Spread the tomato sauce on the pizza. Once the spoon is empty, it should be refilled by dipping it in the bowl once again. Keep on adding tomato sauce until the pizza is fully covered (Both users can perform this task Both users can reach the spoon and pizza crust).
(7) Use the knife to cut four pieces of the sausage and four pieces of the bell pepper on the chopping board (User 1). To enable this, User 2 passes both the sausage and the pepper to User 1.
(8) Place the eight topping pieces on the pizza (User 1). This can also be done by passing the topping pieces to User 2.
(9) Open the oven by pressing the button (User 1).
(10) User 2 passes the pizza shovel to User 1. User 1 uses it to pick up the pizza and place it in the oven.
(11) Close the oven (User 1).
(12) Once the pizza is baked, open the oven (User 1).
(13) User 1 removes the pizza from the oven with the pizza shovel and passes it to User 2. User 2 places it on the plate.

3.4 User and Performance Monitoring

In terms of network monitoring, the NGO package includes some limited support for network measuring. To augment these capabilities, we integrated Wireshark [5], a widely recognized network protocol analyzer. Wireshark provides comprehensive capabilities for deep network traffic analysis and monitoring. It allows us to capture and interactively browse the traffic running on a computer network. Further enhancing our network monitoring framework, the HMD data flows are inspected using the 'USBPcap' plugin for Wireshark. This plugin is instrumental in capturing raw USB traffic, allowing for a detailed data transmission and reception analysis in the system.

[4]https://jagt.github.io/clumsy/

[5]https://www.wireshark.org/

Table 1: Overview of the four scenarios considered in this research.

Scenario	Latency (ms)	Burst (50% chance, ms)
A	0	0
B	0	500
C	500	0
D	500	500

To monitor the device requirements imposed by Unity on the HMD, the 'ADB dumpsys' tool is utilized. This diagnostic tool, running on Android devices, provides comprehensive insights into system services operational on the connected device, these logs of CPU, memory, and network usage could be used for further analysis, thus aiding in the assessment of the VR system's performance on the HMD.

For the laptop used in our setup, Unity's built-in 'Profiler' and 'Profile Analyzer' tools are deployed to track and evaluate essential performance metrics. These tools are key to ensuring that the laptop's hardware and software configurations are optimally tuned for the demands of the VR platform.

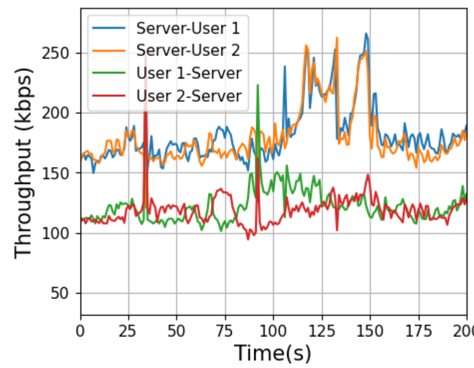
Regarding physiological signal recording from the subjects, we have incorporated two shimmer devices [6] that pair with the server via Bluetooth. A custom-developed program facilitates communication with these devices and streams their data onto the Lab Streaming Layer (LSL) middleware [7]. This middleware is crucial for synchronizing the physiological signal recording with the progression of the game's tasks.

To enhance the granularity of the physiological data analysis, we implemented a system to send markers to the LSL middleware in every state transition of the game. Specifically, upon completing each stage within the game, the game server program communicates with the application responsible for recording physiological signals. This communication is done through LSL, ensuring that physiological signals are accurately marked at the corresponding timestep of each game state transition. These markers are pivotal in extracting and analyzing physiological data with greater precision, correlating it closely with specific events and stages within the VR game. Furthermore, our platform can load a pre-configured file containing explicit assessment questions, storing the responses alongside the physiological data. This integrated approach to physiological monitoring, in conjunction with robust network and performance analytics, forms a comprehensive framework for evaluating and enhancing the user experience in our VR platform.

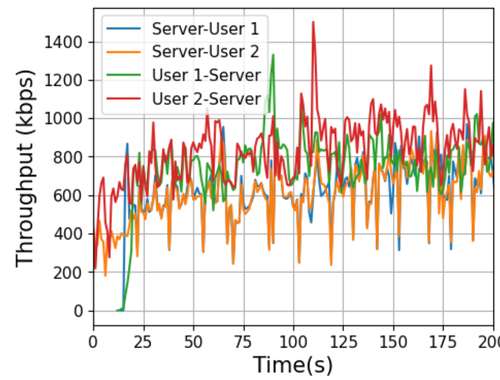
3.5 Evaluating Objective and Subjective measures under Latency related Impairments

The framework has been preliminary examined with three pairs of participants across four distinct network configurations to assess its efficacy in exploring the impact of network impairments. A total of four possible scenarios are outlined, as depicted in Table 1.

[6] https://shimmersensing.com/product/shimmer3-gsr-unit/
[7] https://labstreaminglayer.org/



(a) Configuration A (latency = 0 ms, burst = 0 ms)



(b) Configuration D (latency = 500 ms, burst = 500 ms)

Figure 4: Throughput comparison under Network Configurations A and D.

Figure 4 illustrates the system's throughput evolution across two sessions: one under control Configuration A (no impairments) and another under Configuration D (maximum impairment). Figure 4a demonstrates a similarity in user roles' behavior, underscored by significant throughput difference between uplink and downlink. Notably, between 70 and 80 seconds into the experiment, User 2's interactions are conveyed to the server (red line), which then forwards this information to User 1 (blue line). Similarly, the period between 90 and 110 seconds highlights User 1's activities being relayed to the server (green line) and onward to User 2 (orange line). The uplink is characterized by pronounced, but brief, peaks, particularly around the 25- and 120-second marks, corresponding to intensive vertex manipulation events (e.g., spreading sauce) that demand increased data synchronization.

Figure 4b demonstrates that network throughput under network-impaired Configuration D (latency = 500 ms, burst = 500 ms), distinctive patterns that were previously seen are less pronounced here, yet the throughput requirement surges two to threefold due to latency and burst, hampering immediate packet propagation. For an in-depth exploration of these effects and their implications, the reader is referred to our complementary study [10], which delves further into this subject.

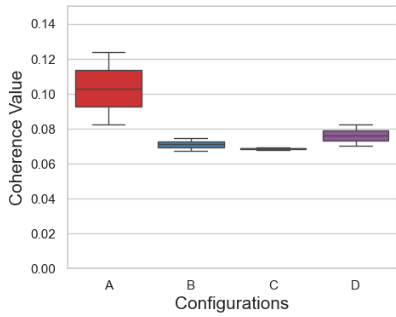


Figure 5: Participant Heart Rate Coherence

Figure 6: Participants Engaged in the Virtual Reality Experiment Setup

Preliminary analysis of implicit subjective measures from these participants indicates nuanced differences across network configurations. Figure 5 displays the coherence in heart rate between participant pairs under each condition. While overall coherence levels are modest, Configuration A exhibits notably higher coherence than other settings. This suggests that desynchronization in user behavior under network-impaired conditions, impacts synchrony. These observations underscore the need for further investigation into this phenomenon and the development of online QoE assessment methods based on physiological data.

4 DEMONSTRATION

To demonstrate our platform, we propose a session that requires two participants (conference attendees) to engage in the collaborative pizza-making VR game. Initially, they will be equipped with physiological monitoring devices – shimmer sensors on their hands, a heart rate sensor on their earlobes, and a skin conductivity measure sensor on their skin. These sensors will be connected to our server for real-time data collection, with physiological responses marked automatically through the LSL middleware. A meta quest 2 HMD will also be placed on each of the participants.

In the CVR session, the users will first enter a lobby in the VR environment, offering two options that can be selected by the laser pointer in the environment. The options are a local play mode for familiarization with the game mechanics and the server-connected collaborative mode (detailed in Section 3.3).

During the collaborative play, specific network distortions will be introduced for the users to perceive their impact. These include varying degrees of delay and burst latency. Delay is implemented at intervals of 0 ms (no delay), 100 ms, and 500 ms, effectively holding packets for these durations to emulate network lag and potentially prolong task completion.

Burst latency, on the other hand, is a more dynamic form of network impairment. Applied at intervals of 0 ms (no burst latency), 100 ms, and 500 ms, burst latency involves deliberately restricting network traffic flow at regular intervals and then releasing it in concentrated bursts. This can lead to a jerky user experience and may cause temporary desynchronization between users' perceptions of the virtual environment. A visual interface will provide live-monitoring capabilities for us and other conference attendees to observe and analyze in real-time how network quality influences collaborative interaction in the VR setting.

5 CONCLUSIONS

This paper has introduced a demonstrator for CVR. This demonstrator serves as a test bed to assess how different network configurations impact user performance in a CVR game focused on pizza making. It allows for a comparative analysis of these effects, providing valuable insights for future development in collaborative VR environments. Additionally, the capture of physiological signals such as heart rate and skin conductivity paves the way for novel approaches in modeling QoE implicitly, enhancing our understanding of user interaction in CVR settings.

ACKNOWLEDGMENTS

Sam Van Damme is funded by the Research Foundation Flanders (FWO), grant number 1SB1822N. This research is also partially funded by the FWO WaveVR project, grant number G034322N.

Furthermore, the authors want to acknowledge the contributions of Violeta Mediavilla and Fangio Van de Velde, who have assisted in the development of the framework discussed in this paper.

REFERENCES

[1] Dario Antonelli, Doriana M D'Addona, Antonio Maffei, Vladimir Modrak, Goran Putnik, Dorota Stadnicka, and Chrysostomos Stylios. 2019. Tiphys: an open networked platform for higher education on industry 4.0. *Procedia CIRP* 79 (2019), 706–711.
[2] Vuthea Chheang, Patrick Saalfeld, Tobias Huber, Florentine Huettl, Werner Kneist, Bernhard Preim, and Christian Hansen. 2019. Collaborative virtual reality for laparoscopic liver surgery training. In *2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*. IEEE, San Diego, CA, USA, 1–17.
[3] David Concannon, Ronan Flynn, and Niall Murray. 2019. A quality of experience evaluation system and research challenges for networked virtual reality-based teleoperation applications. In *Proceedings of the 11th ACM workshop on immersive mixed and virtual environment systems*. ACM, Amherst, MA, USA, 10–12.
[4] Lorenzo Damiani, Melissa Demartini, Guido Guizzi, Roberto Revetria, and Flavio Tonelli. 2018. Augmented and virtual reality applications in industrial systems: A qualitative review towards the industry 4.0 era. *IFAC-PapersOnLine* 51, 11 (2018), 624–630. https://doi.org/10.1016/j.ifacol.2018.08.388 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018.
[5] Fengxian Guo, F Richard Yu, Heli Zhang, Hong Ji, Victor CM Leung, and Xi Li. 2020. An adaptive wireless virtual reality framework in future wireless networks: A distributed learning approach. *IEEE Transactions on Vehicular Technology* 69, 8 (2020), 8514–8528.
[6] R Jinjia and X Dongliang. 2021. Networked VR: State of the Art, Solutions, and Challenges. *Electronics* 10, 2 (2021), 18 pages.
[7] Giuseppe Riva. 2005. Virtual Reality in Psychotherapy: Review. *CyberPsychology & Behavior* 8, 3 (2005), 220–230. https://doi.org/10.1089/cpb.2005.8.220 arXiv:https://doi.org/10.1089/cpb.2005.8.220 PMID: 15971972.

[8] Kristen M Triandafilou, Daria Tsoupikova, Alexander J Barry, Kelly N Thielbar, Nikolay Stoykov, and Derek G Kamper. 2018. Development of a 3D, networked multi-user virtual reality environment for home therapy after stroke. *Journal of neuroengineering and rehabilitation* 15 (2018), 1–13.

[9] Konstantinos Koumaditis Unnikrishnan Radhakrishnan and Francesco Chinello. 2021. A systematic review of immersive virtual reality for industrial skills training. *Behaviour & Information Technology* 40, 12 (2021), 1310–1339. https://doi.org/10.1080/0144929X.2021.1954693 arXiv:https://doi.org/10.1080/0144929X.2021.1954693

[10] Sam Van Damme, Javad Sameri, Susanna Schwarzmann, Qing Wei, Riccardo Trivisonno, Filip De Turck, and Maria Torres Vega. 2024. Impact of Latency on QoE, Performance, and Collaboration in Interactive Multi-User Virtual Reality. *Applied Sciences* 14, 6 (2024), 2290.

[11] Biao Xie, Huimin Liu, Rawan Alghofaili, Yongqi Zhang, Yeling Jiang, Flavio Destri Lobo, Changyang Li, Wanwan Li, Haikun Huang, Mesut Akdere, Christos Mousas, and Lap-Fai Yu. 2021. A Review on Virtual Reality Skill Training Applications. *Frontiers in Virtual Reality* 2 (2021), 19 pages. https://doi.org/10.3389/frvir.2021.645153

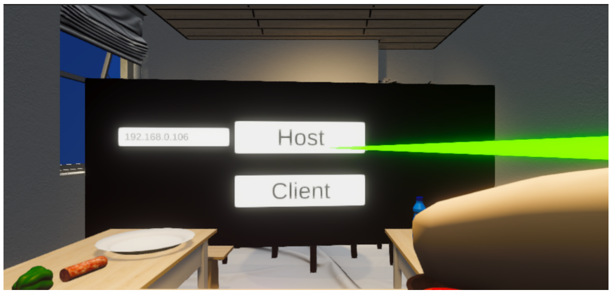


Figure 7: Visual depiction of server IP configuration and Welcome Menu

A ARTIFACT DESCRIPTION

This appendix elucidates the procedural steps essential to operationalize and examine the Collaborative Virtual Reality (CVR) cooking platform. The process starts with the loading of source code, followed by the configuration of software and hardware components as the core elements. Moreover, the guide expands upon the steps required to induce network distortions utilizing the Clumsy application.

A.1 Access and Requirements

A.1.1 Repository Access. The platform's source code is accessible via the BSD-3-Clause open-source license and can be downloaded from the following GitHub repository: https://github.com/mj-sam/CVR_cooking. For network distortion simulations, the Clumsy application is also available at https://jagt.github.io/clumsy/.

A.1.2 Software and Hardware Requirements. Certain hardware and software requirements are necessary for the operational efficiency of the CVR cooking platform. The hardware setup necessitates three Personal Computers (PCs) or laptops equipped with a Graphics Processing Unit (GPU) capable of rendering virtual reality environments on the Head-Mounted Display (HMD)s. Furthermore, two Meta Quest HMDs are essential for the immersive user experience, along with an access point to facilitate network connectivity among the devices.

On the software front, these PCs/laptops must be installed with the Unity engine, version 2021.3.17f1. The Oculus PC application is also required to enable the operation and integration of the Meta Quest HMDs within the virtual environment. For the networked communication and synchronization of entities within the CVR platform, Unity's Netcode for GameObjects, version 1.1.0, must be included in the Unity packages. Additionally, the PCs/laptops should have a minimum of 10 GB of disk space available to accommodate the installation and runtime data requirements.

A.2 Installation Procedures

The setup of the CVR cooking platform involves a two-step process for its deployment. The initial step entails cloning the project's GitHub repository to a local machine. Subsequent to cloning, the project must be activated within the Unity environment. At this step, verifying the Unity engine's version compatibility with the project's requirements is imperative. Upon project initiation in Unity, the engine starts an automatic download of the necessary packages. Users are advised to await the completion of this download process before advancing to further steps.

A.3 System Configuration

A.3.1 Hardware Configuration. The hardware setup requires the integration of two client systems and one host server, all connected to the same network and configured to be mutually accessible. Subsequently, each HMD should be connected to its respective client system. Given the platform's compatibility with Airlink and Cable connection methods, the HMDs may be connected using either option, based on availability or preference.

A.3.2 Platform Configuration. Upon successful installation, the platform requires further configuration to align with the operational requirements of the CVR environment.

Server Configuration: Locate and open the config.xml file within the platform's asset folder. Update the <ServerIp> element with the IP address of the host server, ensuring that all devices within the network can communicate with the server.

Client Configuration: Navigate to the Project Settings within the platform's interface and modify the HMD settings to select between Oculus or OpenXR, depending on the preference or user permission in use. Prepare for server-client interaction by configuring the Meta Quest HMD and its connection to the Oculus application on the PC/Laptop. Given the platform's tested compatibility with both AirLink and Meta Cable connections, select the connection method that best suits the operational needs. The server IP address must be entered into the designated text box within the game interface or the Unity network manager entity, as illustrated in Figure 7.

A.4 Running the Platform

Launching the CVR cooking platform involves initiating the server followed by connecting the clients.

Server Initiation: Begin by activating the server through the NetworkManager entity. This can be done by either selecting the Server button or choosing 'Host' from the welcome menu, the latter accessed via the VR controller's laser ray.

Client Connections: Once the server is active, clients can join the environment by engaging with the same NetworkManager entity or opting for 'Join' from the welcome menu. This flexibility facilitates user access based on individual preference or experimental settings.

A.5 Introducing Network Distortion

To simulate various network conditions impacting the CVR cooking platform, we utilize Clumsy. This tool allows for the introduction of network distortions through a variety of mechanisms, enumerated below:

- **Artificial Lag:** Mimics delays in data transmission to simulate slow network conditions.
- **Packet Drop:** Simulates the loss of packets during their transmission across the network.
- **Burst Behavior:** Generates sudden increases and blocks in network traffic to test the system's handling of data bursts.

- **Out-of-Order Delivery:** Emulates the scenario where packets arrive in a different order than sent.
- **Duplicate Delivery:** Simulates the condition of receiving the same packet more than once.
- **Tampering:** Alters packet data to simulate interference or data corruption.

Clumsy is initiated on the server to deploy these distortions. Given that the platform utilizes port 7777 for client-server communication, the following Clumsy filter is applied to target traffic through this port specifically:

```
tcp.DstPort == 7777 or tcp.SrcPort == 7777 or
udp.DstPort == 7777 or udp.SrcPort == 7777
```

This filter ensures that only the traffic to and from port 7777, whether TCP or UDP, is affected by the simulated network conditions.

Received ; revised ; accepted