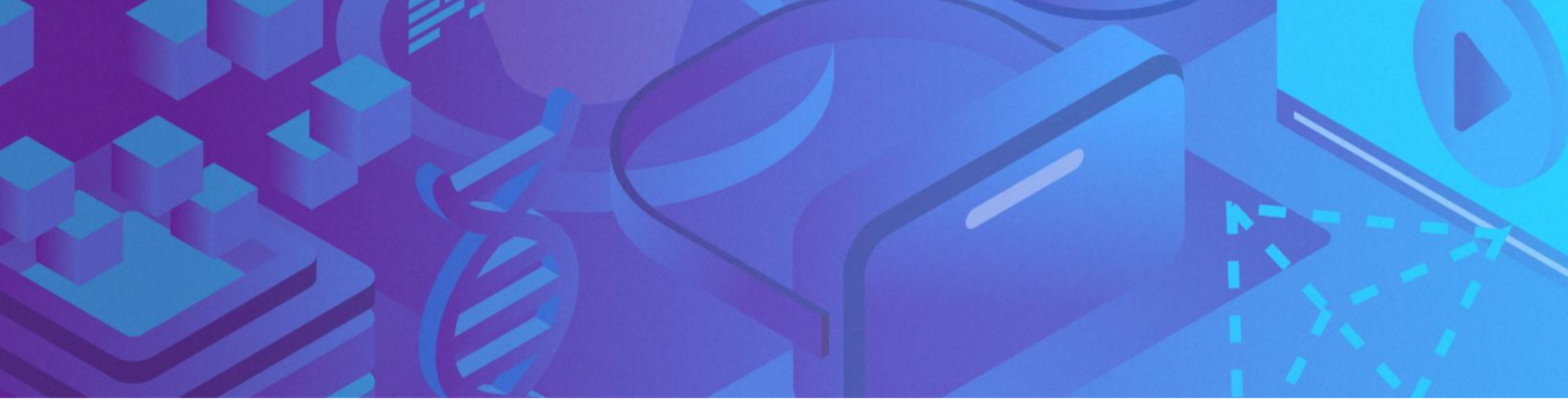


D4.3 SPIRIT PLATFORM (FINAL VERSION)

Revision: v.1.0

Work package	WP 4
Task	Task 4.1, 4.2, 4.3, and 4.4
Due date	30/09/2025
Submission date	30/09/2025
Deliverable lead	Deutsche Telekom
Version	1.0
Authors	Sven Wischnowsky - Editor (DT), José Santos, Casper Haems, Matthias De Fré (IMEC), Nick Turay (EDD), Christoph Stielow (TSI), Peter Hofmann (DT-Sec), Sergio Tejeda Pastor (Fraunhofer), Hermann Hellwagner, Shivi Vats (UNI-KLU), Peng Qian (SURREY), Carl Udora (UoB)
Reviewers	Tim Wauters (IMEC)
Abstract	This report documents the integration activities of the project, with a focus on the actual software development of the third version of the SPIRIT system platform for supporting heterogeneous telepresence use case applications and components. This process is driven by the development and integration of a set of project-designated use cases, system validation and testing.
Keywords	Development, telepresence, holographic communication, testing, evaluation.

www.spirit-project.eu

Grant Agreement No.: 101070672
Call: HORIZON-CL4-2021-HUMAN-01

Topic: HORIZON-CL4-2021-HUMAN-01-25
Type of action: HORIZON-RIA

Document Revision History

Version	Date	Description of change	List of contributor(s)
V1.0	30/09/2025	Third published version	José Santos, Casper Haems, Matthias De Fré (IMEC), Nick Turay (EDD), Vivien Helmut, Sven Wischnowsky (DT), Christoph Stielow (TSI), Peter Hofmann (DT-Sec), Sergio Tejeda Pastor (Fraunhofer), Hermann Hellwagner, Shivi Vats (UNI-KLU), Peng Qian (SURREY), Carl Udora (UoB)

DISCLAIMER

Funded by the Horizon Europe Framework Programme of the European Union

COPYRIGHT NOTICE

Project co-funded by the European Commission in the Horizon Europe Programme		
Nature of the deliverable:	**to specify R, DEM, DEC, DATA, DMP, ETHICS, SECURITY, OTHER***	
Dissemination Level		
PU	*Public, fully open, e.g. web (Deliverables flagged as public will be automatically published in CORDIS project's page)*	✓
SEN	*Sensitive, limited under the conditions of the Grant Agreement*	
Classified R-UE/ EU-R	*EU RESTRICTED under the Commission Decision No2015/ 444*	
Classified C-UE/ EU-C	*EU CONFIDENTIAL under the Commission Decision No2015/ 444*	
Classified S-UE/ EU-S	*EU SECRET under the Commission Decision No2015/ 444*	

* R: Document, report (excluding the periodic and final reports)

DEM: Demonstrator, pilot, prototype, plan designs

DEC: Websites, patents filing, press & media actions, videos, etc.

DATA: Data sets, microdata, etc.

DMP: Data management plan

ETHICS: Deliverables related to ethics issues.

SECURITY: Deliverables related to security issues

OTHER: Software, technical diagram, algorithms, models, etc.

Funded by the Horizon Europe Framework Programme of the European Union

EXECUTIVE SUMMARY

Telepresence can be seen as the next generation of communication applications that will significantly enrich the human-to-human and human-to-machine experience, blurring the boundaries between the physical and virtual worlds. Until now, telepresence solutions have been high-end, expensive, and quite conventional audio and video conferencing systems. However, the last decade has seen intensive research and development on VR/AR/XR technologies and applications that have significantly improved the state of the art.

The goal of the SPIRIT project is to realise Europe's first multi-site, interconnected framework dedicated for supporting the operation of heterogeneous collaborative telepresence applications at large scale through relevant technology innovations.

This report is a SPIRIT deliverable of the work package 4 “Platform Development, Integration and Validation”. It presents the third version of the SPIRIT Platform. The document is based on the second version report extended by enhancements and improvements added to the second version of the SPIRIT platform.

The work mainly focuses on the actual software development of the SPIRIT system platform for supporting heterogeneous telepresence use case applications, and also the integration of the platform with the distributed, interconnected network infrastructures underneath. This process is driven by the development and integration of a set of project-designated use cases.

The initial work consisted of integrating the partner components into a common testbed environment resulting in the first version of the SPIRIT platform. Based on this first version, a first Open Call was launched in early 2024, inviting third parties to realise their telepresence use cases experimentally. In parallel, development in the consortium continued, leading to the second version of the SPIRIT platform (see D4.2 [1]). For this, the second Open Call was launched end of 2024, inviting third parties to realise their telepresence use cases experimentally.

The platforms and components described in this document represent the final version of the SPIRIT offering.

This report

➲ provides an overview of the SPIRIT testbeds, in particular the local testbeds in Surrey, Bristol, Ghent and Berlin, e.g. the available 5G network infrastructure and the computing power of the Edge Cloud located there.

  In the first version of the SPIRIT platform, only Surrey and Berlin were available as local testbeds, providing 5G network and edge cloud computing resources. These resources may not be sufficient for some computationally intensive and graphically complex many-to-many telepresence scenarios using volumetric video. For this reason, imec's Virtual Wall (Ghent) was included in the project as an additional testbed to enable open call applications with high computing requirements in a controlled networking environment.

  Bristol was added as a testbed only recently, when Prof. Ning Wang moved from Surrey to Bristol and continued working on SPIRIT from there.

  A further expansion of the SPIRIT testbed is the secure interconnection of the local testbeds in Berlin, Surrey, and Bristol, so that use cases can now be realised via the connected testbeds. This interconnection was successfully tested with the use case Avatar: “Real-Time Animation and Streaming of Realistic Avatars” and with the “Holographic Human-to-Human Communication” use case.

Funded by the Horizon Europe Framework Programme of the European Union

- lists partner components that were integrated into the testbeds during the project and documents the findings of this integration process.

  Components that are independent of the testbed's infrastructure were deployed, validated, and tested in the local testbeds in Berlin, Surrey, and Bristol, e.g., the components enabling the use cases Hologram: "Holographic Human-to-human Communication" and Avatar: "Real-Time Animation and Streaming of Realistic Avatars". However, the autonomous mobile robot, for example, is still only available in Berlin.
- covers the implementation of partner use cases, integration of the platform components, and the description of the necessary adjustments for a smooth integration into the testbeds, as well as a look ahead to how these components might find use in various scenarios, such as third-party applications.

  For the Hologram use case, a USB device server component has been integrated into the test beds, which now also allows USB-only cameras to be connected to the Edge servers of a testbed.

  The containerisation of components has been optimised to simplify the integration into other operational environments or testbeds.

  The use case sections are concluded with an overview of their requirements from the D2.3 [2] report and the values realised in the testbeds.
- specifies an approach for deployment of Confidential Computing-protected VMs to ensure data owner's control of personal data.
- describes measures taken to investigate and implement methods for intrusion detection.
- presents two subjective Quality of Experience (QoE) studies to assess the impact of quality, quality switching, viewing distance, content characteristics, and compression algorithms on the perception of point clouds in AR/MR environments.
- additionally presents, based on eye-tracking data acquired in the second subjective study, head and gaze movement behaviour of participants and develops models for QoE prediction/estimation of point clouds in MR.
- finally, describes QoE assessments for project-specific use cases using objective QoE estimation models.

Funded by the Horizon Europe Framework Programme of the European Union

TABLE OF CONTENTS

Funded by the Horizon Europe Framework Programme of the European Union

Funded by the Horizon Europe Framework Programme of the European Union

Funded by the Horizon Europe
Framework Programme of the European Union

LIST OF FIGURES

Funded by the Horizon Europe Framework Programme of the European Union

Funded by the Horizon Europe Framework Programme of the European Union

Funded by the Horizon Europe Framework Programme of the European Union

Funded by the Horizon Europe Framework Programme of the European Union

Funded by the Horizon Europe Framework Programme of the European Union

LIST OF TABLES

Funded by the Horizon Europe Framework Programme of the European Union

ABBREVIATIONS

5G CPE	5G Customer Premises Equipment
5G NR	5G New Radio
5G SA	5G Stand Alone
5G/6GIC	5G/6G Innovation Centre
ACME	Automatic Certificate Management Environment
ANOVA	Analysis of Variance
API	Application programming interface
ASA	Adaptive Security Appliance
AR	Augmented Reality
BBU	Broadband Unit
CO	Confidential
CTC	Common Test Conditions
CSV	Comma-Separated Values
C-V2X	Cellular Vehicle-to-Everything
DCMS	Department for Digital, Culture, Media, and Sport
DT	Deutsche Telekom
EFI	Extensible Firmware Interface
ESA	European Space Agency
FoV	Field of View
G-QP	Geometry Quantization Parameter
GCC	Google Congestion Control
GUI	Graphical User Interface
HAS	HTTP Adaptive Streaming
HEVC	High Efficiency Video Coding
HPC	High-Performance Cluster
HMD	Head-Mounted Display

Funded by the Horizon Europe Framework Programme of the European Union

HSD	Honestly Significant Difference
HW	Hardware
ID	Identifier
IP	Internet Protocol
ITU	International Telecommunication Union
JANET	Joint Academic Network
JISC	Joint Information Systems Committee (UK)
KVM	Kernel-based Virtual Machine
LTE	Long-Term Evolution
MEC	Multi-access Edge Computing
MOQ	Media-over-QUIC
MOS	Mean Opinion Score
MPEG	Moving Picture Experts Group
MR	Mixed Reality
MRTK	Mixed Reality Toolkit
MSE	Mean Squared Error
NAS	Network Attached Storage
NTN	Non-Terrestrial Network
OS	Operating System
OVA	Open Virtualization Appliance
OVF	Open Virtualization Format
OVMF	Open Virtual Machine Firmware
PBKDF2	Password-Based Key Derivation Function 2
P2P	Peer-To-Peer
PC	Point cloud
PCC	Point cloud compression
PLCC	Pearson Linear Correlation Coefficient
PLMN	Public Land Mobile Network

Funded by the Horizon Europe Framework Programme of the European Union

PSP	Platform Secure Processor
PU	Public
QCOW2	QEMU-Format Copy On Write 2
QMP	Qemu Management Protocol
QoE	Quality of Experience
QP	Quantization Parameter
RAM	Random Access Memory
RAN	Radio Access Network
RAT	Radio Access Technology
RGB	Red Green Blue
RE	Restricted
REST	Representational State Transfer
RMSE	Root Mean Squared Error
SA	Standalone
SEV	Secure Encrypted Virtualization
SDK	Software Development Kit
SDN	Software-Defined Networking
SFU	Selective Forwarding Unit
STUN	Session Traversal Utilities for NAT
SME	Small and Medium Enterprises
SNMP	Simple Network Management Protocol
SRCC	Spearman’s Rank Correlation Coefficient
SSH	Secure Shell
T-QP	Texture Quantization Parameter
TAR	Tape Archive
TCB	Trusted Computing Base
TCP	Transmission Control Protocol
TMC2	Test Model Category 2

Funded by the Horizon Europe Framework Programme of the European Union

3D	Three-dimensional
TLS	Transport Layer Security
2D	Two-dimensional
UAV	Unmanned Aerial Vehicle
UEFI	Unified Extensible Firmware Interface
UI	User Interface
UKRI	UK Research and Innovation
ULL-DASH	Ultra-Low Latency Dynamic Adaptive Streaming over HTTP
URL	Unified Resource Location
URLLC	Ultra-Reliable Low-Latency Communications
UNI-KLU	University of Klagenfurt
VM	Virtual Machine
VMM	Virtual Machine Manager
VPN	Virtual Private Network
VR	Virtual Reality
WebRTC	Web Real-Time Communication
YAML	YAML Ain't Markup Language

Funded by the Horizon Europe Framework Programme of the European Union

1 INTRODUCTION

Telepresence can be seen as the next generation of communication applications that will significantly enrich the human-to-human and human-to-machine experience, blurring the boundaries between the physical and virtual worlds. Such systems are expected to fundamentally change the way people communicate and collaborate with each other in various sectors such as education, training, entertainment, retail, healthcare, manufacturing and many others. The further development of telepresence services will contribute significantly to increasing society's resilience to environmental disasters, boosting industrial productivity and improving energy efficiency, thanks to changes in people's lifestyles and work habits.

Until now, telepresence solutions have been high-end, expensive, and quite conventional audio and video conferencing systems. However, the last decade has seen intensive research and development on VR/AR/XR technologies and applications that have significantly improved the state of the art. This has led to interesting immersive telepresence and/or collaboration systems, some of which are still in the research stage. Due to their complexity, cost, data compression, and bandwidth requirements, these solutions have not scaled yet.

The mission of the SPIRIT project is to put real-time immersive telepresence into practice by researching, integrating, and further developing state-of-the-art immersive telepresence technologies, components and platforms to create Europe's first multi-site and interconnected framework able to support the operation of heterogeneous collaborative telepresence applications at large scale.

1.1 PURPOSE OF THE DOCUMENT

This report is about the final version of the tested and validated SPIRIT platform with all currently available components. It is one part of a bundle of deliverables providing insight into the project's use cases, requirements, architecture, available components, upcoming enablers, and the integrated SPIRIT platform:

- D2.3 “Use Case Requirements, System Architecture and Interface Definition (Final Version)” [2]
- D3.3 “Innovation Platform Enablers (Final Version)” [3]
- D4.3 “SPIRIT Platform (Final Version)” [4]

On the one hand it reflects the work done within the project, especially the tasks:

- **Platform development** - implement and/or enhance all platform components and interfaces.
- **Technical integration and validation** – integrate the components of the different project partners to form an overall platform demonstrator.
- **Use case development and integration -** ensures the development of the project-designated use cases and their integration with the underlying platform available.
- **User experience evaluation and usability validation -** provide and test quality of experience (QoE) metrics and procedures to assess immersive telepresence solutions and support of Open Call experiments.

Funded by the Horizon Europe Framework Programme of the European Union

On the other hand, it provides the applicants of the Open Calls with insights regarding the components available to enhance their use cases, and identify complementary components to the SPIRIT platform, which they may want to provide to the overall SPIRIT goal.

1.2 STRUCTURE OF THE DOCUMENT

The sections of the report at hand are organised in the following manner:

The chapter 2 “Testbeds”, provides an overview of the testbeds in Surrey, Bristol and Berlin, e.g. the available 5G network infrastructure and the computing power of the EdgeCloud located there. A fourth testbed, the Virtual Wall, has been added to the project infrastructure to enable resource intensive experiments in the SPIRIT platform on many-to-many tele-conferencing scenarios and innovations.

The chapter 3 “Platform components” lists partner components that were integrated into the testbeds during the project and documents the findings of this integration process.

The chapter 4 “Implemented Use Cases” covers the implementation of project-specific use cases, which are described in detail in [2]. On the one hand, the integration of the platform components into the use cases and the description of the necessary adjustments for a smooth integration into the testbeds are provided and, on the other hand, a look ahead to how these components might find use in various scenarios, such as third-party applications, is presented.

The chapter 5 “Security Development and Technical Integration” specifies an approach for the deployment of Confidential Computing-protected virtual machines (VMs) on bare metal servers (either on-premise or in the cloud) where private keys for managing VMs never leave the data owner’s control and specifically do not need to be uploaded to cloud provider systems. Appendix A is a manual to set up and make use of the Confidential Computing infrastructure.

The chapter 6 "Quality of Experience Evaluation” presents the subjective tests conducted by partners to assess the impact of quality, quality switching, viewing distance, and content characteristics on the perception of point clouds in AR environments. The output of this work includes (i) a platform for subjective quality assessment in AR environments, (ii) a dataset of rating scores that can be used for training and validating future QoE models as well as the results (findings) of the subjective tests that produced these rating scores, and (iii) a machine learning based QoE model. In appendix B, the Subjective Test platform is described. Additionally, the results of QoE assessments for project-specific use cases using objective QoE estimation models can be found in this chapter.

The chapter 7 “Conclusions” concludes the document.

Funded by the Horizon Europe
Framework Programme of the European Union

2 TESTBEDS

This section provides an overview of the testbeds in Surrey, Bristol and Berlin, e.g. the available 5G network infrastructure and the computing power of the EdgeCloud located there. It also provides information on imec's Virtual Wall infrastructure, which is used for resource intensive many-to-many conferencing scenarios and other Open Call projects.

2.1 SURREY

2.1.1 General Information on University of Surrey Testbed

The first UK site facility is managed by the 5G/6G Innovation Centre at the University of Surrey, in Guildford (Surrey). The testbed offers 5G infrastructure including campus-wide 4G/Long-Term Evolution (LTE) and 5G/New Radio (NR) based radio access network, virtualised 4G and 5G core network, managed Software-Defined Networking (SDN) and fibre external connectivity, as well as Unmanned Aerial Vehicle (UAV) and satellite systems.

The testbed supports domestic (UK) projects as well as European and International projects. Recent examples thereof are European Space Agency (ESA) 5G-TINA and SUNRISE 5G Pilot, Department for Digital, Culture, Media, and Sport (DCMS) Flex5G and FONRC TUDOR and UK Research and Innovation (UKRI) UK India (UKI-FNI). Small and medium enterprises (SMEs) are also in a partnership with the centre, in order to use the testbed to test different aspects of 5G-and-beyond technologies, such as Non-Terrestrial Network (NTN) communications, high accuracy time synchronization, network slicing, management and orchestration, flexible network functions disaggregation and energy efficiency.

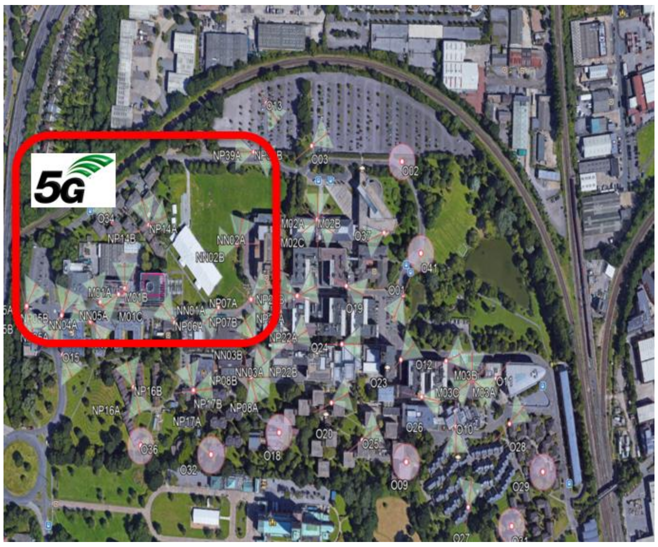


Figure 1: 5G Testbed in University of Surrey

The 5G/6GIC site offers a multi-Radio Access Technology (RAT) radio access network covering the University Surrey campus over four square kilometres and several off-campus locations where coverage is provided. It features both outdoor and indoor base radio units, supporting a mix of 4G/LTE and 5G/NR technologies, as shown in Figure 1. These base stations are used to showcase both 5G standalone (SA) and non-standalone (NSA) mobile network deployment scenarios and are currently used to support projects and research activities requiring traditional Radio Access Network (RAN) deployments.

Funded by the Horizon Europe Framework Programme of the European Union

2.1.2 Testbed Infrastructure

In the 5G testbed, the outdoor coverage consists of 40 LTE sites with 23 centralised BBUs hosted in 5G/6GIC, supporting 2.6GHz band 28 and 38 for a total of 120 cells. Furthermore, there are 3 outdoor NR sites in the 3.5GHz band n78 for a total of 9 cells and another 3 sites in the 3.7GHz band n77 consisting of 3 cells, with respectively, 3 and 2 centralised BBUs hosted in machine rooms in 5G/6GIC. Concerning infrastructure, as shown in Figure 2, the UK trial network hosts two machine rooms hosting physical computing infrastructure and broad-band units (BBUs) for both 4G and 5G, as well as both hardware and software-defined net-working equipment. The UK trial network is interconnected with several UK and international sites via JISC's JANET network, as well as via a satellite link. Finally, the UK trial network features additional equipment such as connected autonomous vehicles for C-V2X scenarios, UAV systems for scenarios of emergency pop-up networking, a prototype URLLC integrated setup used to demonstrate low latency use cases of up to 6K users.

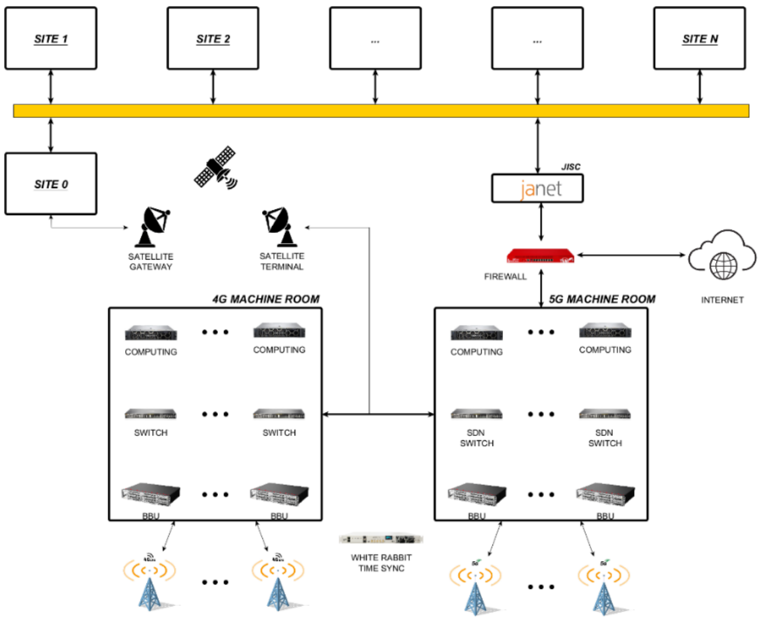


Figure 2: Infrastructure and Connectivity

Regarding the Holographic server and client connecting via 5G network, Figure 3: Holographic server and client connected via 5G network displays their hardware and software components. At left side, the holographic server application is deployed on 5G MEC server which is a Dell Precision 7960 XCTO base machine. In this machine, it has two Nvidia GeForce RTX A6000 video cards with 48 GB memory, a system memory of 256GB and three hard disk drives. On this machine, Windows 11 Pro for workstations is installed with the CUDA library to provide hardware acceleration to the application server software. On the system containers like Ubuntu can be enabled in the embedded containers, with the same ability to access video card re-sources. This machine is directly connected to the 5G network, and the 10G network interface can effectively receive raw hologram frames from clients. At the client side, two Dell Alienware R15 PCs are deployed with the livescan3d client application to fetch raw hologram frames from the camera. The CPU is an Intel Core i9 13900KF with 24 cores. The video card is a NVIDIA GeForce RTX 4090 with 24GB of memory. The system memory is 32 GB and hard disk space is 1TB + 1TB. With the local connectivity to 5G Customer Premises Equipment (5G CPE), the raw data can be streamed to the 5G MEC server.

Funded by the Horizon Europe Framework Programme of the European Union

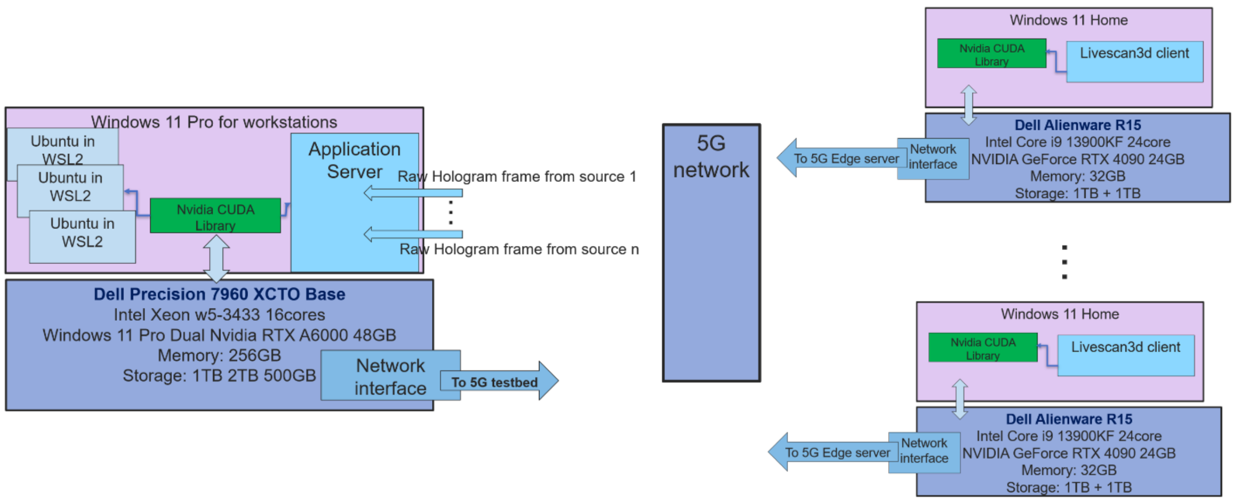


Figure 3: Holographic server and client connected via 5G network

2.1.3 Onsite Use Cases

During the SPIRIT project, the University of Surrey has worked together with the other SPIRIT partners to bring the use case “multi-source live teleportation with MEC support” to the testbed and make it available for Open Call participants. Detailed information can be found in in the use case descriptions of D2.3 “Use Case Requirements, System Architecture and Interface Definition (Final Version)” [2].

Continuous collaboration with other SPIRIT partners during the SPIRIT project has enabled the delivery of additional use cases and contributions to the Surrey testbed, as well as facilitating their availability for Open Call participants. These are:

- Real-Time Animation and Streaming of Realistic Avatars
- Holographic Human-to-Human Communications
- Network aware Kubernetes Scheduler

2.1.4 Onsite Collaborative Opportunities

There are two ways to collaborate with Surrey’s 5G testbed.

1) Deploy a container-based application at a specific 5G core slice with a MEC server to utilize the GPU resources, and 5G access and a public internet connection.

2) Deploy a standalone application and just use the 5G connectivity for the user to access the public internet.

For the first case, an executable application server program can be deployed in one of the containers. The user device (e.g., 5G phone or HoloLens with 5G phone tethering) of the application which is the content receiver can then connect to the 5G radio or another public Internet address to exchange application messages and receive application data from the container server.

Funded by the Horizon Europe Framework Programme of the European Union

In the second case, one or multiple devices can access the public Internet through the 5G radio network and can therefore connect to any remote application server deployed at the public internet.

Furthermore, to aid collaborative efforts as well as testing, SPIRIT project participants will be able to leverage the testbed facilities both remotely and onsite. In furtherance of this, additional computing resources including but not limited to VMs, GPUs and client devices (which include intel Realsense D435(i) depth camera, Azure Kinect DK depth camera, and 5G compatible smartphones) have been made available. External devices can also be supported following relevant compatibility validation.

2.2 BRISTOL

2.2.1 General Information on University of Bristol Testbed

The University of Bristol's 5G testbed represents a cutting-edge research and innovation platform, enabling exploration across a broad spectrum of next-generation technologies. Strategically located within Bristol city centre, the testbed's interconnected nodes facilitate investigations into classical datacentre infrastructure, AI, edge computing, and even quantum computing. This report provides a comprehensive overview of the testbed's status, capabilities and research focus areas.

2.2.2 Testbed Infrastructure

The testbed infrastructure comprises key locations essential to our research, with the core infrastructure centralized within the Smart Internet Lab (SIL) at the University of Bristol (UoB). This facility houses critical equipment, including routers, switches, and hardware necessary for conducting research activities. It also supports the critical network infrastructure required for ongoing projects at UoB. SIL has developed a private cloud network spanning multiple edge locations, such as WTC and MSHED, to advance Mobile Edge Computing (MEC) research and explore innovative concepts within the Open Radio Access Network (O-RAN) infrastructure, including midhaul and fronthaul communication over long distances (Figure 4).

Funded by the Horizon Europe Framework Programme of the European Union

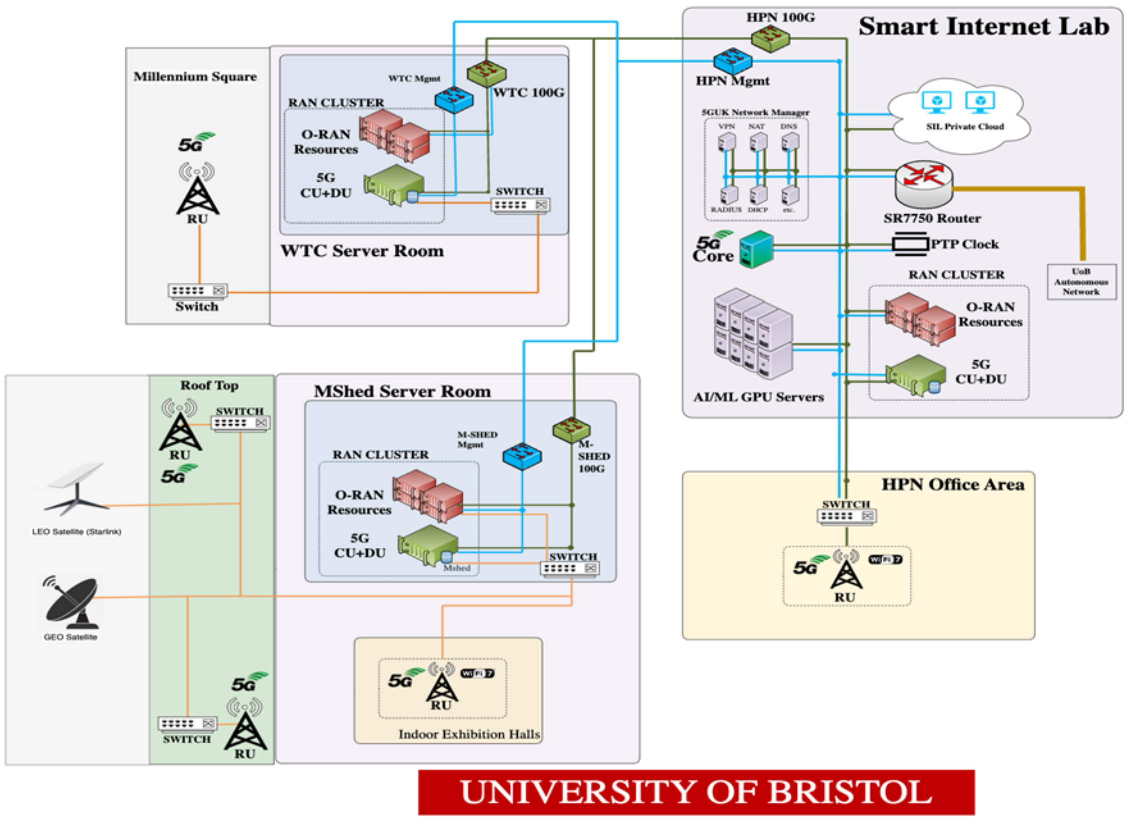


Figure 4: Smart Internet Lab testbed architecture

The cloud infrastructure is based on OpenStack, an open-source platform for managing and orchestrating large pools of compute, storage, and networking resources within our data centre. This system combines older servers, previously used for earlier projects, with new, state-of-the-art servers equipped with GPUs for machine learning (ML) and artificial intelligence (AI) tasks, resulting in the creation of an extensive private cloud platform across Bristol. These servers are typically equipped with multi-core CPUs, large amounts of RAM, and ample hard drive storage. However, supporting the needs of our research would require a substantial number of servers, making it challenging to set up and maintain the infrastructure across both our lab and edge locations.

To address this, we have invested in tools that provide high availability and automation, such as MaaS (Metal as a Service) and a dedicated CEPH storage system. MaaS is a tool that automates the provisioning and management of physical servers, allowing us to deploy new machines quickly and efficiently, while reducing manual intervention. CEPH is a distributed storage system that offers scalability and redundancy, enabling us to perform live virtual machine (VM) migration between servers without disrupting researchers, ensuring continuous operation without downtime or interruptions.

Additionally, the University of Bristol has developed a series of containerized environments utilizing Kubernetes and microK8S, with each cluster dedicated to specific functions, ranging from 5G cores to specialized AI platforms that require tailored environments with Nvidia driver support and GPU passthrough.

This private cloud is offered as a collaborative platform for research groups and project partners, providing secure and isolated workspaces to facilitate seamless collaboration and experimentation. The lab is equipped with cutting-edge technologies, including multiple 5G O-RAN

Funded by the Horizon Europe Framework Programme of the European Union

deployments with Radio Intelligent Controllers (RIC). The University of Bristol has been actively collaborating with local Bristol-based companies to implement these deployments across the city centre, enabling urban 5G testing and the evaluation of realistic use-case scenarios in areas with high mobile device usage. We have deployed two 5G radios in the outdoor area of the MShed Museum, each covering a distinct location while a third outdoor radio is situated in Millennium Square. This configuration forms a coverage loop in the city center, enabling comprehensive experimentation across various locations and facilitating handover testing between the sites.

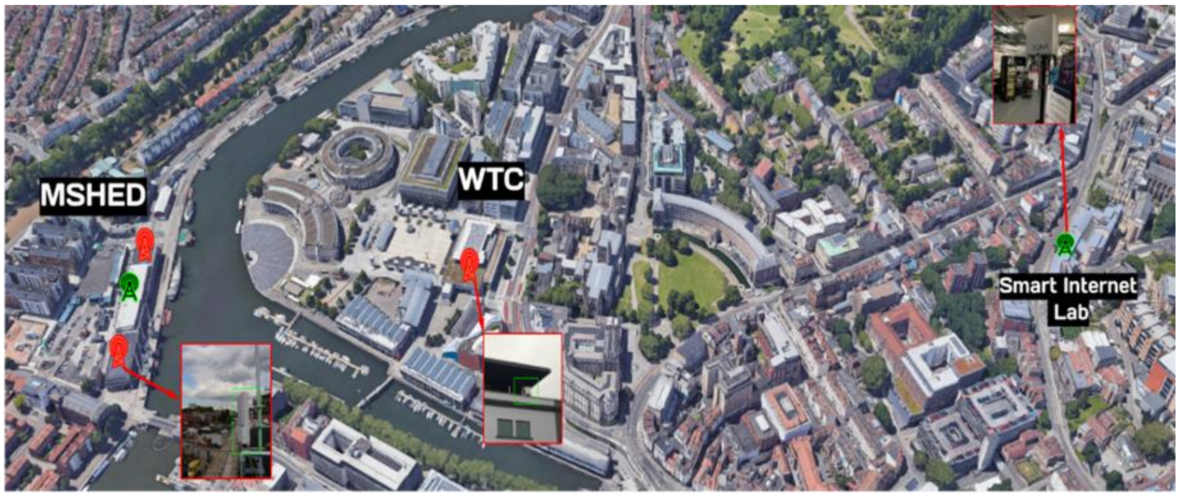


Figure 5: Configuration of the outdoor radios

Furthermore, SIL is focused on developing multi-access technologies within a unified solution to serve as a foundation for Beyond 5G (B5G) research and experimentation. A notable achievement in this area is the development of the first 5G Standalone (SA) multi-technology Customer Premises Equipment (CPE), which integrates 5G, Wi-Fi 7, and Li-Fi, among other technologies, to provide a more reliable and higher-performing client for experimental research.

2.2.2.1 Nomadic Node

One of the key initiatives pursued by the University of Bristol (UoB) is the mobile Beyond 5G (B5G) solution, termed the "Nomadic Node" (Figure 6). This concept involves integrating and deploying a range of both established and emerging technologies—traditionally confined to laboratory and data centre environments—into a mobile, adaptable, and compact format. Technologies incorporated into this solution include wireless technologies such as 5G (O-RAN), Wi-Fi 7, and Li-Fi, as well as MEC computing, cloud and containerized infrastructure, and machine learning and AI data processing servers. The Nomadic Node has demonstrated significant advantages for various use cases, particularly in scenarios with limited connectivity. It enables the deployment of numerous applications closer to the edge or client without compromising reliability and wireless bandwidth, providing usability even at more extreme cases.

Funded by the Horizon Europe Framework Programme of the European Union

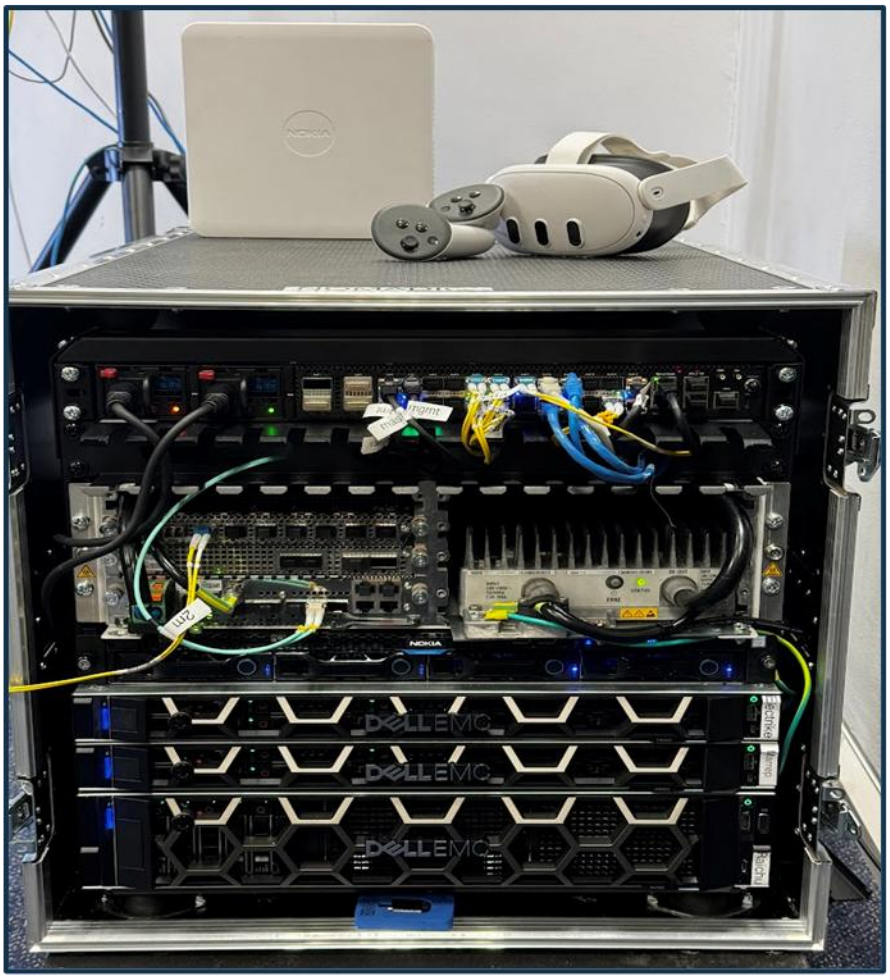


Figure 6: Nomadic Node with Nokia indoors 5G

2.2.2.2 Core Capabilities

The testbed empowers research in a wide array of technology enablers:

- Programmable Radio Networks: Open-RAN architecture supports 4G, 5G, and beyond 5G (B5G) air interfaces, fostering flexibility and innovation.
- Service-Based Architecture: Standalone 5G core network functions facilitate the creation of service-centric network architectures.
- Network Slicing: SDN-enabled switches enable network slicing, providing isolated and tailored connectivity between nodes.
- Flexible Resource Allocation: Open-Source MANO (OSM) orchestrates the allocation of compute, network, and storage resources, while project spaces retain the freedom to explore alternative virtualization technologies.
- Secure Remote Access: A carrier-grade service router provides research partners with secure VPN access to testbed resources.

Funded by the Horizon Europe Framework Programme of the European Union

- In-House Developed Devices: Multi-RAT CPE devices and IoT sensors/actuators enable a wide range of service experimentation and data collection.
- Comprehensive Monitoring: Network parameters and performance are monitored across core, transport, and RAN functions, providing valuable insights for optimization.
- AI and Machine Learning: The testbed supports research into machine learning techniques and AI algorithms for specific use cases.
- IoT Data Platform: IoT devices generate a rich data platform for various use cases.
- Quantum Key Distribution: The testbed enables research into quantum key distribution using dark fibre across multiple sites.

2.2.3 Onsite Use Cases

Similar to the University of Surrey, the University of Bristol has worked together with the other SPIRIT partners to bring the use case “multi-source live teleportation with MEC support” to the testbed and make it available for open call participants. Detailed information can be found in in the use case descriptions of D2.3 “Use Case Requirements, System Architecture and Interface Definition (Third Version)” [2].

Collaboration and Integration efforts with other SPIRIT partners during the SPIRIT project has enabled the delivery of additional use cases and contributions to the Bristol testbed, as well as facilitating their availability for open call participants. These are:

- Real-Time Animation and Streaming of Realistic Avatars
- Holographic Human-to-Human Communications

2.2.4 Onsite Collaborative Opportunities

To collaborate with the University of Bristol, participants can access the Bristol testbed to deploy their solutions in the relevant environment. To access the testbed, VPN access managed by the Bristol testbed will be provided to the users allowing access to relevant VMs and resources. Remote access, and allocation of computational resources are options available to enable easy and efficient collaboration. This allows users to collaborate with no limitations on physical location.

Users upon gaining access to the testbed can proceed with the setup of their dependencies and the deployment of their relevant applications or solutions. Additional steps are taken at the Bristol testbed to ensure privacy and security.

Furthermore, to aid collaborative efforts, SPIRIT project participants will be able to leverage the testbed facilities both remotely and onsite similar to the Surrey testbed. In furtherance of this, additional computing resources including but not limited to VMs, GPUs and client devices (which include intel Realsense D435(i) depth camera, Azure Kinect DK depth camera, and 5G compatible smartphones) have been made available. External devices can also be supported following relevant compatibility validation.

Funded by the Horizon Europe Framework Programme of the European Union

2.3 BERLIN

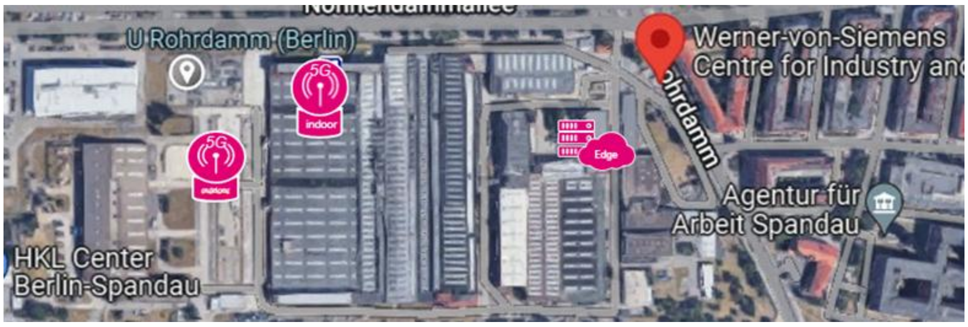

U Rohrdamm (Berlin)
5G
indoor
5G
outdoor
Edge
Werner-von-Siemens Centre for Industry and
Rohrdamm
Agentur für Arbeit Spandau
HKL Center Berlin-Spandau

Figure 7: Location of Deutsche Telekom Testbed

2.3.1 General Information Deutsche Telekom Testbed

The Deutsche Telekom 5G Testbed is located in Berlin, Germany at the “Siemensstadt Square” district where Deutsche Telekom collaborates with the Werner von Siemens Centre and various other partners coming from industrial, public and educational sectors. The research factory is situated in the same old building as the Siemens dynamo factory, which started its production in the early 1900s and is still active up to today. Many different partners experiment and cooperate on innovative topics right next to the actual factory. We are looking into a wide range of topics, including new and innovative manufacturing techniques, advanced transportation and mobility solutions, the shift towards more sustainable energy sources and many other interesting areas.

Figure 8: Location of Testbed in Berlin

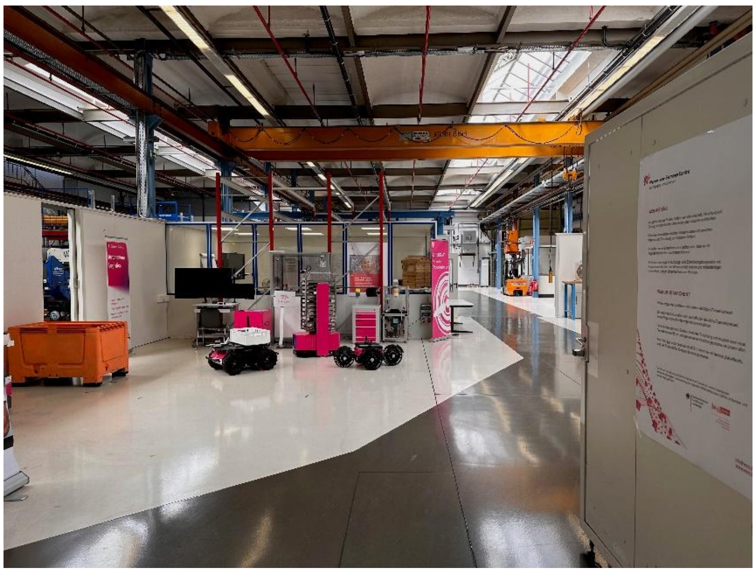


Figure 9: Lab Floor Werner von Siemens Centre

Funded by the Horizon Europe Framework Programme of the European Union

2.3.2 Testbed Infrastructure

Together with the SPIRIT Project, Deutsche Telekom is opening its Future Factory for partners. The site offers an area of around 500 m² outdoor space and 1000 m² indoor space. The indoor area as well as the outdoor area is covered by a private 5G Standalone Network. The 5G network is configured for the 3.7- 3.8 GHz industrial spectrum and has been approved by the German Federal Network Agency. The virtualized 5G core supports the 3GPP standard up to Release 16. The indoor space is also partly covered with WIFI5 and WIFI6 and can be used to connect devices that aren't capable of 5G. Computing power is provided by an edge server that is located on premise and is connected via fibre to the 5G Network. The server runs containerized applications in a Kubernetes cluster. There are two Nvidia T4 16GB Graphic Cards, 96 GB of RAM and 112 Cores (3 x Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz) available in the cluster that are being shared between all tenants.

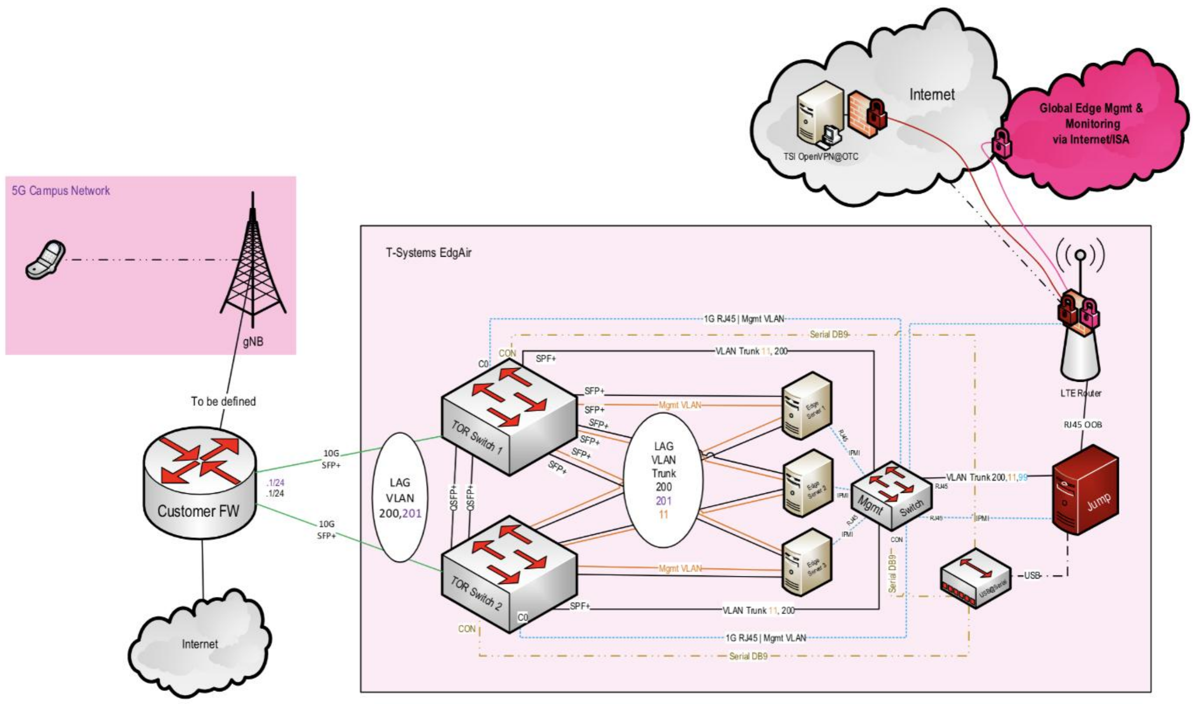


Figure 10: Network Infrastructure

2.3.3 Onsite Use Cases

During the SPIRIT Project, Deutsche Telekom has worked together with the other SPIRIT partners to bring three contributions and use cases to the testbed and make them available for open call participants:

- Real-Time Animation and Streaming of Realistic Avatars
- Holographic Human-to-Human Communications
- Network aware Kubernetes Scheduler

Furthermore, the "Distributed Steering of Autonomous Mobile Robots (AMRs)" use-case is exclusively available in the Deutsche Telekom testbed. Detailed descriptions of these use cases can be found in the use case descriptions of D2.3 [2].

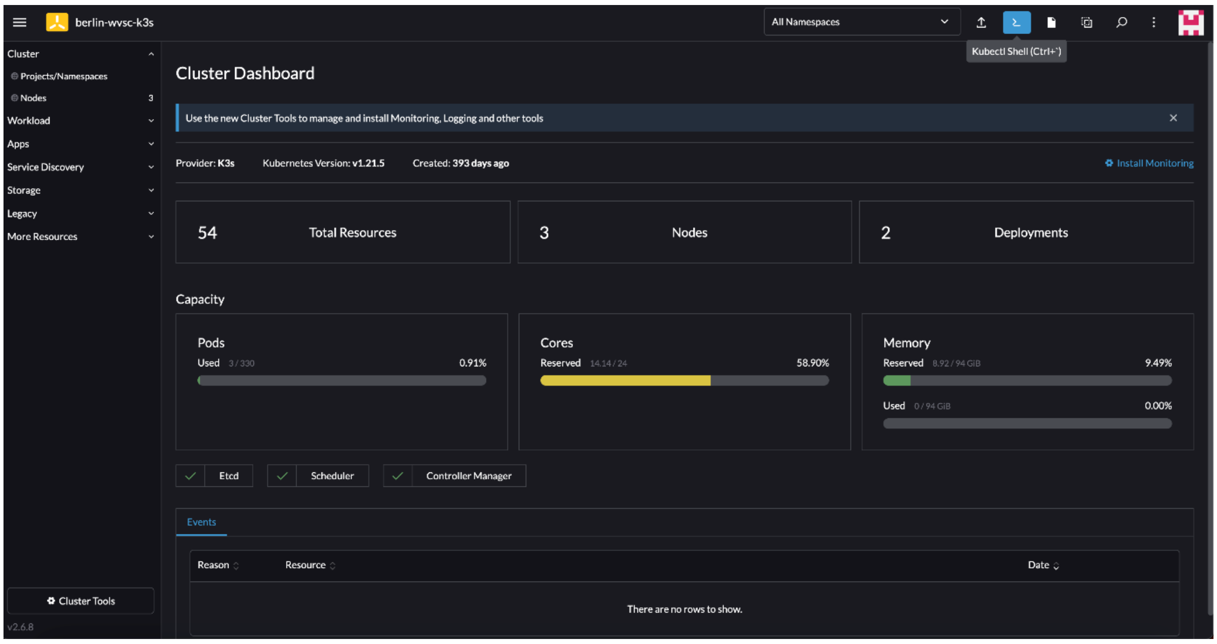
Funded by the Horizon Europe Framework Programme of the European Union

2.3.4 Onsite Collaborative Opportunities

Participants can deploy their software on our local edge cloud. To do so, they get a dedicated access to the Rancher UI, our Kubernetes management platform, to remotely deploy and manage their applications in the cluster. The platform is accessible from the internet and gives partners the opportunity to deploy their software without the need of being onsite. The platform uses multitenancy for ensuring both privacy and security. For this we use namespaces, which are divided into Rancher projects. Each project contains users divided by access levels within the project (we can configure an individual set of permissions or choose from predefined: Owner, Member, Readonly). Access to the storage is also possible only within the namespace/project.

Also, for each project we can limit the allocation of resources (CPU, RAM, etc.).

After identifying that excessive CPU and RAM usage by some partners led to resource shortages and the subsequent shutdown of the cluster, consumption limits have been implemented to prevent future disruptions and maintain system stability. These limits are configured per namespace in the Kubernetes cluster and can also be changed briefly for the purpose of experimentation.



Cluster Dashboard

Use the new Cluster Tools to manage and install Monitoring, Logging and other tools

Provider: **K3s** Kubernetes Version: **v1.21.5** Created: **393 days ago** Install Monitoring

54 Total Resources | 3 Nodes | 2 Deployments

Capacity

Pods — Used 3 / 330 — 0.91%

Cores — Reserved 14.14 / 24 — 58.90%

Memory — Reserved 8.92 / 94 GiB — 9.49%; Used 0 / 94 GiB — 0.00%

✓ Etcd ✓ Scheduler ✓ Controller Manager

Events

Reason	Resource	Date
There are no rows to show.		

Figure 11: Kubernetes management Platform

There is a managed firewall active inside the cluster, which is why network traffic needs to be approved and configured.

New network hardware has been installed at the Berlin testbed that allows us to configure project-specific domains. As part of this setup, we have created a SPIRIT-specific domain, see Figure 12. Within the SPIRIT domain, we plan to configure different VLANs for OpenCall subscribers to allow separation of partners and use cases. These VLANs can be tailored to the requirements of the OpenCall participants. In addition, a management process has been introduced to document the network topology, allowing a clearer overview and easier management in the future.

Funded by the Horizon Europe Framework Programme of the European Union

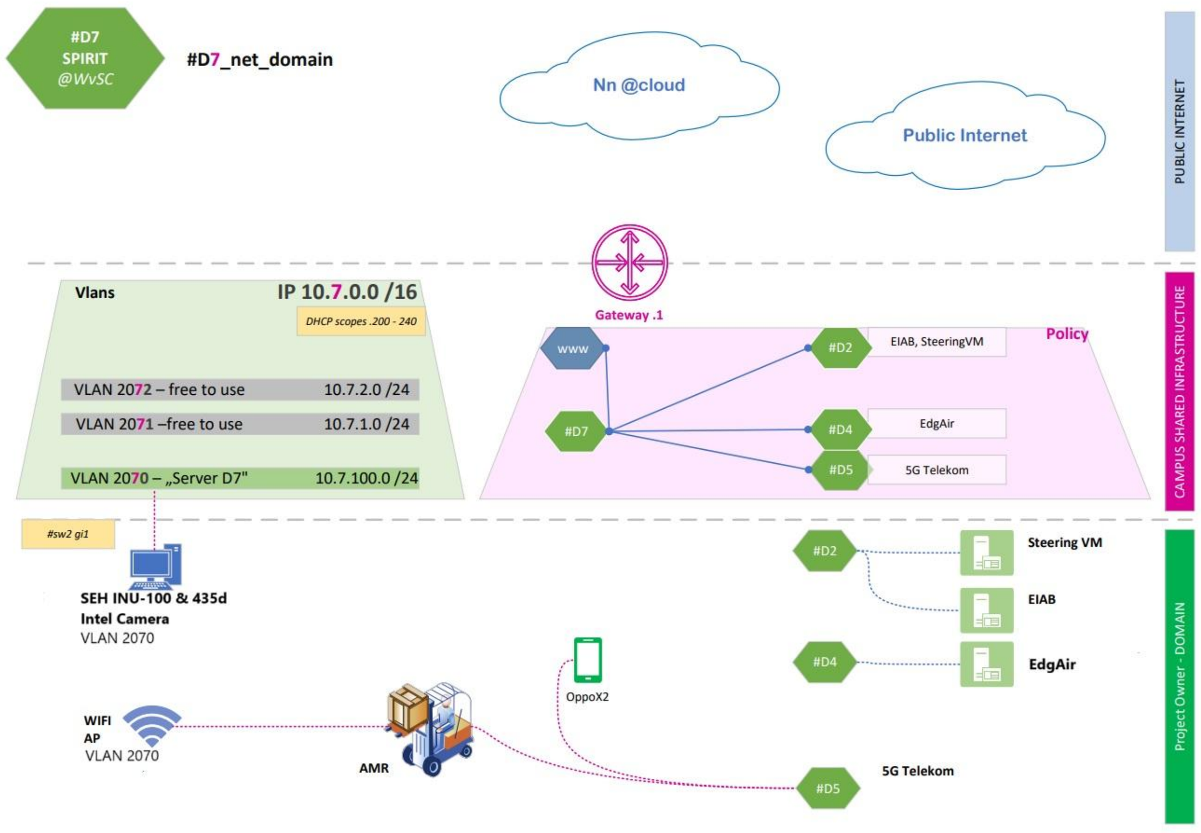


Figure 12: SPIRIT Network Domain

For testing and collaboration, participants of the SPIRIT project will have the possibility to make appointments for onsite visits. In general, except when otherwise stipulated, participants are supposed to bring their own devices to the testbed to test their use case. We offer three ways to connect these devices to our network:

- 5G: We will provide a SIM Card that is configured for the local 5G network
- WIFI: We help connecting to an access-controlled WiFi
- LAN: We will provide an ethernet port to connect to the cluster

The 5G Standalone (5GSA) Network now supports an expanded range of devices, allowing a broader variety of experiments to be carried out for open call participants. Two 5G Oppo phones have successfully been connected to 5GSA and are available for Open Call participants. In addition to the smartphones, iPads have been introduced as compatible end-user devices capable of utilizing the 5GSA network. Moreover, a USB-Device-Server has been tested with Ericsson allowing USB devices to connect to the edge server over the testbed's network, providing even greater flexibility and accessibility.

As not all 5G devices are able to connect to our private 5G network, you can find a list of officially supported devices on this website:

- https://hardware.iot.telekom.com/Hardware/Applications?id=44

We have tested more devices than those on the list and are also open to testing further devices, but the devices on the list are those that are officially supported by our 5G network.

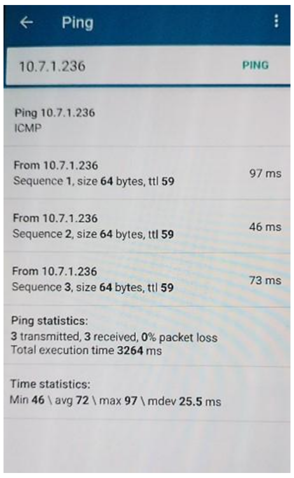
Funded by the Horizon Europe Framework Programme of the European Union

2.4 INTERCONNECTION OF SURREY/BRISTOL AND BERLIN TESTBEDS

In the course of the SPIRIT project, the collaborative efforts of the project partners enabled the interconnection of the testbeds to further enhance the use cases by facilitating the integration and validation of interconnected use cases. In this regard, an IPSec/VPN tunnel was established between the testbeds, enabling the routing of relevant traffic between them. Further collaborative validation efforts were conducted to provide sufficient determination of the interconnection.

Ping Test

Networking validation efforts include an initial ping test between both endpoints. This was conducted multiple times from both testbeds. From the Figure 13, it is evident that the ping tests were successful, indicating that communication between both testbeds was stable.



Ping

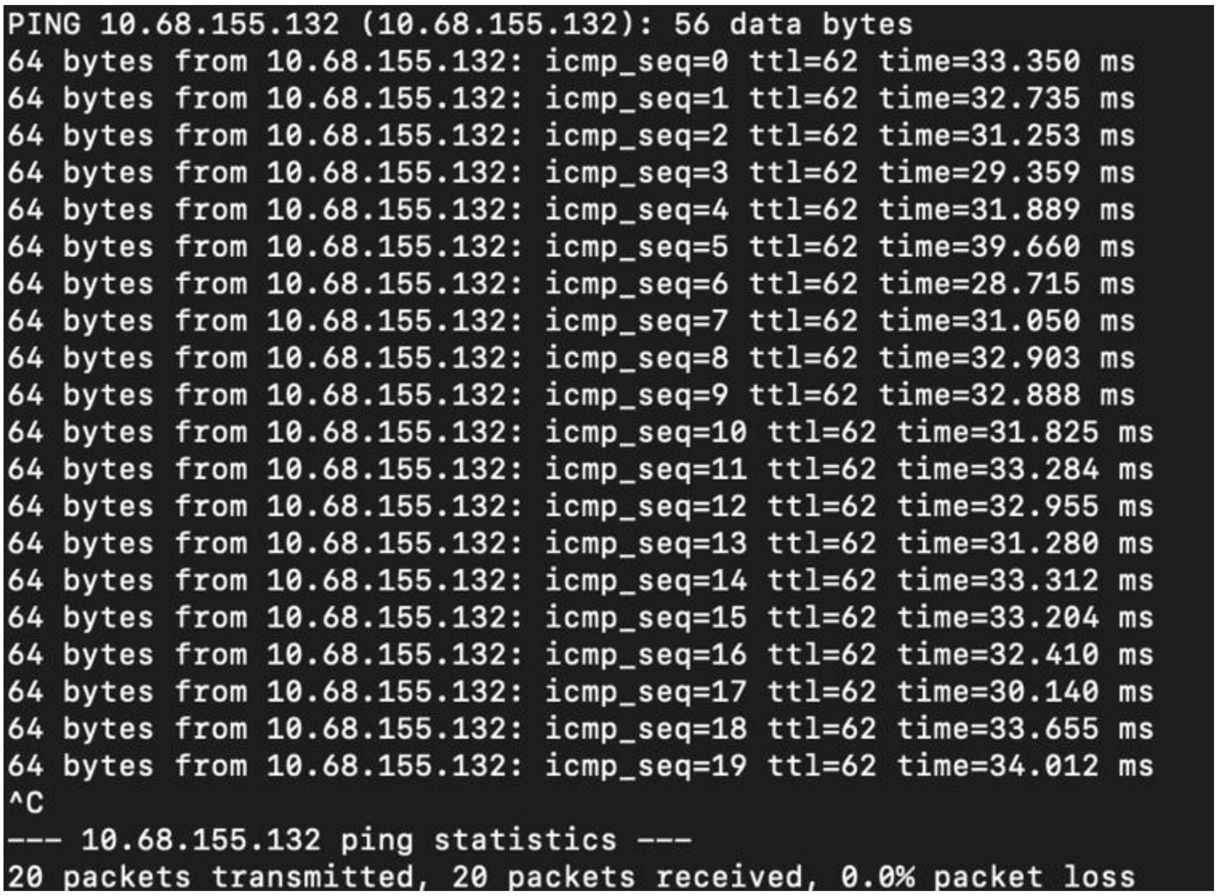
10.7.1.236 PING

```
Ping 10.7.1.236
ICMP

From 10.7.1.236
Sequence 1, size 64 bytes, ttl 59        97 ms

From 10.7.1.236
Sequence 2, size 64 bytes, ttl 59        46 ms

From 10.7.1.236
Sequence 3, size 64 bytes, ttl 59        73 ms

Ping statistics:
3 transmitted, 3 received, 0% packet loss
Total execution time 3264 ms

Time statistics:
Min 46 \ avg 72 \ max 97 \ mdev 25.5 ms
```

Figure 13: Interconnection Ping Test

Similar network validation efforts were explored between the Bristol and Berlin testbeds, with a ping test conducted between both points. From Figure 14 it is evident that the ping tests were successful, indicating stable communication between both testbeds.

Funded by the Horizon Europe Framework Programme of the European Union



```
PING 10.68.155.132 (10.68.155.132): 56 data bytes
64 bytes from 10.68.155.132: icmp_seq=0 ttl=62 time=33.350 ms
64 bytes from 10.68.155.132: icmp_seq=1 ttl=62 time=32.735 ms
64 bytes from 10.68.155.132: icmp_seq=2 ttl=62 time=31.253 ms
64 bytes from 10.68.155.132: icmp_seq=3 ttl=62 time=29.359 ms
64 bytes from 10.68.155.132: icmp_seq=4 ttl=62 time=31.889 ms
64 bytes from 10.68.155.132: icmp_seq=5 ttl=62 time=39.660 ms
64 bytes from 10.68.155.132: icmp_seq=6 ttl=62 time=28.715 ms
64 bytes from 10.68.155.132: icmp_seq=7 ttl=62 time=31.050 ms
64 bytes from 10.68.155.132: icmp_seq=8 ttl=62 time=32.903 ms
64 bytes from 10.68.155.132: icmp_seq=9 ttl=62 time=32.888 ms
64 bytes from 10.68.155.132: icmp_seq=10 ttl=62 time=31.825 ms
64 bytes from 10.68.155.132: icmp_seq=11 ttl=62 time=33.284 ms
64 bytes from 10.68.155.132: icmp_seq=12 ttl=62 time=32.955 ms
64 bytes from 10.68.155.132: icmp_seq=13 ttl=62 time=31.280 ms
64 bytes from 10.68.155.132: icmp_seq=14 ttl=62 time=33.312 ms
64 bytes from 10.68.155.132: icmp_seq=15 ttl=62 time=33.204 ms
64 bytes from 10.68.155.132: icmp_seq=16 ttl=62 time=32.410 ms
64 bytes from 10.68.155.132: icmp_seq=17 ttl=62 time=30.140 ms
64 bytes from 10.68.155.132: icmp_seq=18 ttl=62 time=33.655 ms
64 bytes from 10.68.155.132: icmp_seq=19 ttl=62 time=34.012 ms
^C
--- 10.68.155.132 ping statistics ---
20 packets transmitted, 20 packets received, 0.0% packet loss
```

Figure 14: Bristol-Berlin interconnection ping test

Interconnected Use Cases

Building on the successful validation tests, collaboration between the SPIRIT project partners facilitated the establishment of an integrated interconnected use case. In this context, the real-time animation and streaming of realistic avatars was integrated to utilize the interconnection. The server was deployed at the Surrey testbed, and the client was set up at the Berlin testbed. Further collaborative efforts from partners assisted in the configuration of the server and client connections, allowing the client in Berlin to connect to the server, which was started at the Surrey testbed. This allowed for the exchange of audio and video data, allowing for real time viewing of the avatar and as such the establishment of the interconnected real-time animation and streaming of realistic avatars use case. Figure 15 depicts the interconnected use case setup.

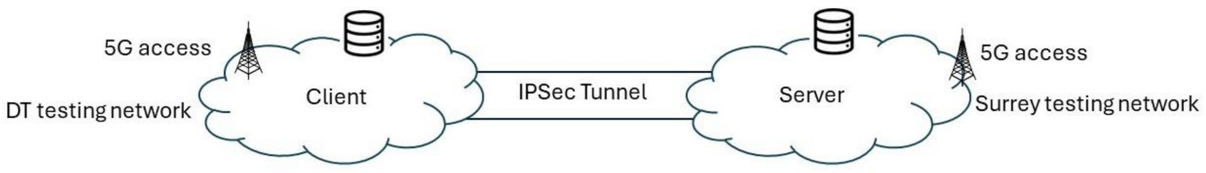


Figure 15: Interconnected Use Case Setup

In addition, the “Holographic Human-to-Human” use case is also available as a networked use case. Here, the receiver client (mobile phone and AR glasses) is located in the Surrey test

Funded by the Horizon Europe Framework Programme of the European Union

network, while the sender client (high-performance cluster) is located in the DT test network. As shown in Figure 15, the networked "Holographic Human-to-Human" use case enables 3D streaming of a user captured in real time on the DT side via the public Internet (using an IPSec tunnel) to a user on the Surrey side wearing AR glasses connected to the mobile phone.

Further collaboration between the SPIRIT project partners enabled the further validation of the interconnected use case. In this context, the Holographic Human-to-Human/Real-time Streaming of Realistic Avatars use cases were enabled using the interconnection. Here, the server was deployed at the Berlin testbed, and the client was deployed at the Bristol testbed. This allowed for the exchange of relevant data and the utilization of the use case in real-time. Figure 16 shows the additional interconnected use case setup.

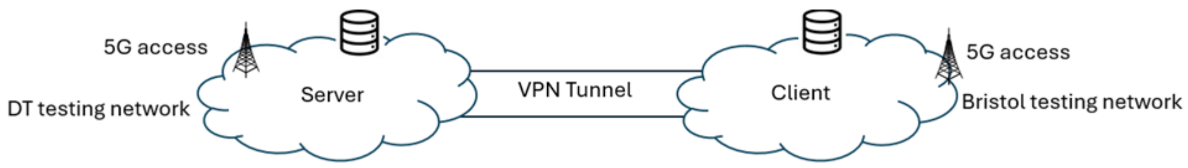
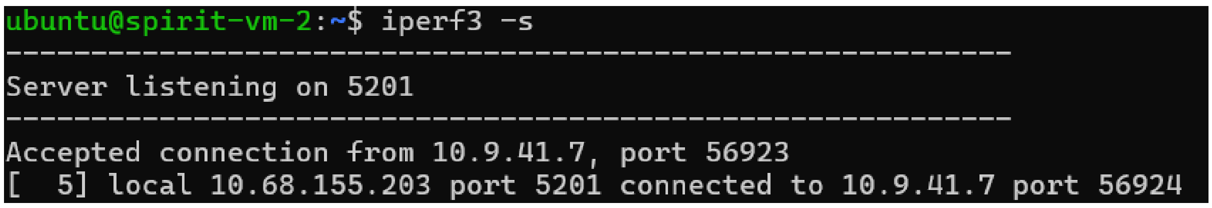


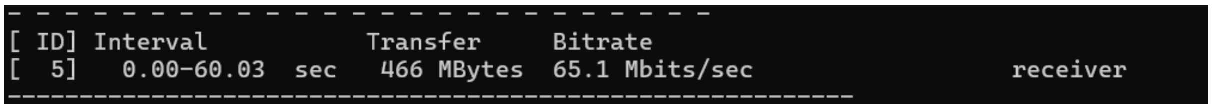
Figure 16: Bristol-Berlin interconnected use case

To further validate the interconnection between both testbeds, additional connectivity tests were conducted. In this context, a server instance hosted at Bristol and a Client instance at Berlin were used for the tests. Iperf was the network tool used at both testbeds, with basic throughput and jitter tests conducted. For more accurate evaluation, both UDP and TCP protocols were tested, providing testing variation and accounting for protocols utilized in the interconnected use cases.



```
ubuntu@spirit-vm-2:~$ iperf3 -s
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
Accepted connection from 10.9.41.7, port 56923
[  5] local 10.68.155.203 port 5201 connected to 10.9.41.7 port 56924
```

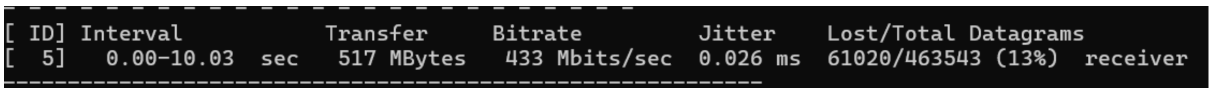
Figure 17: Server instance at the Bristol testbed



```
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate
[  5]   0.00-60.03  sec   466 MBytes  65.1 Mbits/sec                  receiver
-----------------------------------------------------------
```

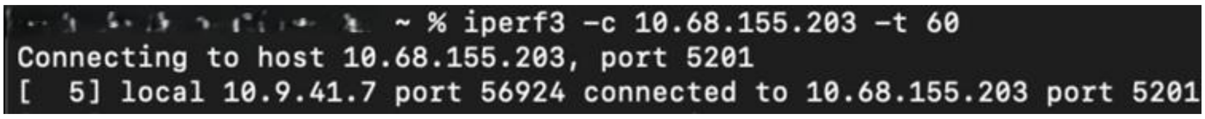
Figure 18: TCP Throughput test summary at the Bristol testbed

Funded by the Horizon Europe Framework Programme of the European Union



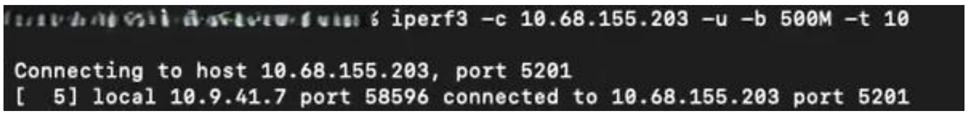
```
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bitrate         Jitter    Lost/Total Datagrams
[  5]   0.00-10.03  sec   517 MBytes   433 Mbits/sec  0.026 ms  61020/463543 (13%)  receiver
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Figure 19: UDP throughput and Jitter test summary at the Bristol testbed



```
[redacted] ~ % iperf3 -c 10.68.155.203 -t 60
Connecting to host 10.68.155.203, port 5201
[  5] local 10.9.41.7 port 56924 connected to 10.68.155.203 port 5201
```

Figure 20: Client instance at Berlin for the TCP throughput test



```
[redacted] iperf3 -c 10.68.155.203 -u -b 500M -t 10

Connecting to host 10.68.155.203, port 5201
[  5] local 10.9.41.7 port 58596 connected to 10.68.155.203 port 5201
```

Figure 21: Client instance at Berlin for the UDP throughput and Jitter test

From the figures above, it is evident that the interconnection is successful, as the connection between Berlin and Bristol testbed is established, allowing for the exchange of relevant data. Furthermore, the throughput and jitter results inform on the connectivity performance from the tests conducted, highlighting the suitability of the testbed interconnection. Moreover, when examining the UDP results which account for the interconnected use cases, it is evident that the interconnection performance is satisfactory as the jitter is quite low at 26ms and the bitrate is 433Mbits/sec for a 500M transfer. As such, it can be determined based on the results, that the interconnection between the Bristol and Berlin testbeds is suitable for facilitating the deployment and utilization of the interconnected use cases.

Funded by the Horizon Europe Framework Programme of the European Union

2.5 IMEC'S VIRTUAL WALL INFRASTRUCTURE

In case the Surrey or Berlin testbed may not provide sufficient resources for some computationally intensive and graphically complex many-to-many telepresence scenarios using volumetric video, imec's Virtual Wall (Ghent) was included in the project as an additional testbed to enable open call applications with high computing requirements in a controlled networking environment.

2.5.1 General Information on imec's Virtual Wall Testbed

The Virtual Wall is a large-scale testbed hosted in the iGent building on Ghent University's Ardoyen campus, for advanced networking, distributed software, cloud, big data and scalability research and testing.

The testbed contains 550+ bare metal and GPU servers which are fully configurable both in terms of their software installation (choice of operating systems, drivers, applications, etc.) as well as how their network interfaces are physically interconnected. The nodes can be assigned different functionalities ranging from terminal, server, network node, and impairment node. The nodes can be connected to test boxes for wireless terminals, generic test equipment, simulation nodes (for combined emulation and simulation) etc.

The testbed was integrated in a worldwide federation through Fed4FIRE, a European initiative that provides a large-scale federated testbed infrastructure for experimentation in the field of Future Internet research[1].

2.5.2 Testbed Infrastructure

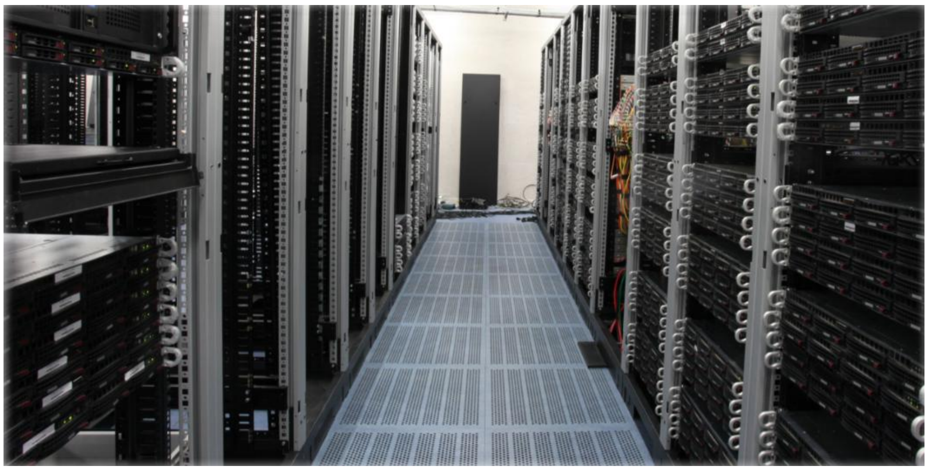


Figure 22: Server rack in imec's Virtual Wall infrastructure.

Full documentation of the Virtual Wall is available online[2], including all technical details of the available nodes (e.g., in terms of memory and RAM). To access the testbed, any researcher

[1] https://portal.fed4fire.eu/

[2] https://doc.ilabt.imec.be/ilabt/virtualwall/

Funded by the Horizon Europe Framework Programme of the European Union

can simply log in through their research institution[3]. They can then use the jFed tool[4] to create and manage their experiments. Among others, this tool allows to:

- Select specific hardware nodes or virtual machines
- Select the preferred operating system
- Request public IPv4 addresses
- Enable network emulation (e.g., fixed bandwidth capacity between two nodes)
- Connect through SSH.

Rspec files are used to request, manifest and advertise experiments. These files are structured XML files that contain, among others, a list of nodes (e.g., bare-metal servers or virtual machines), the type of image on each of these nodes, etc.

Once the experiment has been set up, experimentation can begin. Nodes are interconnected through 1 Gb/s links, meaning that relatively high bandwidth use cases can be considered. Tasks can be run from the command line but can also be scheduled as needed. The jFed GUI can be used to follow up on the experiment's status, extend the experiment's reservation, etc.

2.5.3 Onsite Use Cases

Because of the number of available hardware nodes and high-capacity (and configurable) network connections, the Virtual Wall is particularly suitable for experiments that need scale and network adaptivity.

In the SPIRIT project, the testbed is used for many-to-many scenarios in teleconferencing, linked to the use case:

➲ Holographic Human-to-Human Communications

[3] https://portal.fed4fire.eu/

[4] https://jfed.ilabt.imec.be/

Funded by the Horizon Europe Framework Programme of the European Union

3 PLATFORM COMPONENTS

The section lists partner components that were integrated into the testbeds during the project and documents the findings of this integration process.

3.1 LIVE MULTI-SOURCE HOLOGRAPHIC STREAMING

This section is about the components supporting applications using live teleporting people from remote internet locations to a common virtual space of the audience such that the audience can have the immersive and multisensory perception that everyone is located in the common physical scene.

3.1.1 Deployment of the Frame Render and Synchronisation Function

There are two key components for enabling a multi-source live volumetric streaming application: a frame synchronization function and a RESTful interface to accept runtime parameter settings and queries (as shown in the orange block in Figure 23). The frame synchronization function is responsible for rendering holographic frames from multiple independent sources. For frames from different sources, the MEC server can distinguish them based on their source IP addresses and record their timestamps separately. Based on a predefined synchronization threshold (e.g., 30ms) and the real-time timing of each source, the MEC server can perform operations such as pairing, buffering, and discarding of multi-source frames to ensure that the synthesized frame exhibits optimal performance. Meanwhile, this frame synchronization function can also be configured through RESTful interfaces. The frame synchronization threshold and required key performance metrics (e.g., throughput, playback latency) can be set using the POST method and queried using the GET method through an authorized network management interface. Figure 23 displays the application of the frame synchronization function on Surrey's testbed. The rendering function can directly call the NVIDIA Compute Unified Device Architecture (CUDA) library to utilize the graphic card resources of the platform, enabling efficient computation and display of multi-source frames on this platform.

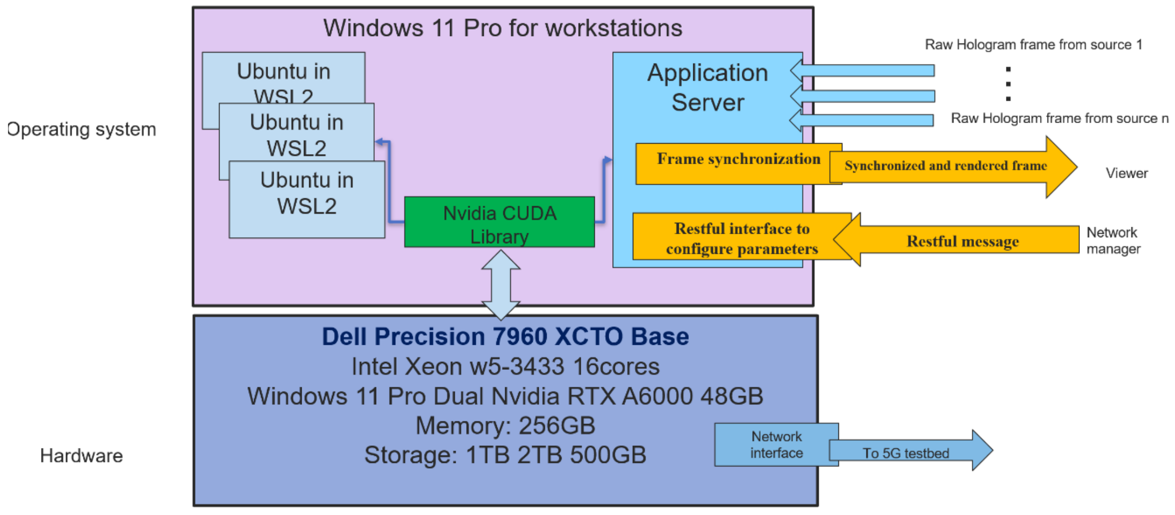

Windows 11 Pro for workstations
Operating system
Ubuntu in WSL2
Ubuntu in WSL2
Ubuntu in WSL2
Nvidia CUDA Library
Application Server
Raw Hologram frame from source 1
Raw Hologram frame from source n
Frame synchronization
Synchronized and rendered frame
Viewer
Restful interface to configure parameters
Restful message
Network manager
Hardware
Dell Precision 7960 XCTO Base
Intel Xeon w5-3433 16cores
Windows 11 Pro Dual Nvidia RTX A6000 48GB
Memory: 256GB
Storage: 1TB 2TB 500GB
Network interface
To 5G testbed

Figure 23: Deployment of frame production and synchronization function on Surrey's platform

Funded by the Horizon Europe Framework Programme of the European Union

3.1.2 Validation

Figure 24 shows a captured frame from a demonstration of the Surrey live multi-source holographic platform with the frame render and synchronization function. There are two sources located at location A and B, with dedicated cameras to capture different people in a live streaming manner. The virtual space in the middle is the real-time screen projected from the edge server. It shows the merged and synchronized frame, which can validate the efficacy of the remote production and frame synchronization function.

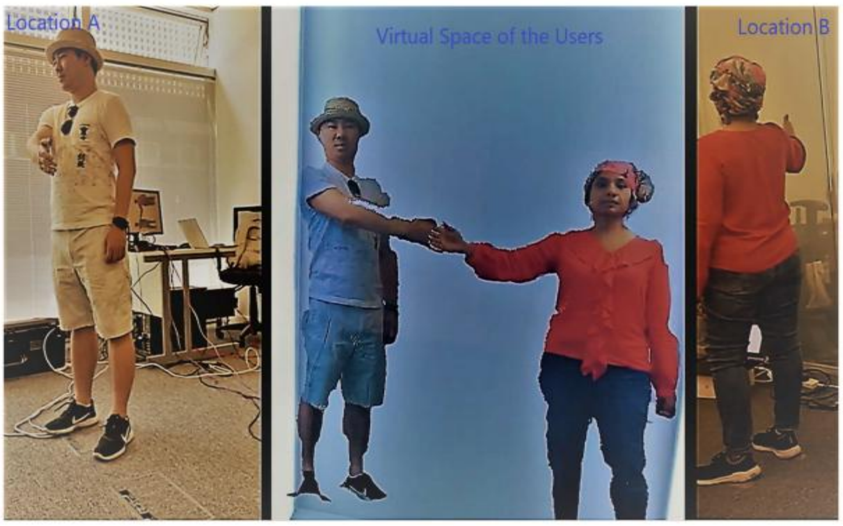


Figure 24: Validation of the frame production and synchronization function on Surrey's platform

The frame synchronization mechanism features certain key parameters which are critical to its implementation. These parameters are the synchronisation window (Δ) and the Frame paring approximation threshold (γ) [5]. Additional details relating to these parameters and other key terminology are provided below. Detailed text on the mechanism can be found in D3.3.

Synchronisation window (Δ): The tolerable window relevant to multiple sources that allows for the arrival of late or delayed frames.

Frame paring approximation threshold (γ): The threshold that provides an additional level of approximation to pair the frames which have sufficiently small-time offset between their timestamps together.

Fresh frames: Frames with the same timestamp that arrive at the server within the synchronisation window (Δ) from all connected sources.

Half fresh frames: If a frame from one of the sources does not arrive within the synchronisation window, instead of discarding the frame, the system pairs a previously cached frame with the waiting frame.

Dropped frames - Frames that arrive at the server which are neither fresh nor half-fresh frames are dropped and are not used. [5]

Figure 25 depicts the frame distribution with the Frame paring approximation threshold (γ) at a stringent value of 10ms and a varying synchronisation window from 5ms to 50ms. From the

Funded by the Horizon Europe Framework Programme of the European Union

figure, it is clear that in the consideration of the synchronisation window at 5 ms and 10 ms, the distribution of half fresh frames is quite similar to the fresh frames. This is due to the rather stringent nature of synchronisation window in such scenarios, guaranteeing the limited waiting period for frames from other sources. After the expiration of the synchronisation window, the waiting frame searches and pairs with a qualified cached frame based on the approximation threshold. However, when considering a more lenient window of 50 ms, the proportion of fresh frames is only around 60%. This pattern is associated with the reduced FPS at the receiver side, as one of the sources is 50ms away. As such, it is evident that stringent *γ* coupled with the reduced FPS affects the performance of the application.

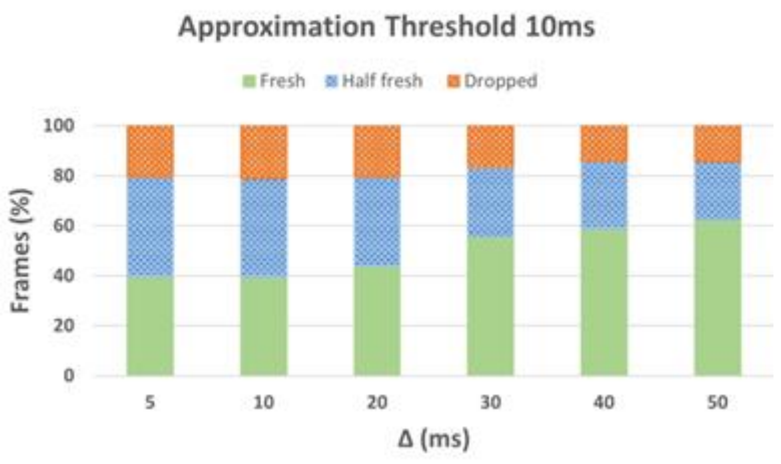


Figure 25: Frame Distribution at γ =10ms

Figure 26 depicts the frame distribution with the Frame paring approximation threshold (γ) at a rather lenient value of 50ms and a synchronisation window which varies from 5ms to 50ms. When compared to the strict case of γ =10ms, it is evident that the distribution of fresh frames increases significantly. This pattern is attributed to the reduced sensitivity to the synchronisation window, due to the relaxation of γ. This is further reflected with slight fresh frames increases for higher values of Δ like 40 ms and 50 ms.

Additional details on the mechanisms as well as further results can be found in [5].

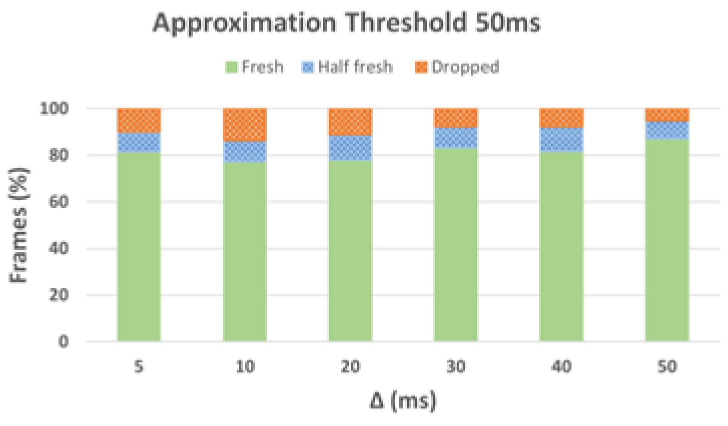


Figure 26: Frame Distribution at γ =50ms

Funded by the Horizon Europe Framework Programme of the European Union

Platform Deployment

Following explorative efforts, the multisource platform was deployed and integrated on the Surrey and Berlin testbeds. Building on this, the validation of the server can be detailed below.

Figure 27 depicts a cut section of the server panel showing the server in an inactive state. From the figure, it is evident that the server has not been started and as such, is not in an active state.

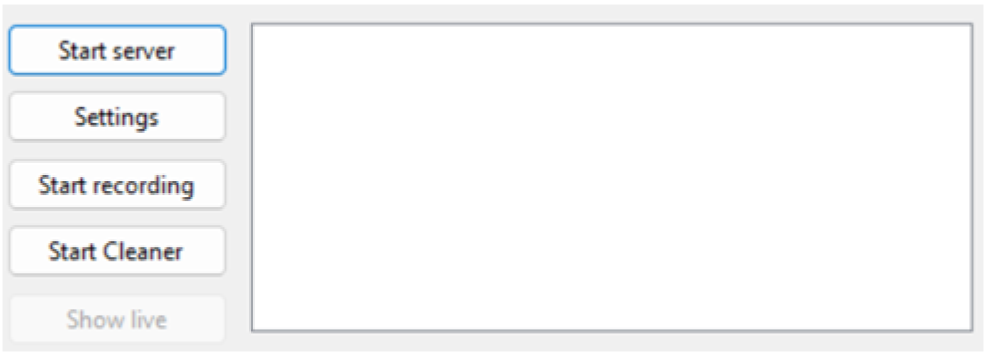


Figure 27: Section of the Server Panel during an inactive session

By clicking on the start sever button, the server is initialised and moves from an inactive state to an active state, enabling it to receive and transmit relevant data, facilitating live multi-source holographic communication. This is shown in Figure 28.

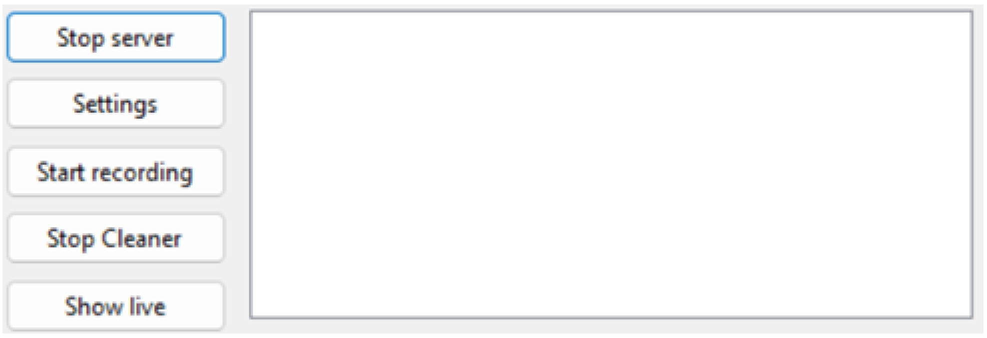
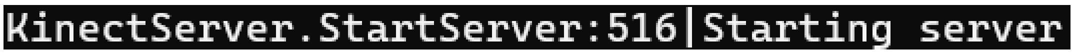


Figure 28: Section of the Server Panel during an active session

From the figure, it is evident that the server has been started and is in an active state. However, whilst the server is active, no connections have been made to it as such, a complete live multisource session has not begun.



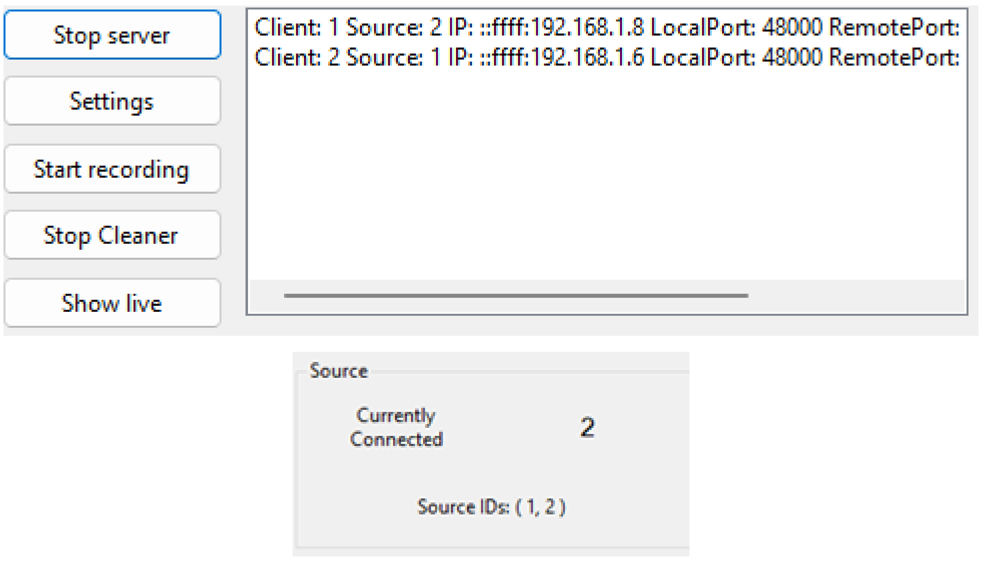
```
KinectServer.StartServer:516|Starting server
```

Figure 29: Server active state

Figure 29 further indicates that the server is now in an active state. This notice further confirms that the server has been successfully deployed within the testbed.

Figure 30 depicts the server during an active multi-source session. Here, two unique sources (source 1 and 2) are connected to the server simultaneously. The server maintains these connections by continuously making relevant frame requests to the unique producers/sources, whilst delivering the aggregated content to the relevant receiver.

Funded by the Horizon Europe Framework Programme of the European Union



```
Client: 1 Source: 2 IP: ::ffff:192.168.1.8 LocalPort: 48000 RemotePort:
Client: 2 Source: 1 IP: ::ffff:192.168.1.6 LocalPort: 48000 RemotePort:
```

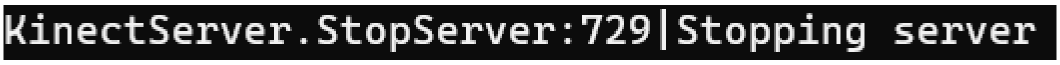
Source

Currently Connected 2

Source IDs: (1, 2)

Figure 30: Section of the Server Panel during a complete active multi-source session

After the conclusion of the live session, clicking the stop server button terminates the session, closing all connections and rendering the server inactive. At this stage, the server no longer requests or transmits any relevant data and cannot be connected to. Figure 31 provides further confirmation of the inactive server state.



```
KinectServer.StopServer:729|Stopping server
```

Figure 31: Server inactive state

Many-to-many support

The multisource use case features the generation and delivery of volumetric media from different/multiple sources of origin to a single receiver, enabling the viewer/consumer to view and interact with both volumetric sources in real time.

Augmentation efforts in relation to the initial configuration features the bolstering of the configuration to support multiple receivers/consumers. In this context, an additional receiver/consumer can connect to the server instance and receive volumetric streams from the sources within the same session. Moreover, the utilization of source identifiers also enables the possibility of receiving streams from different sources as opposed to a unified scenario (i.e. a client can receive streams from one source whilst the other client receives streams from a different source). Here both combined or separate streams can be accessed by viewers connected to the same server instance, allowing for flexible multi viewer support and utilization.

This multi-receiver based many-to-many augmentation facilitates the possibility of multiuser driven interactions, allowing for expanded flexibility, further promoting virtual shared experiences and paving the way for additional group driven interactions, enhancements and scenarios. Figure 32 shows the updated configuration of the extension.

Funded by the Horizon Europe Framework Programme of the European Union

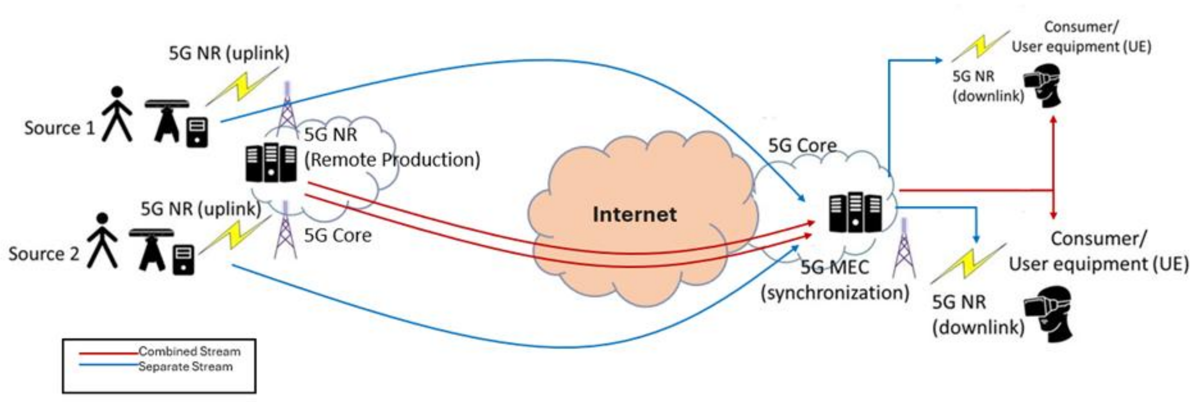


Figure 32: Many-to-many augmentation support architecture

The following points summarize the multi-client many-to-many support:

- Real time addition of clients: New receivers/consumers can connect to the server, facilitating the delivery of live volumetric media.
- Independent viewer experience: Users can freely interact with the media without restrictions on orientation and movement.
- Distinct source viewing: Users can view different sources individually or unified within the stream for more variability. This is managed and configured at the server.

Multidimensional adaptation

Figure 33 shows framework of the proposed Multidimensional adaptation framework. The framework accounts for source object behaviour, viewer behaviour and the network environment, leveraging machine learning to make intelligent adaptation decisions in order to satisfy user intent. For example, when considering a live holographic streaming scenario, where both the user and the source object can freely undertake varying behaviours, the uncertainty associated with such behaviours present a challenge particularly in relation to the satisfaction of user intent. To address this, the Multidimensional adaptation framework, accounts for the various behaviours and makes relevant adaptations to satisfy the user intent. More details on the proposed framework can be found in D3.3 [3].

Funded by the Horizon Europe Framework Programme of the European Union

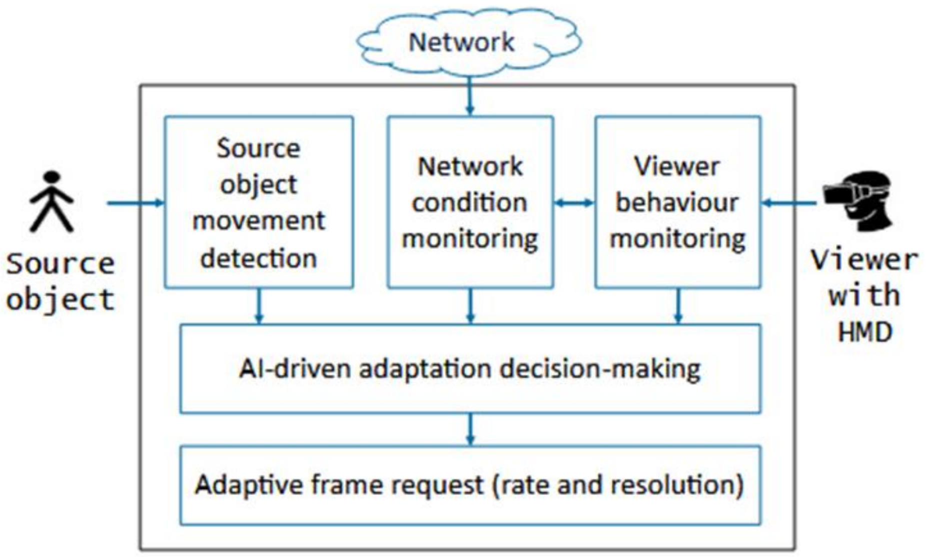

Network
Source object
Source object movement detection
Network condition monitoring
Viewer behaviour monitoring
Viewer with HMD
AI-driven adaptation decision-making
Adaptive frame request (rate and resolution)

Figure 33: Multidimensional Adaptation Framework

To explore the suitability of the proposed solution, further investigation has been conducted in relation to the performance of the solution to explore and detail its capabilities. In this context, additional experiments were conducted, with controlled and varying network parameters introduced in the environment.

To further inform on the performance, comparisons against an anchor implementation have been carried out. Here, the anchor implementation refers to basic adaptation implementation without the proposed solution. This base version of the adaptation only considers the viewer behaviour when attempting to satisfy the user intent.

Figure 34 illustrates the solution performance with dynamic source object behaviour present, under the ideal conditions of 0 ms delay and 0% packet loss in the streaming session. From the figure (the first row of plots), the results show that the proposed solution, leveraging its intelligent adaptation strategies can effectively satisfy user intent, accounting for varying distinct behaviours in the streaming session. The FPS plot further informs on this, as it indicates that upon the determination of the dynamic object behaviour, the maximum FPS compatible with the corresponding resolution is selected. Moreover, the resolution plot further highlights the suitability and flexibility of the proposed solution in its attempt to satisfy user intent, displaying its ability to freely choose between the WQHD resolution, which provides the highest quality but not the best FPS performances, and the FHD resolution, which provides higher quality than the HD level but lower quality than the WQHD level with better FPS performances in an effort to satisfy user intent. The throughput plots further validate these findings, depicting corresponding increases in throughput, aligned with the decisions made by the proposed solution, further confirming that the solution can effectively satisfy user intent under such ideal conditions.

Comparing the results of the proposed solution against the anchor implementation (the second row of plots), where the only behaviour factored is the viewer movement, it is clear that under such ideal circumstances, the anchor implementation can also satisfy the user intent to a certain extent. In this regard, the anchor implementation resolution plot shows that corresponding changes to the resolution can be made based on the viewer forward and backward movement. However, the capability of the anchor implementation even under such stable circumstances

Funded by the Horizon Europe Framework Programme of the European Union

is still somewhat limited, as it lacks the flexibility to provide suitable adaptations to provide the most optimal performances. Furthermore, the base or anchor implementation lacks the intelligent capabilities of the proposed solution, limiting its ability to effectively account for the source object behaviour, resulting in limitations in its adaptation approach.

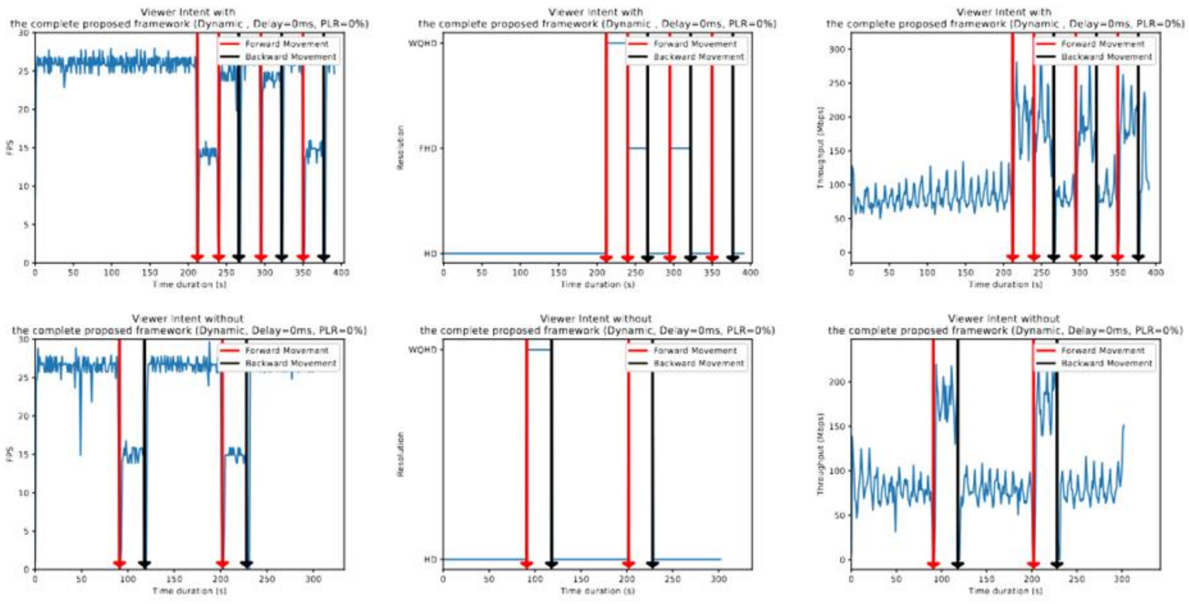


Figure 34: Solution performance with dynamic source object behaviour under 0 ms delay and 0% packet loss

Figure 35 depicts the solution performance with dynamic source object behaviour, 50 ms delay and 0.1% packet loss as the prevalent conditions in the streaming session. Under such erroneous conditions, the results indicate that the proposed solution can still provide satisfactory performances. The FPS plots inform on this as they show that whilst occasional FPS drops occur due to the network conditions, the proposed solution achieves a fairly sustained performance. This is further inferred from the plots showing the relevant adaptations made to account for the less favourable conditions. In this context, the FPS and resolution plots show that the proposed solution by leveraging its network awareness can provide the most relevant recommendations for suitable FPS and resolution values, ensuring that the user intent can still be satisfied effectively. For example, the results show that FPS can ben reduced to 5 whilst increasing the resolution to the FHD level. Similarly, the solution can decrease the resolution to the HD level and simultaneously increase or decrease the FPS to target somewhat stable performances. This further highlights the flexibility of the solution in its approach to satisfy user intent. The throughput results provide additional context on this as they indicate that the framework can achieve somewhat stable performances while satisfying user intentions.

Comparing the results against the performances of the base or anchor implementation which is a framework that only factors the user behaviour in adaptation decision making, it is clear that the anchor implementation can not effectively satisfy the user intent under such adverse circumstances. This is inferred from the anchor implementation FPS results, which shows unreliable and undesired FPS performances, with frequent FPS drops and unstable performances ranging from 0-30 FPS occurring in the streaming session. Moreover, the FPS plot contains grey regions which are of major concern, as such areas indicate the incompatibility of the user intent with the anchor implementation adaptation outcome. This determination is due to the inability of the network to support the anchor implementation-based adaptations and

Funded by the Horizon Europe Framework Programme of the European Union

is depicted by frequent zero or near zero FPS levels. The FPS results of the anchor implementation, in comparison with the results of the proposed solution indicate that the anchor implementation cannot provide suitable performances when attempting to satisfy user intent, and as such significantly underperforms. Additionally, the reduced and near zero or zero FPS levels of the anchor implementation raises further concerns as such FPS performances could potentially negatively impact the user experience under such less favourable circumstances. The resolution plot of the base or anchor implementation indicates that during forward viewer movement, the base or anchor implementation can provide an increase in quality. However, the FPS and throughput plots provide additional context, with the throughput plot further confirming the deteriorative performances, highlighting irregular throughput observed and unstable levels when the highest quality level is adapted.

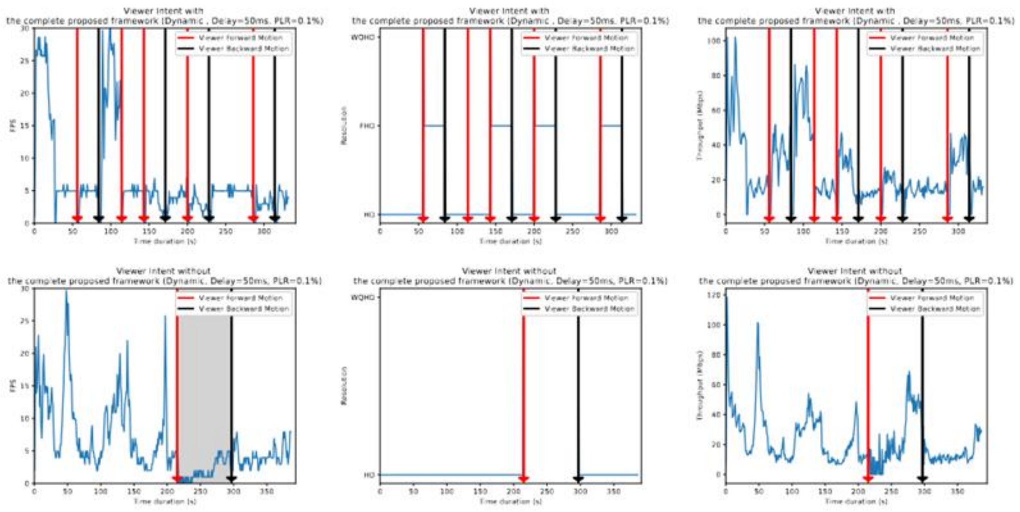


Figure 35: Solution performance with dynamic source object behaviour under 50 ms delay and 0.1% packet loss

3.2 NETWORK-AWARE RESOURCE SCHEDULER (DIKTYO)

To avoid monolithic architectures, telepresence solutions typically consist of multiple individual microservices or components, in the form of containers, that together represent a service function chain. In order to ensure that the deployments of such chains still meet the latency and throughput requirements of the application, resource management and orchestration platforms should be made aware of the network characteristics as well as of computational resource usage. Vanilla Kubernetes (K8s) deployments do not take networking characteristics into account and primarily focus on CPU and RAM optimization.

The Diktyo network-aware scheduler has been developed to alleviate this problem, as described in more detail in deliverable D3.3. It has been proven to improve network throughput considerably for several benchmarking applications.

Diktyo supports easy deployment in a Kubernetes (K8s) cluster as an additional scheduler in the system without needing to remove the default scheduler. A helm-chart has been developed to automatically deploy all Diktyo components, available here:

➲ https://github.com/diktyo-io/helm-chart/tree/main.

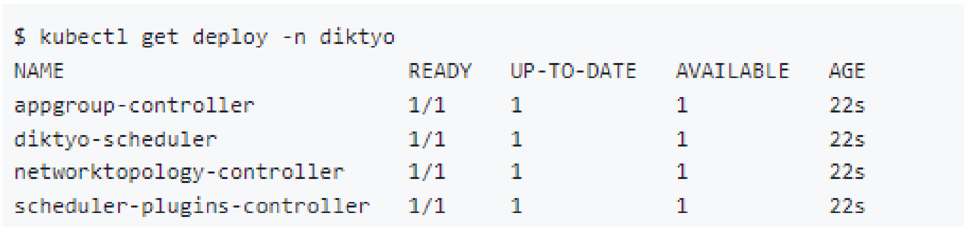
Funded by the Horizon Europe Framework Programme of the European Union

Documentation with required steps and verification that all pods are running properly as seen in Figure 36 is provided. Different versions of Diktyo exist since K8s v1.24 in the K8s scheduling community available as different releases:

- https://github.com/kubernetes-sigs/scheduler-plugins/releases.

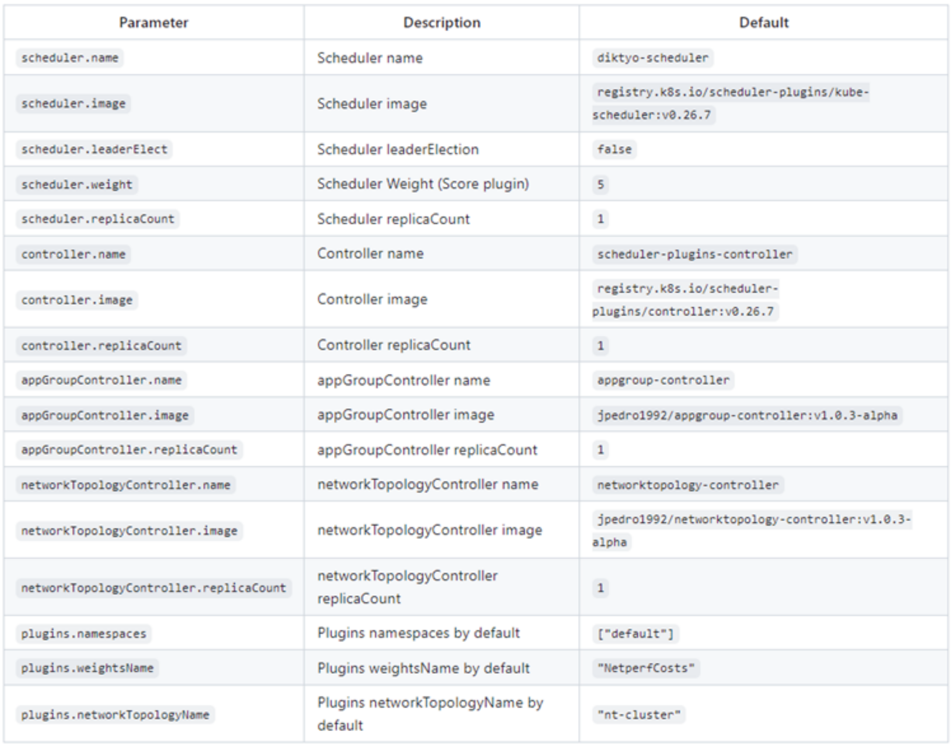
The Diktyo scheduler can be configured with different parameters as shown in Figure 37. Additional information on how to deploy additional schedulers in K8s clusters can be found in the K8s scheduling community:

- https://github.com/kubernetes-sigs/scheduler-plugins/blob/master/doc/install.md



```
$ kubectl get deploy -n diktyo
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
appgroup-controller            1/1     1            1           22s
diktyo-scheduler               1/1     1            1           22s
networktopology-controller     1/1     1            1           22s
scheduler-plugins-controller   1/1     1            1           22s
```

Figure 36: Diktyo scheduler running properly



Parameter	Description	Default
`scheduler.name`	Scheduler name	`diktyo-scheduler`
`scheduler.image`	Scheduler image	`registry.k8s.io/scheduler-plugins/kube-scheduler:v0.26.7`
`scheduler.leaderElect`	Scheduler leaderElection	`false`
`scheduler.weight`	Scheduler Weight (Score plugin)	`5`
`scheduler.replicaCount`	Scheduler replicaCount	`1`
`controller.name`	Controller name	`scheduler-plugins-controller`
`controller.image`	Controller image	`registry.k8s.io/scheduler-plugins/controller:v0.26.7`
`controller.replicaCount`	Controller replicaCount	`1`
`appGroupController.name`	appGroupController name	`appgroup-controller`
`appGroupController.image`	appGroupController image	`jpedro1992/appgroup-controller:v1.0.3-alpha`
`appGroupController.replicaCount`	appGroupController replicaCount	`1`
`networkTopologyController.name`	networkTopologyController name	`networktopology-controller`
`networkTopologyController.image`	networkTopologyController image	`jpedro1992/networktopology-controller:v1.0.3-alpha`
`networkTopologyController.replicaCount`	networkTopologyController replicaCount	`1`
`plugins.namespaces`	Plugins namespaces by default	`["default"]`
`plugins.weightsName`	Plugins weightsName by default	`"NetperfCosts"`
`plugins.networkTopologyName`	Plugins networkTopologyName by default	`"nt-cluster"`

Figure 37: Diktyo scheduler configuration parameters

Funded by the Horizon Europe Framework Programme of the European Union

3.2.1 Integration with CloudNativeLab

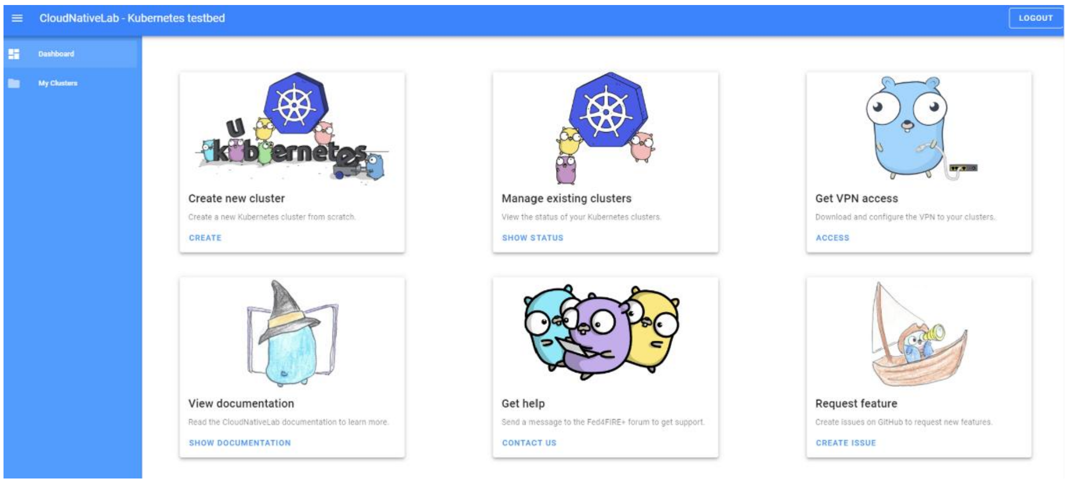
During the SPIRIT project, Diktyo has been successfully integrated into the CloudNativeLab at IMEC-IDLab:

➲ https://practicum.cloudnativelab.ilabt.imec.be/.

CloudNativeLab is a testbed for experimenting with K8s and cloud native technologies. With only a few clicks, experimenters can spin up their own K8s cluster in the IMEC-IDLab data center (Figure 38 and Figure 39).

Diktyo has been integrated into K8s clusters created with CloudNativeLab and can be used to deploy applications in the system by changing the *schedulerName* field in deployment files (Figure 40) as shown in the K8s documentation:

➲ https://kubernetes.io/docs/tasks/extend-kubernetes/configure-multiple-schedulers/.



Create a new Kubernetes cluster from scratch.

View the status of your Kubernetes clusters.

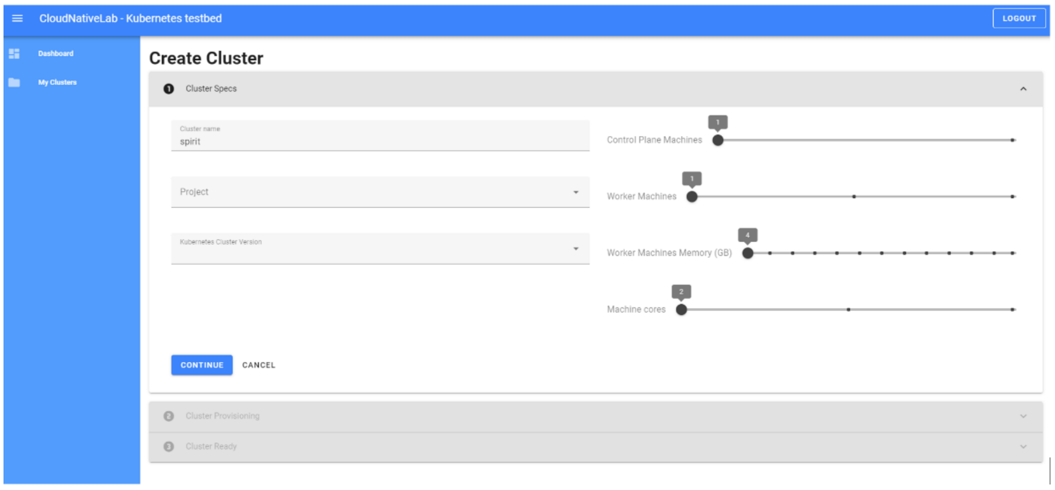
Download and configure the VPN to your clusters.

Read the CloudNativeLab documentation to learn more.

Send a message to the Fed4FIRE+ forum to get support.

Create issues on GitHub to request new features.

Figure 38: CloudNativelab main dashboard



Cluster name: spirit

Control Plane Machines: 1

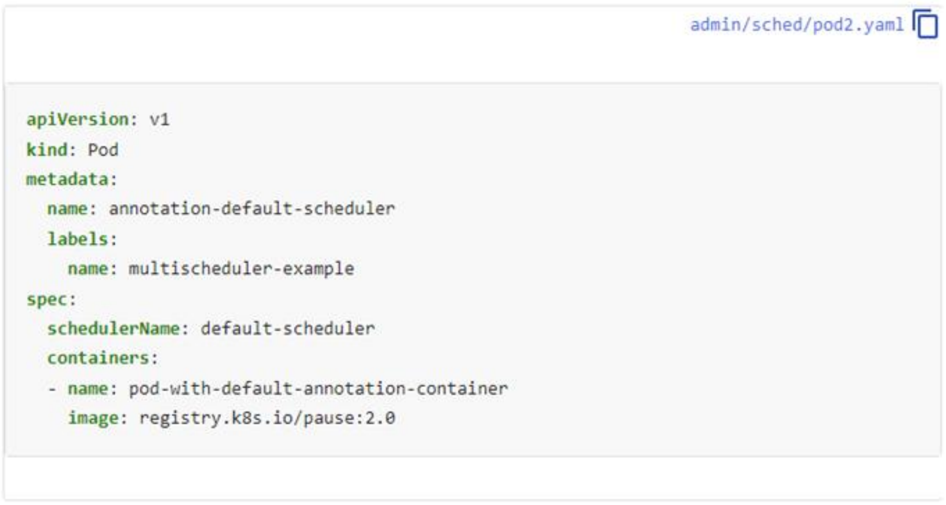
Worker Machines: 1

Worker Machines Memory (GB): 4

Machine cores: 2

Figure 39: CloudNativelab K8s cluster creation page

Funded by the Horizon Europe Framework Programme of the European Union


admin/sched/pod2.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: annotation-default-scheduler
  labels:
    name: multischeduler-example
spec:
  schedulerName: default-scheduler
  containers:
  - name: pod-with-default-annotation-container
    image: registry.k8s.io/pause:2.0
```

Figure 40: example that specifies A scheduler for pod deployment

3.2.2 Integration with Berlin Testbed

During the SPIRIT project, integration efforts have taken place to test and validate Diktyo in the T-Systems testbed. The integration has been challenging since the T-Systems testbed currently supports K8s v1.21 and the most recent version of Diktyo is available for K8s v1.26. The mismatch of K8s API versions has hindered the complete integration of the most recent version of Diktyo in the T-Systems testbed.

Nonetheless, an older version of Diktyo (v1.22) has been currently deployed and successfully integrated into the T-Systems testbed, being able to deploy microservices in the system and providing the required functionality to support experimenters with network-awareness when scheduling application containers.

3.2.3 Integration with Surrey Testbed

The recent version of Diktyo has been successfully integrated in Surrey based on the described helm chart. Example applications such as Online Boutique (link below) have been deployed in Surrey to confirm that Diktyo is able to deploy pods in the system as an additional Kubernetes scheduler.

➲ https://github.com/GoogleCloudPlatform/microservices-demo

Funded by the Horizon Europe Framework Programme of the European Union

3.3 HOLOGRAPHIC HUMAN-TO-HUMAN COMMUNICATION

The third version of the SPIRIT platform aims to support human-to-human communication use cases for immersive telepresence applications, as described in D2.3 [2]. In addition, within the SPIRIT project the goal is to incorporate innovative platform enablers presented in D3.3 [3] to enhance the immersive telepresence experience.

Prior to exploring the technical considerations of the holographic human-to-human communication use case it is crucial to emphasize the profound significance of capturing authentic experiences.

For example, in professional settings, holographic communication transforms collaborative endeavours by offering lifelike depictions of remote participants. The immersive nature of 3D representation fosters a collaborative environment where individuals can share ideas, engage in discussions, and collaborate on projects as if they were physically co-located. The ability to visualise the spatial arrangement of team members enhances the collaborative process, possibly contributing to more innovative and human-centric outcomes.

Within the SPIRIT project, the components of the application platform supporting holographic human-to-human communications by Ericsson are implemented into the network testbeds provided by DT, UoS and UoB. The procedure mainly involves the integration of the components primarily responsible for data processing into the network testbeds.

This process entails importing the producer application into a high-performance cluster (HPC) detailed in Section 2, while maintaining the consumer application on a mobile end device.

Leveraging the computational capabilities of an HPC available at the testbeds facilitates expedited data processing by the producer application software, reducing the end-to-end latency. This approach is particularly significant for digital human-to-human communication use cases with real-time requirements.

The required hardware components of the application platform will be provided to third parties, this includes:

- Depth camera
- HPC
- WebRTC Component(s)
- 5G Android mobile phone
- AR glasses

Other components that may be required to extend the application platform to support additional use cases must be provided by third parties.

The following sections describe the platform components provided by Ericsson and highlights some aspects of the integration procedure that are needed to implement the holographic human-to-human communications use case for the third version of the SPIRIT platform.

Funded by the Horizon Europe Framework Programme of the European Union

3.3.1 Common Prerequisites for Integration

Initially, it is essential to establish a common set of requirements for both the application platform and the network testbed for the integration procedure. The network testbed is designed to facilitate the deployment and execution of applications within a container-based environment. This environment enables seamless connectivity between diverse containers and provides access to tangible resources like GPU cards and computational power.

The producer application is available in two formats, namely as container and as stand-alone executable. The choice of offering the software as container is motivated by containers ensuring consistent, portable, and efficient software deployment. They encapsulate applications and dependencies, facilitating easy scaling, versioning, and streamlined management across diverse environments. The orchestration of these containers is managed by a Kubernetes Cluster. A 5G network environment is in place to establish connections between SPIRIT platform devices, such as mobile phones, within a controlled network scenario.

3.3.2 Compatibility Test of Client Devices

The consumer application is provided as an Android application. To integrate the consumer application, the software is downloaded and installed on a mobile end device equipped with a SIM card. The SIM card is provided by DT, UoS, or UoB and allows to connect the mobile end device to a controlled 5G test network within their respective testbeds. It is noteworthy that the mobile end device must meet certain compatibility criteria. This requires knowledge about the public land mobile network (PLMN) identification number of the testbed and compatibility with augmented reality (AR) glasses, more specifically the XReal Light glasses used to develop and test the application. The latter criterion necessarily requires an Android operating system.

A range of Android test devices was available for SPIRIT to assess connectivity with the 5G environment within the testbed and compatibility with the holographic communication consumer application that makes use of the AR glasses. In our compatibility tests for suitable client devices, the mobile end devices Oppo Find X2 Pro and Samsung S20 were identified as suitable consumer application devices as they satisfy the criteria for the AR glasses and network testbeds. It is important to note, that the use of other phones is also possible. However, mobile phones provided by third parties need to meet the test criteria. Therefore, we suggest selecting mobile end devices in consultation with Ericsson and DT/UoS/UoB to provide seamless integration of mobile end devices that have not been tested with the platform.

3.3.3 Integration Efforts

The holographic human-to-human communications use case requires a depth camera, which is the source to capture the media and spatial information of an object of interest. In this case the object of interest is a human face or torso. The data obtained from the depth camera is used by the producer application to compute 3D representations in the form of holograms, that are streamed to the consumer application once a WebRTC connection has been set up exchanging meta data in signalling, as described in D3.3 [3].

It is crucial to highlight that the depth camera only provides a USB 3.0 port and does not offer a network interface. Hence, a solution that bridges the connection between the depth camera and the Edge Cloud system is needed. Two solutions have been considered:

Funded by the Horizon Europe Framework Programme of the European Union

1. Solution using a dedicated device server, as seen in Figure 41:

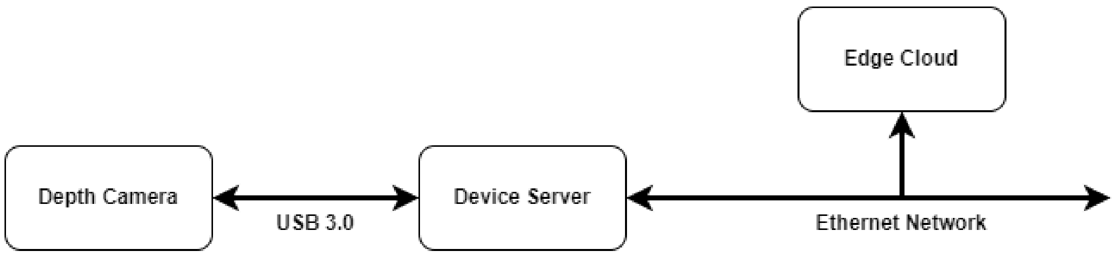


Figure 41: Connectivity Between Depth Camera and Edge Cloud with a Device server [5]

The device server provides two USB 3.0 interfaces that are compatible with the depth camera hardware. In addition, the device server offers a network interface based on the IEEE 802.3 (1000BaseT, 100BaseTX, and 10BaseT) standard, which can be used to connect the Edge cloud to the device server. The hardware and the pre-installed software of the device server enable a data connection between the depth camera and Edge Cloud system without additional implementation efforts on the application or testbed side.

In addition, the device server comes with additional features such as freely selectable power supply between 12V and 48V, integrated change-over relay with automatic or event-controlled switching in the event of defined status changes (e.g. power or net-work failure) or manual switching via web browser, and numerous notification options in the event of faults (via SNMP (Simple Network Management Protocol) traps, e-mail and relay control). However, one disadvantage of this solution is the additional cost of procuring this hardware.

2. Solution using a Raspberry Pi:

The setup for the Raspberry Pi solution is similar to the set up proposed in Figure 41. The difference is that the dedicated device server is exchanged by a Raspberry Pi computer. Hence, this solution approach can be seen as more cost-effective solution with lower costs but higher integration complexity.

The idea is that the Raspberry Pi serves as a USB/Ethernet data converter that bridges a connection between the depth camera via the USB 3.0 interface and the Edge Cloud system via the Ethernet interface. This approach follows a client-server architecture, where the Raspberry Pi is the server and the HPC is the client. However, this approach requires the implementation of a sophisticated communication mechanism in order to minimise the additional latency caused by the Raspberry Pi, given its relatively weak computing power.

To initialize a connection between the client and server, both need to detect each other. To do so, the server listens for a “start” signal. To start the acquisition of data using the depth camera, the client sends a “start” signal to the IP address of the server. Upon receiving the “start” signal, a TCP connection between client and server is established, RGB and Depth frames are captured and synchronized using the implemented capture

[5] INU-100

Funded by the Horizon Europe Framework Programme of the European Union

pipeline on the server. Subsequently, the frames are resized and sent in smaller chunks to the client, in order to comply with TCP transmission requirements.

3.3.3.1 Device Server

For the deployment at DT the device server solution was determined to be the more robust solution for the use case, as presented in Figure 41. For the deployment at UoS and UoB no device server is needed since the testbeds support USB 3.0 interfaces making it compatible with the depth camera in use.

Internally, the device server creates a virtual USB interface within the cloud by mapping the ethernet interface to a virtual USB interface through the USB driver of the operating system. The process is managed by the USB-to-Network (UTN)-manager firmware. Figure 42 depicts the described behaviour.

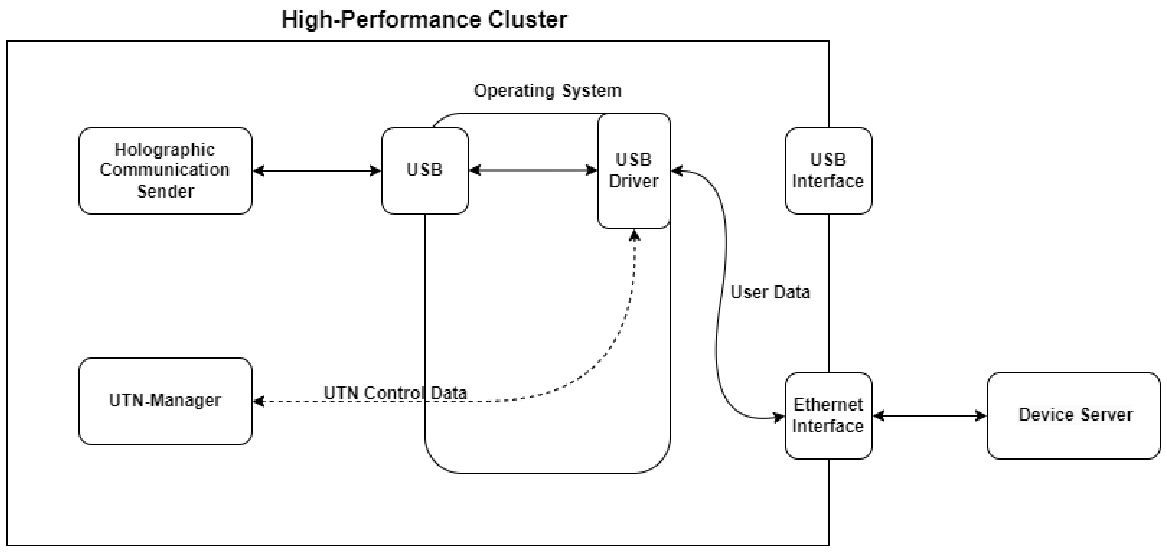


Figure 42: Device Server Firmware

The device server comes with several firmware files that need to be installed on the host system.

For the deployment at DT the installation of the UTN manager on a Linux OS without a graphical interface is conducted, i.e., the installation for the minimal UTN manager version which can be found here[6].

The firmware consists of three components which must be installed in the right order to comply with their dependencies.

1. driver
2. service
3. clitool

Before installing the tools, the requirements given in Table 1 must be met.

[6] https://www.seh-technology.com/services/downloads/industrial/inu-100.html

Funded by the Horizon Europe Framework Programme of the European Union

Table 1: UTN Manager Requirements

Requirement	Version (recommended)
Linux kernel	2.6.32 >
glibc	2.15 >
OpenSSL	1.0.1 >
Dynamic Kernel Module Support (DKMS)	/

In addition, the logged-in user must have root access, i.e., can use the command *sudo.*

Once the requirements are met, the headers of the host system kernel must be installed. Here, one must make sure that the version numbers of the kernel and headers match exactly. Otherwise, a version conflict can occur preventing the successful installation of the firmware tools.

- *sudo apt -get install linux-headers- 'uname-r'*

Next, the downloaded tools must be installed, which can be done using the *dpkg.*

- *sudo dpkg -i <tool_package>*

Using the above steps, the UTN manager (minimal) version was installed successfully.

3.3.4 Additional Components

The HPC is linked to both the controlled 5G network environment and the Internet. In scenarios where peers are situated in distinct networks, Internet access becomes imperative for initiating the WebRTC connection.

In this particular use case, the signalling, and Session Traversal Utilities for NAT (STUN) servers could be located in DT's 5G test network or they reside in an Ericsson network that is openly accessible to the peer devices participating in the use case. Nevertheless, upon request, there is the possibility of exploring the deployment of signalling and STUN instances within the 5G network testbed to mitigate delays in the connection set up procedure.

During the integration process, it was finally decided to host the signalling server within the DT testbed, since the testbed does not offer connectivity to Ericsson's data centre without additional effort. For UoS and UoB, the signalling server remains at Ericsson's network, as the UoS and UoB testbeds do offer connectivity without additional effort.

Funded by the Horizon Europe Framework Programme of the European Union

3.3.5 Deployment

The application platform consists of the following code components

- main
- server
- lib/

Here, *main* includes the WebRTC client and the *server* is the WebRTC signalling server necessary to create a data channel between the WebRTC clients. The lib directory includes important helper functions to run the application.

3.3.5.1 Platform (Producer) Modes

The platform producer application can be run in different modes depending on the given environment specification. Here, one can differentiate between two modes, namely mode 1 and mode 2. Table 2 provides an overview of the mode and the specific hardware requirements.

Table 2:Platform Modes and Producer Hardware Requirements

Mode	Hardware
Mode 1	High-Performance Cluster Depth Camera
Mode 2	High-Performance Cluster Depth Camera Device Server

The modes consider whether the HPC on which the sender application is running supports a USB 3.0 interface or requires a device server that converts the USB 3.0 data traffic from the depth camera into IP data traffic to be compatible with the HPC. This requires that the HPC supports at least an Ethernet interface.

For the deployment at DT the platform runs in mode 2 and for UoS/UoB the platform runs in mode 1.

Funded by the Horizon Europe Framework Programme of the European Union

3.3.5.2 Platform (Consumer Setups)

Figure 43 presents the (consumer) setups deployed at DT, UoS and UoB.

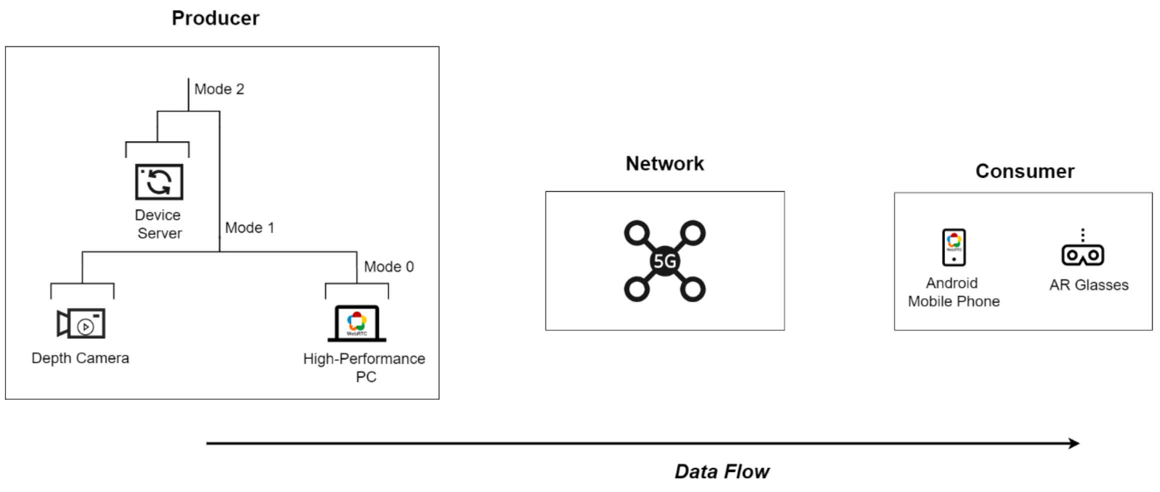


Figure 43: Holographic Communication SPIRIT Setup

As can be seen, the producer application is hosted on a high-performance PC. The SPIRIT setup supports each of the producer modes presented previously. At the receiver side the consumer application runs on a mobile phone connected to AR glasses. The producer side and consumer side are connected via a 5G network in the given SPIRIT testbeds.

Table 3 summarises the setup and the receiver hardware requirements. It should be noted that the total hardware setup is given by considering the hardware requirements on the producer side presented in Table 2.

Table 3: SPIRIT Holo Setup and Consumer Hardware Requirements

Setup	Hardware
SPIRIT	Mobile Phone AR Glasses

3.3.6 Validation

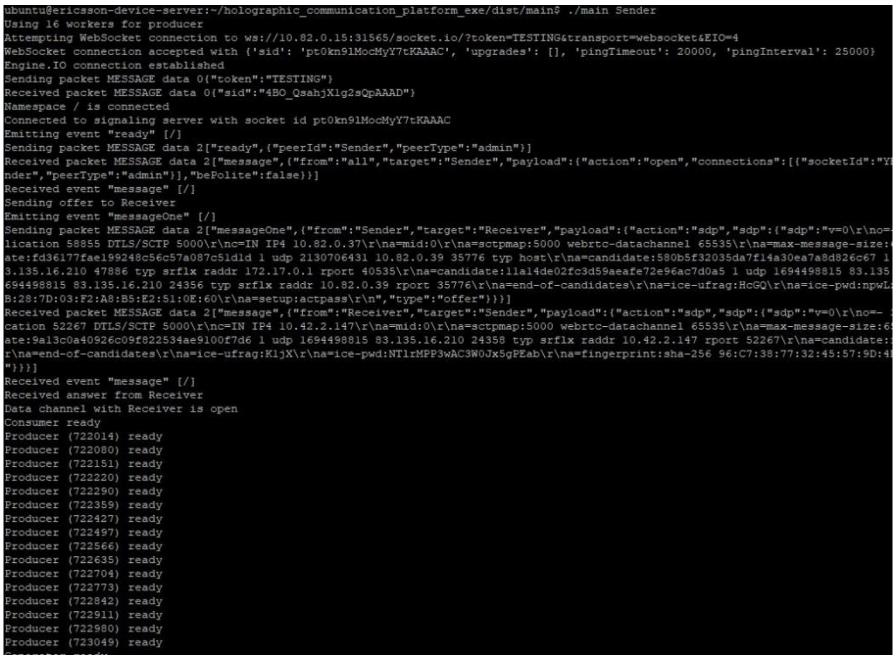
The validation steps described next refer to functional tests of crucial components of the application platform to confirm the successful integration within a testbed.

Producer Application as Executable

The binary is validated within a cloud environment that hosts a VM.

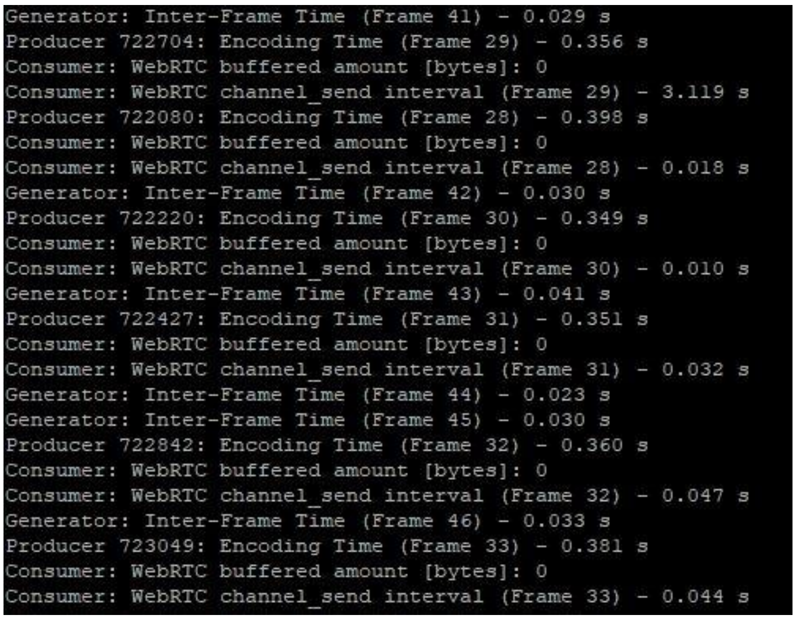
After transferring the binary directory to the HPC and starting the holographic communication sender binary, the output looks like in Figure 44 and Figure 45.

Funded by the Horizon Europe Framework Programme of the European Union



```
ubuntu@ericsson-device-server:~/holographic_communication_platform_exe/dist/main$ ./main Sender
Using 16 workers for producer
Attempting WebSocket connection to ws://10.82.0.15:31565/socket.io/?token=TESTING&transport=websocket&EIO=4
WebSocket connection accepted with {'sid': 'pt0kn9lMocMyY7tKAAAC', 'upgrades': [], 'pingTimeout': 20000, 'pingInterval': 25000}
Engine.IO connection established
Sending packet MESSAGE data 0{"token":"TESTING"}
Received packet MESSAGE data 0{"sid":"4BO_QsahjXlg2sQpAAAD"}
Namespace / is connected
Connected to signaling server with socket id pt0kn9lMocMyY7tKAAAC
Emitting event "ready" [/]
Sending packet MESSAGE data 2["ready",{"peerId":"Sender","peerType":"admin"}]
Received packet MESSAGE data 2["message",{"from":"all","target":"Sender","payload":{"action":"open","connections":[{"socketId":"Y
nder","peerType":"admin"}],"bePolite":false}}]
Received event "message" [/]
Sending offer to Receiver
Emitting event "messageOne" [/]
Sending packet MESSAGE data 2["messageOne",{"from":"Sender","target":"Receiver","payload":{"action":"sdp","sdp":{"sdp":"v=0\r\no=
lication 58855 DTLS/SCTP 5000\r\nc=IN IP4 10.82.0.37\r\na=mid:0\r\na=sctpmap:5000 webrtc-datachannel 65535\r\na=max-message-size:
ate:fd36177fae199248c56c57a087c51d1d 1 udp 2130706431 10.82.0.39 35776 typ host\r\na=candidate:580b5f32035da7f14a30ea7a8d826c67 1
3.135.16.210 47886 typ srflx raddr 172.17.0.1 rport 40535\r\na=candidate:11a14de02fc3d59aeafe72e96ac7d0a5 1 udp 1694498815 83.135
694498815 83.135.16.210 24356 typ srflx raddr 10.82.0.39 rport 35776\r\na=end-of-candidates\r\na=ice-ufrag:HcGQ\r\na=ice-pwd:npwL
B:28:7D:03:F2:A8:B5:E2:51:0E:60\r\na=setup:actpass\r\n","type":"offer"}}}]
Received packet MESSAGE data 2["message",{"from":"Receiver","target":"Sender","payload":{"action":"sdp","sdp":{"sdp":"v=0\r\no=-
cation 52267 DTLS/SCTP 5000\r\nc=IN IP4 10.42.2.147\r\na=mid:0\r\na=sctpmap:5000 webrtc-datachannel 65535\r\na=max-message-size:6
ate:9a13c0a40926c09f822534ae9100f7d6 1 udp 1694498815 83.135.16.210 24358 typ srflx raddr 10.42.2.147 rport 52267\r\na=candidate:
r\na=end-of-candidates\r\na=ice-ufrag:K1jX\r\na=ice-pwd:NTlrMPP3wAC3W0Jx5gPEab\r\na=fingerprint:sha-256 96:C7:38:77:32:45:57:9D:4
"}}}]
Received event "message" [/]
Received answer from Receiver
Data channel with Receiver is open
Consumer ready
Producer (722014) ready
Producer (722080) ready
Producer (722151) ready
Producer (722220) ready
Producer (722290) ready
Producer (722359) ready
Producer (722427) ready
Producer (722497) ready
Producer (722566) ready
Producer (722635) ready
Producer (722704) ready
Producer (722773) ready
Producer (722842) ready
Producer (722911) ready
Producer (722980) ready
Producer (723049) ready
```

Figure 44: Validation of Holographic Communication Application as Binary (Initial Phase)



```
Generator: Inter-Frame Time (Frame 41) - 0.029 s
Producer 722704: Encoding Time (Frame 29) - 0.356 s
Consumer: WebRTC buffered amount [bytes]: 0
Consumer: WebRTC channel_send interval (Frame 29) - 3.119 s
Producer 722080: Encoding Time (Frame 28) - 0.398 s
Consumer: WebRTC buffered amount [bytes]: 0
Consumer: WebRTC channel_send interval (Frame 28) - 0.018 s
Generator: Inter-Frame Time (Frame 42) - 0.030 s
Producer 722220: Encoding Time (Frame 30) - 0.349 s
Consumer: WebRTC buffered amount [bytes]: 0
Consumer: WebRTC channel_send interval (Frame 30) - 0.010 s
Generator: Inter-Frame Time (Frame 43) - 0.041 s
Producer 722427: Encoding Time (Frame 31) - 0.351 s
Consumer: WebRTC buffered amount [bytes]: 0
Consumer: WebRTC channel_send interval (Frame 31) - 0.032 s
Generator: Inter-Frame Time (Frame 44) - 0.023 s
Generator: Inter-Frame Time (Frame 45) - 0.030 s
Producer 722842: Encoding Time (Frame 32) - 0.360 s
Consumer: WebRTC buffered amount [bytes]: 0
Consumer: WebRTC channel_send interval (Frame 32) - 0.047 s
Generator: Inter-Frame Time (Frame 46) - 0.033 s
Producer 723049: Encoding Time (Frame 33) - 0.381 s
Consumer: WebRTC buffered amount [bytes]: 0
Consumer: WebRTC channel_send interval (Frame 33) - 0.044 s
```

Figure 45: Validation of Holographic Communication Application as Binary (Final Phase)

The first figure illustrates the start-up phase of the holographic communication sender, where the generator, producer, and consumer instances are initialized. During this phase, signalling data is exchanged between clients via the signalling server using Session Description Protocol (SDP) messages. The second figure depicts the final phase of the holographic communication sender application. At this stage, the consumer component functions as the WebRTC sender, transmitting 3D data to the receiver through an established WebRTC data channel.

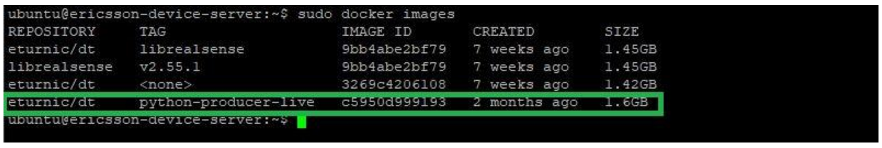
Funded by the Horizon Europe Framework Programme of the European Union

The terminal output confirms the successful deployment of the holographic communication system, including the sender, receiver, and signalling applications.

Producer Application as Container

The container is validated within an HPC environment that hosts a VM.

After pulling the image and creating a compose.yaml when using *sudo docker images,* the docker engine should display the pulled images similar to Figure 46.

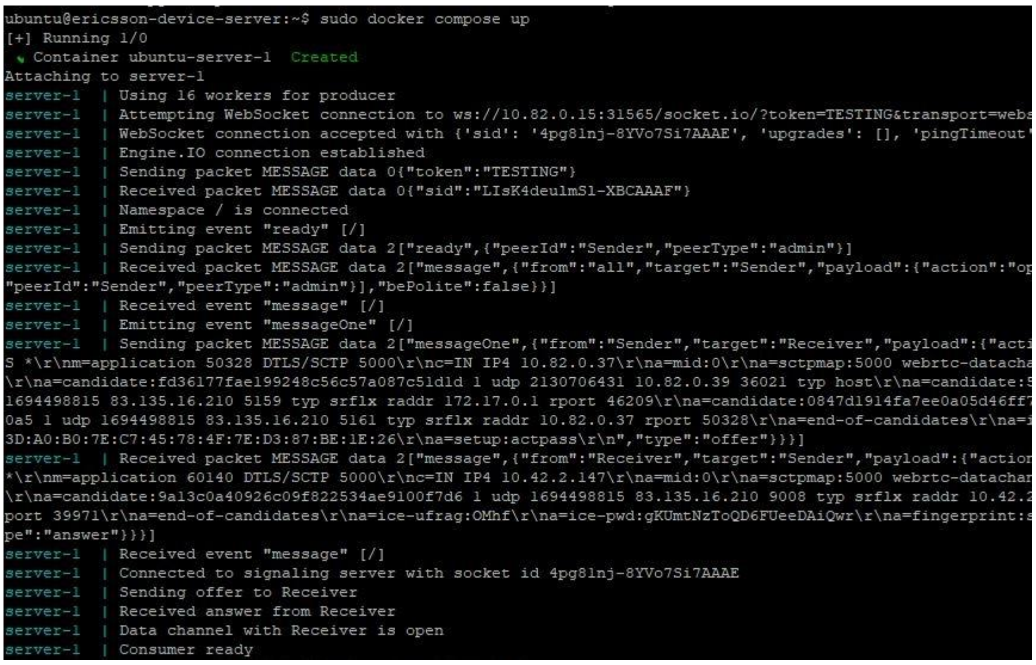


```
ubuntu@ericsson-device-server:~$ sudo docker images
REPOSITORY     TAG                    IMAGE ID       CREATED        SIZE
eturnic/dt     librealsense           9bb4abe2bf79   7 weeks ago    1.45GB
librealsense   v2.55.1                9bb4abe2bf79   7 weeks ago    1.45GB
eturnic/dt     <none>                 3269c4206108   7 weeks ago    1.42GB
eturnic/dt     python-producer-live   c5950d999193   2 months ago   1.6GB
ubuntu@ericsson-device-server:~$
```

Figure 46: Validation of Docker Image Pull

Next, the holographic communication sender application is started. The receiver and signalling application are started on a separate machine.

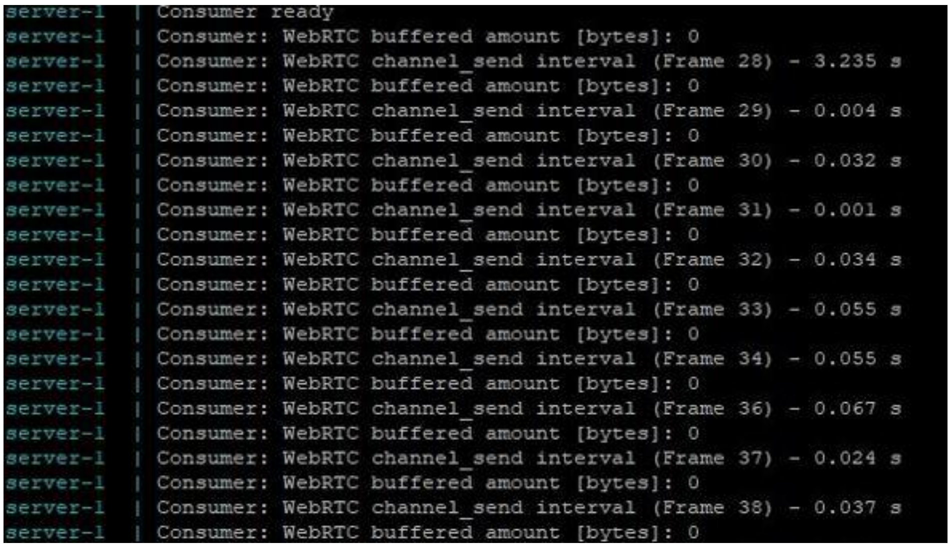
Figure 47 and Figure 48 show terminal output.



```
ubuntu@ericsson-device-server:~$ sudo docker compose up
[+] Running 1/0
 Container ubuntu-server-1  Created
Attaching to server-1
server-1  | Using 16 workers for producer
server-1  | Attempting WebSocket connection to ws://10.82.0.15:31565/socket.io/?token=TESTING&transport=webs
server-1  | WebSocket connection accepted with {'sid': '4pg8lnj-8YVo7Si7AAAE', 'upgrades': [], 'pingTimeout'
server-1  | Engine.IO connection established
server-1  | Sending packet MESSAGE data 0{"token":"TESTING"}
server-1  | Received packet MESSAGE data 0{"sid":"LIsK4deulmSl-XBCAAAF"}
server-1  | Namespace / is connected
server-1  | Emitting event "ready" [/]
server-1  | Sending packet MESSAGE data 2["ready",{"peerId":"Sender","peerType":"admin"}]
server-1  | Received packet MESSAGE data 2["message",{"from":"all","target":"Sender","payload":{"action":"op
"peerId":"Sender","peerType":"admin"}],"bePolite":false}}]
server-1  | Received event "message" [/]
server-1  | Emitting event "messageOne" [/]
server-1  | Sending packet MESSAGE data 2["messageOne",{"from":"Sender","target":"Receiver","payload":{"acti
S *\r\nm=application 50328 DTLS/SCTP 5000\r\nc=IN IP4 10.82.0.37\r\na=mid:0\r\na=sctpmap:5000 webrtc-datacha
\r\na=candidate:fd36177fae199248c56c57a087c51d1d 1 udp 2130706431 10.82.0.39 36021 typ host\r\na=candidate:5
1694498815 83.135.16.210 5159 typ srflx raddr 172.17.0.1 rport 46209\r\na=candidate:0847d1914fa7ee0a05d46ff7
0a5 1 udp 1694498815 83.135.16.210 5161 typ srflx raddr 10.82.0.37 rport 50328\r\na=end-of-candidates\r\na=i
3D:A0:B0:7E:C7:45:78:4F:7E:D3:87:BE:1E:26\r\na=setup:actpass\r\n","type":"offer"}}}]
server-1  | Received packet MESSAGE data 2["message",{"from":"Receiver","target":"Sender","payload":{"actior
*\r\nm=application 60140 DTLS/SCTP 5000\r\nc=IN IP4 10.42.2.147\r\na=mid:0\r\na=sctpmap:5000 webrtc-datachar
\r\na=candidate:9a13c0a40926c09f822534ae9100f7d6 1 udp 1694498815 83.135.16.210 9008 typ srflx raddr 10.42.2
port 39971\r\na=end-of-candidates\r\na=ice-ufrag:OMhf\r\na=ice-pwd:gKUmtNzToQD6FUeeDAiQwr\r\na=fingerprint:s
pe":"answer"}}}]
server-1  | Received event "message" [/]
server-1  | Connected to signaling server with socket id 4pg8lnj-8YVo7Si7AAAE
server-1  | Sending offer to Receiver
server-1  | Received answer from Receiver
server-1  | Data channel with Receiver is open
server-1  | Consumer ready
```

Figure 47: Validation of Holographic Communication Application within a Container (Initial Phase)

Funded by the Horizon Europe Framework Programme of the European Union



```
server-1   | Consumer ready
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 28) - 3.235 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 29) - 0.004 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 30) - 0.032 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 31) - 0.001 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 32) - 0.034 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 33) - 0.055 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 34) - 0.055 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 36) - 0.067 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 37) - 0.024 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
server-1   | Consumer: WebRTC channel_send interval (Frame 38) - 0.037 s
server-1   | Consumer: WebRTC buffered amount [bytes]: 0
```

Figure 48: Validation of Holographic Communication Application within a Container (Final Phase)

The first figure shows the start-up phase of the holographic communication sender. Here the generator, producer, and consumer instances are initiated. In addition, the signalling data is exchanged between the clients through the signalling server in the form of session description protocol (SDP) messages. The second figure shows the final phase of the holographic communication sender application. Here, the consumer part of the sender application acts as a WebRTC sender transmitting the 3D data to the receiver through an established WebRTC data channel.

The terminal output indicates the successful deployment of the holographic communication application, i.e., sender, receiver, and signalling application.

Device Server

It is important to note that the firmware of the device server must be installed on the host OS. In the case of a container, installing the firmware within the container is not sufficient. This is because the UTN manager must communicate to the kernel of the OS within the host system, which is hardly possible from inside a container.

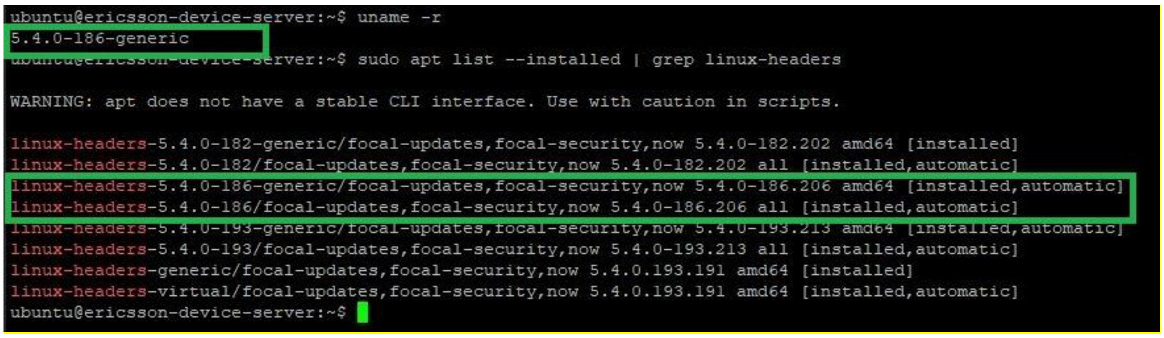
Device Server - Matching Versions Numbers of Linux Kernels and Headers

To verify the matching version numbers of kernel and headers, the following commands are used

- Kernel Version: *uname -r*
- Kernel Headers: *sudo apt list –installed* | *grep linux-headers*

The result of the terminal output is depicted in Figure 49. As seen, the kernel and headers show a matching version number marked in green.

Funded by the Horizon Europe Framework Programme of the European Union

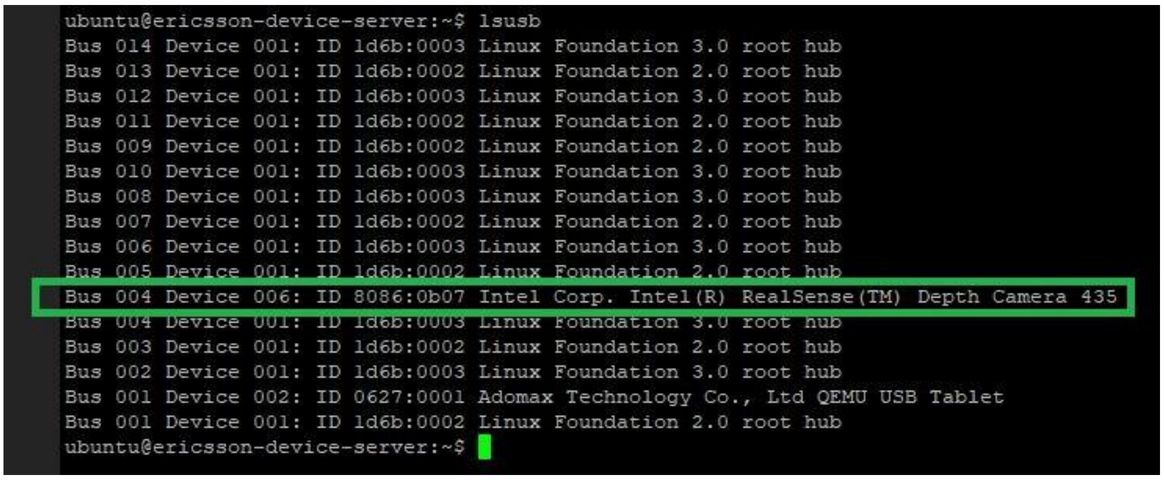


```
ubuntu@ericsson-device-server:~$ uname -r
5.4.0-186-generic
ubuntu@ericsson-device-server:~$ sudo apt list --installed | grep linux-headers

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

linux-headers-5.4.0-182-generic/focal-updates,focal-security,now 5.4.0-182.202 amd64 [installed]
linux-headers-5.4.0-182/focal-updates,focal-security,now 5.4.0-182.202 all [installed,automatic]
linux-headers-5.4.0-186-generic/focal-updates,focal-security,now 5.4.0-186.206 amd64 [installed,automatic]
linux-headers-5.4.0-186/focal-updates,focal-security,now 5.4.0-186.206 all [installed,automatic]
linux-headers-5.4.0-193-generic/focal-updates,focal-security,now 5.4.0-193.213 amd64 [installed,automatic]
linux-headers-5.4.0-193/focal-updates,focal-security,now 5.4.0-193.213 all [installed,automatic]
linux-headers-generic/focal-updates,focal-security,now 5.4.0.193.191 amd64 [installed]
linux-headers-virtual/focal-updates,focal-security,now 5.4.0.193.191 amd64 [installed,automatic]
ubuntu@ericsson-device-server:~$
```

Figure 49: Validation of Matching Linux Kernel and Headers

Device Server - Connected USB Devices

After successful installation of the UTN manager, the VM should recognise the camera device as a USB device. To verify this, the *lsusb* command can be used within the terminal. The result is shown in Figure 50. The figure shows that the VM recognises the depth camera connected to the device server.



```
ubuntu@ericsson-device-server:~$ lsusb
Bus 014 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 013 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 012 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 011 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 009 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 010 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 008 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 007 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 006 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 005 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 006: ID 8086:0b07 Intel Corp. Intel(R) RealSense(TM) Depth Camera 435
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 002: ID 0627:0001 Adomax Technology Co., Ltd QEMU USB Tablet
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
ubuntu@ericsson-device-server:~$
```

Figure 50: Validation of Depth Camera and Cloud Connectivity

Device Server - Functionality

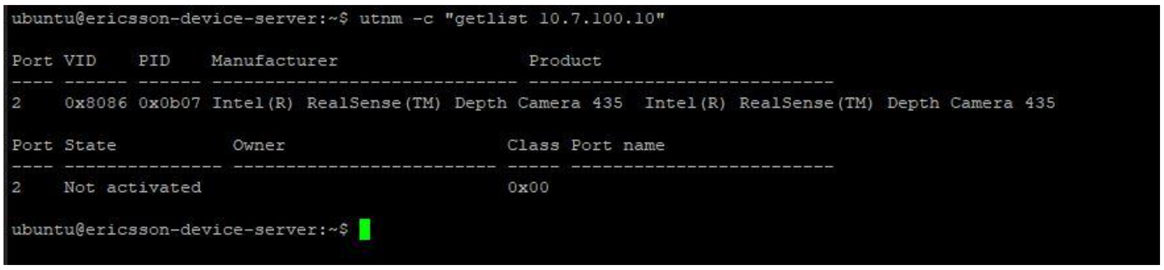
The device server comes with various functionalities. One important feature is the ability to activate and deactivate the camera remotely, which can also be automated. For further information on the device server, we recommend referring to the device server manual, which can be found here[7].

First, it is verified that the device server and the attached depth camera are visible by using the UTN manager command as seen in Figure 51.

- *utnm -c "getlist 10.7.100.10"*

[7] https://www.seh-technology.com/services/downloads/industrial/inu-100.html

Funded by the Horizon Europe Framework Programme of the European Union



```
ubuntu@ericsson-device-server:~$ utnm -c "getlist 10.7.100.10"

Port VID    PID    Manufacturer                  Product
---- ------ ------ ----------------------------- -----------------------------
2    0x8086 0x0b07 Intel(R) RealSense(TM) Depth Camera 435  Intel(R) RealSense(TM) Depth Camera 435

Port State           Owner                     Class Port name
---- --------------- ------------------------- ----- -------------------------
2    Not activated                             0x00

ubuntu@ericsson-device-server:~$
```

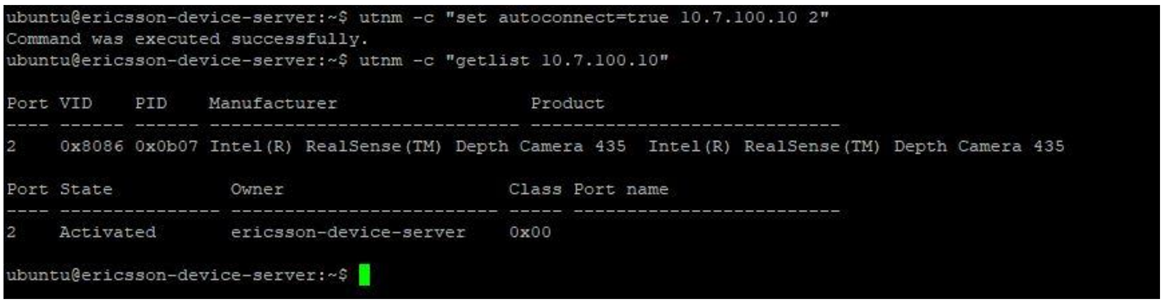
Figure 51: Validation of Device Server List Functionality

As expected, the device server displays the connected depth camera, otherwise the *lsub* command in the previous validation step would not have displayed the depth camera. However, it can be noted that the depth camera is not activated as indicated by the state tab in the terminal.

To turn on the camera one can either use the *connect* or *autoconnect* method. We recommend the *autoconnect* method, which activates the depth camera automatically when the device server it booted up.

- *utnm -c "activate 10.7.100.10 2"*
- *utnm -c "set autoconnect=true 10.7.100.10 2"*

The second term after the IP address of the device server indicates the desired USB port, in this case USB port 2. The result after activating the depth camera and using the list command of the device server is depicted in Figure 52 .



```
ubuntu@ericsson-device-server:~$ utnm -c "set autoconnect=true 10.7.100.10 2"
Command was executed successfully.
ubuntu@ericsson-device-server:~$ utnm -c "getlist 10.7.100.10"

Port VID    PID    Manufacturer                  Product
---- ------ ------ ----------------------------- -----------------------------
2    0x8086 0x0b07 Intel(R) RealSense(TM) Depth Camera 435  Intel(R) RealSense(TM) Depth Camera 435

Port State           Owner                     Class Port name
---- --------------- ------------------------- ----- -------------------------
2    Activated       ericsson-device-server    0x00

ubuntu@ericsson-device-server:~$
```

Figure 52: Validation of Device Server Activate Functionality

As it can be seen, the state of the depth camera changed to activated. It should be noted that activate does not mean that the camera starts streaming but rather that the camera interface is ready to be used by the cloud through the device server.

In summary, the device server was successfully validated when installing the firmware directly on the host machine or the VM on top.

Funded by the Horizon Europe Framework Programme of the European Union

3.3.7 Summary

To summarise, the application platform supporting the holographic communication use case is successfully deployed at DT, UoS, and UoB. To integrate the producer application into the testbed of DT, additional hardware in the form of a device server is needed to connect the depth camera on the producer side to the HPC hosting the producer application. The additional hardware mainly serves the purpose of relaying USB data from a depth camera to an IP network where the HPC resides. For UoS, there is no need for an additional device server since the HPC and the depth camera both support a common interface in the form of USB 3.0. The same hardware requirements apply to the integration efforts for UoB.

The consumer application integration demands the use of a mobile phone that can be connected to the network testbed using a SIM card provided by the testbed owners, namely DT, UoS, and UoB. In addition, the mobile phone hosting the consumer application must be supported by the corresponding AR glasses of this use case.

Table 4 highlights important specifications of the implemented use case:

Table 4: Use case: Holographic Communication Specifications

	Holographic Communications
Users	Producer: a user who is captured by a depth camera. The captured information is used to generate a digital 3D representation which is streamed to a receiving consumer. Consumer: a user who receives, processes, and displays the human 3D representation of the user on the producer side.
Network	Real-time communication peer-to-peer network (WebRTC)
Network Access	Wireless, e.g., Wi-Fi / 5G Wired, e.g., Ethernet
Components	Depth Camera (Intel RealSense): capture of media and spatial information. (Device Server: relay instance between camera and cloud.) HPC (Windows/Linux): generation and encoding of meshes for each RGB-D frame and transmission of the data. Mobile phone (Android): receiving, decoding, and rendering of data. AR glasses (XReal Light): displaying of 3D object to a user.

The use case includes two users engaged in digital communications. The user on the producer side provides data to be processed and ultimately transmitted to a receiver on the consumer side for the presentation of volumetric content, manifested as a human hologram on AR glasses.

For data exchange, users establish a real-time communication peer-to-peer (P2P) connection employing WebRTC, as described in D3.3 [3]. The essential devices required for leveraging the presented application platform can be supplied by the respective application platform or testbed provider. This provision aims to facilitate smooth development, integration, and testing of innovations by third parties during the first Open Call. The use of other components by third parties is possible but needs to be considered in collaboration with Ericsson and the testbed providers, i.e. DT, UoS, and UoB.

Funded by the Horizon Europe Framework Programme of the European Union

3.3.8 Extension: many-to-many conferencing through imec's Virtual Wall

In D3.3, imec's demonstrator for many-to-many video delivery is presented. This demonstrator has first been validated in a local demo setup. For scalability purposes, the setup has been rebuilt on imec's Virtual Wall and made accessible for external parties.

We recreated our local demo setup on the Virtual Wall using the star-shaped network shown in Figure 53. Five nodes are available with public IPv4 address: four client nodes and one server node. All clients are connected to the server through a dedicated switch, which allows for advanced network emulation using traffic control. This allows to dynamically change the available bandwidth, network latency and jitter, packet loss, etc.

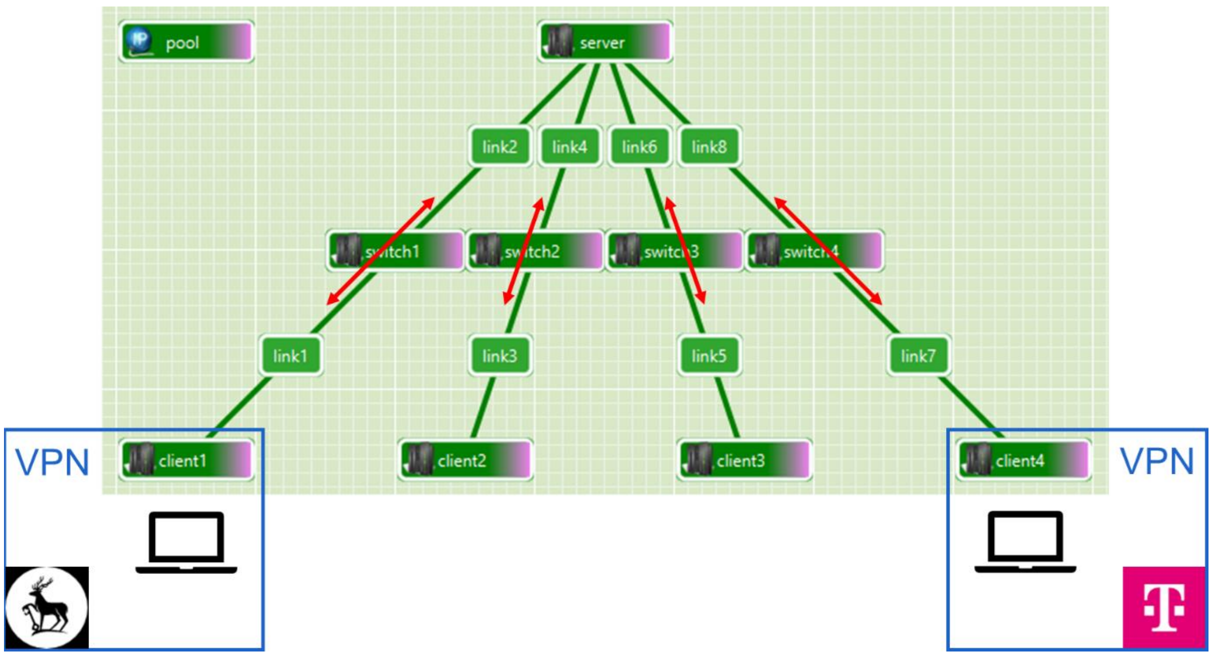


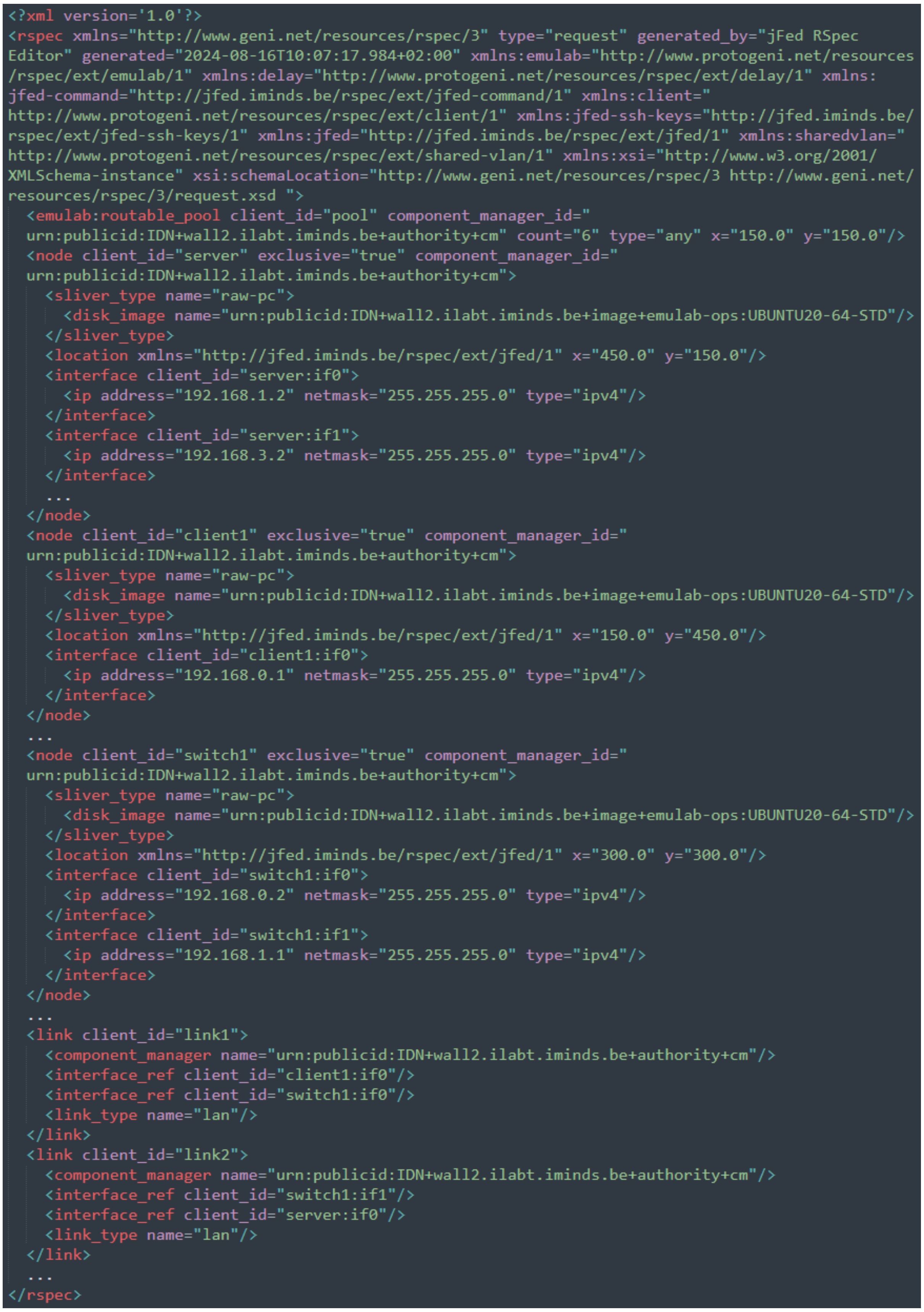
Figure 53: Experimental setup using jFed hosted on imec's Virtual Wall infrastructure

Fed4FIRE uses Rspec files to create, manage and share experiments. Figure 54 shows part of the Rspec file, which corresponds to the experimental setup in Figure 53. Different elements specify the requested nodes and the links between them, the requested OS, etc.

Because hardware specifications on the Virtual Wall are limited, our aim is to make sure that remote high-tier machines, equipped with a powerful GPU and an external HMD, can be used to run the experiments. These devices can connect through VPN, since a custom VPN is set up through OpenVPN on each of the client nodes. Since all nodes have a public IP address, remote machines (belonging to e.g., one user at Surrey and one user at Berlin) can easily connect to the local network.

The selective forwarding unit (SFU) described in D3.3 [3] is hosted on the server node, so that clients can connect over WebRTC. While these clients can technically connect over any interface (including the one used for public access), they are forced to set up a connection over the local network. This way, the impact of emulated network conditions on the system performance can be examined. The demo's dashboard (see Figure 55) is also made available on the server node and can be accessed through its public IPv4 address.

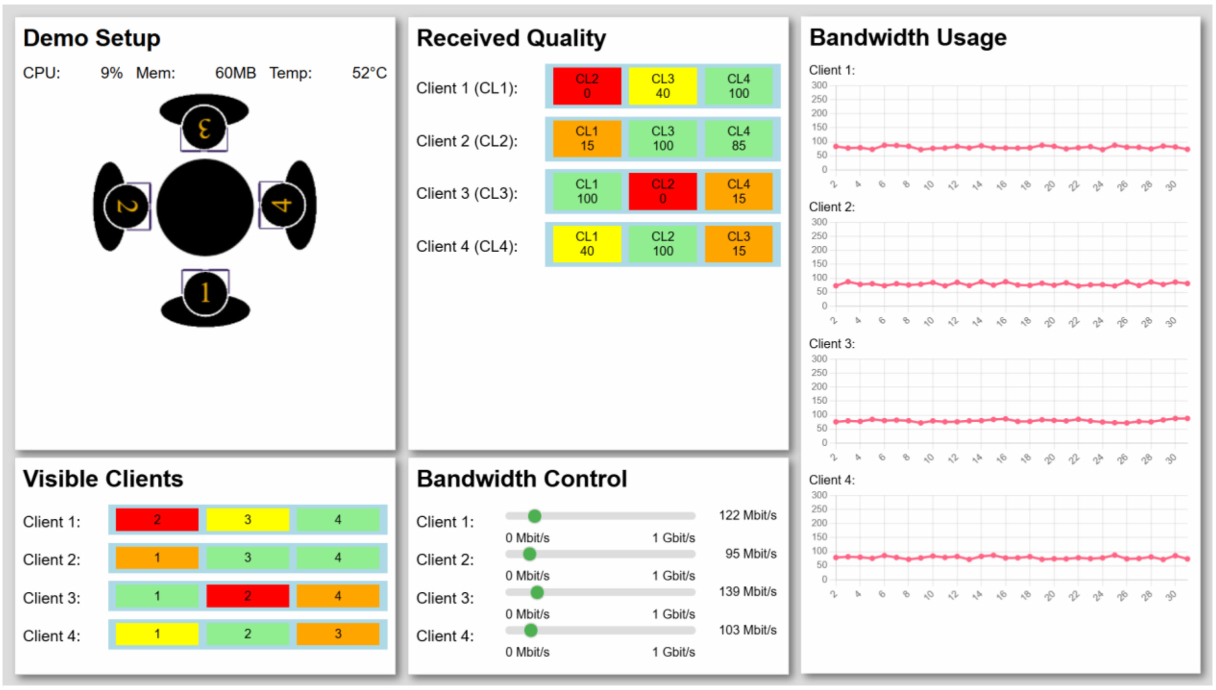
Funded by the Horizon Europe Framework Programme of the European Union



```
<?xml version='1.0'?>
<rspec xmlns="http://www.geni.net/resources/rspec/3" type="request" generated_by="jFed RSpec
Editor" generated="2024-08-16T10:07:17.984+02:00" xmlns:emulab="http://www.protogeni.net/resources
/rspec/ext/emulab/1" xmlns:delay="http://www.protogeni.net/resources/rspec/ext/delay/1" xmlns:
jfed-command="http://jfed.iminds.be/rspec/ext/jfed-command/1" xmlns:client="
http://www.protogeni.net/resources/rspec/ext/client/1" xmlns:jfed-ssh-keys="http://jfed.iminds.be/
rspec/ext/jfed-ssh-keys/1" xmlns:jfed="http://jfed.iminds.be/rspec/ext/jfed/1" xmlns:sharedvlan="
http://www.protogeni.net/resources/rspec/ext/shared-vlan/1" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:schemaLocation="http://www.geni.net/resources/rspec/3 http://www.geni.net/
resources/rspec/3/request.xsd ">
  <emulab:routable_pool client_id="pool" component_manager_id="
  urn:publicid:IDN+wall2.ilabt.iminds.be+authority+cm" count="6" type="any" x="150.0" y="150.0"/>
  <node client_id="server" exclusive="true" component_manager_id="
  urn:publicid:IDN+wall2.ilabt.iminds.be+authority+cm">
    <sliver_type name="raw-pc">
      <disk_image name="urn:publicid:IDN+wall2.ilabt.iminds.be+image+emulab-ops:UBUNTU20-64-STD"/>
    </sliver_type>
    <location xmlns="http://jfed.iminds.be/rspec/ext/jfed/1" x="450.0" y="150.0"/>
    <interface client_id="server:if0">
      <ip address="192.168.1.2" netmask="255.255.255.0" type="ipv4"/>
    </interface>
    <interface client_id="server:if1">
      <ip address="192.168.3.2" netmask="255.255.255.0" type="ipv4"/>
    </interface>
    ...
  </node>
  <node client_id="client1" exclusive="true" component_manager_id="
  urn:publicid:IDN+wall2.ilabt.iminds.be+authority+cm">
    <sliver_type name="raw-pc">
      <disk_image name="urn:publicid:IDN+wall2.ilabt.iminds.be+image+emulab-ops:UBUNTU20-64-STD"/>
    </sliver_type>
    <location xmlns="http://jfed.iminds.be/rspec/ext/jfed/1" x="150.0" y="450.0"/>
    <interface client_id="client1:if0">
      <ip address="192.168.0.1" netmask="255.255.255.0" type="ipv4"/>
    </interface>
  </node>
  ...
  <node client_id="switch1" exclusive="true" component_manager_id="
  urn:publicid:IDN+wall2.ilabt.iminds.be+authority+cm">
    <sliver_type name="raw-pc">
      <disk_image name="urn:publicid:IDN+wall2.ilabt.iminds.be+image+emulab-ops:UBUNTU20-64-STD"/>
    </sliver_type>
    <location xmlns="http://jfed.iminds.be/rspec/ext/jfed/1" x="300.0" y="300.0"/>
    <interface client_id="switch1:if0">
      <ip address="192.168.0.2" netmask="255.255.255.0" type="ipv4"/>
    </interface>
    <interface client_id="switch1:if1">
      <ip address="192.168.1.1" netmask="255.255.255.0" type="ipv4"/>
    </interface>
  </node>
  ...
  <link client_id="link1">
    <component_manager name="urn:publicid:IDN+wall2.ilabt.iminds.be+authority+cm"/>
    <interface_ref client_id="client1:if0"/>
    <interface_ref client_id="switch1:if0"/>
    <link_type name="lan"/>
  </link>
  <link client_id="link2">
    <component_manager name="urn:publicid:IDN+wall2.ilabt.iminds.be+authority+cm"/>
    <interface_ref client_id="switch1:if1"/>
    <interface_ref client_id="server:if0"/>
    <link_type name="lan"/>
  </link>
  ...
</rspec>
```

Figure 54: Part of the Rspec file, specifying the required nodes and links in the experiment

Funded by the Horizon Europe Framework Programme of the European Union



Demo Setup

CPU: 9% Mem: 60MB Temp: 52°C

Received Quality

Client 1 (CL1):	CL2 0	CL3 40	CL4 100
Client 2 (CL2):	CL1 15	CL3 100	CL4 85
Client 3 (CL3):	CL1 100	CL2 0	CL4 15
Client 4 (CL4):	CL1 40	CL2 100	CL3 15

Bandwidth Usage

Visible Clients

Client 1:	2	3	4
Client 2:	1	3	4
Client 3:	1	2	4
Client 4:	1	2	3

Bandwidth Control

Client 1:	0 Mbit/s	1 Gbit/s	122 Mbit/s
Client 2:	0 Mbit/s	1 Gbit/s	95 Mbit/s
Client 3:	0 Mbit/s	1 Gbit/s	139 Mbit/s
Client 4:	0 Mbit/s	1 Gbit/s	103 Mbit/s

Figure 55: Dashboard of the WebRTC demonstrator described in D3.2

The functionality of this dashboard has been extended recently to include computational metrics about the server node. CPU and memory usage, as well as the temperature, are included in the top left corner.

To help other researchers set up an experiment of their own, we created a GitHub repository[8] that explains exactly how to do that. This repository contains the following:

- A README.md file with detailed instructions on how to set up an experiment
- The Rspec file to recreate the experiment on imec's Virtual Wall
- Install scripts that are automatically retrieved and executed when launching the experiment, making sure that client nodes set up a custom VPN, that traffic control is enabled on all switches, and that the SFU and corresponding dashboard are started on the server node
- A sample configuration file used for experimentation, specifying e.g. the available bandwidth

The provided documentation enables external users to create an experimental setup, connect several remote machines, and start up a WebRTC session between the involved clients. Parameter settings can easily be configured, specifying the number of clients (2, 3 or 4), the number of quality representations, and several network-related parameters.

Currently, the Rspec file provides a setup with four client nodes only. The setup can be extended to contain more clients, but this is not straightforward for a number of reasons:

[8] https://github.com/idlab-discover/pc-webrtc-vwall

Funded by the Horizon Europe Framework Programme of the European Union

- Physical machines in the Virtual Wall infrastructure are interconnected through 1 Gb/s links, which in practice result in bandwidths between 500 to 940 Mb/s. While this is significant, it is important to realise that our solution for a single RGBD camera results in a compressed video bitrate of approximately 75 Mb/s. When ten client devices actively send content to the SFU, network limitations quickly result in an impeded video quality.
- Even though bitrate adaptation is used to respect network capacity, it is possible for a client to retrieve multiple representations of all users involved in the immersive video conference. Decoding the content requires significant computational resources, which makes it unpractical to consider video at 30 FPS for e.g. ten other participants on commodity hardware, even with parallel decoding of incoming descriptions.

Nevertheless, the provided Rspec file can be modified to consider more than four clients if needed. The setup can also be extended with a limited number of virtual "senders", headless clients that send pre-captured static or dynamic point cloud video (e.g., objects from the 8i dataset[9]) to the SFU. While the content is not captured live, it does require receiving clients to decode and render the content.

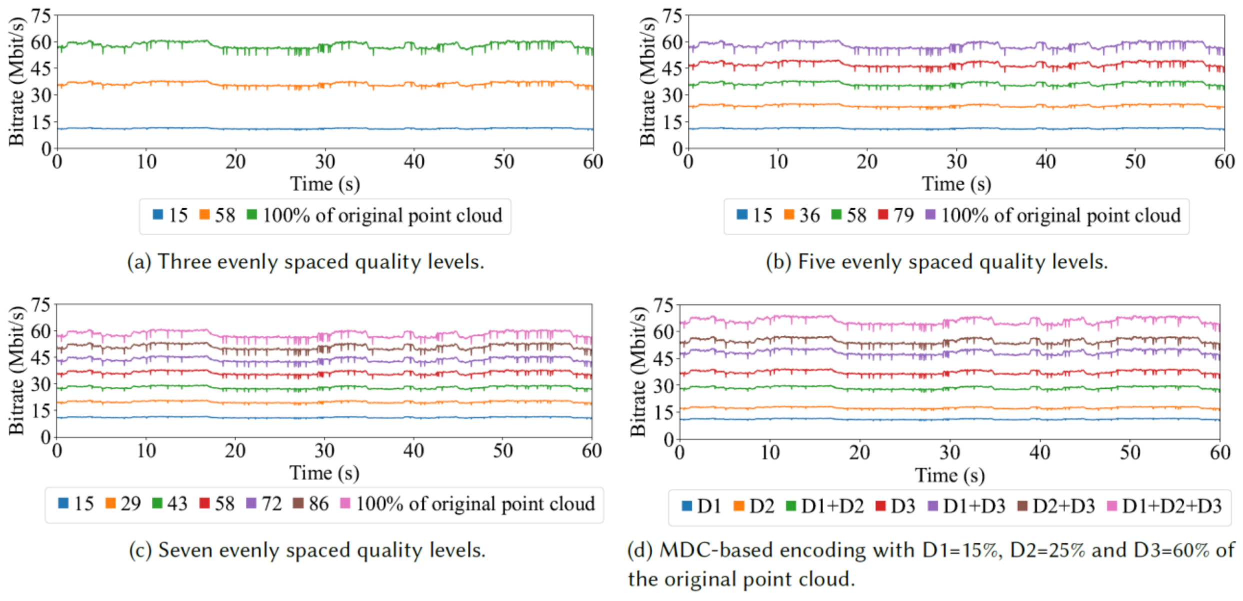
3.3.8.1 Validation tests on imec's Virtual Wall

The goal of our pipeline is being able to adapt its content based on the available bandwidth, while still maintaining a latency suited for real-time communication. In order to test the achievable throughput and WebRTC latency on the Virtual Wall testbed we use two different link capacities, 50 Mbit/s and 100 Mbit/s. We motivate the use of the 50 Mbit/s capacity due to it being below the maximum required bandwidth, while still allowing for a sufficient number of quality representations (see Figure 56). In this experiment, we employ the fixed-size MDC encoder due to it having the most consistent resulting bitrates, as a stable bandwidth makes it easier for the congestion controller to converge.

Figure 57 depicts the results of this experiment, we observe that the 100 Mbit/s link reaches a stable bitrate of 70.6 Mbit/s. Referring to Table 5, we observe that this corresponds to the bitrate of the highest quality representation. Contrary, the 50 Mbit/s link stabilizes at 30.6 Mbit/s, corresponding with the 40% representation. However, even with the 90% safeguard this link should be able to achieve bitrate of the 60% representation. Figure 57 indicates that this behaviour is due to the GCC algorithm not recovering after an overestimation which subsequently caused packet loss. By analysing both figures, we can conclude that the GCC algorithm takes a significant time before converging to a stable estimation. This phenomenon is related to the initial bitrate of GCC, which was set to the minimum required bandwidth. Increasing the initial bitrate speeds up the convergence at the cost of causing packet loss for low bandwidth links. In the case of the 50 Mbit/s link, convergence is observed around 30 Mbit/s. Looking at Table 5, this corresponds to a quality level of 40% with an average VMAF score of 67.56.

[9] https://plenodb.jpeg.org/pc/8ilabs

Funded by the Horizon Europe Framework Programme of the European Union



(a) Three evenly spaced quality levels.

(b) Five evenly spaced quality levels.

(c) Seven evenly spaced quality levels.

(d) MDC-based encoding with D1=15%, D2=25% and D3=60% of the original point cloud.

Figure 56: Having only three fixed quality representations causes large jumps in the bitrate ladder compared to the smooth behavior of having seven fixed representations. Available bitrates for the MDC-based approach have a similar spread as the seven qualities.

Table 5: Resulting average bitrates (Mbit/s) for the different MDC quality representations, with 15%, 25% and 60% being the base descriptions used to generate the other representations

	15%	25%	40%	60%	75%	85%	100%
Fixed Size	11.9	18.6	30.6	39.9	51.8	58.5	70.5
Fixed Bitrate	12.6	19.6	32.1	41.2	53.6	60.8	73.3
Percentage	12.7	19.8	32.0	42.4	55.0	62.1	74.8

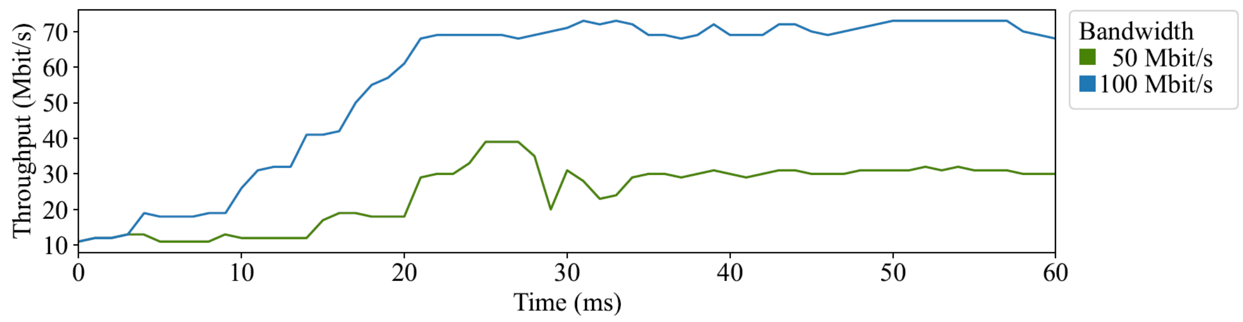


Figure 57: Throughput obtained when using the fixed size MDC approach for 50 Mbit/s and 100 Mbit/s

In terms of latency, the 100 Mbit/s link achieves an average transport latency of 30.9 ms (std dev=10.2 ms), compared to 24.4 ms (std dev=5.7 ms) for the 50 Mbit/s link. Both results are acceptable to use in a real-time environment. The lower latency of the 50 Mbit/s link is attributed to the lower throughput, indicating that the GCC algorithm ensures a good balance between estimated bandwidth, latency and potential packet loss.

As illustrated by Figure 58, it is necessary to have at least five evenly distributed quality levels to achieve the throughput needed for the highest quality level. This requirement arises from significant jumps in required bitrates, which cause sudden latency spikes. The GCC implementation in use cannot manage these spikes effectively, because it relies on inter-packet latency to detect sudden congestion and cannot discern whether the increase is caused by the application or network congestion. Consequently, having a higher number of quality levels gradually increases the latency, ensuring the inter-packet latency increase remains stable, as shown in Figure 59d. Reducing the available bandwidth to 50 Mbit/s shows similar behaviour and also requires at least five fixed quality levels before acceptable bitrates are achieved. For the encoders with five or more qualities the gaps between the qualities have decreased enough to allow for a smoother increase of latency.

With the 100 Mbit/s link, the encoder with five fixed quality levels achieves a throughput of 65.2 Mbit/s (std dev=3.5 Mbit/s) with an average latency of 29.4 ms (std dev=2.6 ms). These results are comparable to the results of the MDC-based approach. This is also observed when reducing the bandwidth to 50 Mbit/s, which results in a throughput of 43.8 Mbit/s (std dev=1.5 Mbit/s), which corresponds to a sampling rate of 60% or a VMAF score of 75.8.

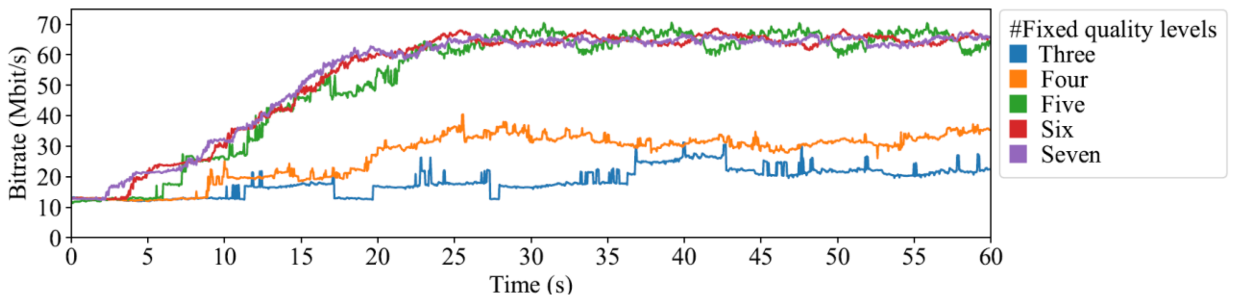


Figure 58: Achieved bitrates when using WebRTC together with GCC, indicating that three and four quality levels are insufficient to achieve the throughput required for high quality

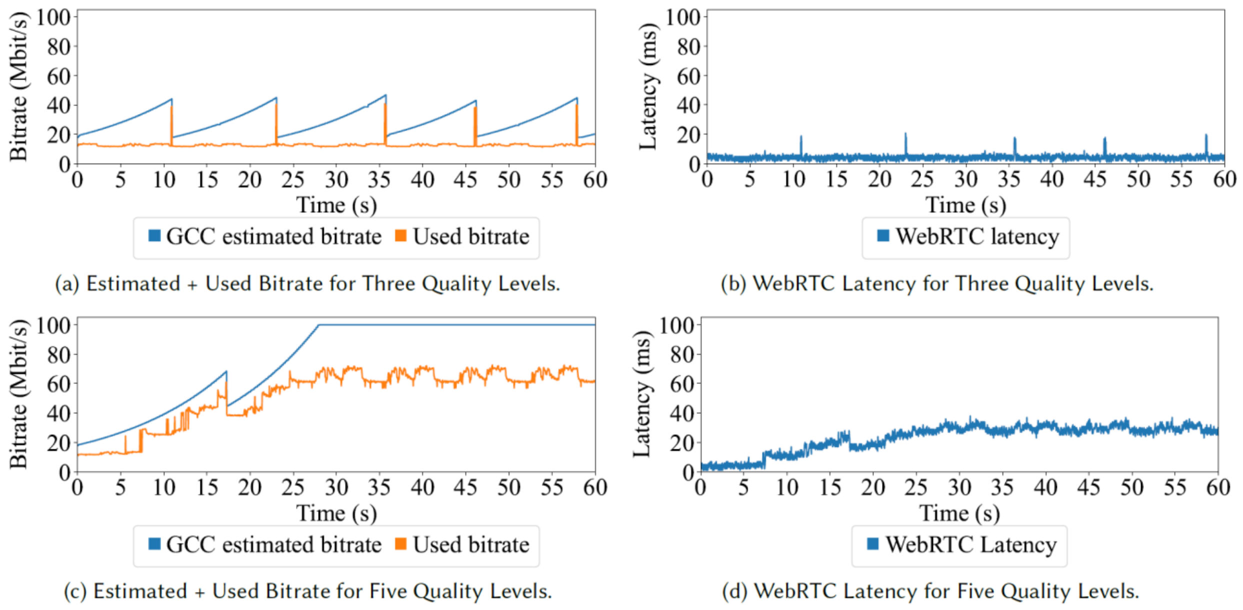


Figure 59: Resulting bitrate, estimated bitrate and latency for a single WebRTC run, illustrating the impact of the latency spikes on the throughput and bitrate estimated by GCC when using a 100 Mbit/s network link

We refer to our published paper for more detailed presentation of the results [6].

Funded by the Horizon Europe Framework Programme of the European Union

3.4 REAL-TIME ANIMATION AND STREAMING OF REALISTIC AVATARS

The integration of the “Real-Time Animation and Streaming of Realistic Avatars” application platform in the testbeds provided by T-Systems (as described in Section 2.3) and the Universities of Surrey and Bristol (Section 2.1) has been carried out in a sequence of steps, from the definition of the requirements to the evaluation and assessment of the performance. A detailed description of the proposed scenario has been described in other deliverables of the project, such as D2.3 [2], where the general architecture of the framework is included or D3.3 [3], where a list of the innovations developed within the use case is detailed.

3.4.1 Common Prerequisites for Integration

The requirements of both the application platform and the testbed first had to aligned. The testbed in Berlin allows the deployment and execution of applications in a container-based environment. This environment offers connectivity options among different containers and access to physical resources, such as GPU cards and computational power. The orchestration of the containers is carried out by a Kubernetes Cluster that uses the Rancher management platform. On the other hand, a 5G internal network is also available to connect the servers with different client devices, such as mobile phones, tablets, and AR glasses. No external connection is needed at this step of the integration, since the whole application platform can be tested internally.

On the other hand, the testbed in Surrey requires the deployment of the server application on a virtual machine equipped with Ubuntu 22.04 LTS. This machine has access to a Nvidia GPU and the required connectivity to its internal 5G network. Again, no external connection is needed.

3.4.2 Containerisation of the Application

The original server application platform was based on a central Unity app in which different external plugins provided the required functionalities. To integrate it in the testbed in Berlin, an effort has been carried out to build a set of three containers from the application. These containers are self-sufficient and include integral environments in which software can run without the need of additional components. This makes this kind of deployments very portable across a variety of scenarios.

Three containers have been created:

- **Audio animation server:** audio processing application written in Python that accepts a set of audio samples and generates a list of the visemes that corresponds to the speech contained in them.
- **Video animation server:** video processing application written in Python that accepts a video streaming signal and generates a list of the visemes that corresponds to the facial expressions contained in them.
- **Avatar animation, rendering and streaming:** Unity application that acts as a central hub with which the rest of the components interact to generate the mesh and texture of the avatar, render it and stream the resulting audio and video to the clients. This application must run in “server” or “headless” mode. This means that no display can be used on the server. To overcome this obstacle, a virtual display is configured within the container to allow Unity to render the necessary views.

Funded by the Horizon Europe Framework Programme of the European Union

3.4.3 Deployment and Interconnection of Containers

Once the containers were created, they were deployed in the Rancher Kubernetes orchestrator. Three images, corresponding to both necessary containers, were created and uploaded to Docker Hub. Then, a deployment element was created in Rancher. This deployment has three containers, each of them using one of the images.

The audio animation server exposes the port 6666 through the container. The main container uses this port to insert the received audio samples so they can be processed. This container, on its side, is exposing the port 8080. This allows the client devices to connect to the running server.

A schema of the architecture of the whole integration can be seen in Figure 60, simplified by showing only the rendering and the audio animation servers, since the container for the video animation server is similar to the audio animation server.

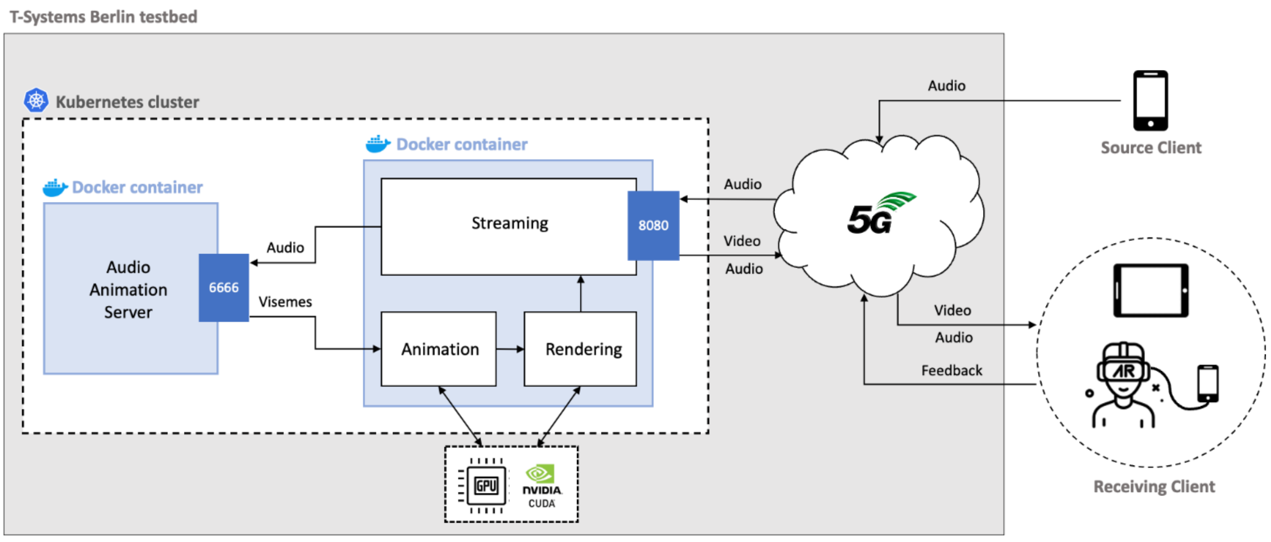


Figure 60: Integration of the real -time animation and streaming system in the Berlin testbed

One of the main challenges that the deployment of the server component has presented is the need of the central Unity application (see D2.3 [2]) to have access to a display while running in headless mode. In this working mode, there is no visible rendering of the Unity scene at the server since the whole rendering process takes place in the background and only the clients show the corresponding video. To solve this, a reference to an X virtual server has been included within the Docker image. This way, the container can access the GPU and the Unity application can run without a real display.

The startup of the container corresponding to the avatar animation and streaming server required extra configuration to guarantee that the application could make use of the hardware resources of the machine. The following command is used to run this container:

```
docker run -it \
      --gpus all \
      -p 8080 \
      -e PULSE_SERVER=unix:${XDG_RUNTIME_DIR}/pulse/na-
      tive:${XDG_RUNTIME_DIR}/pulse/native \
      -v ${XDG_RUNTIME_DIR}/pulse/native:${XDG_RUNTIME_DIR}/pulse/native \
      -v ~/.config/pulse/cookie:/root/.config/pulse/cookie \
      -v /tmp/.X11-unix:/tmp/.X11-unix \
```

Funded by the Horizon Europe Framework Programme of the European Union

```
-v $XDG_RUNTIME_DIR:$XDG_RUNTIME_DIR \
-e XAUTHORITY \
-e DISPLAY \
-e XDG_RUNTIME_DIR \
--net="host" \
hhivca/avatar_server \
1
```

This configuration was applied to the Kubernetes cluster, to ensure the correct deployment of the application and to perform the necessary tests.

3.4.4 Compatibility Test of Client Devices

In the testbed in Berlin, a set of testing devices were available to test the connectivity of the 5G network present in the testbed and the overall performance of the scenario. This set included mobile phones such as Samsung Galaxy S20, Samsung Galaxy S8, Oppo Find X2 Pro, Oppo Find X3 and Motorola Edge +.

In a first attempt, important difficulties were met when trying to use the 5G network, since none of these devices were able to connect. This presents a serious challenge, since only specific phones are compatible with the AR glasses that are being used (Xreal Light). Only one phone model belonging to Ericsson, the Oppo Find X2, could successfully connect to the network using a 5G SIM card provided by T-Systems, as specified in Section 3.3.2. Two of these devices were used to test the different execution modes of the use case.

Regarding the testbed in Surrey, the phones that were used as producer and consumer were the Samsung S23 and the Samsung S23 Ultra, respectively. The tests could be carried out without major issues.

3.4.5 Validation and Performance Evaluation

An initial validation of the deployments has been carried out successfully in both testbeds. The following configurations and functionalities have been tested:

Video encoding configurations

- *Software encoding*: for this test, the Gstreamer pipeline in charge of the processing, encoding and transmission of video and audio was configured to use the “x264enc” element. This uses software encoding to generate the video that is sent to the client.

- *Hardware encoding*: in this case, the faster and more advanced hardware encoding provided by the Nvidia GPUs available in the testbed was tested by using the element “nvh264enc” in the Gstreamer pipeline.

The use of these encoding configurations allows to carry out a variety of latency and throughput evaluations that are described later in this document.

Functionalities

Pre-configured speech script: as a first test, the capabilities of the animation module to process text was evaluated. A JSON script with predefined sentences was created and audio was recorded for them. This script contains the specific phonemes corresponding to each sound, as

Funded by the Horizon Europe Framework Programme of the European Union

well as information about their timings. Then, the avatar was animated using this script. The following lines show an example of this:

```
{
  "sentences":[
    {
      "audioFile":"scripts/story/S1",
      "s2tData":{
        "text":"hello",
        "start":0.000000,
        "end":0.990000,
        "result":[
          {
            "conf":1.000000,
            "end":0.560000,
            "phones":[
              {
                "end":0.560000,
                "phone":"SIL",
                "start":0.000000
              }
            ],
            "start":0.000000,
            "word":""
          },
          {
            "conf":1.000000,
            "end":0.990000,
            "phones":[
              {
                "end":0.810000,
                "phone":"HH_B",
                "start":0.660000
              },
              {
                "end":0.870000,
                "phone":"EH_I",
                "start":0.810000
              },
              {
                "end":0.930000,
                "phone":"L_I",
                "start":0.870000
              },
              {
                "end":0.990000,
                "phone":"OW_E",
                "start":0.930000
              }
            ],
            "start":0.660000,
            "word":"hello"
          }
        ]
      }
    }
  ]
}
```

Funded by the Horizon Europe Framework Programme of the European Union

Real-time audio capture: in this working mode, the full pipeline of the use case was tested. One of the participants, the so-called producer, uses one of the phones to send his/her voice to the server. Then, these audio samples are used to animate the face of the avatar. The second participant (consumer) receives audio and video of the animated avatar on a tablet/XR glasses.

Real-time video capture: as an additional mode, the possibility of capturing a 2D video signal and using it as the source of the animation for the face of the avatar has been added. In this mode, in which audio is also streamed to produce a full communication scenario, the facial features of the producer user are extracted from the video and transferred to the avatar. More information on this innovation can be found in D3.3.

Playback methods

2D video: Android phones such as the Oppo Find X3 and the Samsung S23 and tablets such as the Samsung Galaxy Tab S7+ have been used to act as consumer devices. This kind of devices provides full interactivity with the avatar and a 360° view of the model, although it cannot provide any 3D effect. Figure 61 shows this scenario being successfully tested in Berlin.

XR glasses: the use of the XReal Light glasses has also been tested successfully. In this case, the video of the avatar received by the consumer is stereoscopic. In this case, two images, one per eye, is shown, creating a more immersive 3D effect. On top of this, an OpenXR client for the Meta Quest3 glasses has been implemented and tested at the testbeds.

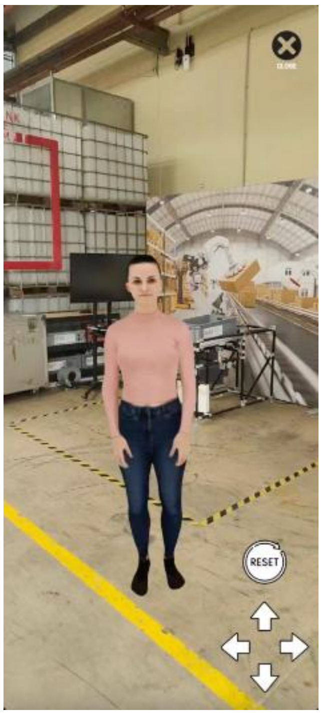


Figure 61: Avatar use case running in the berlin testbed

On top of these evaluations, as mentioned in section 2.4, cross-testbed tests have been carried out as well. They showed that the use case also offers a remote-access scenario in which the producer and consumer users are in different locations.

The integration and deployment work carried out in the testbeds allow the Open Call participants to use a variety of functionalities while, at the same time, enabling qualitative and quantitative evaluation of the performance.

One-to-many capabilities

The original configuration of the Avatar use case was based on a one-to-one communication. Throughout the course of SPIRIT, this setup has been reworked and extended to allow multiple consumer users to participate simultaneously. This opens a new set of applications for the use case, such as common lectures in educational environments or multi-user conferences.

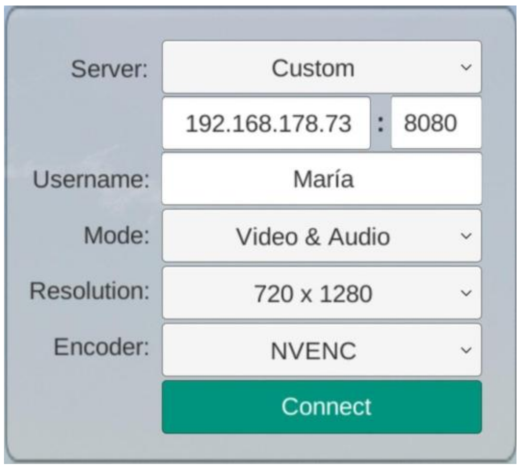
To enable the connection of multiple concurrent users, and to improve the general user experience, a series of changes and new functionalities have been implemented:

<u>Dynamic user and session creation</u>

The internal structure of the session handler has been modified to dynamically generate the resources associated to a user when a new one connects. This includes the creation of a WebSocket session, an individual Gstreamer pipeline and a virtual camera in Unity. Once the users disconnect, all these elements are safely removed.

<u>Customizable video streaming parameters</u>

The UI of the consumer application has been reworked to increase the customization of the communication for each user, as can be seen in Figure 62. It includes, in the first place, the selection of the resolution of the streamed video, so it can be individually chosen by each user depending on his/her device and network conditions. On top of this, the encoder to use (software or hardware with NVENC) can also be selected when initiating the communication. The Gstreamer pipeline will then be configured accordingly.



Server: Custom
192.168.178.73 : 8080
Username: María
Mode: Video & Audio
Resolution: 720 x 1280
Encoder: NVENC

Figure 62: Consumer Application UI

Funded by the Horizon Europe Framework Programme of the European Union

Customizable username

Each user can also select a username that will identify him/her in the communication.

Client placeholder icons

When multiple users are collaborating in the communication, new placeholder icons have been included to mark the position of each participant. These icons will move in real time with the users to which they belong, and they will show the username. An example of such icons can be seen in Figure 63.

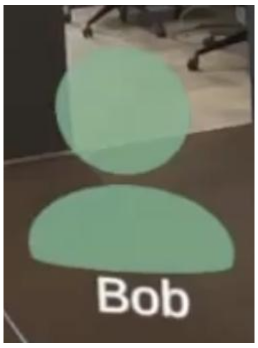


Figure 63: Client placeholder icon

Concurrency lock for the interaction with the avatar

The possibility to drag and drop the avatar and reposition it in the scene has been kept. However, a new lock has been implemented to guarantee that no race conditions appear and only one user can interact with the avatar at the same time. These actions will be then perceived by all users.

Dynamic avatar gaze direction

The gaze direction of the avatar is now adapted automatically to the relative position of the users in the scene. The avatar will look at the user that is closer to it.

Support for concurrent heterogeneous devices

The dynamic and independent nature of the streaming pipelines allow for the simultaneous use of different devices, being able to have 2D (phones/tablets/browsers) and 3D (VR/XR glasses) in the same communication session.

Volume control

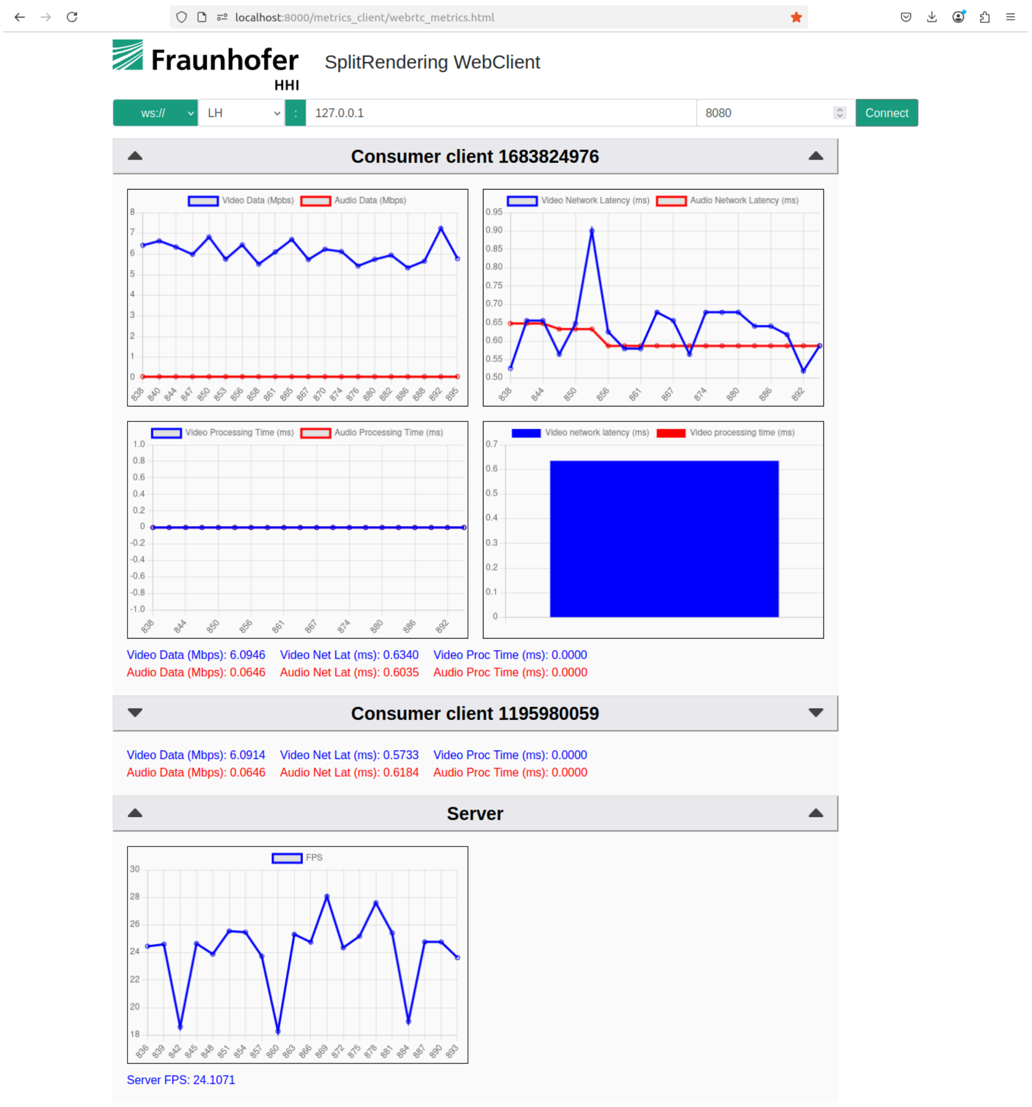
The audio of the avatar (generated in the server or captured by the producer user) is broadcast to all the active users. New volume controls have been added to the UI of all the client applications, including a button to mute/unmute the audio.

Multi-user metrics website

In order to extract performance indicators and quantitative results of the different configurations that the new setup allows, a new metrics HTML/Javascript website has been developed. This

Funded by the Horizon Europe Framework Programme of the European Union

website receives real-time data about the throughput, latency, jitter and FPS from the clients and the server. This information is updated constantly and adapted to the addition/removal of new users. Figure 64 shows an example of this new metrics website.


localhost:8000/metrics_client/webrtc_metrics.html
Fraunhofer HHI
SplitRendering WebClient
ws://
LH
127.0.0.1
8080
Connect
Consumer client 1683824976
Video Data (Mpbs)
Audio Data (Mbps)
Video Network Latency (ms)
Audio Network Latency (ms)
Video Processing Time (ms)
Audio Processing Time (ms)
Video network latency (ms)
Video processing time (ms)
Consumer client 1195980059
Server
FPS

Video Data (Mbps): 6.0946 Video Net Lat (ms): 0.6340 Video Proc Time (ms): 0.0000
Audio Data (Mbps): 0.0646 Audio Net Lat (ms): 0.6035 Audio Proc Time (ms): 0.0000

Video Data (Mbps): 6.0914 Video Net Lat (ms): 0.5733 Video Proc Time (ms): 0.0000
Audio Data (Mbps): 0.0646 Audio Net Lat (ms): 0.6184 Audio Proc Time (ms): 0.0000

Server FPS: 24.1071

Figure 64: Metrics website developed for the avatar use case

Metrics evaluation

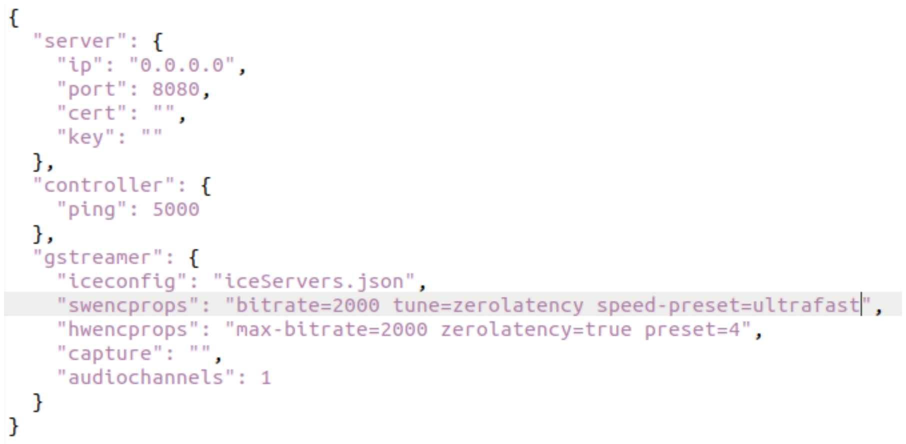
In the last stage of the project, a set of metrics and KPIs have been evaluated in order to analyse the performance of the implementation and to identify possible areas for improvement. For this, the already mention metrics client has been used. This website acts as a special client and retrieves data from the Gstreamer pipelines and the Unity application itself to show the evolution of different parameters such as the data throughput and the network latency. The information about the users is displayed dynamically as they connect, showing data about those (both consumers and producer) currently present in the scene. At the same time, the frame rate of the Unity application is also shown.

Funded by the Horizon Europe Framework Programme of the European Union

From this live data, it has been possible to extract information related to the performance of the pipeline and the variety of configuration options that it offers. The following metrics have been measured:

Data bandwidth

The use of GStreamer pipelines to stream audio and video offer great flexibility in the encoding and streaming process. When talking about the amount of data sent from the server to the clients, it is important that it can adapt to the network conditions and the requirements of the hardware involved. To achieve that, the properties of the pipeline can be customized when starting the server in a configuration file, as shown in Figure 65. Individual configurations for SW and HW encoders are included.



```
{
  "server": {
    "ip": "0.0.0.0",
    "port": 8080,
    "cert": "",
    "key": ""
  },
  "controller": {
    "ping": 5000
  },
  "gstreamer": {
    "iceconfig": "iceServers.json",
    "swencprops": "bitrate=2000 tune=zerolatency speed-preset=ultrafast",
    "hwencprops": "max-bitrate=2000 zerolatency=true preset=4",
    "capture": "",
    "audiochannels": 1
  }
}
```

Figure 65: Configuration file with encoding properties

The video bandwidth can therefore be chosen depending on the available resources. Figure 66 shows the evolution of the used bandwidth for two different configurations. On the other hand, Figure 67 shows the bandwidth used by the producer application when sending audio. It can be seen that, in this case, the amount of data sent by the application decreases to near to zero when audio is not being transmitted, making it more efficient.

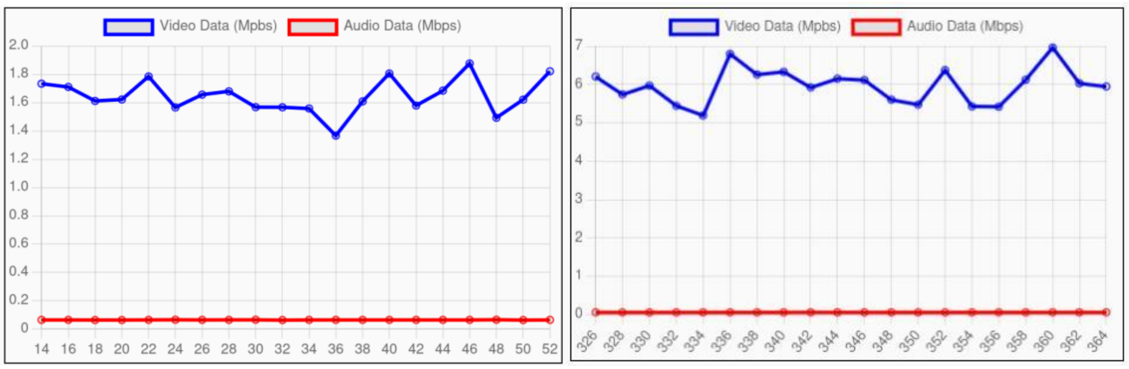


Figure 66: Evolution of the used data bandwidth for different configurations of the encoder (left: 1.5 Mbps, right: 6 mbps)

Funded by the Horizon Europe Framework Programme of the European Union

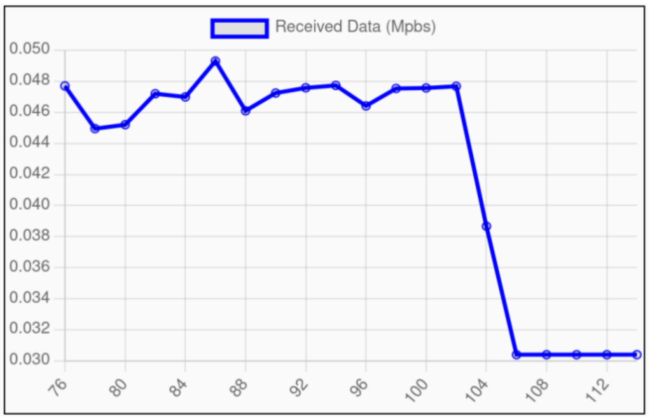


Figure 67: Evolution of the data bandwidth sent by the producer user

<u>Jitter</u>

The jitter is defined as the variation in the arrival time of data packets over a network connection. In the avatar use case, it has been measured in real time from the WebRTC statistics provided by GStreamer. Figure 68 shows an example.

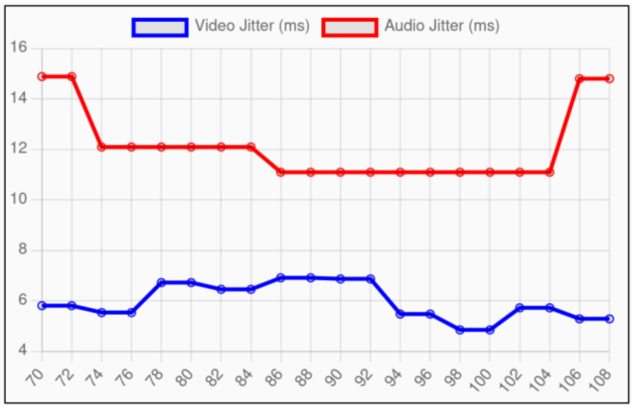


Figure 68: Evolution of the audio and video jitter

<u>Latency</u>

In a complex system like the one corresponding to the Avatar use case, there are several latencies that can be measured, depending on the scope of the analysis. The following have been defined as the most important ones:

- Network latency

  This metric is provided by the GStreamer pipeline (at the webrtcbin element) and is defined as the time that a packet needs to be transmitted from the sender to the receiver. In this case, it has been measured as half of the round-trip time that the WebRTC library states. It is included in the metrics client as shown in Figure 69.

Funded by the Horizon Europe Framework Programme of the European Union

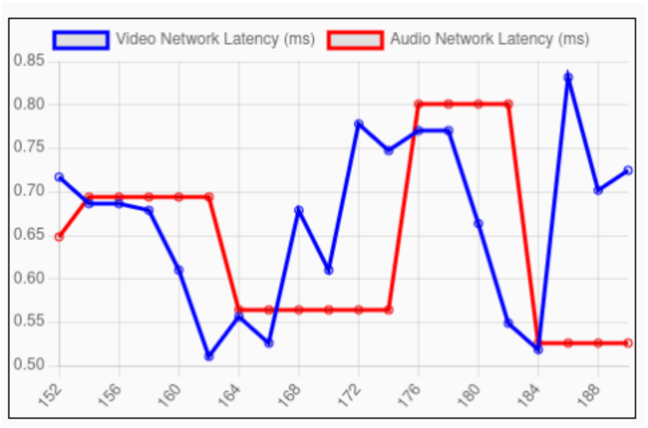


Figure 69: Evolution of the audio and video network latencies

- Interaction latency

The interaction latency is a metric that appears as a result of the Split Rendering approach used in the Avatar application. It has been defined as the time that passes since a user interacts with the avatar (by modifying its position) until the change is visible on the screen. This includes informing the server of the action, the adaptation of the virtual camera, the rendering of the view from the new position and the modification of the viewpoint in the 3D space managed by the consumer application.

This latency is crucial for the functioning of the Split Rendering mechanism, as well as for the perception of the avatar by the user as a 3D object and not as a 2D texture. It has been measured directly on the client device.

- End-to-end latency

The end-to-end latency has been defined, in relation to the Avatar use case, as the time that it takes from an audio sample that has been generated by the producer user to be played at the consumer application. This includes its generation, the streaming to the server, the animation of the avatar, the rendering in Unity, the encoding of the video and audio, the streaming towards the consumer user and the decoding and rendering at the consumer application.

To measure this metric, a reference sound (a clean artificial tone produced by a sound tool) has been used. This sound, produced on an external computer with speakers, has been used as input to the producer application. After a moment, the same sound has been played at the consumer application. A third device, a phone, has been used to record the soundwave, that has been later analysed with the help of the open-source program Audacity. An example of the result appears in Figure 70. To eliminate random errors associated to this process, the same operation has been performed a number of times (at least 10) and an average value of the end-to-end latency has been obtained.

Funded by the Horizon Europe Framework Programme of the European Union

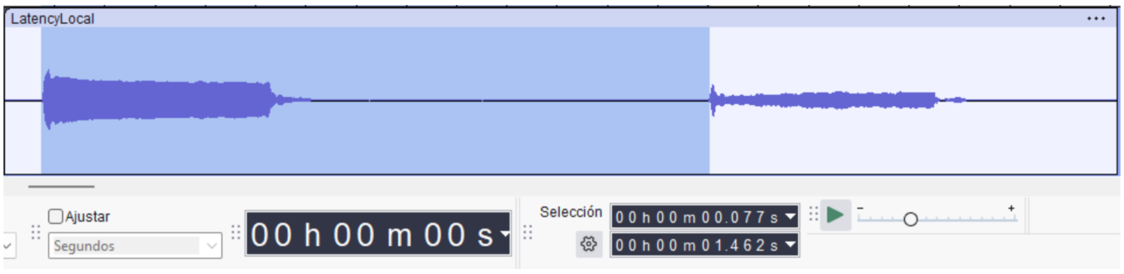


Figure 70: Sound wave analysed to study the end-to-end latency

<u>Framerate</u>

One of the crucial metrics when using telepresence systems is the framerate of the video signal that the consumer user receives. This can be measured in both the server and the client applications, since the two of them perform rendering actions. However, the Split Rendering approach offloads a big part of this process to the server. Because of this, the FPS provided by Unity have also been included in the metrics website, as shown in Figure 71.

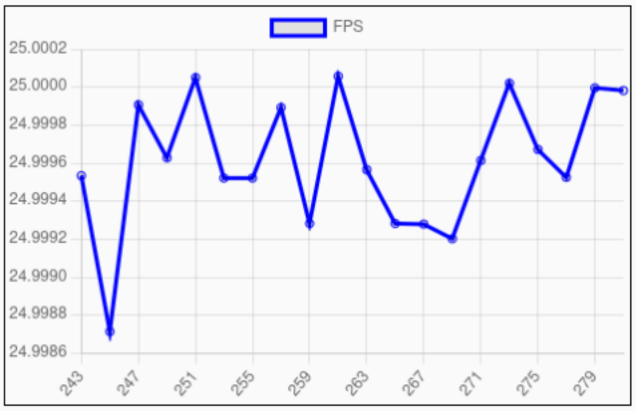


Figure 71: Evolution of the framerate of the server

After performing a set of tests in a local environment and using the network available at the testbed in Berlin, the values gathered in Table 6 show the performance of the Avatar use case.

Funded by the Horizon Europe Framework Programme of the European Union

Table 6: Values of the metrics analyzed for the avatar use case

Metric	Measured value
Video bandwidth consumer (Mbps)	Variable (encoding max bitrate selectable). Between 1.5 and 4 for acceptable quality
Audio bandwidth consumer (Mbps)	0.06
Audio bandwidth producer (Mbps)	0.045
Video jitter (ms)	6
Audio jitter (ms)	12
Video network latency (ms)	0.6
Audio network latency (ms)	0.6
Interaction latency (ms)	79
End to end latency (s)	1.3
Number of simultaneous consumer users	> 5

During the tests, it has been proved that at least 5 simultaneous consumer users can be connected at the same time when using resolutions up to 1080p, which already provides good quality for the display of the avatar.

The interaction latency presents values of around 79 ms, providing a smooth user experience and prevents the avatar from losing its 3D appearance.

The biggest improvement spot appears at the end-to-end latency, which, at the writing of this document, has values of over one second. The main reason for this is the necessary delay that the face animation library requires, since it uses a window of previous audio samples to generate the mesh and texture of each frame. This fact, however, does not seem to affect the overall experience, since the use cases contemplated in these tests (unidirectional communication with one producer user and multiple consumers) do not rely on low end-to-end latency, but on low interaction latency.

Funded by the Horizon Europe Framework Programme of the European Union

3.4.6 Summary

The deployment of the application corresponding to the Real-Time Animation and Streaming of Realistic Avatars use case is contained completely within the infrastructure of the available testbeds. A list of the agents that participate in the system as well as the network components used at the testbed can be found in Table 7.

Table 7: Use Case: real-time animation and streaming of realistic avatars

	Real-Time Animation and Streaming of Realistic Avatars
Users	Producer: a user whose voice is captured by the media capture device and streamed over the network to the server, where it will be processed and used as input to animate the avatar. Consumer: a user that receives audio and video from the server that is displayed on the client device. This user can interact with the avatar and modify its position, scale and rotation.
Network	RTC P2P (WebRTC)
Network Access	Wireless, e.g., Wi-Fi / 5G
Components	Edge Cloud System (Linux 20.04 LTS): generation of the animated avatar, rendering, streaming, Websockets/WebRTC connection management. Media capture device: Android mobile device that captures media (audio) and streams it to the server through the network. Consumer device: Android mobile device that displays audio and video received from the server and can interact with the avatar. AR glasses (Xreal Light or Meta Quest 3): Augmented Reality glasses that displays audio and video received from the server and can interact with the avatar.

The extensive compatibility of the system with a variety of devices, especially Android mobile phones or tablets, reduces the number of obstacles encountered when trying to connect a media capture/consumer device. The number of mobile phones that are compatible with the testbeds has now been extended, which facilitates the access to the applications by potential users. On the other hand, only a few mobile phones are compatible with the investigated AR glasses, so a potential user must convey with this information when using the application. Because of this limitation, a new client working on the Meta Quest 3 headset has been implemented.

Funded by the Horizon Europe Framework Programme of the European Union

4 IMPLEMENTED USE CASES

This section covers the implementation of use cases, which are described in detail in D2.3 [2]. On the one hand, the integration of the platform components into the use cases and the description of the necessary adjustments for a smooth integration into the testbeds and, on the other hand, a look ahead to how these components could be used in various scenarios, such as third-party applications. For each use case, there is a table at the end of the respective section that summarises the relationship to the requirements and KPIs from the document D2.3 [2], together with the validations carried out so far.

4.1 UC HOLOGRAM: HOLOGRAPHIC HUMAN-TO-HUMAN COMMUNICATIONS

This section presents the application pipeline and possible extensions of the holographic human-to-human communication use case implemented into the network testbed representing one part of the third version of the SPIRIT platform. The use case can be extended within the SPIRIT project to offer a wide compatibility with applications from third parties.

4.1.1 Overview

Figure 72 presents the implemented holographic human-to-human use case using the application platform:

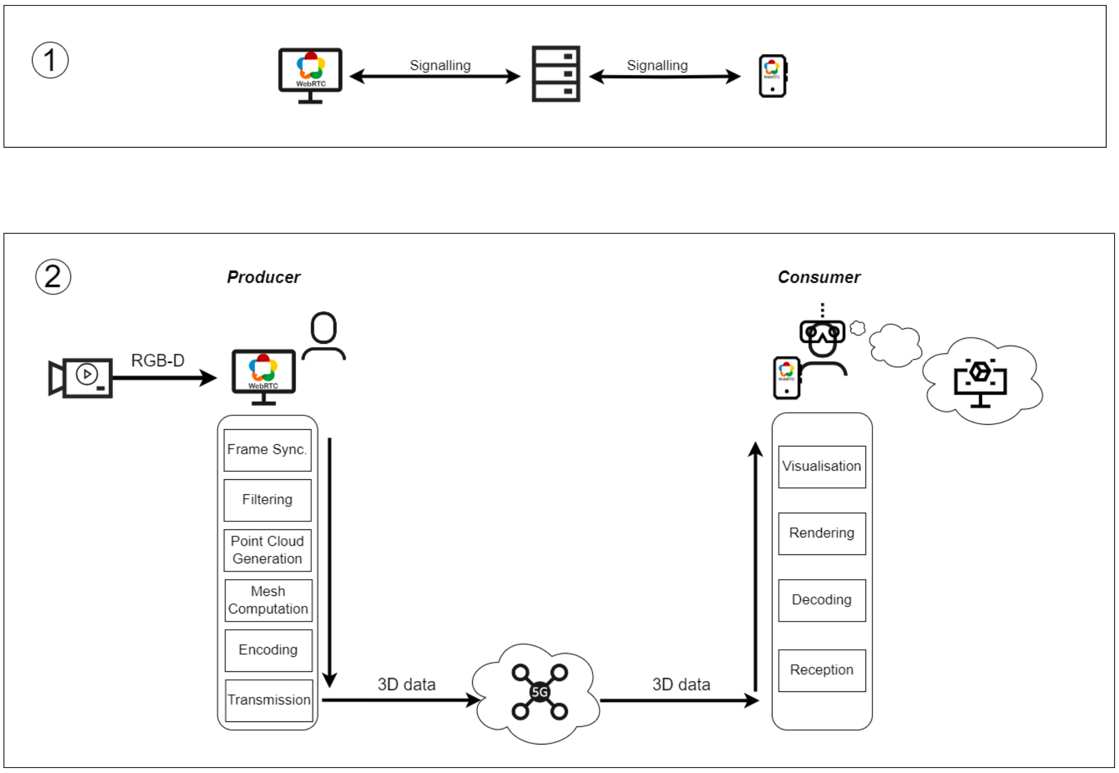


Figure 72: Application Pipeline of the Holographic Human-to-human communication

Funded by the Horizon Europe Framework Programme of the European Union

The use case is implemented by means of two applications, i.e., a producer and a consumer application:

➲ **Producer application:** Containerised Python-based application hosted on a high-performance PC connected to a depth camera.

1. Upon starting the application, the HPC connects to the signalling server.
2. Once the consumer device joins the signalling server, a WebRTC channel is set up as described in D3.3 [3].
3. Producer application initiates the acquisition of RGB-D information data of an object of interest (human torso/face) using a depth camera.
4. Producer application generates human 3D representations using the acquired RGB-D information.
5. Encoded human 3D representations are streamed to the consumer application using a 5G network.

➲ **Consumer application:** Unity (C#)-based application running on an Android mobile phone connected to AR glasses via USB-C.

1. Upon starting the consumer application, the mobile phone connects to the signalling server.
2. Once, the producer device joins the signalling server, a WebRTC channel is set up as described in D3.3 [3].
3. Consumer application receives the 3D data stream through a 5G network.
4. The 3D data stream is decoded and rendered.
5. Human 3D representation of the user on the producer side is displayed on AR glasses in real time.

4.1.2 Extension of Use Case

The implemented use case offers real-time holographic streaming delivery with the flexibility of manual configuration for both quality and data size of holographic content. This includes the ability to achieve a 3D representation through filtering and compression techniques that can be tailored to meet network and Quality of Experience (QoE) requirements. Here, a producer client application functions as a versatile tool for media and spatial information acquisition and processing, compatible with applications supporting either network or USB connections. On the other end, the consumer client application serves as a decoding and rendering component. It allows users to visualize encoded 3D data transmitted from the producer client application. Both application parts are adaptable to wireless network access technologies such as Wi-Fi or 5G, providing a seamless and efficient holographic communication experience.

For the SPIRIT project the goal is to extend this implemented use case by leveraging immersive telepresence application provided by third parties in the scope of two Open Call waves.

General

- **Research on 3D data processing**
  External parties are given the opportunity to utilise the use case to drive development in the processing of 3D content. This includes researching and testing novel processing methods in the areas of point cloud generation and mesh calculation as well as encoding/compression algorithms. Assessments by third parties can be made on various aspects such as encoded data size, processing time, image quality, or overall QoE. In corporation with University of Klagenfurt, the QoE of the holographic communication platform has been calculated using QoE models trained on third-party datasets incorporating knowledge of 3D compression parameters. The results of the QoE evaluation can be found in section 6**.**

- **Implementation of various streaming technologies**
  The existing iteration of the application employs WebRTC for streaming 3D data between the clients asymmetrically. There have been developments in extending these technologies, e.g., in terms of scalability (c.f. Section 3.3.8). This offers a promising chance to explore the system's performance. In addition, alternative streaming frameworks such as LL-DASH have been explored and compared to WebRTC solutions.

- **Use of new devices and new ways of interaction**
  The market of AR devices evolves rapidly and new options with improved features are constantly made available. This presents a good chance to extend the compatibility of the application to a wider range of equipment. New ways of interaction between the user and the holographic content could be implemented using new device technologies to provide a richer immersive experience. This for example may include the incorporation of the Microsoft HoloLens 2 or the Apple Vision Pro through third parties.

Based on the feedback of the SPIRIT partners and the Open Calls, the following optional extensions were implemented for the holographic communication use case.

- Point Clouds: An option has been added to display the human 3D model as a point cloud instead of a polygon mesh. In general, the underlying 3D compression technique used for holographic communication works faster with point clouds than with polygon meshes. However, as described in D3.3 [3], using point clouds instead of meshes usually comes at the expense of immersion quality. It is up to the user/experimenter to decide which option to use.

- Built-In Congestion Control: The holographic communication architecture, or more precisely the underlying WebRTC component, has been extended to support media tracks for streaming content in addition to the more generic data channel of earlier holographic communication versions. Unlike the data channel, the WebRTC media track is more specialised for traditional media such as 2D video and has built-in support for 2D compression codecs such as VP8 and H264. It also features Google Congestion Control (GCC). These characteristics enable efficient streaming in congested networks. For holographic communication use cases, the WebRTC media track option streams synchronised depth and colour images via the media track. The receiver has been modified to generate 3D content from the received colour and depth images for an immersive experience. The user/experimenter can choose which option to use.

Funded by the Horizon Europe Framework Programme of the European Union

- Audio Support: The holographic communication use case was extended to also support real-time streaming of audio. At the receiver side the audio can be used in parallel with the real-time hologram presentation. The streaming of audio is optional and can be turned on or off depending on the needs.

- Support for other protocol suites: While WebRTC is the de facto transport protocol in this use case for holographic human-to-human communication, other protocol suites have been added in collaboration with Open Call participants to enrich the SPIRIT offering towards novel usage scenarios:

    - Together with the OPEN-DASH-PC project, an open-source implementation of ultra-low latency dynamic adaptive streaming over HTTP (ULL-DASH) for point clouds is now supported in both the Surrey and Ghent testbeds.
    - In cooperation with the QUEST project, scalable real-time holographic streaming through Media-over-QUIC (MoQ) enables many-to-many scenarios through a relay-based mesh network, instantiated and publicly accessible on the Virtual Wall testbed in Ghent.

Furthermore, in the third version of the SPIRIT platform, the following extensions are conceivable but should not be understood as limiting the immersive telepresence application of third parties:

- *Incorporating edge computing (split-rendering)*: The rendering operation is done by the consumer application supposed to be running on a mobile end device such as a commercial off-the-shelf mobile phone. However, rendering is GPU- intensive, as the processes involved in creating realistic and visually appealing images or animations are very complicated. The complexity arises from factors such as intricate scene geometry, high-resolution textures, advanced lighting effects and complicated shading models. Meticulous calculations for light interactions, reflections, refractions, and shadows are required to achieve photorealistic results as demanded by the digital human-to-human use cases presented in D2.3 [2]. One solution to mitigate the impact of rendering on hardware with low GPU power is to (partially) off-load the rendering operation on more sophisticated hardware, such as the HPC provided in the implemented use case.

  Similar to [7], third parties could investigate cloud solutions leveraging the provided HPC for the use case with a special focus on rendering within immersive telepresence scenarios. Excluding cloud service monetisation, computational off-loading of heavy software operations from a mobile end device to an HPC not only optimises end consumer battery usage but also contributes to possibly reducing end-to-end latency in the overall system originating from faster software processing time.

Funded by the Horizon Europe Framework Programme of the European Union

4.1.3 Summary Validation

Table 8: Use case Holograms: development/integration and performance

Use case Holographic Communication development and integration	Relevant requirements	KPIs (metrics achieved)	Evaluation (determination of KPIs)

The use-case was developed by EDD and initially tested locally at 5G testbed in Aachen.
 	
Integration in the DT testbed is done.
 	
Integration in the UoS testbed is done.
 	
Integration in the UoB testbed is done.
 	
The extension for many-to-many conferencing was developed by imec, tested in a local setup and integrated in the Virtual Wall testbed in Ghent.
	RLat = 200 ms
	
	RDown = 20 Mbps
	
	RClients = 2
	
	RFPS = 30
	
	RRes = 1280x720
	
	RQoE = 4
		RLat = 300 ms
		
		RDown = 10 Mbps
		
		Clients = 2 (extension to 10 clients in the many-to-many scenario)
		
		RFPS = 60
		
		RRes = 848x480
		
		RQoE = 4
			E2E latency (Lat) is in this case the time between the capturing and displaying on AR glasses.
			
			Downlink bandwidth corresponds to the 3D video stream (compressed mesh).
			
			In our tests satisfactory results were achieved with a resolution of 848x480@60FPS.
			
			Two clients were used in the tests, one producer and one receiver (up to 10 clients in the many-to-many scenario on the Virtual Wall testbed).
			
			A signalling server included in the application was used to test the communication.

Funded by the Horizon Europe Framework Programme of the European Union

4.2 UC MULTI-SOURCE: LIVE TELEPORTATION WITH 5G MEC

4.2.1 Overview

This use case is about live teleporting people from remote internet locations to a common virtual space of the audience such that the audience can have the immersive and multisensory perception that everyone is located in the common physical scene. One application scenario is distributed virtual performances where actors can physically perform (e.g. dancing) at different locations, but their live holograms can be simultaneously teleported to a "virtual stage" where the audience can enjoy the entire performance event constituting the virtual holograms of real performers from different remote locations.

4.2.2 Use of the System within the Surrey/Bristol Testbeds

The overall application platform has been implemented based on open-source platform of LiveScan3D [38]. At content source side (e.g. person to be captured and teleported), multiple Kinect Azure DK cameras are used to capture the object from different directions, and each camera is connected to a local PC called clients which is responsible for local processing of the raw content data. The raw content data is in point cloud data format.

At the 5G Multi-access Edge computing (MEC) server side, the pre-processed local data are further streamed to the production server responsible for integrating frames produced from different clients for the same captured object. The functionality of the production server is supported by a 5G MEC attached to the 5G testbed network at University of Surrey. In this case local frames from individual clients are streamed in real-time to the local 5G MEC through 5G new radio (NR) uplink. A similar approach was applied at the Bristol testbed.

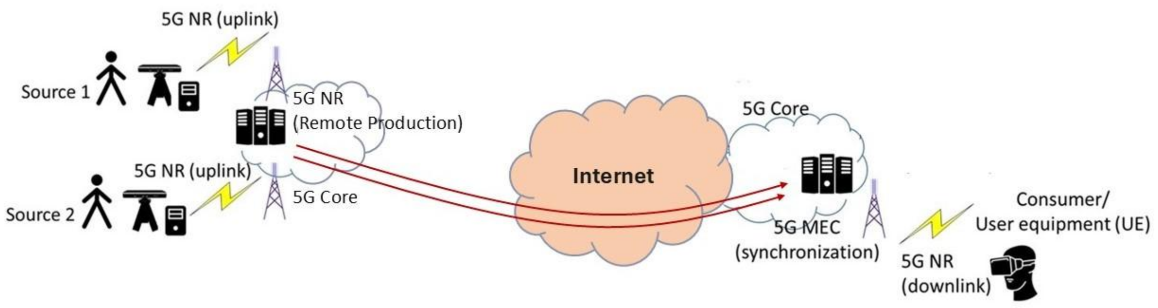


Figure 73: Implemented framework for live teleportation

The use of this platform can be described as follows:

1) **Start LiveScan3D server at MEC server**: on the MEC server, first the LiveScan3D server should be started. Then at the control panel, calibration parameters including offset for each source can be tuned as well as point density, request interval, buffering control parameters.

2) **Start LiveScan3D clients at different sites**: once one or multiple Microsoft Azure Kinect DK cameras can be deployed on different sites and connected to local laptop, LiveScan3D client can be launched at each laptop. Local resolution level and depth

Funded by the Horizon Europe Framework Programme of the European Union

mode can be selected on the application panel and local source ID should be configured. Especially, the source ID should be predefined to avoid collision with other clients.

3) **Connect clients to MEC server and streaming Live teleportation**: The client can set the remote MEC server IP as the server IP to be connected, and once the connect button is clicked, a TCP connection will be requested and then established from client to MEC server. Then the live hologram can be requested from remote MEC server. On the MEC server panel, by click “show live” button, real-time hologram from multiple sites can be merged and displayed on the screen.

4) **Start Hololens 2 and receive the holographic streaming:** once the live teleportation from clients to server is running, the user can start the HoloLens 2 to connect to the same MEC server via TCP connection to request live teleportation streaming. Each HoloLens 2 will have dedicated user ID, then the server can decide how many sources should be streamed to this specific user. During the streaming, a user can perform several actions to interact with the hologram including moving closer or far away, manipulating the hologram to change its size, orientation.

4.2.3 Extension of the Use Case

The use case facilitates the realisation of live multisource teleportation, allowing for the delivery of photorealistic holographic content to viewers. The implementation and use of the key elements (producer/source, server, receiver) shown in the architecture, coupled with the employment of 5G connectivity, ensures that the network requirements are met while also considering the viewer experience. Building on this, and factoring the key elements of the architecture, there are options for developing and extending the use case and its functionality. This is detailed below.

Many-to-many extension

The architecture shown in Figure 73 illustrates the multi-source use case. Although implemented, the diagram displays only 2 sources. A possible extension of the current use case functionality therefore includes the potential support for a many-to-many scenario. In this context, the corresponding functionality could be used to support the introduction of additional elements such as more producers/sources and additional receivers. The introduction of such elements would enable the realisation of a many-to-many scenario while laying the foundation for further extensions. Figure 74 shows an outline of the potential extension to enable a many-to-many scenario.

From the Figure 74, the potential extension of the use case displays the support for additional sources and receivers. In this regard, the computing capabilities of the testbed coupled with the 5G network could be leveraged to meet the demands for an increase in producers/sources and receivers, allowing for the satisfaction of requirements, and as such facilitating a many-to-many multi-source use case scenario.

Funded by the Horizon Europe Framework Programme of the European Union

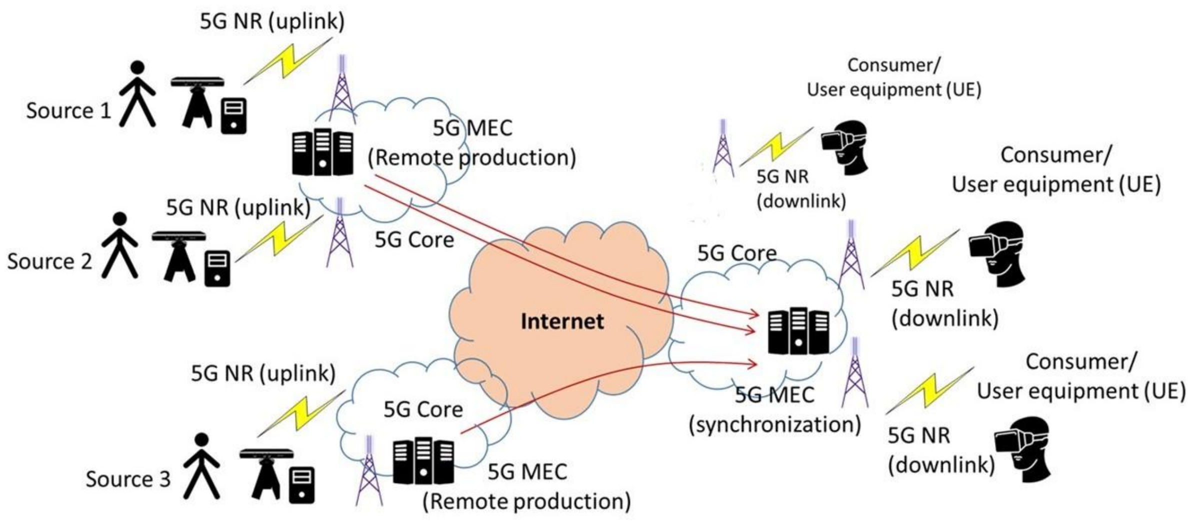


Figure 74: Extended framework for Live Teleportation

Multi-dimensional Adaptation

The use case allows photorealistic holographic content to be viewed in a 3D space. This opens up the possibility for uncertainties associated with the behaviour of the source, the viewer and, in some cases, the network. Such uncertainties could be of concern within the context of viewer satisfaction. As such, a further extension of the use case could be the potential introduction of the multi-dimensional media adaptation detailed in D3.3 [3], to account for such uncertainties and facilitate satisfactory experiences.

Utilisation of different client devices

Growing interest in mixed reality has led to further variability in the consumer market. Therefore, another potential extension would be to consider the use of different devices such as holographic displays and other head mounted displays.

Further source image considerations

The use case facilitates multi-source teleportation via the use of camera sensors as shown in the Figure 73. Based on this, a possible extension of the use case could also include additional considerations regarding the source and source image. In this context, a possible extension could be in the introduction of additional camera sensors per source, allowing denser representations. Furthermore, the variability of point densities particularly with regards to different sources could further be explored, representing the potential for further improvements.

Funded by the Horizon Europe Framework Programme of the European Union

4.2.4 Summary Validation

Table 9: Use case Multi-source: development/integration and performance

Use case Multi-Source development and integration	Relevant requirements	KPIs (metrics achieved)	Evaluation (determination of KPIs)

The use case was developed by University of Surrey and integrated into Surrey's/Bristol's testbed. The current UC status is tested under the Surrey's 5G network testbed.
 	
Each source consists of a unique client which encompasses a Microsoft Azure Kinect Dk/Kinect-2, and corresponding computing resource, utilized by the source client application for the generation and transmission of the relevant point cloud data to the server.
 	
The primary viewer client device utilized at the receiver is the Hololens 2.
 	
Major challenges in integrating the UC were in relation to computing resources and network configurations. This indicates that performances could be impacted to some degree by such factors. Also, integration and development are not inclusive of the WebRTC protocol. ongoing actionable item.
	RLat=200ms
	
	RDown=50Mbps
	
	RUp=50Mbps
	
	RClients=4
	
	RQoE=4
	
	RFPS=30
	
	RRes= ≤ 1280 x720(512x288), 1920x1080(512 x288)
	
	RSyn=50ms
	
	ROutDev= Hololens 2
		Lat = 300 ms
		
		RDown = 50-70Mbps
		
		RUp = 50-70Mbps
		
		RClients = 3 (planned extension to 4 clients in the many to many scenario)
		
		RQoE = ~3-3.7
		
		RFPS=25-30
		
		RRes= ≤ 1280 x720(512x288)
		
		RSyn=50-100ms
		
		ROutDev=Hololens 2, Proto M Holographic Display.
			RLat is determined by the processing capability of the server running LiveScan3D platform as well as the network conditions (e.g. network delay). This can be up to 300ms in challenging network conditions.
			
			The 3 clients used in the tests represent 2 unique producers/sources and a receiver. (The planned extension could feature the introduction of an additional source/producer as well as an additional receiver).
			
			The QoE metric incorporates subjective driven model-based QoE evaluation for volumetric media.
			
			Downlink and uplink bandwidth (Down and Up) in the 5G network of up to 70Mbps respectively, as measured in separate experiments.
			
			Regarding synchronization performance, this also depends on the network distance from individual sources to the viewer side.
			
			FPS is within the 25-30 range and corresponds to the resolution level at ≤ 1280 x720(512x288) resolution which is indicative of the relative point density. The resolution level at 1920x1080 (512x288) is also supported with FPS between 18-30.
			
			The use case has been tested with 2 consumer/user equipment. These are the Hololens 2 and the Proto M display.

Funded by the Horizon Europe Framework Programme of the European Union

4.3 UC AVATAR: REAL-TIME ANIMATION AND STREAMING OF REALISTIC AVATARS

4.3.1 Overview

The Real-Time Avatar Animation and Streaming application proposes a communication scenario between a producer user, whose voice is captured and used as input to animate a volumetric avatar, and one or more consumer users, that receive video and audio information of the animated avatar and have the possibility of interacting with it. Once again, the reader is invited to refer to the documents D2.3 [2] and D3.3 [3] for a more detailed explanation.

This section describes the steps needed to run and test the application platform within the testbed provided by T-Systems in Berlin, as well as a list of possible extensions of the application that could be carried out by third parties.

4.3.2 Use of the System within the Berlin Testbed

A step-by-step description of how the application can be used within the Berlin testbed is described here. At the time of writing, and as mentioned in previous sections, there are still several issues that prevent the system from being fully integrated in the infrastructure provided by T-Systems.

The use of the Real-Time Animation and Streaming of Realistic Avatars application involves the following steps:

1. **Start audio/video animation server**: the container associated to the audio/video animation servers must be running before starting the main container. These servers receive a stream of audio or video samples and generate the visemes (speech sound descriptions) used to animate the avatar in a further step.
2. **Start rendering and streaming server**: the main container must be started by selecting the necessary parameters. These include, among other options, the working mode (single image or stereoscopic), and the desired resolution of the rendered image.
3. **Connect media capture device to server**: the producer user must start the media capture Android application, enter the IP of the server within the network and connect using Websockets.
4. **Connect consumer devices to server**: the consumer devices can consist of a single Android mobile phone or tablet or a set of AR glasses. In both cases, the user can interact with the avatar.
    a. Single Android device: the user must start the corresponding Unity application, enter the IP of the server and connect using Websockets. The image of the avatar will appear directly on the screen, and the audio will be played on the device.
    b. AR glasses: the user must connect the glasses to the corresponding Android phone, start the Unity application, enter the IP of the server and connect to it. The image of the avatar will be displayed on the glasses, and the audio will also be played on them.

Funded by the Horizon Europe Framework Programme of the European Union

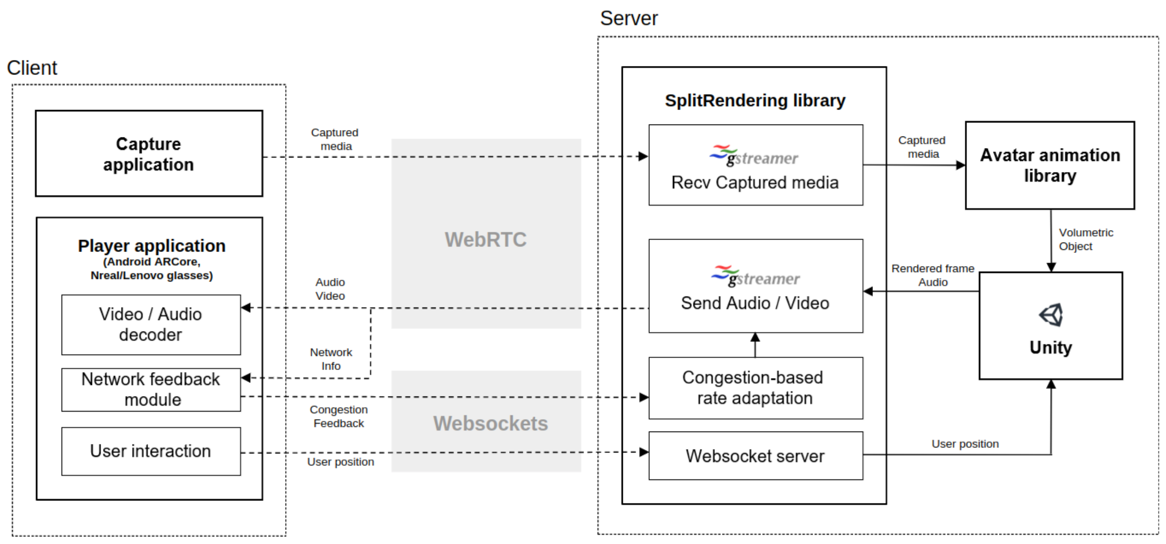


Figure 75: Components of the real-time animation and streaming of realistic avatars application

On the other hand, the flow of data of the whole scenario can be described in the following steps (as shown in Figure 75):

1. A certain type of media (audio or video) is captured on the producer device.
2. The media samples are streamed over the network to the server using a WebRTC connection.
3. A set of media samples (corresponding to a frame) is used as input for the avatar animation library.
4. The avatar animation library sends the media samples to the audio animation server and visemes are generated.
5. The produced visemes are used to generate the mesh and texture of the avatar for the current frame.
6. The mesh and texture of the avatar are updated within the Unity scene.
7. The position of the client device with respect to the avatar is synchronized on the server and the cameras in the Unity scene are placed at the right position.
8. The transformations resulting of the interaction of the consumer user with the avatar (position, rotation, scale) are applied at the server.
9. One or two 2D images (depending on the working mode) are rendered by the corresponding cameras.
10. The 2D images are streamed over the network to the consumer device, together with the audio corresponding to the current frame.
11. Both audio and video are displayed on the consumer devices.

Funded by the Horizon Europe Framework Programme of the European Union

4.3.3 Extension of the Use Case

The architecture of the application makes it possible to extend its functionalities in a variety of ways within the development of the SPIRIT project. Each of the main components (server application, producer client and consumer client) provide a series of subsystems that can be further developed to improve the already-present functionalities or to add new ones. This section includes some entry points for third parties to do this.

Use of different avatars or animation methods

The avatar included in the application platform can be switched to a new one with different characteristics. The architecture of the system allows any object placed in a Unity scene to be rendered and shown on the client producer device. This opens up a number of possibilities regarding the aspect of the avatar and the way it is animated frame by frame. New animation algorithms could be used as libraries within the existing infrastructure.

Manipulation of the rendered images and audio

The nature of the rendering procedure used in this application makes it possible to manipulate the generated 2D image in an infinite number of ways. New techniques based on the use of AI (Artificial Intelligence) and Neural Networks can be used to improve the quality of the renders or to modify its aspect.

The same can be applied to the audio captured on the producer client device. The samples can be processed in different ways, providing better sound quality, noise removal or voice manipulation.

Use of different streaming technologies

The current version of the application uses Gstreamer and WebRTC to stream audio and video information from the server to the client and vice versa. New extensions of these technologies are being developed at the moment. This presents a good opportunity to investigate how the system behaves when being integrated with other streaming frameworks.

Use of new client devices and new ways of interaction

The market of AR devices evolves rapidly and new options with improved features are constantly being made available. This presents a good chance to extend the compatibility of the application to a wider range of equipment. New ways of interaction between the user and the avatar could be implemented as well in order to provide a richer experience.

Measure of Quality of Experience and Performance

The existing capabilities of the system, together with the possible new additions described above, provide an interesting playfield to analyse and compare the behaviour of different rendering and streaming technologies in the context of the communication in Mixed Reality environments.

Some of the experiments carried out by the participants of the Open Calls of SPIRIT have addressed several of these aspects of the Avatar use case, with the use of other 3D objects together with the Split Rendering module, improvements in the rendering and streaming pipeline and analysis of the overall quality of experience of this application.

Funded by the Horizon Europe
Framework Programme of the European Union

4.3.4 Summary Validation

Table 10: Use case Avatar: development/integration and performance

Use case Avatar development and integration	Relevant requirements	KPIs (metrics achieved)	Evaluation (determination of KPIs)

This use case was developed by Fraunhofer HHI and integrated in the Berlin DT server in collaboration with T-Systems, as well and in the Surrey and Bristol testbeds.
 	
The necessary applications were containerized and deployed in the Kubernetes cluster of the testbed. Then, the main functionality (animation and streaming of the avatar) was tested.
 	
The main difficulties found during the integration process were related to the compatibility of the client devices with the 5G network. One of the Android smartphones (Oppo Find X3) could finally be connected to the testbed in Berlin and used for the tests with two different configurations: mono video shown directly on the phone and a stereoscopic mode using the XReal Light and Meta Quest 3 glasses. In Surrey and Bristol, the application was deployed in a virtual machine and tested with Samsung S24 phones.
	RLat = 200 ms
	
	Rup > 5 Mbps
	
	RDown > 5 Mbps (mono), 10 Mbps (stereo)
	
	RClients = 3
	
	RFPS = 25
	
	RRes = 1920x1080
	
	RInDev = Android smartphone/tablet
	
	ROutDev = Android smartphone/tablet, XR glasses
	
	RQoE=4
		RLat ~= 300 ms in Scripted mode, 1.3 s in Audio2Animation and Video2Animation
		
		Rup ~= 0.1 Mbps for audio, 0.5 Mbps for video
		
		RDown ~= 1.5-4 Mbps for mono, 3-8 Mbps for stereo (customizable and adaptable)
		
		RClients > 6 (tested with 5 consumers and 1 producer)
		
		RFPS = 25
		
		RRes = 1080x1920 with up to 5 consumer users. 2560x1440 with less than 5 users.
		
		RInDev = Android smartphones/tablets
		
		ROutDev = Android smartphones/tablet, XReal Light/Meta Quest 3 glasses.
		
		RQoE ~= 4
			E2E latency (RLat) is, in the Audio2Animation and Video2Animation cases, the time between the moment in which the producer user sends audio from the phone and the instant that the receiving user hears the audio. The animation procedure for the avatar can be optimized to reduce this latency.
			
			The value of Rup corresponds to audio or video.
			
			Downlink bandwidth corresponds to the video and audio information. The mono mode was tested, and the bandwidth was as expected.
			
			Up to six clients were used in the tests, one producer and five receivers.
			
			Android smartphones have been tested as input devices at the testbeds.
			
			Android smartphones and XR glasses (XReal Light and Meta Quest 3) have been tested as output devices at the testbeds.
			
			The QoE value has been approximated from the results of the evaluation performed in section 6.
			
			The achieved framerate is limited by the current procedure used to animate the avatar.

Funded by the Horizon Europe Framework Programme of the European Union

4.4 REAL-TIME HUMAN-MACHINE INTERACTIONS

4.4.1 UC Robot: Distributed Steering of Autonomous Mobile Robots (AMRs)

Alongside the SPIRIT Platform the Deutsche Telekom Testbed provides a teleoperation use case that is available on premise. The use case is about the remote manual navigation of autonomous mobile robots and is further described in the D2.3 [2] document.

The use case runs centralized on a local edge server and can be used by any device (for example phone, tablet, controller and computer) with a web browser (preferably chrome) inside the campus. Within the web application one can view the robot’s surroundings by observing video streams from the AMR as well as sending out drive commands to the robot. In general, the web application is capable of manually and autonomously steering multiple different AMR’s however only one robot was made available for the project, the “Husky” Robot, see Figure 76: Husky Robot from ClearPath Robotics.

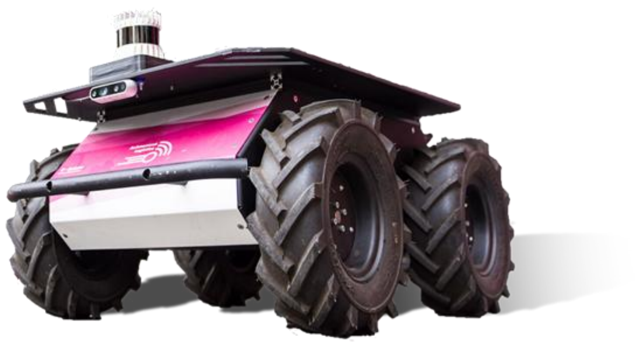


Figure 76: Husky Robot

In order to enable the robot for autonomous driving and distributed steering, it had to be equipped with multiple different sensors. We use IMU sensors to detect acceleration and pose, use Lidar sensors to perceive the robot’s environment, and cameras to provide assistance for remote workers. The Lidar sensor that is being used on the Husky is the 3D Ouster Sensor OS0. For the cameras we are using four Intel depth cameras d435, however they are implemented as normal RGB cameras for now.

To communicate with our fleet management system and distributed steering use case that is running on the edge cloud, the Husky is connected via a 5G modem.

For the presentation of the use case a simplified control centre has been setup with six 1920x1080p displays and an Intel Nuc computer which can be seen in Figure 77.

Funded by the Horizon Europe Framework Programme of the European Union

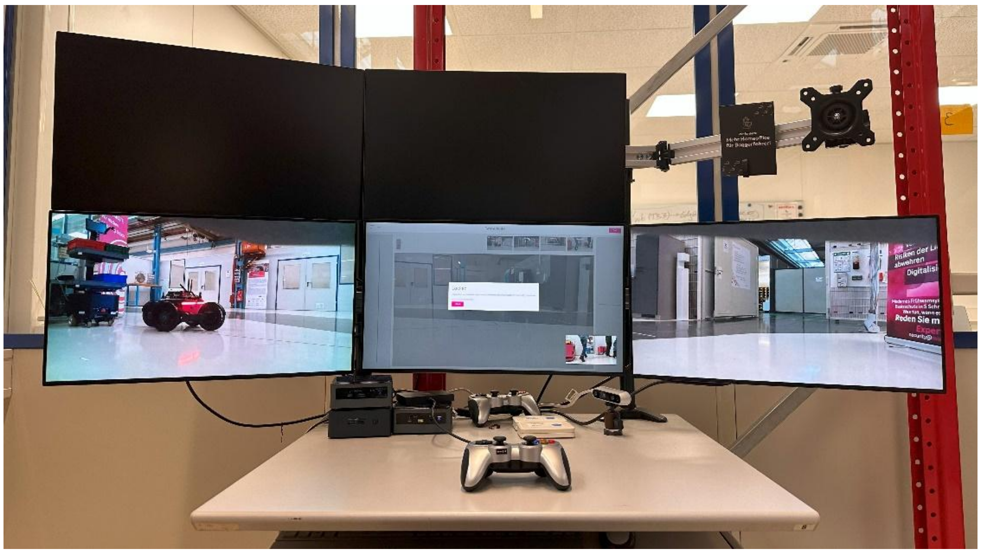


Figure 77: Simplified control center

Side by side with the web application we also offer an API to control the robot to participants of the SPIRIT project. For safety reasons the API can only be used in agreement and with the guidance of the Autonomous Logistics Team from the Deutsche Telekom. We have a variety of APIs available. However, for the interaction with our use case, we're providing APIs for the following functions:

➲ Sending Drive commands

➲ Post/get a map for the navigation of one specific robot

➲ Defining POI's/Goals in a robot specific map

➲ Post autonomous driving order for a specific robot to a specific POI/Goal

➲ Accessing the video stream of Husky robot

However, during the execution of the experiment's special usages and extensions of the API were discussed.

4.4.2 Extension of the Use Case

The Robot use case has been extended with input from several Open Call projects:

- One project deployed their simulation API on the testbed's edge server and utilized a digital twin of the facility to model pollution dispersion from a nearby power plant. These simulation insights were delivered to a robot operator equipped with virtual reality (VR) glasses. Within the VR environment, the operator was presented with an optimal path to reach a predefined target. The system integrated the Robot UC API to facilitate robot movement and localize the robot within the VR scene. In addition, the system established a connection to the Robot UC WebRTC Server, enabling the operator to view a live video stream alongside the virtual reality environment.
- Another project deployed their mixed reality interface for robot teleoperation on the testbed's edge server. During an on-site visit, they integrated their interface with the Robot UC API to enable manual navigation of the robot. Additionally, they connected all four onboard cameras to their video streaming solution. Ultimately, they achieved full control of the Husky robot using their mixed reality interface on a HoloLens 2 device, connected to the testbed's infrastructure.

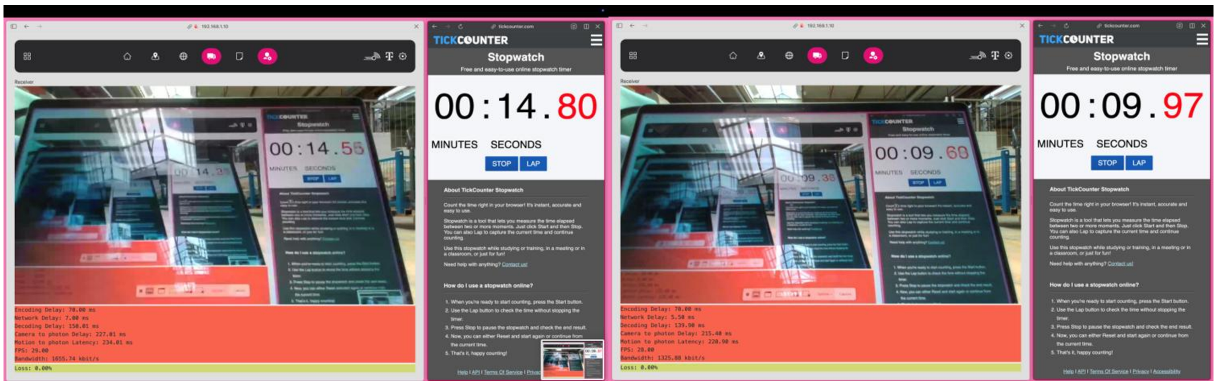
Funded by the Horizon Europe Framework Programme of the European Union

- Finally, in the second Open Call, a project is working on a distributed, fully containerization-compatible platform designed to enable robot operators to monitor and control mobile robots or manipulators via immersive, bi-directional interfaces that is planned to be integrated into the Berlin testbed together with the Husky Robot. Its capabilities are currently being tailored for the integration in the testbed in Berlin.

4.4.3 Validation

For validation purposes, a new page was implemented in the beta section of our web application. This page provides live metrics and includes a color-coded indicator to quickly inform operators about adverse latency and packet loss conditions. The goal is to give immediate feedback on the current state of the connection. The new UI with live metrics can be seen in Figure 78.

The displayed metrics include Encoding Delay, Network Delay, Decoding Delay, Camera-to-Photon Delay, Motion-to-Photon Delay, Frames per Second (FPS), Bandwidth, and Packet Loss. Among these, Network Delay, Decoding Delay, FPS, and Bandwidth are obtained directly via WebRTC internals. WebRTC internals is the browser's built-in debugging tool (e.g., chrome://webrtc-internals), which provides real-time traces and statistics of active WebRTC sessions, offering insights into connection behaviour, media transport, and performance.



```
Encoding Delay: 70.00 ms
Network Delay: 7.00 ms
Decoding Delay: 150.01 ms
Camera to photon Delay: 227.01 ms
Motion to photon Latency: 234.01 ms
FPS: 29.00
Bandwidth: 1655.74 kbit/s
Loss: 0.00%
```

```
Encoding Delay: 70.00 ms
Network Delay: 5.50 ms
Decoding Delay: 139.90 ms
Camera to photon Delay: 215.40 ms
Motion to photon Latency: 220.90 ms
FPS: 28.00
Bandwidth: 1325.88 kbit/s
Loss: 0.00%
```

Figure 78: Stopwatch test with new metrics UI

Figure 79 illustrates the distribution of latency components in the form of a bar chart. The most limiting factor observed during testing was the Decoding Delay, which reached 150 ms. Network latency remained relatively stable, varying between 7 and 13 ms. Since no direct data is available for the encoding stage, the encoding delay was estimated as the mean value by subtracting all known latency contributions from the end-to-end latency, based on multiple measurements.

Funded by the Horizon Europe Framework Programme of the European Union

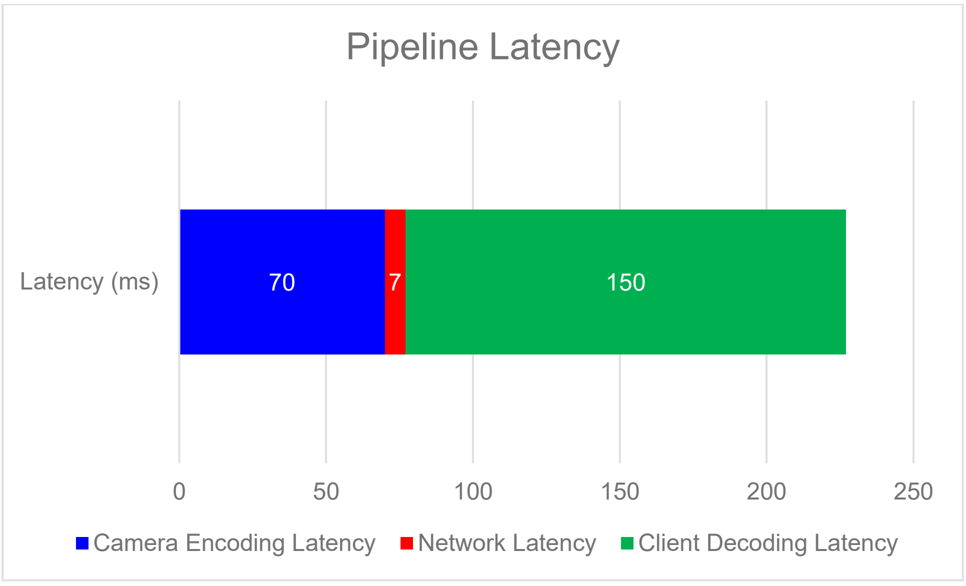


Figure 79: Pipeline latency distribution

The FPS during the tests ranged between 25 and 31 fps at a resolution of 1280 × 720 pixels, which meets the defined performance requirements.

To measure end-to-end latency, a practical test was conducted by capturing a stopwatch displayed on a client PC through the robot's camera. The resulting latency measurements varied between 200 and 300 ms. An image of this test setup is shown in Figure 78.

Figure 80 below illustrates the entire latency pipeline. Green arrows represent the components with measured latency values, whereas red arrows indicate segments where latency could not be directly measured.

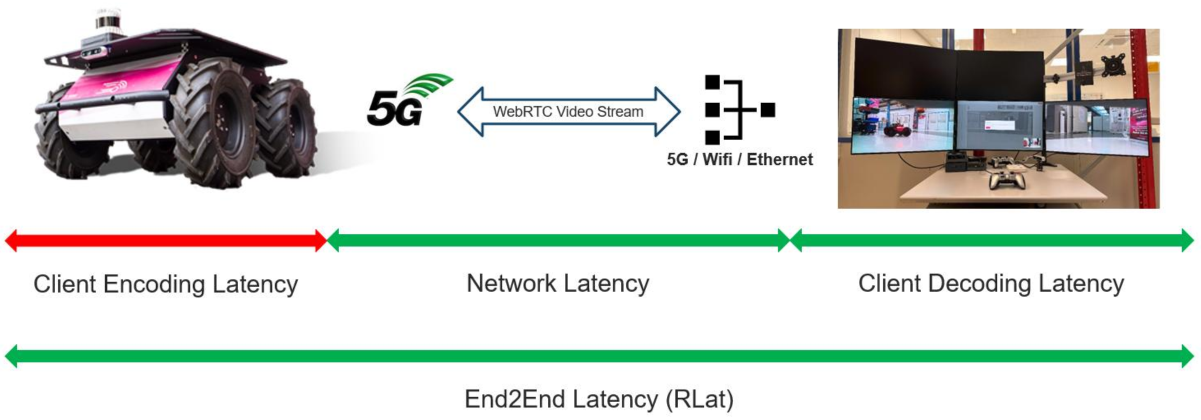


Figure 80: Elements of the latency pipeline

Funded by the Horizon Europe Framework Programme of the European Union

4.4.4 Summary Validation

Table 11: Use case Robot: development/integration and performance

Use case Robot development and integration	Relevant requirements	KPIs (metrics achieved)	Evaluation (determination of KPIs)

This use case was developed by T-Systems and is fully integrated in the Berlin Testbed.
 	
A robot that is available to the testbed was equipped with four cameras and a producer client application which forwards video streams to a local edge server. To make use of the local 5G network, the robot was also equipped with a 5G modem.
 	
A containerized application was deployed to the edge server providing a web interface for clients to connect to the video streams.
 	
For the demonstration of the use case a demo “control center” consisting of an Intel NUC and multiple monitors has been set up.
	RLat = 200
	
	RDown = 2Mbps
	
	RUp = 2Mbps
	
	RFPS = 30 fps
	
	RRes ≤ 1280x720
	
	RInDev = Cameras supporting video4linux
	
	ROutDev = Any common device running a web browser
		RLat = 200 – 300ms
		
		RDown = 1.45 Mbps
		
		RUp = 1.45 Mbps
		
		RFPS = 25-31 fps
		
		RRes = 1280x720 px
		
		RInDev = Intel Realsense 435
		
		ROutDev = MacBook Pro 2021
			E2E latency (RLat) was determined with a simple test capturing a stopwatch on the client pc with a camera mounted on the robot. The resulting latency varied between 200-300ms.
			
			Due to the peer-to-peer nature of the connection between client and consumer, downlink and uplink bandwidth requirements are roughly equivalent. Across multiple measurements, an average bandwidth of 1.45 Mbps was observed for a single camera stream. The Robot was connected via 5G while the receiver of the video streams during the measurement was connected via wifi.
			
			The frame rate achieved during the experiment varied between 25 and 31 fps.
			
			The final measurements presented in this table were conducted at the target resolution of 1280x720 pixels.
			
			The Robot is equipped with 4 Intel Realsense 435 cameras, which were used for these measurements.
			
			The video streams can be accessed via a web interface available in the local network. A measurement page has been setup that shows live metrics for the respective video stream. For these specific measurements a Mac Book Pro 2021 with Google Chrome was used.

Funded by the Horizon Europe Framework Programme of the European Union

5 SECURITY DEVELOPMENT AND TECHNICAL INTEGRATION

5.1 PROBLEM STATEMENT

In the SPIRIT project, we propose the implementation and deployment of an open architecture to foster vendor interoperability and easy re-use of our toolbox by partners. This approach leads to specific challenges that must be addressed in our work, specifically from our security viewpoint:

- The different components are disaggregated, i.e. they are functionally and spatially distributed.
- Components might be deployed on cloud or edge devices beyond the control of the users and communication partners.
- The components are provided by different vendors with different run-time requirements.
- Some of the components might run in public cloud environments or edge servers in physically exposed locations that are not under direct control of project partners.

Such a highly dynamic environment creates new attack surfaces and security problems. To this end, we have developed the new approach of end-to-end security (see Figure 81), where data in transit is encrypted and only provably trust-worthy cloud applications can break up this encryption to process and store the confidential data. To exclude the cloud provider itself from the trusted computing base (TCB) of the application, any solution must provide robust security guarantees that can also be verified by remote parties before delivering their potentially confidential data to the cloud.

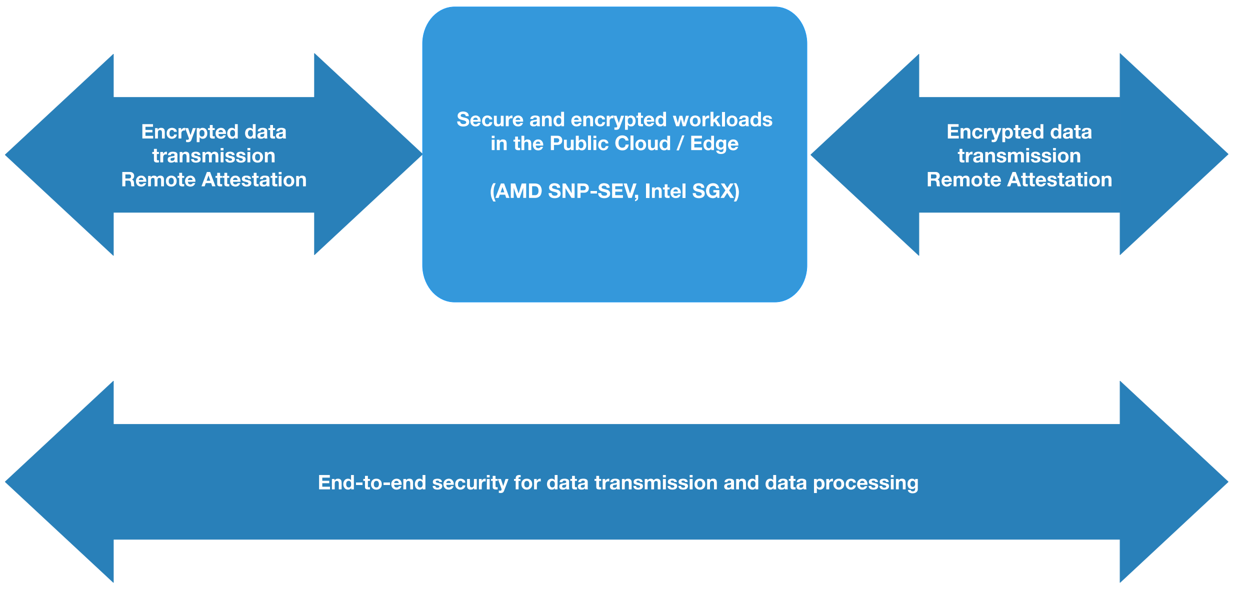

Encrypted data transmission Remote Attestation
Secure and encrypted workloads in the Public Cloud / Edge
(AMD SNP-SEV, Intel SGX)
Encrypted data transmission Remote Attestation
End-to-end security for data transmission and data processing

Figure 81: End-to-end security

Funded by the Horizon Europe Framework Programme of the European Union

In previous reports [8] we have identified cloud-based solutions and their limitations as well as cloud-specific vulnerabilities, such as

- mishandling of customer private keys by cloud providers,
- opaque processes in the cloud, and
- reliance on security certifications instead of hard security guarantees rooted in hardware properties.

In this document we specify an approach for deployment of Confidential Computing-protected virtual machines (VM) on bare metal servers (either on-premise or in the cloud) where private keys for managing VMs never leave the data owner's control and specifically do not need to be uploaded to cloud provider systems. The specification is enhanced with hands-on documentation to help project partner's use the provided software.

Computing resources as well as project partner support is provided by the Deutsche Telekom Cloud Security Lab in Berlin.

5.2 OVERVIEW OF THIS SECTION

This section serves both as a specification of the approach to Confidential Computing we have designed and implemented for the project as well as a user manual for project partners. The section structure supports this dual-purpose approach.

In section 5.3 we introduce our approach to Confidential Computing on a bare-metal Linux system, giving a high-level overview of the processes as well as an application example to further explain the system.

In section 5.4 we specify the different components we have implemented for our solution, notably the tool virtual machine (VM), the REST API for remote management, and the accompanying Docker image to simplify remote management for guest owners. The specification parts are interspersed with some hands-on instructions to support the dual role of user manual of this document.

In the appendix section "A.4 End-to-end Example" everything is put together to give users of our software an end-to-end example of the whole process, from obtaining the necessary software from our download site until the final operation of the trusted VM on our infrastructure.

Finally, in the Appendix A: Security we document specific aspects of our setup that did not fit into the previous.

Funded by the Horizon Europe Framework Programme of the European Union

5.3 OVERVIEW OF LINUX-BASED CONFIDENTIAL COMPUTING ON BARE METAL SERVER

After a recap of previous deliverables, introducing the trust model and some basic terminology of Confidential Computing, we explain in this section what Confidential Computing facilities are available on a fresh Linux installation (in our case Centos 9 Stream) and what is needed to achieve a practical and secure implementation that addresses the attacker model where the hardware operator of the VM host system is potentially untrustworthy.

5.3.1 Trust Model

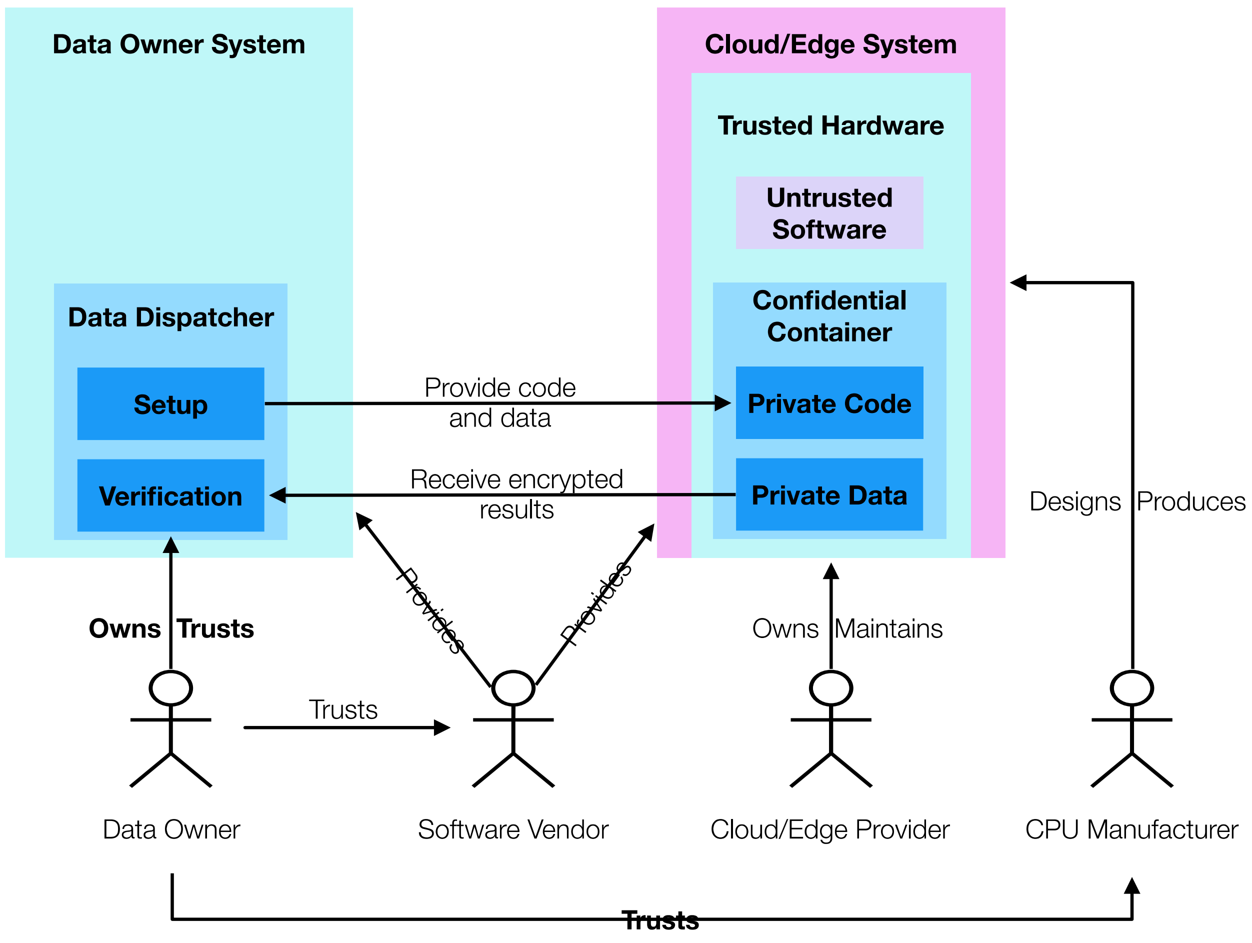


Figure 82: Trust Relationship in Confidential Computing

Confidential or Trusted computing in the context of this document refers to technologies which aim to solve the secure remote computation problem by leveraging trusted hardware in the remote computer. The trusted hardware establishes a secure container, and the remote computation service user uploads the desired computation and data into the secure container. The trusted hardware protects the data's confidentiality and integrity while the computation is being performed on it.

Figure 82 shows how the users trust the manufacturer of a piece of hardware in the remote computer and entrust their data to a secure container hosted by the secure hardware.

Funded by the Horizon Europe Framework Programme of the European Union

The Data Owner (also called Guest Owner later on) sets up the remote computation on the cloud or edge server by preparing code and data to be sent to the remote system.

On the remote side, the code and data are run in a Confidential Container (or Confidential VM). The guest or data owner only interacts with the Confidential VM after verifying its integrity through the process of measurement or remote attestation.

5.3.2 Basic Approach to Measured Start of VM

In our host system (representing the cloud/edge system) we run the standard Linux KVM virtualization manager (VM), with the qemu and libvirt software stack. The qemu component allocates resources and starts the open virtualization manager firmware (OVMF, basically an UEFI-based BIOS of the VM), which in turn invokes the grub boot manager which starts the kernel with the initial ram disk image (initramfs) which then mounts the root disk image of the Linux guest (Figure 83).

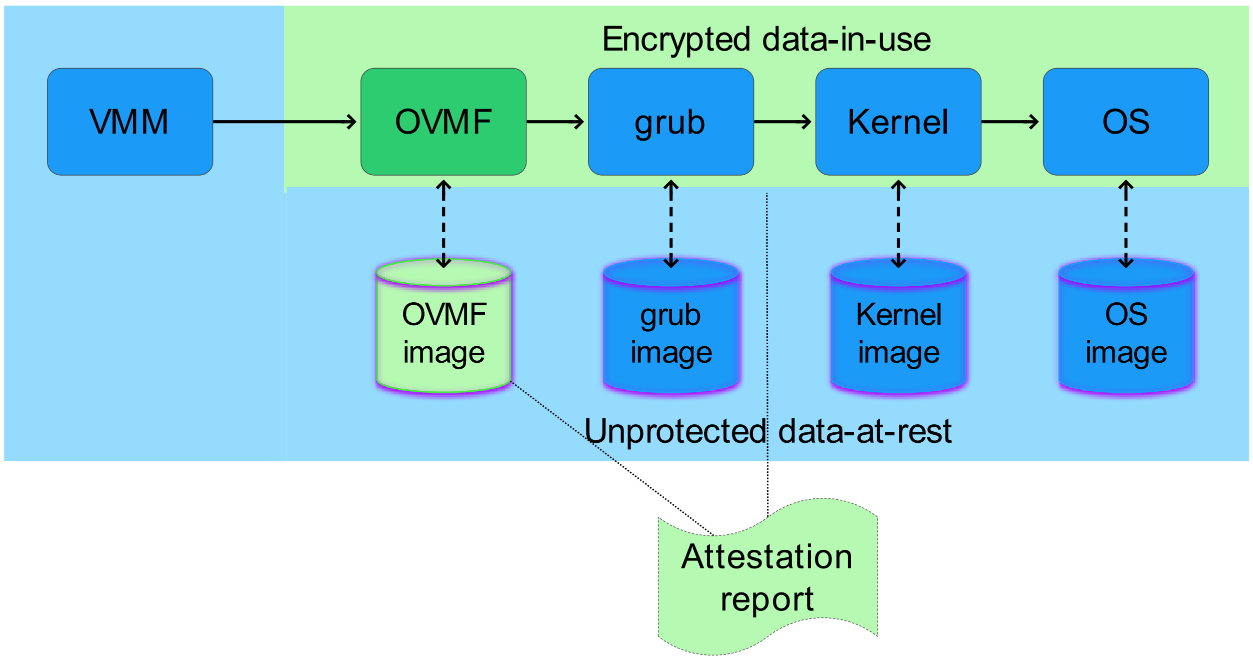


Figure 83: Basic measured start on Linux

The challenge in confidential computing is the following: the guest VM operates within a secure envelope, while the host hypervisor (including the QEMU process) operates outside this secure boundary, making it untrusted. Even with the new Luks (*Linux Unified Key Setup*, the Linux standard for disk encryption) format, QEMU handles image encryption, resulting in the encryption key residing outside the secure envelope. As a result, there is a need for a new format that ensures the encryption key (and the encryption mechanism) remains within the secured guest VM to enhance security.

Additionally, the attestation report obtainable with standard host tools only reports the integrity of the host processor, its firmware (in our case AMD Platform Secure Processor firmware), as well as the OVMF firmware image. The grub, Kernel, and OS image reside unprotected on the host system disks and are open to tampering.

Funded by the Horizon Europe Framework Programme of the European Union

5.3.3 Measured Start with Fully Encrypted Disk Images

In order to fully protect the remote environment, grub, Kernel, and OS images must be securely encrypted so that even the cloud provider cannot read or alter data. This, however, leads to the problem of bootstrapping this encryption key in a secure manner. In this section we describe how we achieved that.

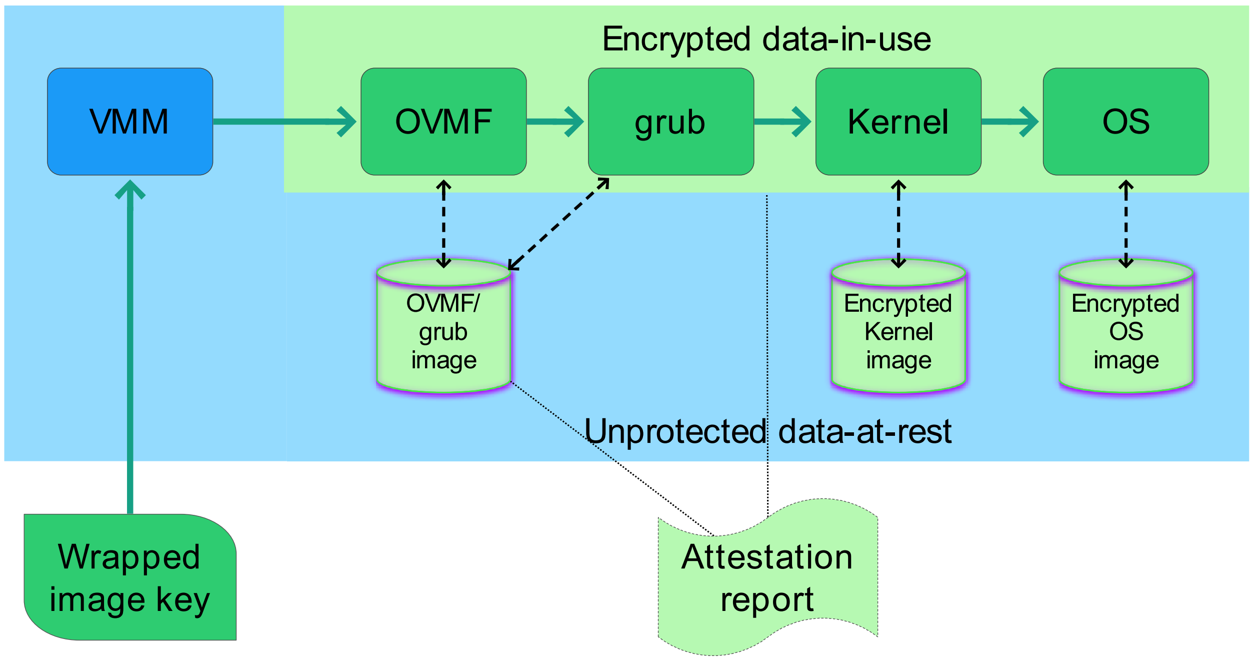


Figure 84: Measured start with fully encrypted disk images

Figure 84 shows the boot process after everything is set up:

1. The guest owner instructs the cloud provider to start the remote VM, but first only in paused state. The remote VMM (virtual machine manager – libvirt/qemu) starts the VM with a combined OVMF/grub image.
2. The guest owner requests a measure via the QMP protocol (Qemu Management Protocol). If the measurement meets expectations (OVMF/grub hash, AMD-SEV HW revision, AMD PSP firmware revision), the guest encrypts the Kernel/OS image key for the remote hardware and created a so-called “wrapped image key”.
3. The key is sent via QMP to the remote system and passed on the PSP.
4. The PSP decrypts the key and puts it into the encrypted RAM of the VM.
5. Here it is picked up by the embedded grub bootloader (which was also part of the measurement) and passed to the Kernel.
6. The Kernel decrypts the combined boot/root OS image and mounts it.
7. The VM is now running in encrypted RAM space with encrypted OS images (boot, root, and home partitions) and the hardware and VMM operator never has access to the protected data in RAM or disk image.

The fully encrypted disk image needed some preparation to support the sequence of steps outlined above. It can be created as follows:

Funded by the Horizon Europe Framework Programme of the European Union

1. Extracting the root partition: Extract the complete root partition of an existing VM’s image to a tar file. It needs to contain essential tools needed, such as cryptodisk and grub-efi.
2. Creating a two-partition raw image file: A new disk image file is generated, consisting of two partitions: the boot (UEFI) and the root/home partition.
3. Loopback mounting the image file: The newly created image file is mounted as a loop device, with 2 partitions: the EFI partition (p1) dedicated to the EFI system partition, and the Encrypted root partition (p2) used for the encrypted root filesystem.
4. Setting up EFI and cryptsetup: The EFI bootloader is configured to recognize the system's boot process and interact with the encrypted root partition. The root partition is encrypted using cryptsetup. During the initial setup, a simple password is used for the encryption (low-entropy or recovery password with many PBKDF2 iterations), however it can later be replaced by a high-entropy key for fast-booting, since grub is very slow in computing the PBKDF2 iterations.
5. Setting up grub for encrypted boot: grub configuration needs to be edited to enable boot from an encrypted partition.
6. Setting up fast-booting: to avoid 2 password prompts when booting the system (for boot and root partitions), a high-entropy password is added to cryptsetup. The initramfs configuration file is then edited, enabling the system to bypass the prompt for the encrypted root partition password.
7. Starting confidential VM: the initial ramdisk is rebuilt with all its partitions, and the system can boot using either the recovery or high-entropy password, enabling confidential computing with a fully encrypted disk image.

The process described in this section is based on previous work of the open-source community, [9], [10]. We found that many of the proposed patches are already present in an up-to-date Centos 9 Stream operating system. We had to modify and apply patches to some components, however, to make the whole system work (e.g. re-building OVMF with embedded grub, among other tasks).

5.3.4 Application Example: Secure Certificate Provisioning

To further deepen the reader’s understanding of the trusted VM image creation and remote management processes outlined in the previous sections we would like to introduce a concrete application scenario for Confidential Computing in the project’s context.

As has been outlined in our previous deliverables, relying parties (e.g. guest owner systems) should only send sensitive data to a remote system after successful remote attestation of the integrity of the remote system.

In this section we describe how our approach can be used to support trustworthy SSH (secure shell) and TLS (transport layer security) sessions involving trusted VMs running in the edge or public cloud.

Funded by the Horizon Europe Framework Programme of the European Union

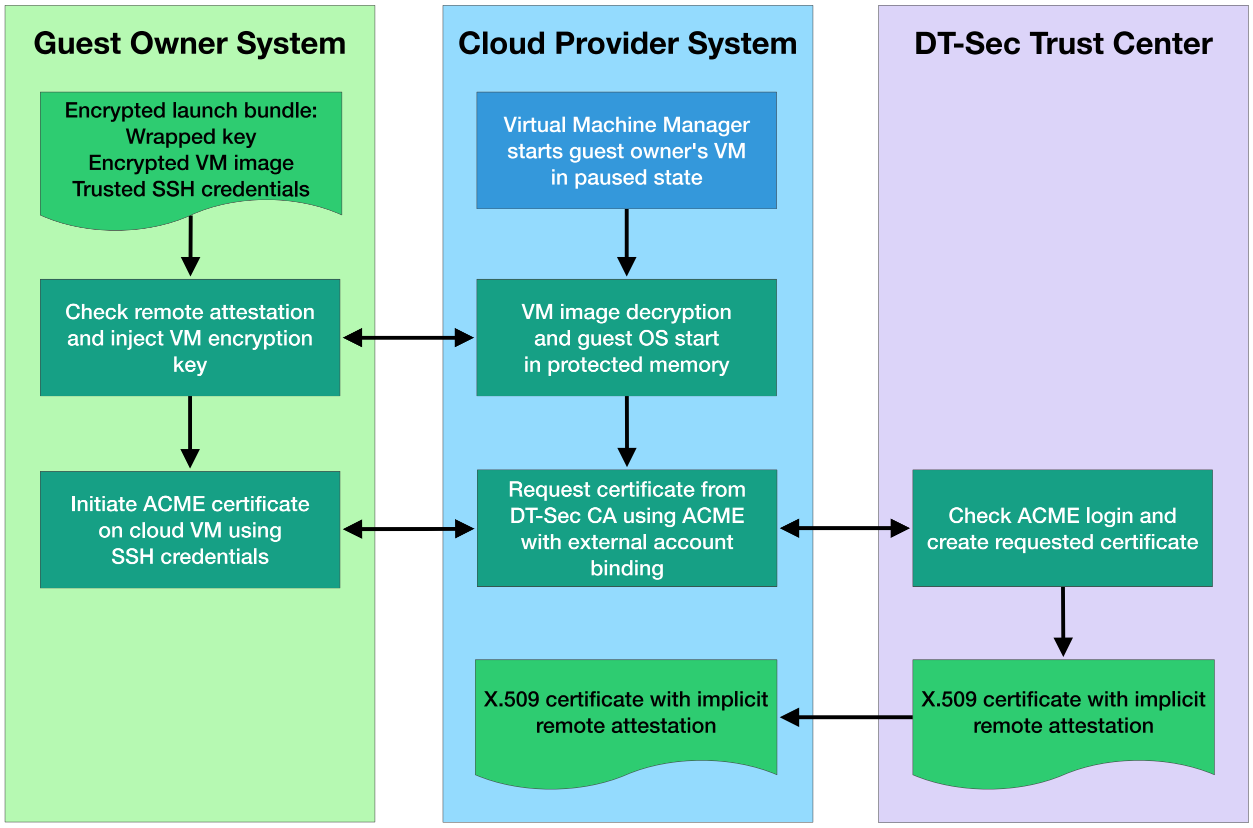

Guest Owner System
Cloud Provider System
DT-Sec Trust Center
Encrypted launch bundle: Wrapped key Encrypted VM image Trusted SSH credentials
Virtual Machine Manager starts guest owner's VM in paused state
Check remote attestation and inject VM encryption key
VM image decryption and guest OS start in protected memory
Initiate ACME certificate on cloud VM using SSH credentials
Request certificate from DT-Sec CA using ACME with external account binding
Check ACME login and create requested certificate
X.509 certificate with implicit remote attestation
X.509 certificate with implicit remote attestation

Figure 85: Application example - secure certificate provisioning

Figure 85 shows the three involved parties with their respective activities:

- On the guest owner system, the initial image is prepared: It is encrypted with a disk key only known to the guest owner. Additionally, the guest prepares the SSH host key and loads software and data specific to the application.
- The cloud (or edge) provider receives the encrypted keys as well as parameters generated by the guest owner (the so-called “launch bundle”) to setup the VM in the VMM.
- The DT-Sec trust center is contacted by the trusted VM on first run (and later for certificate renewal) in order to create certificates.

The certificates originating from this process depend implicitly on the successful remote attestation between guest owner and cloud provider, because the guest must follow these steps:

1. Create an SSH host key in a trustworthy environment while setting up the image.
2. Start the VM only after successful remote attestation.
3. Log into the VM depending on the trusted host key. This way, the guest owner can be sure to log into a trusted environment.
4. Only then start the ACME client to request the certificate using the ACME login credentials (for external account binding). This step in turn depends on the trust into the SSH host key.

As a result, the trustworthiness of the certificate depends inherently on the successful remote attestation. Clients connecting to the application on this VM using TLS don’t have to do the

Funded by the Horizon Europe Framework Programme of the European Union

remote attestation step themselves, since this has been done already in the process of obtaining the certificate. Additionally, this approach enables standard SSH-based management of the remote VM without doing any additional attestation step.

5.4 SPECIFICATION OF COMPONENTS

In this section we describe the tools and processes for guest owners to create individually encrypted, trusted VMs as well as the remote management and remote attestation/measurement facilities that we have implemented.

Using the Guest VM setup process, Guest owners can configure an encrypted VM template with their own keys. They can then provide the resulting encrypted image to the Cloud provider and initiate the startup of the VM in paused mode (necessary state for secure launch). Afterwards, Guest owners can initiate measured launch by checking the remote attestation from the Cloud provider and inject their VM's encryption key. The VM can then start in protected mode.

5.4.1 Setup Phase

5.4.1.1 Overview

The Guest VM Setup tool is a tool VM based on CentOS 9 allowing guests to create and set up their own encrypted VM.

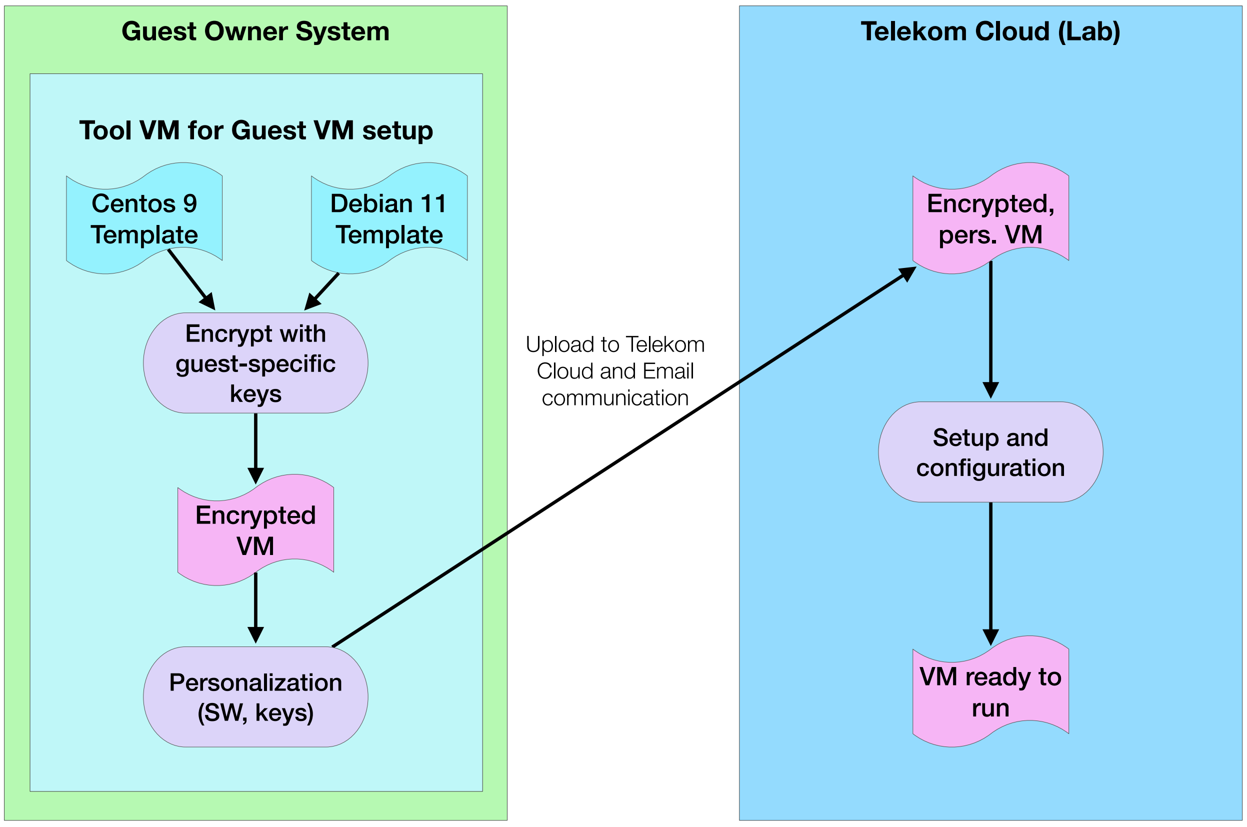


Figure 86: Setup of the guest VM

Funded by the Horizon Europe Framework Programme of the European Union

Guests can set up their encrypted VM using the VM-tool provided by DT. There are currently 2 available OS templates for encrypted VM, Debian 11 and CentOS 9. After choosing a template, guests can encrypt it with their own keys to replace the DT default one. Afterwards, the VM image is transmitted to the Telekom Cloud Lab, to ensure that the encrypted VM is ready to run.

5.4.1.2 Commands Available in the VM for Guest VM Setup

The Guest VM Setup Tool contains all the necessary tools for guests to:

- select an encrypted VM image template (Debian 11 or CentOS 9),
- encrypt that image with 2 different keys (low entropy password as a recovery key, and high entropy password dynamically generated for faster boot of the VM), and
- mount the image on the tool VM (to perform administrative tasks, such as editing configuration, install updates and proprietary software, create an initial SSH key...).

The Guest VM Setup tool was originally supposed to be a docker container, but it was later replaced by a CentOS 9 based-VM, as it would have required too many privileges to run on Guest. Pre-configured VMs are available as well as the tools as a separate download for partners that want to use their own Guest setup VM.

The pre-configured images as well as intermediate files are located in directories below the /workdir directory in the VM root filesystem. The structure of the /workdir in the VM is as follows:

```
/workdir
        images/
                debian-template.qcow2 (with pre-defined recovery key)
                centos9-template.qcow2 (with pre-defined recovery key)
        transfer/
                debian_XXXXX.qcow2 (encrypted with new keys)
                centos_XXXXX.qcow2 (encrypted with new keys)
        keys/
                YYYYY_XXXXX_recovery-password.txt (user-given recovery password)
                YYYYY_XXXXX_high-entropy-password.txt
```

Present in the original directory Created by commands

All the commands described below are already set up in the Guest VM Setup tool. They need to be run with root privilege.

```
prepare [--debian/--centos] <VM name> <recovery-password>
```

This command is used to create a new encrypted image of a chosen template with personalized password. It works as follow:

- Arguments:
  - `--debian/--centos` : the chosen base template (CentOS 9 or Debian 11)
  - `<VM_name>`: the name of the created image. It will be renamed as “template_name.qcow2” (eg: “centos_name.qcow2”)
  - `<recovery-password>:` password of choice which will be used to encrypt the image
- Generate a high entropy password, store recovery password and high-entropy password in the “/workdir/keys directory”.
- Make a copy of the chosen template to “/workdir/transfer” directory and rename it
- Encrypt the image with both recovery-password and high-entropy password using `cryptsetup`.

`mount <VM name>`

This command is used to mount the encrypted VM image from the /workdir/transfer directory and give the user a command line in the VM /root directory. It works as follow:

- Argument:
  - `<VM_name>:` name of the image to mount, can be written either as "template_name" or "name" (eg : “centos_name” or “name”).
- Connect the image as NBD (Network Block Device) to a free NBD device (located in “/dev/nbd*”).
- Open the encrypted luks partition using the recovery password in “/workdir/keys” directory.
- Mount the associated cryptsetup device (“dev/mapper/encrypted-image”) and all its contained directories at “/mnt/root”.
- Give the user a command line from “/mnt/root” using `chroot.`

`unmount-images`

This command is used to unmount and disconnect every image.

- Unmount all partitions.
- Close every cryptsetup devices.
- Disconnect every NBD device.

Funded by the Horizon Europe Framework Programme of the European Union

`reset-images`

This command is used to reset the VM setup tool environment. Use this command with care since it deletes all image personalizations.

- Unmount all images (the same as `unmount-images` command).
- Delete all previously created files in the workdir (/workdir/transfer containing created images, /workdir/keys containing keyfiles.

The complete REST API is described in appendix A.9.

5.4.2 Operation Phase

After the prepared image from the setup phase has been transferred to the host system in the DT-Sec Berlin lab and pre-configured by the support personnel, the guest owner needs to perform a few last steps to prepare the confidential remote operation, after which he or she can start using the remote computing resource.

5.4.2.1 Overview

Before the actual operation phase, the following two steps must be performed by the guest owner:

- Creating a so-called “launch bundle”, consisting of AMD-SEV-related key material for remote management.
- Sending two public keys from this launch bundle to the remote system via the REST API described below.
- Notifying the host admin about the completion of these two steps.

After these steps the remote VM is ready to run.

Funded by the Horizon Europe Framework Programme of the European Union

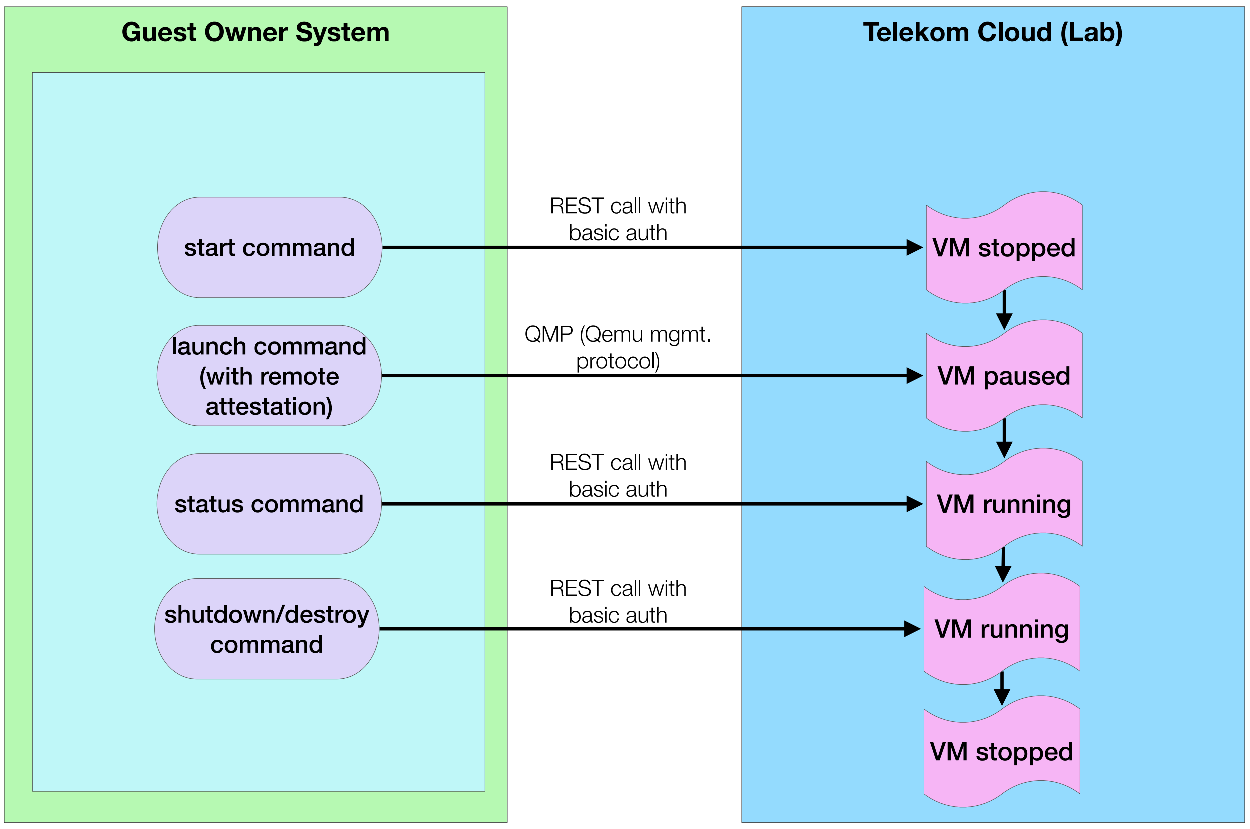

Guest Owner System
Telekom Cloud (Lab)
start command
launch command (with remote attestation)
status command
shutdown/destroy command
REST call with basic auth
QMP (Qemu mgmt. protocol)
REST call with basic auth
REST call with basic auth
VM stopped
VM paused
VM running
VM running
VM stopped

Figure 87: Managing the guest VM

Figure 87 shows the available remote management commands as well as the life cycle of the VM:

- With the start command the guest owner starts the remote VM in paused mode.
- The launch command performs the actual remote attestation. If the remote attestation passes (i.e. the measurement is equal to the locally computed value) the launch command also transmits the encrypted disk key to the remote VM. The VM then starts and is ready to perform its function.
- At any time, the guest owner can request the status of the remote VM using the status command.
- When the VM is not needed anymore, the guest owner should issue a shutdown command, followed later by a destroy command to stop and switch off the remote VM. Note: destroy should be issued about 30 s after the stop command to allow for clean shutdown. Some VMs need the destroy command because the Linux kernel is not switching off the VM properly. Best practice would be to check with the “status” command and issue the destroy only if the VM is still running after 30 s.

Detailed configuration steps are given in appendix A and in README files on our download site.

Funded by the Horizon Europe Framework Programme of the European Union

5.5 CLOUD SECURITY LAB IN BERLIN

To allow for integration of and experimentation with the Confidential Computing technologies described in this document we have set up a lab environment for project partners and Open Call partners to run confidential work loads in a cloud-like environment in our Deutsche Telekom Cloud Security Lab in Berlin (see Figure 88).

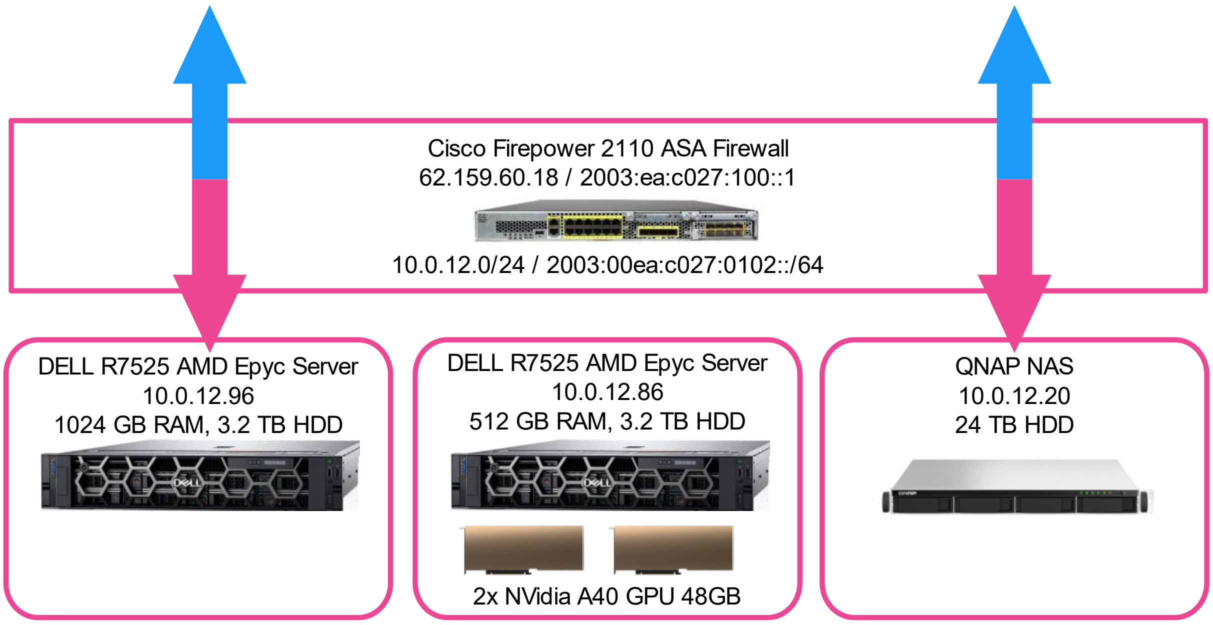

Cisco Firepower 2110 ASA Firewall
62.159.60.18 / 2003:ea:c027:100::1
10.0.12.0/24 / 2003:00ea:c027:0102::/64
DELL R7525 AMD Epyc Server
10.0.12.96
1024 GB RAM, 3.2 TB HDD
DELL R7525 AMD Epyc Server
10.0.12.86
512 GB RAM, 3.2 TB HDD
2x NVidia A40 GPU 48GB
QNAP NAS
10.0.12.20
24 TB HDD

Figure 88: Cloud security lab infrastructure

The setup in our lab consists of two Confidential Computing-enabled servers, one of them with Nvidia-based GPU resources.

The test environment offers the following facilities:

1. 300 MB Telekom Business Fiber Connection 8h response
2. Access via Cisco AnyConnect or open source openconnect
3. 10.0.12.0.XXX addresses only available via VPN
4. Ports can be exposed via NAT on 62.159.60.18 in the 5 digit port range
5. Individual IPv6 addresses (no NAT), ports to be exposed an request
6. Large datasets can be hosted on NAS and made available as ISCSI targets

Funded by the Horizon Europe Framework Programme of the European Union

5.6 INTRUSION DETECTION DEPLOYMENT IN THE BERLIN TESTBED

In this section we describe the intrusion detection approach we have pursued in experimentation in the Berlin testbed.

We describe the systems we have analysed and give an overview the final demonstration setup in the testbed environment. Our approach consists of the following three steps:

- Analyze existing open-source and commercial IDS
- Select a system for deployment in a demonstrator
- Deployment of the selected system in a demonstrator

5.6.1 Overview of Intrusion Detection Systems

We have looked at several different intrusion detection systems as well as supporting software needed for our final demonstration setup which are shortly described in the following sections.

5.6.1.1 Grafana

Grafana is an open-source software used for the visualization, monitoring, and analysis of real-time measurement data. It allows users to create dashboards with panels that offer a variety of different visualizations such as graphs, histograms, heatmaps, and others.

The data visualized in Grafana typically comes from a wide range of data sources. Grafana supports many different types of data sources, including, but not limited to, databases (such as MySQL, PostgreSQL, InfluxDB), cloud services (such as AWS CloudWatch, Google Stack-driver), and other monitoring tools (such as Prometheus, Graphite).

In terms of functionality, Grafana retrieves data from the configured data sources, processes the data, and then presents it in an appropriate visual form on the dashboard. Users can customize the dashboards to display the data in the most meaningful way for them. They can also set up alerts to receive notifications when certain conditions are met.

5.6.1.2 Suricata

Suricata is an open-source network threat detection system (IDS), intrusion prevention system (IPS), and network security monitoring tool. It is designed to monitor network traffic and detect and block malicious activities.

Suricata uses a combination of technologies and rules to analyse network traffic and detect potential threats. It utilizes a powerful and flexible rule language to detect complex threats and can also be used for anomaly detection.

In terms of functionality, Suricata inspects network traffic and analyses it based on predefined rules or signatures that identify known threats.

5.6.1.3 Suricata Prometheus Exporter

Prometheus Exporter is a service that converts metrics generated by third-party systems into a format readable by Prometheus. These are specialized programs or scripts that extract metrics from existing systems and services, transform them into the format used by Prometheus, and make them available for retrieval.

Funded by the Horizon Europe Framework Programme of the European Union

These exporters allow Prometheus to collect a wide range of metrics, which can then be used for monitoring, alerting, and analysis. They are a central component of the Prometheus ecosystem and contribute to its flexible and extensible architecture.

5.6.1.4 Prometheus

Prometheus is an open-source system monitoring and alerting toolkit. It is primarily used for monitoring metrics, including the collection of counters, histograms, and time series data. Originally developed by SoundCloud, it is now part of the Cloud Native Computing Foundation project.

Prometheus operates by retrieving metrics from configured targets at given intervals, evaluating expression rules, displaying the results, and optionally triggering alerts when certain conditions are met. Users define alert conditions in Prometheus' query language, PromQL.

Prometheus offers several modes for visualizing data: in its built-in expression browser, through Grafana integration, or via an API that can be used by third-party software.

A key feature of Prometheus is its multidimensional data model: time series are identified by a metric name and key/value pairs. This allows for powerful and precise selection and aggregation of time series data.

5.6.1.5 Promtail

Promtail is an agent used in Grafana Labs' Loki log aggregation platform. It collects logs, particularly from servers and other devices, and sends them to a Loki instance.

Promtail is designed to track log files present on the local system and transmit their data to Loki in real time. It uses service discovery or static configuration to locate log sources and add metadata.

Promtail employs a pipeline concept to parse, format, and prepare the data before sending it to Loki. These pipelines consist of several stages, including timestamp extraction, labeling, and output, allowing users to customize and refine their log data.

Overall, Promtail is an essential component of the Loki system, serving to gather log data, which is then stored centrally for querying and analysis.

5.6.1.6 Loki

Loki is an open-source, horizontally scalable, highly available log aggregation system developed by Grafana Labs. It is designed to work with Grafana to simplify the visualization and management of log data.

Unlike other log aggregation systems, Loki does not index the entire log, but only metadata such as the log source. This makes it more efficient and cost-effective to operate.

Loki’s functionality is straightforward. It receives input from Promtail (the log collection agent), which collects logs and sends them to Loki. Loki stores these logs and metadata and allows users to search through them via Grafana.

Users can then visualize their log data in Grafana by creating queries based on the metadata that Loki has stored. This allows for fast and easy searching of log data without the need to index large volumes of logs.

Funded by the Horizon Europe Framework Programme of the European Union

5.6.2 Integration into our IDS Demonstrator

We use two methods for data collection and transmission in our system:

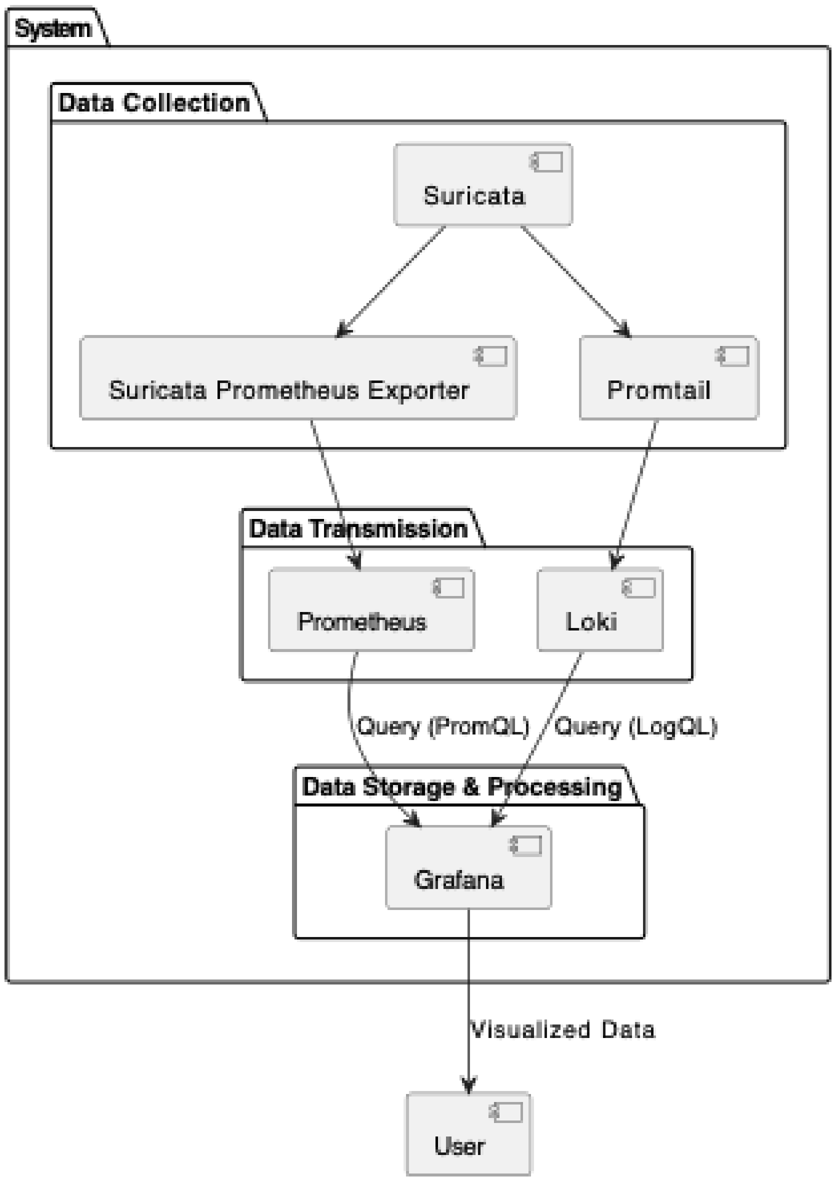


Figure 89: IDS Demonstrator Setup

Explanation:

- **Suricata** collects network data and logs events based on predefined rules.
- **Promtail** collects logs from Suricata and sends them to **Loki**.
- **Suricata Prometheus Exporter** pulls metrics from Suricata and sends them to **Prometheus**.
- **Grafana** queries **Loki** using **LogQL** for log data and **Prometheus** using **PromQL** for metrics.
- **User** interacts with **Grafana** to visualize the data.

The components are described in detail in the following sections:

Funded by the Horizon Europe Framework Programme of the European Union

5.6.2.1 Suricata

Suricata serves as the foundation for our data collection, analysing network traffic and logging certain events based on predefined rules. We use the following two methods for data transmission and processing.

5.6.2.2 Promtail + Loki

Promtail and Loki work together to enable efficient and effective log aggregation and visualization. Promtail acts as the agent running on your servers, responsible for collecting log data. It is configured to track log files from specific locations on the server. When new entries are added to these files, Promtail reads them and sends them to Loki.

Loki is the back-end system that receives the log data and metadata collected by Promtail. Loki stores this data and makes it searchable. Since Loki indexes only the metadata and not the entire log text, it is able to store and make the data searchable more efficiently than some other log aggregation tools.

When users want to search through their log data, they do so via Grafana, which sends a query to Loki. Loki uses the metadata it has stored to quickly find the relevant log data and return it to Grafana.

5.6.2.3 Suricata Prometheus Exporter + Prometheus

Promtail and Loki work together to enable efficient and effective log aggregation and visualization. Promtail acts as the agent running on your servers, responsible for collecting log data. It is configured to track log files from specific locations on the server. When new entries are added to these files, Promtail reads them and sends them to Loki.

Loki is the back-end system that receives the log data and metadata collected by Promtail. Loki stores this data and makes it searchable. Since Loki indexes only the metadata and not the entire log text, it is able to store and make the data searchable more efficiently than some other log aggregation tools.

When users want to search through their log data, they do so via Grafana, which sends a query to Loki. Loki uses the metadata it has stored to quickly find the relevant log data and return it to Grafana.

5.6.2.4 Grafana

Grafana serves as our endpoint, visualizing data with the help of query languages. The query languages used are LogQL (https://grafana.com/docs/loki/latest/query/) for Loki and PromQL (https://prometheus.io/docs/prometheus/latest/querying/basics/) for Prometheus.

5.6.3 Conclusion of IDS Experiments

The experiments have shown how to combine several open source software components into a viable intrusion detection system that we have successfully deployed in the Berlin testbed used internally and by open call partners for SPIRIT experimentation.

Since we have followed a general anomaly detection approach in our demonstrator setup the solution is suitable for a wide variety of different application scenarios.

Funded by the Horizon Europe Framework Programme of the European Union

6 QUALITY OF EXPERIENCE EVALUATION

In recent years, immersive video delivery has improved significantly. This type of video content can be viewed in XR, including VR, AR, and MR, to provide near-lifelike 3D objects and scenes. To represent these 3D entities, PCs are commonly used as the format providing high-fidelity representations without any constraint on the viewpoint and interaction. A PC comprises thousands or even millions of points, including information about colours (e.g., RGB) and geometry (x,y,z coordinates) of each point. Thus, the usage of point clouds (PCs) costs a large amount of storage and network bandwidth. Efficient ways to compress PCs are of importance to solve this issue. The compressed PC is then stored and/or delivered to end users via video streaming techniques (e.g., WebRTC, and HAS [11]).

In HAS, the content is encoded into various quality levels, then temporally split into multiple segments with the same duration. These segments are stored on one or multiple servers in a content delivery network [12]. The end users' media player selects the quality level for each segment based on the network conditions (i.e., observed throughput), the devices' characteristics (i.e., resolution and buffer), and the users' preferences. On the other hand, WebRTC makes its own decisions about the target encoding bitrate based on the estimated throughput, at which the video encoder is called for video compression. In both cases, the video quality can be changed in a streaming session, which leads to quality switches. The quality switches are well-known to affect the QoE in traditional video streaming [13]. However, they have not been fully considered in dynamic point cloud streaming.

Due to extremely high data volumes of uncompressed PCs, point cloud compression (PCC) is necessary to reduce both storage requirements and the amount of data delivered through networks. Initial studies started with compression of static 3D objects [14] [15] [16]. However, recent work focused on dynamic scenarios [17]. The MPEG developed standardized solutions for point cloud compression by leveraging their existing codecs, such as HEVC [18]. Their video point cloud encoder implementation can achieve a compression rate of 125:1. However, PCC comes at the cost of visual quality, determined by the QP. A higher QP provides a lower bitrate but leads to a lower quality.

Therefore, understanding the impact of different factors in video streaming techniques and PCC on the QoE is of importance. PCs have been evaluated in different viewing conditions (i.e., VR HMD and 2D screens) [19] [20]. However, research on the subjective quality assessment of PCs in AR environments is still limited. AR enhances people's perception of physical and virtual environments [21]. Thus, it is an interesting setting for immersive telepresence applications, which we develop and assess in the SPIRIT project.

Most of the effort in building QoE tools has been focused on 2D videos. Petrangeli et al. [22] proposed a parametric statistical model computing QoE based on average bitrate, its standard deviation, stall duration, and stall frequency. Tran et al. [23] introduced BiQPS using a Long-Short Term Memory network to predict the QoE. BiQPS takes into account stall duration, QP, bitrate, resolution, and frame rate. Recently, the ITU defined a standardized QoE model, the P.1203 model [24] for 2D video streaming that is trained and validated with different large databases of QoE. Some effort has been made to predict QoE in point cloud streaming on the basis of mathematical models, such as [25]. However, these models are often validated by subjective tests in VR environments.

UNI-KLU, in collaboration with imec, conducted a subjective test to assess the impact of quality, quality switching, viewing distance, and content characteristics on the perception of point clouds in AR environments. The output of this work includes *(i)* a platform for subjective quality assessment in AR environments, *(ii)* a dataset of rating scores that can be used for training

Funded by the Horizon Europe Framework Programme of the European Union

and validating future QoE models as well as the results (findings) of the subjective tests that produced these rating scores, and *(iii)* a machine learning-based QoE model. In addition, the work also fine-tuned an existing ITU QoE model – that has been originally developed for conventional 2D videos – to predict the QoE for point clouds in AR environments. The details are provided in the following subsections.

The software, dataset, results (findings) from the subjective test, and the QoE models could be used by the project partners and by Open Call participants in use cases (applications) involving point cloud content in AR environments for three purposes:

- to build their own preview and subjective test software informing their use cases,
- to be guided by the subjective test results when deciding about the quality level(s) of the PC content to be captured, transmitted, and presented in their use cases, and
- to predict how their users will perceive the PC content in AR environments.

UNI-KLU also conducted another round of subjective testing, using a different set of point clouds and incorporating eye-tracking into the tests as well [26]. The compressed point clouds and the results from the testing were made public and could be used by the Open Call participants. In fact, the Open Call 2 project "QoE Evaluations for Multi-Cloud Streaming (QEST)" used the dataset captured in that study to perform their own subjective tests using recent HMDs (Meta Quest 3 and Apple Vision Pro).

This final version of the deliverable (D4.3) also reports the main results of the second subjective study, namely: *(i)* an analysis of the eye-tracking data (visual attention, head and gaze movement); *(ii)* the subjective QoE results and the main factors impacting quality perception; and *(iii)* machine learning-based models as well as another fine-tuned ITU model for QoE prediction/estimation.

Finally, QoE assessments for project-specific use cases using objective QoE estimation models are described.

6.1 TESTING PLATFORMS

We have developed testing platforms for subjective testing, and later also eye-tracking testing, of point clouds in mixed reality using the Microsoft HoloLens 2. These testing platforms have been the basis of our subjective studies [27] [28], which in turn have provided the data for us to perform QoE evaluations of the project partners' use cases. The initial version, first presented in [29], is a modular platform that can be used to configure and run subjective testing for point clouds and meshes. The platform can be found on GitHub at:

https://github.com/shivivats/MR-Subjective-Testing-Platform/tree/old-version.

The newer version, now titled STEP-MR [30], can be found on GitHub at:

https://github.com/shivivats/MR-Subjective-Testing-Platform,

and contains an updated version of the subjective quality testing functionality as well as newly added eye-tracking testing functionality. The following subsections present the platforms in detail.

Funded by the Horizon Europe Framework Programme of the European Union

6.1.1 A Platform for Subjective Quality Assessment of Point Clouds in Mixed Reality Environments

We introduced a platform for subjective quality assessment of PCs in MR environments and used this platform for our initial round of subjective quality testing [27]. The platform provides multiple options for configuring the rendering of dynamic PC objects and meshes, including changing the content, quality, viewing distance, and representation, as well as previewing and interacting with the PC objects. Additionally, the proposed platform can be deployed to create subjective tests of the visual quality of dynamic PCs and meshes in AR environments.

The platform is built using Unity[10]. The MRTK 2[11] software development kit (SDK) from Microsoft is utilised to work with the HoloLens 2. Despite the HoloLens 2 having standalone capabilities, we use Holographic Remoting[12] from MRTK 2 to run the platform on a workstation while taking input from and sending output to the HoloLens 2. The architecture is shown in Figure 90. The workstation used for the initial platform contained an Intel Core i9-13900K processor, 64 GB of DDR5-4800 MHz memory and an NVIDIA RTX 4070 Ti GPU.

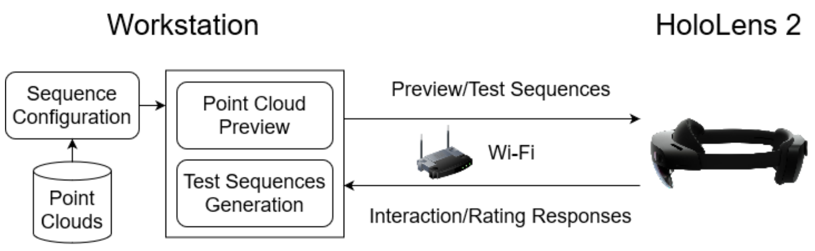


Figure 90: Platform architecture for subjective quality assessment

The platform has two major functionalities: *(i)* point cloud previewing and *(ii)* subjective testing.

6.1.1.1 Point Cloud Previews

The purpose of the preview functionality is to allow the tool users to explore and compare various configurations of the point clouds. These configurations are controlled via a menu in the HoloLens 2 device. Multiple objects can be added and viewed side by side, and each object can be configured individually. As shown in Figure 91, an object can be configured in terms of the following properties: *Representation, Viewing Distance, and Quality*. The *Loot* object[13] (front) is rendered using the *Point* representation at a distance of 3 m from the viewer with quality level 3. The *Soldier* object[13] (at the back) is rendered using the *Square* representation at 5 m distance with the same quality.

[10] Version 2021.3.19f1. https://unity.com/. Accessed: 20 April 2023.

[11] https://github.com/microsoft/MixedRealityToolkit-Unity. Accessed: 20 April 2023.

[12] https://learn.microsoft.com/en-us/windows/mixed-reality/develop/native/holographic-remoting-overview. Accessed: 20 April 2023.

[13] https://plenodb.jpeg.org/pc/8ilabs/. Accessed 09 September 2025.

Funded by the Horizon Europe Framework Programme of the European Union

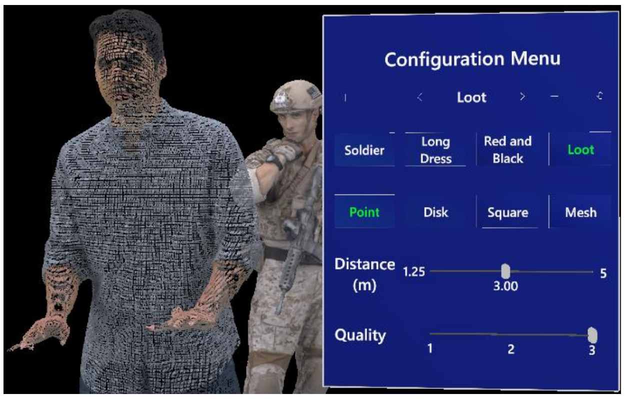
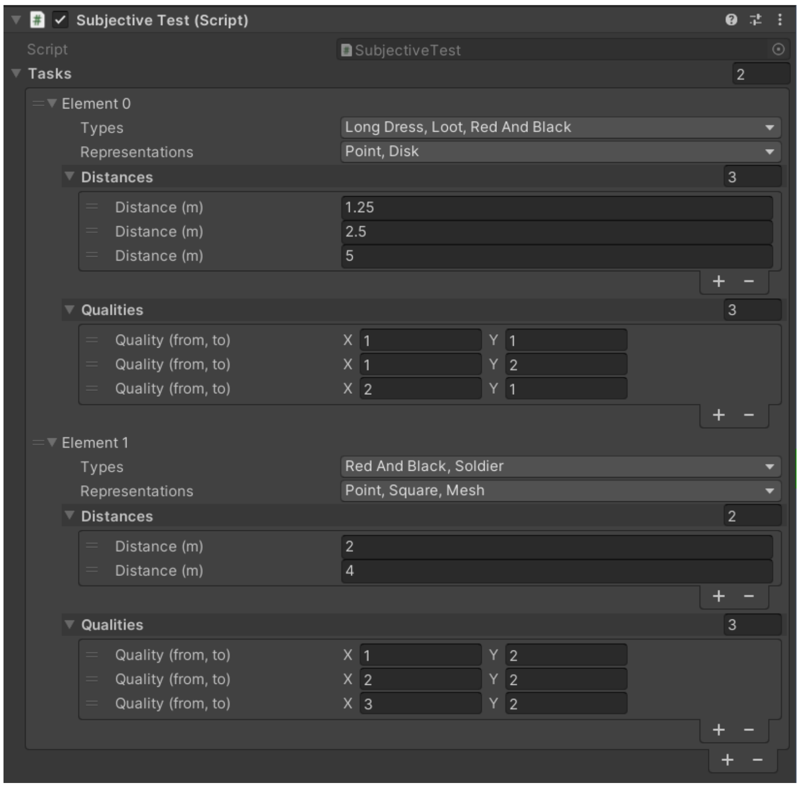


Figure 91: Configuration control panel and PC objects

6.1.1.2 Subjective Testing

The second major functionality of our platform is the ability to compose and perform subjective tests within the given parameter ranges. These are single-stimulus tests designed with the purpose of testing the impact of various factors, such as distance and quality, on the perceived quality of the object and, thus, the QoE of the test participant. These tests can be configured using the Unity UI and are divided into “tasks”. Within a task, the tool user (researcher or test director) can select several configurations for the point clouds, similar to the configuration options in the preview. All the possible permutations from the chosen configurations are determined and displayed to the test participant in random order. Randomization removes any bias the participants might obtain by watching the sequences in a particular order. The Unity UI for the subjective test configuration is shown in Figure 92.



```
Subjective Test (Script)
Script                      SubjectiveTest
Tasks                                                   2
  Element 0
    Types                   Long Dress, Loot, Red And Black
    Representations         Point, Disk
    Distances                                           3
      Distance (m)          1.25
      Distance (m)          2.5
      Distance (m)          5
    Qualities                                           3
      Quality (from, to)    X 1        Y 1
      Quality (from, to)    X 1        Y 2
      Quality (from, to)    X 2        Y 1
  Element 1
    Types                   Red And Black, Soldier
    Representations         Point, Square, Mesh
    Distances                                           2
      Distance (m)          2
      Distance (m)          4
    Qualities                                           3
      Quality (from, to)    X 1        Y 2
      Quality (from, to)    X 2        Y 2
      Quality (from, to)    X 3        Y 2
```

Figure 92: Subjective test configuration UI

Funded by the Horizon Europe Framework Programme of the European Union

The test workflow is designed so that the participant is asked to rate the perceptual quality of each sequence after watching it. This is made possible via an immersive slider which allows the participants to rate the quality from 1 through 10 with textual guidelines to support the ratings (i.e., 1, 2 – very bad, 3, 4 – bad, 5, 6 – fair, 7, 8 – good, 9, 10 – very good). These ratings are stored in a CSV file with a unique numerical ID assigned to the test participant and a string describing the sequence they just watched. The timestamp of this action is also stored and can be used to synchronize with a separate questionnaire that the participants might fill in, among other usages.

Furthermore, while the test participants perform the test using the HMD, the tests can be monitored by simply looking at the Unity application on a computer screen. This includes tracking the participant's progress and any difficulties they might have during the test.

6.1.2 STEP-MR: A Subjective Testing and Eye-Tracking Platform for Dynamic Point Clouds in Mixed Reality

STEP-MR builds upon the initial platform, improving on the subjective testing functionality and adding the ability to conduct eye-tracking tests along with heatmap generation. The platform continues to use Unity and the Mixed Reality Toolkit (MRTK) 2 SDK from Microsoft as its basis.

The architecture of STEP-MR is shown in Figure 93. STEP-MR still utilizes Holographic Remoting, but the performance requirements of the workstation have been reduced thanks to code optimisations and design changes. The workstation used with STEP-MR contained an Intel i7-12700H CPU, 32 GB of DDR5-4800 MHz RAM, and an NVIDIA RTX A2000 GPU.

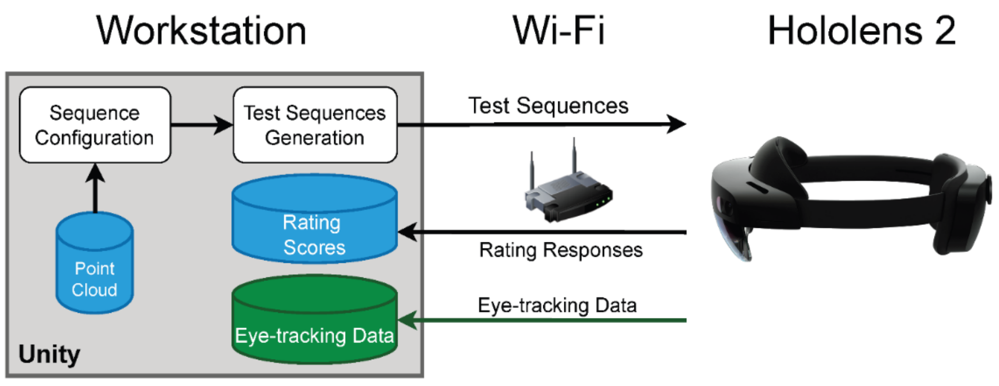


Figure 93: STEP-MR System Architecture

STEP-MR has three major functionalities: *(i)* point cloud previewing, *(ii)* subjective tests, and the newly added *(iii)* eye-tracking tests.

6.1.2.1 Point Cloud Previewing

The point cloud preview functionality has undergone only minor user-facing changes, namely *(i)* the quality slider has been replaced with buttons for ease of use, and *(ii)* the mesh option has been removed, since we focused our research solely on point clouds. These changes can be seen in Figure 94.

Funded by the Horizon Europe Framework Programme of the European Union

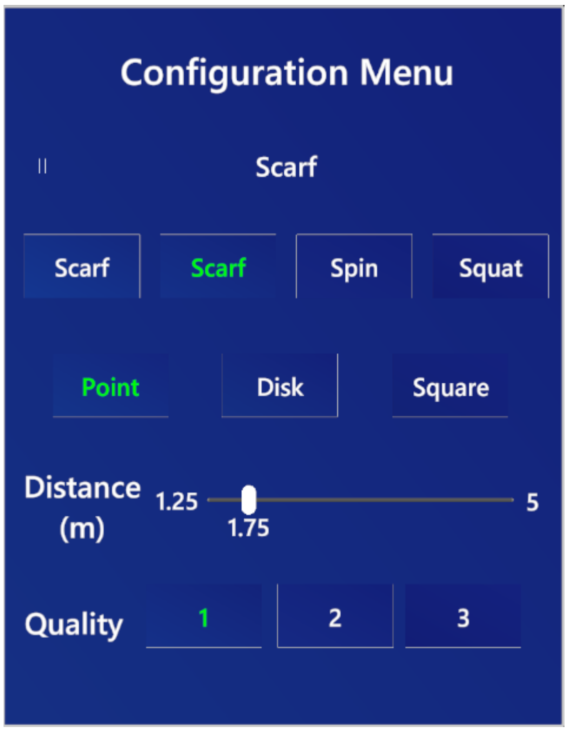


Figure 94: Updated configuration panel for point cloud previewing

Additionally, the back-end functionality of previewing scenes has been modified for demonstration purposes. To this extent, the controls of the configuration menu have also been bound to the keyboard, allowing the demonstration controller to adjust the point cloud (PC) configurations. This enables participants to experience the dynamic PC (DPC) content without needing to interact with the HoloLens 2 controls, which they might not be familiar with.

Finally, a general user experience change to the platform as a whole shows up when previewing scenes as well. The point cloud objects are now represented by Unity ScriptableObjects[14], allowing a PC's data to be stored in the Unity *Assets* folder, enabling easy reusability and adding even more modularity to the design.

6.1.2.2 Subjective Tests

Building upon the subjective testing functionality in the initial platform, STEP-MR now contains a set of buttons from 1 to 5 that allow the user to input their rating. The buttons replace the quality slider, a change that was made in response to participants' feedback indicating that the slider was difficult to use for people not accustomed to the controls of the HoloLens 2.

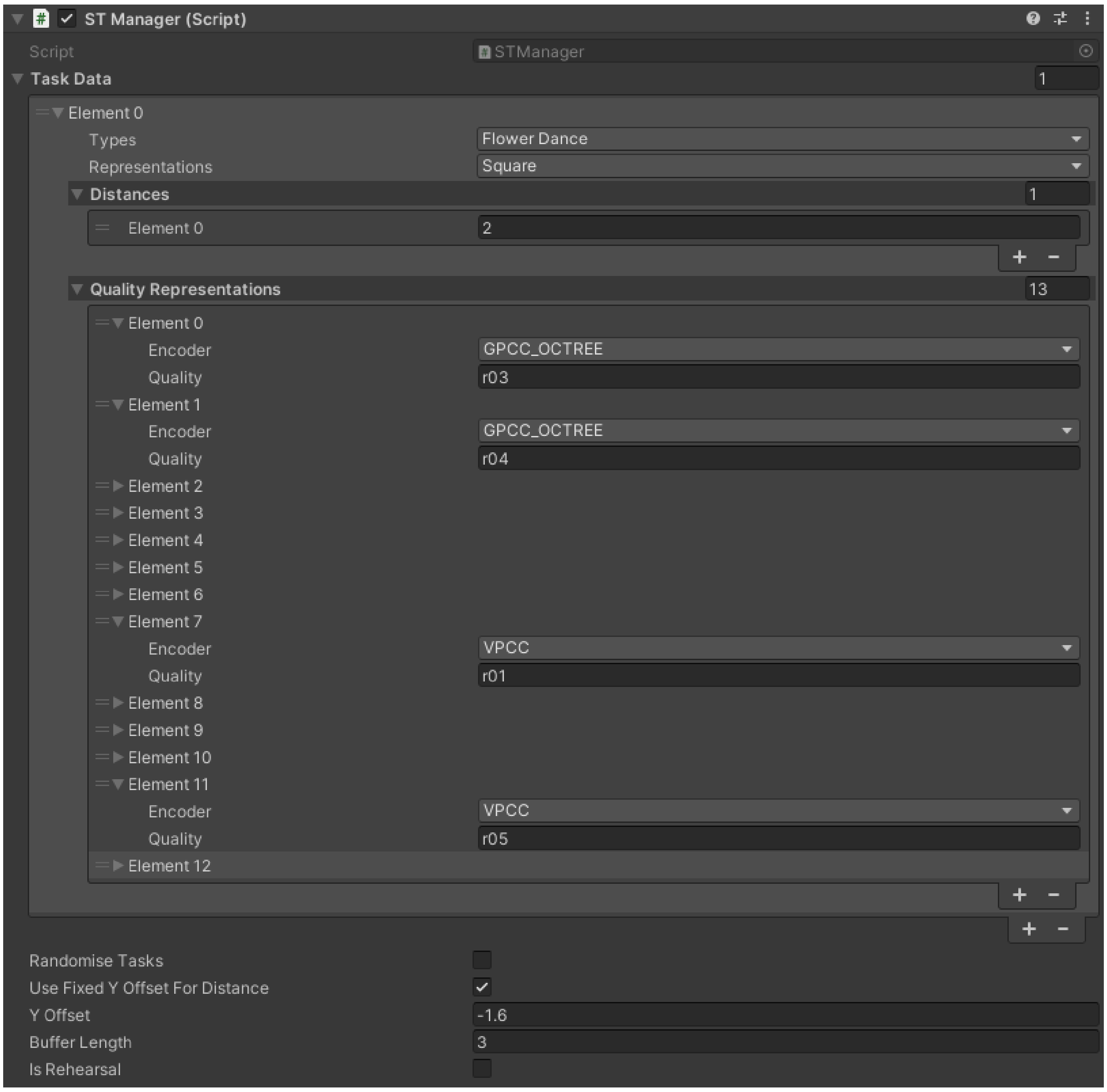
Since we wanted the second round of our subjective tests to include participants moving around the dynamic point cloud (DPC) sequences instead of remaining stationary, the quality rating buttons now appear in front of a participant's position the moment the sequence stops. This allows for immediate feedback from the participants. However, they are still required to go to a fixed location to start viewing the next sequence.

[14] https://docs.unity3d.com/2021.3/Documentation/Manual/class-ScriptableObject.html. Accessed: 06 July 2025.

Funded by the Horizon Europe Framework Programme of the European Union

Additionally, the DPC frames are loaded dynamically into memory. When the subjective test begins, the order of the sequences is randomised, and only the next five sequences are loaded into memory. The last played sequence is unloaded from memory, significantly reducing RAM usage. This optimisation allowed STEP-MR to be run from a laptop with 32 GB of DDR5-4800 MHz memory compared to a workstation with 64 GB memory of similar speed for the initial platform, further adding to the ease-of-use and modularity of the platform.

Finally, the configuration of the subjective testing contains support for various codecs, which was not present in our initial platform. The UI reflecting these changes can be seen in Figure 95.



ST Manager (Script)

Script	STManager
Task Data	1
Element 0	
Types	Flower Dance
Representations	Square
Distances	1
Element 0	2
Quality Representations	13
Element 0	
Encoder	GPCC_OCTREE
Quality	r03
Element 1	
Encoder	GPCC_OCTREE
Quality	r04
Element 2	
Element 3	
Element 4	
Element 5	
Element 6	
Element 7	
Encoder	VPCC
Quality	r01
Element 8	
Element 9	
Element 10	
Element 11	
Encoder	VPCC
Quality	r05
Element 12	
Randomise Tasks	☐
Use Fixed Y Offset For Distance	✓
Y Offset	-1.6
Buffer Length	3
Is Rehearsal	☐

Figure 95: STEP-MR Subjective TEsting COnfiguration UI

More information on the subjective testing methodology and how STEP-MR was used to perform the tests is found in [26].

Funded by the Horizon Europe Framework Programme of the European Union

6.1.2.3 Eye-Tracking Tests

The eye-tracking functionality is the most significant addition to STEP-MR. It consists of error measurement, point cloud viewing, fixation map generation, and heatmap visualisation. A brief overview of these functionalities is given below. The eye-tracking testing methodology and data processing workflow are further described in [26].

Error Measurement. Since the tool GazeMetrics [31] is not available for the HoloLens 2, we implemented this functionality ourselves. Nine 5 cm large spherical markers are placed 2 m away from the participant and displayed for 3 s each. The participant is asked to look at the markers as soon as they appear. The error data is stored as a CSV file per participant. The positioning and duration of the markers can be customised by the platform user.

Point Cloud Viewing. This functionality involves playing back the DPC sequences for the participant to walk around and observe. To minimise bias, the order of the sequences and the PCs' rotation at the start of a sequence are both randomised. The participant's gaze origin and gaze direction are saved every frame to a CSV file per PC. Users of STEP-MR can control which PCs are displayed and how much the rotation of the PC is randomised.

Fixation Map Generation. The fixation maps are generated by processing the eye-tracking data using both Unity C# and MATLAB code. The MATLAB scripts are available in the GitHub repository, and are heavily based on the work done in [32] whose authors also collaborated with us on our work with the subjective tests [26].

Firstly, the eye-tracking data is processed in a separate Unity scene. The Dispersion Threshold Identification (I-DT) [33] method is used, and the parameters for it are made available to STEP-MR users to customise. The results from this step are stored in a folder on the disk which the MATLAB scripts then access to further process the data into fixation maps.

Heatmap Visualisation. The fixation maps are loaded back into the Unity scene for data processing. PCs are played back as DPCs in the highest quality, and fixation heatmaps are visualised on the PC objects per frame. Screenshots of the heatmaps are captured from the front, back, left, and right views of the PCs. Users can customise the heatmap colours to suit their preferences. Example heatmaps are shown in Figure 105 and Figure 106.

Funded by the Horizon Europe Framework Programme of the European Union

6.2 DATASET AND RESULTS OF SUBJECTIVE TESTS FOR POINT CLOUDS IN AUGMENTED-REALITY ENVIRONMENTS

6.2.1 Experiment Tasks

As this work focuses on the usage of PCs in telepresence applications, we used four PC video sequences from the *8i Voxelized Full Bodies Database* [34]: *Loot*, *RedAndBlack*, *LongDress*, and *Soldier* as shown in Figure 96. The first two have a lower contrast content than the other two [19]. Each sequence captures a complete object using 42 RGB cameras operating at 30 fps for 10 seconds. We use the MPEG V-PCC reference software TMC2[15] to create compressed PCs by varying the QPs. This software already includes five sets of QPs defined in MPEG's CTC [35] with the G-QP and T-QP ranging from 16 to 32 and from 22 to 42, respectively. Three such pairs from the MPEG PCC software with the lowest, middle, and highest QPs (i.e., the rates R5, R3, and R1 in MPEG's software TMC2) are selected as follows:

- Q1 (R1): (G-QP, T-QP) = (32, 42)
- Q2 (R3): (G-QP, T-QP) = (24, 32)
- Q3 (R5): (G-QP, T-QP) = (16, 22)

Q3 is thus the best quality level. The bitrate of the objects decreases with increasing QPs (see Table 12).

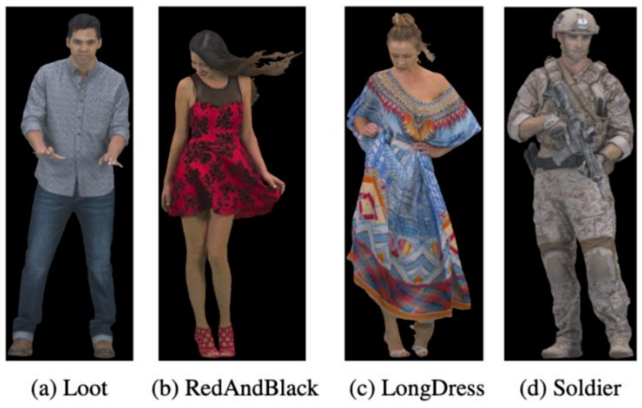


Figure 96: Test objects in 8i Voxelized Full Bodies Database

Table 12: Bitrates in Mbit/s of different quality levels of the PC objects

Video	Quality		
	Q1	Q2	Q3
Loot	2.28	5.63	16.68
LongDress	4.64	14.05	46.78
RedAndBlack	3.39	7.55	22.90
Soldier	4.38	11.58	35.29

[15] https://github.com/MPEGGroup/mpeg-pcc-tmc2. Accessed 16 November 2023.

Funded by the Horizon Europe Framework Programme of the European Union

We designed two tasks for each participant, with both tasks consisting of 18 sequences of length 10 s. Table 13 describes the sequences. Before the experiment, participants are asked to provide some background information, including age, gender, eyesight, and experience in viewing VR, AR, and MR content.

Table 13: Notation and description of the test sequences

Task	Notation	Description	Sequences
1	Qij	The video starts with quality Qi, then switches to Qj after 5 s.	*Loot* and *LongDress*
2	Qi_Dk	The video is watched at quality Qi at distance Dk.	*RedAndBlack* and *Soldier*

6.2.1.1 Task 1: Impact of Video Encoding and Quality Switches

The participant watches nine sequences for each of the two objects, including three sequences with static quality (Q1, Q2, and Q3) and six sequences with a quality switch in the middle of each sequence. The objects are placed 5 m from the participant so that the whole body can be viewed. *Loot* and *LongDress* are used in this task as they belong to low and high contrast levels, respectively.

6.2.1.2 Task 2: Impact of Viewing Distance

The participant watches static-quality sequences of the other two objects (*RedAndBlack* and *Soldier*) at quality levels Q1, Q2, and Q3 at three distances:

- D1: 1.25 m (only face and shoulder in FoV)
- D2: 2.5 m (only upper body in FoV)
- D3: 5 m (full body in FoV)

The order of tasks and sequences for each task are randomized for each participant. After watching each sequence, the participant is asked to rate the perceptual quality (i.e., 1, 2 – very bad, 3, 4 – bad, 5, 6 – fair, 7, 8 – good, 9, 10 – very good) through the immersive slider shown in Figure 97.

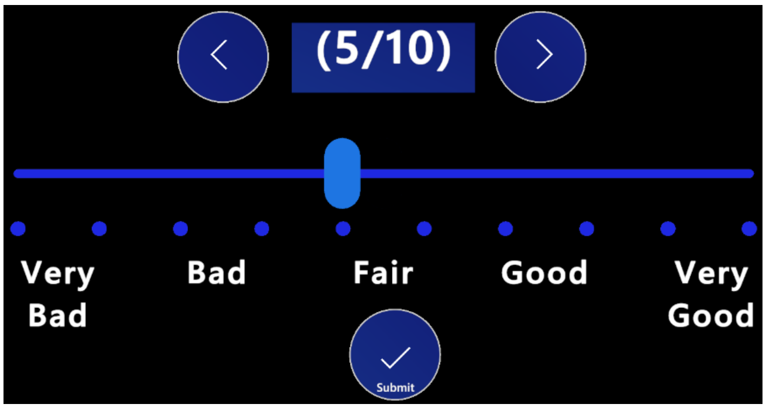


Figure 97: Immersive rating slider within the user interface of the HoloLens 2 as used during the subjective tests

Funded by the Horizon Europe Framework Programme of the European Union

After the experiment, participants are asked to provide feedback on their experience regarding levels of general discomfort, nausea, sweating, headache, or dizziness that they may have experienced. Participants also answer a question of whether they feel the PC objects are part of the real environment by selecting one of five options: *(i)* strongly disagree, *(ii)* disagree, *(iv)* neutral, *(i)* agree, and *(v)* strongly agree. The total duration of a single experiment is approximately 25 minutes.

6.2.2 Dataset Description

Our dataset of the subjective tests is published on GitHub:

➲ https://github.com/minhkstn/QoE-and-Immersion-of-Dynamic-Point-Cloud.

The dataset includes two files: `rating_score.csv` and `questionnaire_responses.csv`. This section describes the content of each file.

6.2.2.1 Rating Scores

Table 14 describes the structure of the data in file `rating_scores.csv`.

Table 14: Structure of Rating scores database

Column Name	Data Type	Description
participant	Int	The index of the participant
object	String	The name of the 3D object in the test sequence
start_quality	String	The quality in the first 5 s of the test sequence
end_quality	String	The quality in the last 5 s of the test sequence
distance	Float	The viewing distance (m)
start_gqp	Int	The G-QP in the first 5 s of the test sequence
start_tqp	Int	The T-QP in the first 5 s of the test sequence
end_gqp	Int	The G-QP in the last 5 s of the test sequence
end_tqp	Int	The T-QP in the last 5 s of the test sequence
qoe	Int	The score rated by the participant

6.2.2.2 Questionnaire Responses

Before the test, we asked the participants to provide some background information and tested their colour vision with an Ishihara test[16].

[16] https://en.wikipedia.org/wiki/Ishihara_test. Accessed 03 November 20223.

Funded by the Horizon Europe Framework Programme of the European Union

After the test, the participants answered a survey about cybersickness and the immersion level of the tested objects. The answers are stored in the table of file `questionnaire_responses.csv`. The structure of the table is described in Table 15.

Table 15: structure of questionnaire answers

Column Name	Data Type	Description
Participant	Int	The index of the participant
Which number do you see below?	Int	The first tested number for Ishihara test (must be 12)
Which number do you see below?	Int	The second tested number for Ishihara test (must be 45)
Which number do you see below?	Int	The third tested number for Ishihara test (must be 74)
During the assignment I felt... [General discomfort]	String	Level of agreement with options: Strongly disagree, Disagree, Neutral, Agree, Strongly agree
During the assignment I felt... [Nausea]	String	Level of agreement with options: Strongly disagree, Disagree, Neutral, Agree, Strongly agree
During the assignment I felt... [Sweating]	String	Level of agreement with options: Strongly disagree, Disagree, Neutral, Agree, Strongly agree
During the assignment I felt... [Headache]	String	Level of agreement with options: Strongly disagree, Disagree, Neutral, Agree, Strongly agree
During the assignment I felt... [Dizziness]	String	Level of agreement with options: Strongly disagree, Disagree, Neutral, Agree, Strongly agree
How would you describe the general experience? [I felt the objects were part of the real environment]	String	Level of agreement with options: Strongly disagree, Disagree, Neutral, Agree, Strongly agree
What is your gender?	String	Gender with options: Male, Female, Non-binary, Prefer not to say
What is your age group?	String	Age group with options: Under 18, 18-24, 25-34, 35-44, 45-54, 55-64, 65-74, 75-84, 85 or older
How often have you watched/experienced extended reality content (e.g, 360 videos/games, holographic content)?	String	Frequency with options: Never, Once, A few times, Monthly, Weekly, Daily

Funded by the Horizon Europe Framework Programme of the European Union

6.2.3 Subjective Test Results

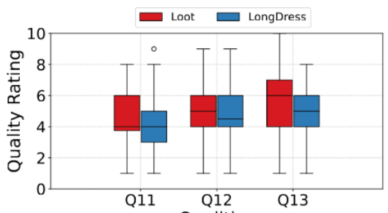
In this section, we provide an overview of the participants and present and analyse the subjective test results. We published our analysis in [27] [28].

6.2.3.1 Participants

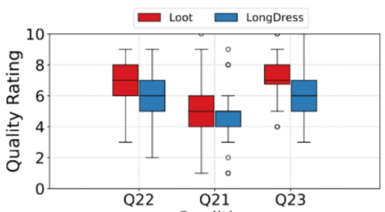
A total of 36 participants, who were recruited from the UNI-KLU, attended the subjective test, including 22 (61%) males, 13 (36%) females, and 1 (3%) non-binary. 3 (8%) were in the age group of 18 to 24 years, 18 (50%) were between 25 and 34, 12 (33%) between 34 and 44, 2 (6%) between 45 and 54, and 1 (3%) between 55 and 64. The colour vision of the participants is evaluated using the Ishihara test [36]. Four participants failed this test, so their ratings are excluded. Hence, the results in this section are gathered from 32 participants which is compliant with ITU-T P.919 [37].

6.2.3.2 Impact of Video Encoding

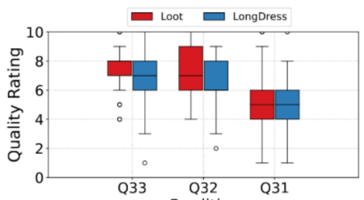
Figure 98 shows the quality ratings of the participants for the test sequences at different quality levels and quality switches. Regarding the sequences with static quality levels (i.e., Q11, Q22, Q33 in Figure 98(a)-(c)), it can be seen that objects encoded with lower qualities have lower scores. At least 75% of the viewers gave *Loot* (*LongDress*) a rating of 6 (5) or less for the lowest quality sequence, Q11. Their medians are both 4, which means a bad experience. With a higher-quality sequence, Q22, the median quality scores improve remarkably to 7 (i.e., good) and 6 (i.e., fair) for *Loot* and *LongDress*, respectively. For the highest-quality sequence, Q33, there is an improvement in quality ratings, but it is less remarkable than for Q22. *Loot* still receives good ratings from participants, with a median of 8 (i.e., good), while *LongDress* achieves ratings ranging from fair (median of 6) for Q22 to good (median of 7) for Q33. Furthermore, though Q33 can achieve very good ratings (9 or 10), the majority (at least 75%) of the participants rate this sequence at no more than 8. To statistically validate these claims, we used one-way ANOVA [38] and post-hoc comparison analysis using Tukey's HSD test [39]. According to the ANOVA results, there is a significant difference ($p < 0.001$) between the three quality levels. Post-hoc pairwise, Tukey's HSD reveals that quality ratings do not differ significantly ($p > 0.05$) between Q22 and Q33 for *Loot*, but do for *LongDress* ($p < 0.05$). Furthermore, there are significant p-values ($p < 0.001$) between Q11 and the others.



(a) Quality $Q1$ and switching from $Q1$.



(b) Quality $Q2$ and switching from $Q2$.



(c) Quality $Q3$ and switching from $Q3$.

Figure 98: Quality ratings for different quality levels and quality switches

Funded by the Horizon Europe Framework Programme of the European Union

6.2.3.3 Impact of Quality Switches

Figure 98 also describes the participant ratings for different quality switches, including switching up when the quality is increased and switching down when the quality is decreased. There is no remarkable improvement in the quality scores when the sequence starts at quality Q1 (i.e., Q11, Q12, and Q13). ANOVA analysis indicates no significant difference ($p > 0.05$) among the quality scores for both *Loot* ($p = 0.07$) and *LongDress* ($p = 0.13$). This can be attributed to the severe distortion of Q1 in the initial 5 s that affects the QoE when watching the entire 10 s video. Regarding switching down, when the quality changes from Q2 or Q3 to Q1 (i.e., Q21 or Q31), the quality ratings are markedly reduced compared to the constant-quality sequences (Q22 and Q33). However, there are no significant differences when the quality changes b tween Q2 and Q3. We conducted a paired samples t-test [40] to further validate this observation. It shows non-significant p-values between Q22 and Q23 (e.g., $p = 0.6136$ for *Loot*) as well as between Q33 and Q32 (e.g., $p = 0.1162$ for *LongDress*).

Combined with the results in the previous section, we claim that the end user hardly recognizes the quality differences between Q2 and Q3. Thus, we recommend that it is unnecessary to change the quality from Q2 to Q3 when the object is viewed at a distance of 5 m. This can remarkably reduce the amount of transferred data.

6.2.3.4 Impact of Viewing Distance

Figure 99 shows the quality ratings of the test sequences at different viewing distances. It is noticeable that distance significantly impacts the visual quality of the objects: the higher the viewing distance, the higher the quality scores. The reason is that, at a higher distance, it is harder for the viewers to recognize some quality distortions; thus, they give higher quality scores, which is comparable to what has been reported for traditional video sequences [41]. Additionally, we observe that to achieve the same visual quality, the object should be encoded with lower QPs (i.e., more data) if viewed closer. For example, *RedAndBlack* at quality Q1 is rated on average 4.8 at 5 m, and this object must be encoded at Q2 to gain a similar score (i.e., 4.9) if it is viewed at 1.25 m ($p = 0.5$ in a paired t-test).

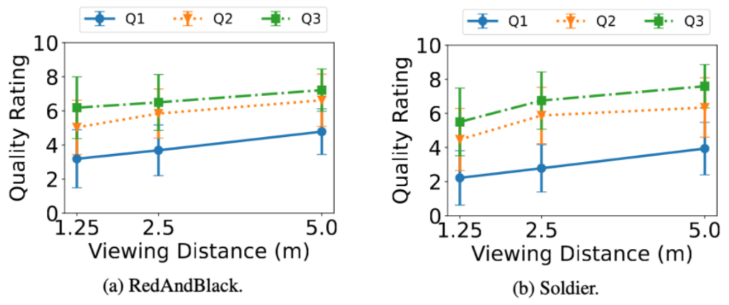


Figure 99: Average quality ratings for different distances

6.2.3.5 Impact of Content Characteristics

We also evaluate the impact of content characteristics on the visual perception of participants for both tasks, as shown in Figure 100. *Loot* and *RedAndBlack* achieve higher quality ratings in most cases. For example, the quality scores of *Loot* and *RedAndBlack* with quality Q1 viewed at distance D3 (i.e., 5 m) are 4.5 and 4.8, respectively. Under the same conditions, these figures for *LongDress* and *Soldier* are 4.2 and 3.9, respectively. This can be explained by the fact that participants are less sensitive to quality distortion/changes for the content with fewer contrast differences. This finding extends the results presented in the work [19] on 2D screens to an AR environment with AR HMDs, in which the texture of the objects is a crucial factor for viewers.

Funded by the Horizon Europe Framework Programme of the European Union

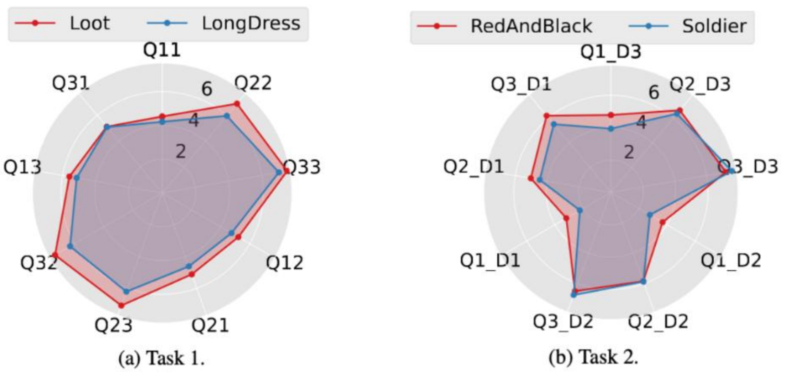


Figure 100: Average quality ratings of participants. It should be noted that the sequence Qii in Task 1 is equivalent to Qi_D3 (i ∈ {1, 2, 3}) in Task 2, encoded at quality Qi and viewed at 5 m.

6.2.3.6 Cybersickness in AR

The cybersickness levels of the participants are illustrated in Figure 101. Figure 101a shows that most of the participants did not feel symptoms of cybersickness in their experiment session that lasted about 25 minutes. 84% and 81% of them did not sweat or feel nauseated, respectively. The most common symptom is dizziness, but only 21% of the participants reported feeling dizzy during the test. Figure 101b provides more details about the symptoms of the participants who received cybersickness. No one suffers from all the symptoms mentioned above. There is only one person who experiences three symptoms, including sweating, headache, and dizziness. Three participants felt two symptoms, and six others received one symptom. On the contrary, in a similar-duration subjective test [42] where participants were watching videos with four characters in a room and dolphins in the ocean with VR HMDs, cybersickness was a serious problem that affected more than 90% of the viewers.

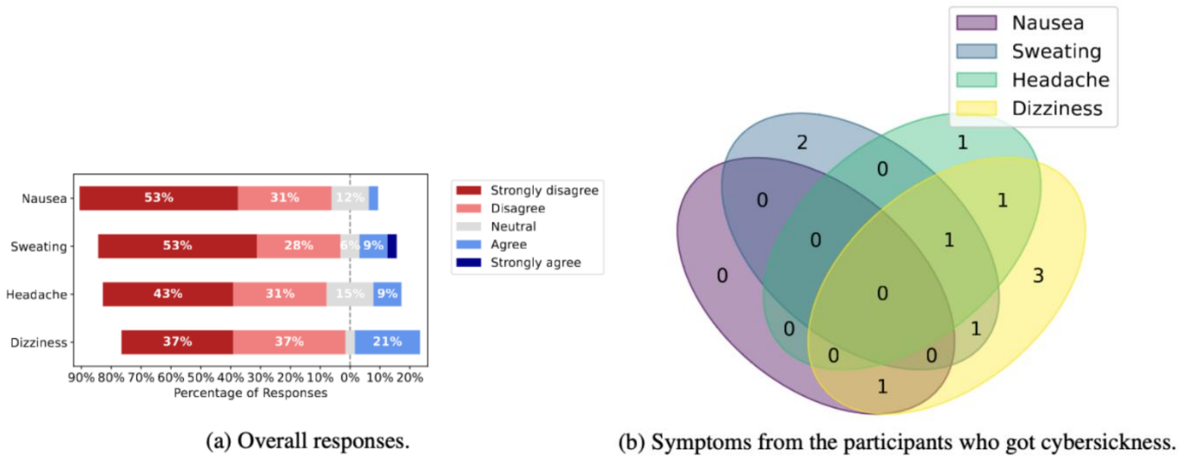


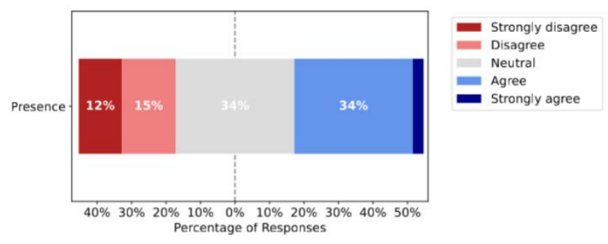
Figure 101: Cybersickness levels of the participants

6.2.3.7 Objects' Immersion Levels

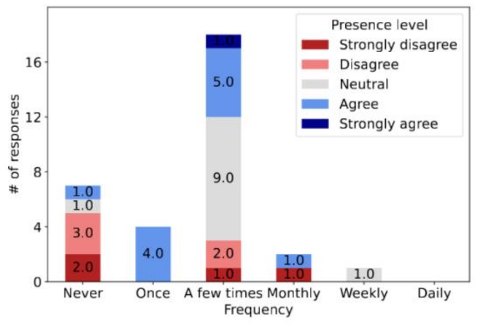
Figure 102 shows the immersion levels of digital objects in the physical world rated by the participants. It can be seen from the figure that 39% of the participants (strongly) agreed that the objects were part of the real environment. Only 27% of the participants (strongly) disagreed with this feeling. Therefore, the tested objects and HoloLens 2 provide the feeling of telepresence to some extent. However, some participants complained about the quality of some parts of the objects, even at the highest quality level. For example, the hair of *RedAndBlack* was perceived as blocky, and the heels of *LongDress* were missing in some

Funded by the Horizon Europe Framework Programme of the European Union

frames (see Figure 96). When we consider the impact of the participants' frequency of watching XR contents on the immersion level rating of the tested objects, there are two findings to be noted. First, most participants (5 out of 7 people) who have never watched XR content do not feel that the test sequences are real. Second, most experienced participants felt neutral or agreed that the objects were real. These people may have a good understanding of how 3D objects look in such environments and thus may have lower expectations in terms of feeling the presence of these digital objects.



(a) Level of feeling the presence of objects in the real environment.



(b) Frequency of watching XR contents vs. level of feeling the presence of objects.

Figure 102: Objects' immersion levels

6.2.4 QoE Models

Using the dataset acquired in this study, we evaluate the performance of supervised machine learning techniques in predicting the QoE for point clouds in AR environments. In addition, we also fine-tuned an existing QoE model, which was originally designed for 2D videos, called ITU-T P.1203 mode 0, to predict the QoE of point clouds.

These models can be integrated in the testbed of Open Call participants to predict the QoE of the end users.

6.2.4.1 Machine Learning-Based QoE Models

6.2.4.1.1 Data Preparation

We consider four influence factors, including encoding parameters, quality switching, viewing distance, and content characteristics. The first two factors are represented by the values of start and end QPs of sequences. The content characteristics can be represented by the bitrate of the encoded bitstream. As observed in our test, the objects have different bitrates even at the same quality level (same QPs). For this evaluation, each input data record comprises six features: *start G-QP, start T-QP, end G-QP, end T-QP, viewing distance, and bitrate*. The corresponding ratings of the participants are used as the learning targets. We received 1152 responses in total from 32 participants. After omitting outliers defined by the interquartile range method, 1107 responses are used as the input data. To receive a reliable and unbiased estimate of model performance, we use leave-one-out cross-validation. The input data is split into k groups, in which k − 1 groups are used as the training dataset, and the remaining one as the testing dataset. The process of splitting the data is repeated k times so that every group is used as a testing dataset once. There are, in total, 36 test sequences; hence, we have k = 36 so that the ratings for 35 test sequences are used for training, and the others are for testing.

Funded by the Horizon Europe Framework Programme of the European Union

6.2.4.1.2 Evaluation Results

Here, we train and evaluate common machine learning model for QoE prediction in AR environments. Their performance is reported in Table 16.

Table 16: Performance of machine learning models in predicting the QoE of point clouds in AR environments. The bold entry signifies the best performance.

QoE Model Type	R2 Score	MSE
Gradient Boosting Regressor	**0.8582**	**0.2874**
Random Forest Regressor	0.8384	0.3273
Decision Tree Classifier	0.8155	0.3738
Decision Tree Regressor	0.7954	0.4146
Polynomial Regression (Degree 2)	0.7650	0.4761

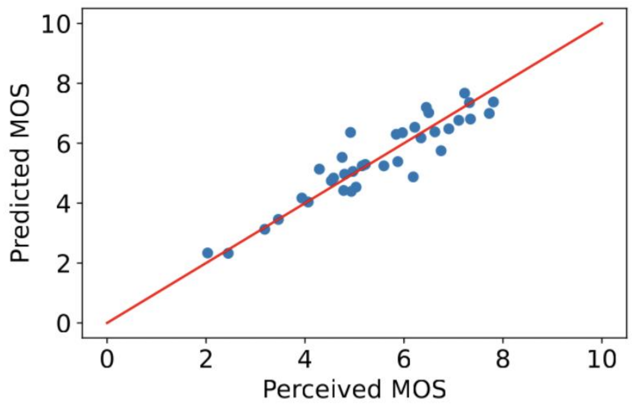


Figure 103: Perceived MOS (from our subjective test) versus predicted MOS using the Gradient Boosting Regressor. The red line represents the y = x line.

Figure 103 shows the correlation of the perceived MOS (from our subjective test) with respect to the predicted MOS using the Gradient Boosting Regressor. The predicted MOS is highly correlated (Pearson correlation coefficient = 0.93) with the perceived MOS, which was gathered from our subjective test.

Figure 104 presents feature importance scores of input features in the Gradient Boosting Regressor model using the Python scikit-learn library. A higher score means more importance when building a predictive model. It is highlighted that the end T-QP plays the most crucial role for the Gradient Boosting Regressor in predicting the QoE, followed by viewing distance and end G-QP. Their importance scores are 0.32, 0.24, and 0.19, respectively. The content characteristics represented by the bitrate are the least relevant feature of the prediction model, with an importance score of 0.05.

Funded by the Horizon Europe Framework Programme of the European Union

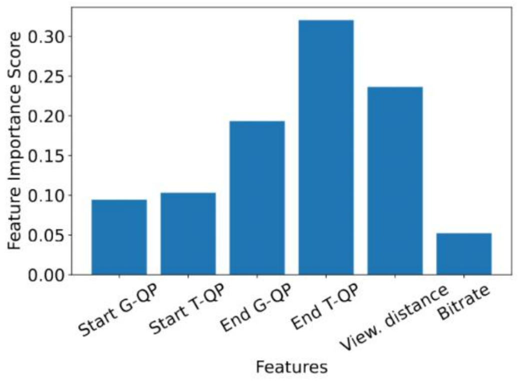


Figure 104: Feature importance scores of input features in Gradient Boosting Regressor.

6.2.4.2 Fine-tuned P.1203 Mode 0 Model

The ITU put in a great deal of effort in video quality estimation models, namely ITU-T P.1203.1[17]. This P.1203 model was implemented and published on GitHub[18]. The model's inputs include video characteristics (i.e., bitrate, framerate, codec, and frame size), streaming parameters (i.e., stall events), and viewing conditions (i.e., device type and viewing distance). Point cloud streaming also shares some parameters that can be used in the P.1203 model, such as bitrate, framerate, stall events, and viewing distance. However, as the coefficients in the original P.1203 model were determined from a training phase based on a subjective database for 2D videos, they need to be re-trained with a new subjective database for point cloud streaming.

We split this dataset into a training set and a validation set. We train the coefficients of the P.1203 model with the former set and validate its performance with the latter one. The results show that our fine-tuned P.1203 model outperforms the original model from the ITU. Our model achieves an RMSE of 0.813, compared to 0.887 of the original P.1203 model with the training set. The PLCC and SRCC of our fine-tuned model are also significantly higher than that of ITU's model (see Table 17).

Table 17: Performance of the original P.1203 mode 0 and our fine-tuned P.1203 with training and validation dataset.

	Training			Validation		
	PLCC ↑	**SRCC ↑**	**RMSE ↓**	**PLCC ↑**	**SRCC ↑**	**RMSE ↓**
ITU P.1203	0.766	0.785	0.887	0.918	0.829	1.032
Fine-tuned ITU P.1203	**0.919**	**0.953**	**0.813**	**0.958**	**0.828**	**0.955**

[17] https://www.itu.int/rec/T-REC-P.1203.1/en. Accessed: 02 November 2023.

[18] https://github.com/itu-p1203/itu-p1203. Accessed: 16 November 2023.

Funded by the Horizon Europe Framework Programme of the European Union

The fine-tuned P.1203 model is published in

➲ https://github.com/minhkstn/itu-p1203-point-clouds.

This work was presented at the Mile-High Video 2024 Conference [43].

6.3 SECOND SUBJECTIVE STUDY AND COMPRESSED POINT CLOUD DATASET WITH EYE TRACKING AND QUALITY ASSESSMENT IN MIXED REALITY

We used an updated version of our subjective testing platform (see Section 6.1) to perform another round of subjective testing, including eye-tracking tests. We compiled the point clouds used for these tests and the results from them in an open-source **Com**pressed **P**oint cloud dataset with **E**ye-tracking and **Q**uality assessment in **M**ixed **R**eality (**ComPEQ-MR**) [26]. The tested dynamic point clouds (DPCs) are made publicly available for reproducibility: https://ftp.itec.aau.at/datasets/ComPEQ-MR/.

The dataset comprises 4 uncompressed (raw) DPCs as well as compressed versions processed by Moving Picture Experts Group (MPEG) reference tools (*i.e.*, VPCC and 2 GPCC variants). The dataset includes eye-tracking data of 41 study participants watching the raw DPCs with six degrees of freedom (6DoF), yielding 164 visual attention maps. We analyse this data and present head and gaze movement results here. We conducted subjective tests to assess the quality of the DPCs, each both uncompressed and compressed with 12 levels of distortion, resulting in 2132 quality scores. This section presents the QoE performance results of the compression techniques, the factors with significant impact on participant ratings, and the correlation of the objective Peak Signal-to-Noise Ratio (PSNR) metrics with Mean Opinion Scores (MOS). The results indicate superior performance of the VPCC codec as well as significant variations in quality ratings based on codec choice, bitrate, and quality/distortion level, providing insights for optimising point cloud video compression in MR applications. Finally, making use of the subjective scores, we developed models for QoE prediction for DPCs compressed using the pertinent MPEG tools. We present the models and their prediction results, noting that the fine-tuned ITU-T P.1203 models exhibit good correlation with the subjective ratings.

6.3.1 ComPEQ-MR Point Cloud Dataset

The study described in [26] comprised two tasks: eye-tracking testing and subjective quality assessment. This section provides an overview of the dataset collected in the ComPEQ-MR study and recaps the testing methodology used. Sections 0 and 6.3.3 discuss expanded results from the findings made in the ComPEQ-MR study.

6.3.1.1 Participants

41 persons participated in this study [26], including 19 (46%) females and 22 (54%) males. 18 (45%) persons were in the age group of 18 to 24 years, 14 (35%) were between 25 and 34, 7 (17.5%) between 35 and 44, and 1 (2.5%) between 55 and 64. 17 persons (41.5%) had never used an AR headset before, 16 (39%) fewer than 5 times, 7 (17%) between 5 and 20 times, and 1 (2.5%) person was a frequent user (more than 20 times). Most of the participants were students or staff of the University of Klagenfurt. All of them passed the Ishihara colour vision test before the experiment.

Funded by the Horizon Europe Framework Programme of the European Union

Using a questionnaire, the participants were asked about demographic data (as partially summarized above), their feelings of immersion in the MR scenes, and potential symptoms of AR sickness (before and after the test).

6.3.1.2 Point Cloud Video Sequences

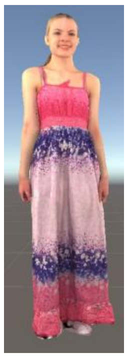
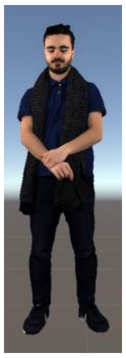
The state-of-the-art uncompressed, voxelised, 10-bit point cloud dataset UVG-VPC [44] was deployed. This dataset comprises 12 video sequences of humans with various content characteristics, captured at a frame rate of 25 fps, each 10 s long. Four DPCs, as seen in Table 18, were selected that have large differences in the criteria *spatial information* (SI), *temporal information* (TI), and *colourfulness* (CF) to cover a wide diversity: *BlueSpin*, *CasualSquat*, *FlowerDance*, and *ReadyForWinter*.

These DPCs were compressed using MPEG's reference software tools for VPCC and GPCC point cloud compression, the latter deploying the specific tool combinations GPCC-Oct-Pred (Octree and Predlift modes) and GPCC-Tri-RAHT (Trisoup and RAHT modes) to process the point clouds. Using different quantisation and other parameter configurations (Table 19), based on the Common Test Conditions for each tool [35] [45] [45] [46], the sequences were compressed into 12 different distortion levels: 5 quality levels for VPCC ($r01 \ldots r05$), 3 for GPCC-Oct-Pred ($r03 \ldots r05$), and 4 for GPCC-Tri-RAHT ($r01 \ldots r04$). Thus, together with the uncompressed version, 13 quality levels per DPC are contained in the ComPEQ-MR dataset, leading to a total of 52 DPC sequences. The bitrates of the 13 representations are shown in Table 20.

.

Funded by the Horizon Europe Framework Programme of the European Union

Table 18: UVG-VPC sequences used in this open dataset.

Name	Description	Snapshot
BlueSpin	A person wearing a blue t-shirt and spinning at a consistent rate. **SI**: 20.8 **TI**: 8.0 **CF**: 8.6	
CasualSquat	A person wearing a striped shirt and jeans in the performance of a squat exercise. **SI**: 53.5 **TI**: 19.0 **CF**: 11.5	
FlowerDance	A person in a long, flowing dress spinning and twirling. **SI**: 43.9 **TI**: 22.3 **CF**: 25.3	
ReadyForWinter	A person donning a beanie and a scarf. **SI**: 20.6 **TI**: 11.5 **CF**: 7.8	

Funded by the Horizon Europe Framework Programme of the European Union

Table 19: Encoder parameters to generate compressed dynamic point clouds

Compression		Quality Levels				
		r01	r02	r03	r04	r05
VPCC	Geometry QP	36	32	28	20	16
	Texture QP	47	42	37	27	22
GPCC-Oct-Pred	QP	-	-	40	34	28
	Depth	-	-	0.5	0.75	0.875
GPCC-Tri-RAHT	QP	40	34	28	22	-
	Level	5	4	3	2	-

Table 20: Bitrates of the compressed DPCs in Mbit/s

Point Cloud	Raw	VPCC					GPCC-Oct-Pred			GPCC-Tri-RAHT			
	vox10	r01	r02	r03	r04	r05	r03	r04	r05	r01	r02	r03	r04
BlueSpin	924.81	1.43	1.88	2.60	6.17	10.34	1.37	4.61	10.99	0.99	2.17	4.87	10.98
CasualSquat	829.47	2.05	3.21	5.46	15.98	25.49	1.79	8.04	23.66	3.83	9.26	19.71	38.78
FlowerDance	1019.65	2.18	3.47	5.84	18.69	32.24	1.98	8.64	25.09	3.03	8.02	19.09	39.91
ReadyForWinter	1072.33	1.62	2.12	2.96	7.65	13.59	1.56	5.29	13.42	1.00	2.29	6.00	14.24

6.3.1.3 Eye-Tracking Data

Before the study began, the in-built eye calibration of the HoloLens 2[19] was performed for each participant, since the eye-tracking services of the HoloLens do not function without calibration. This actual eye-tracking task consisted of two subtasks: *(i)* error measurement and *(ii)* watching the PC videos.

The second subtask consisted of the participants watching 20 s long uncompressed (voxelised 10-bit format) PC sequences of the four DPCs. The participants were allowed to move freely in the space of the test room (6DoF within a 4 m × 4 m area), but were required to return to the starting point before starting the next sequence. The participant's gaze origin and gaze direction were stored once per DPC frame. The order of the DPCs was randomised among the participants to avoid bias. The subtasks were alternated until the participant had watched all four sequences.

The data obtained from this task were processed to generate fixation heatmaps for the DPCs. The detailed information about how the eye-tracking data was processed can be found in [26]. In short, we obtain the accuracy of the eye tracking data per marker and use a threshold of angle and time to discard unintentional gazes and identify fixations. The fixation gazes are then mapped on to the point cloud object, and noisy data is filtered out. This process is performed for each user and overall fixation maps are generated by summing up the fixations for all users. These fixation maps are represented as heatmaps, as can be seen in Figure 105 and Figure 106.

[19] https://learn.microsoft.com/en-us/windows/mixed-reality/design/eye-tracking

Funded by the Horizon Europe Framework Programme of the European Union

The heatmap images for every frame, along with fixation weights for each point of the frames for all four PCs, were made available in the ComPEQ-MR dataset.

6.3.1.4 Quality Ratings

Following eye-tracking tests, participants in the subjective study rated their QoE for DPCs presented via the Microsoft HoloLens 2. Each subject viewed 52 DPCs, as described earlier, each lasting 20 s, in randomised order. They began viewing a DPC (a character) from 2 m away and could then move freely within a 4 m × 4 m area. After each DPC presentation, participants entered their QoE score using a user-friendly HoloLens 2 interface, returned to the initial position, and proceeded to the next DPC. The study employed the single-stimulus Absolute Category Rating (ACR) methodology with a 5-level Likert scale (5–excellent to 1–bad) and adhered to relevant ITU recommendations as outlined in [26]. Each session lasted approximately 30 min.

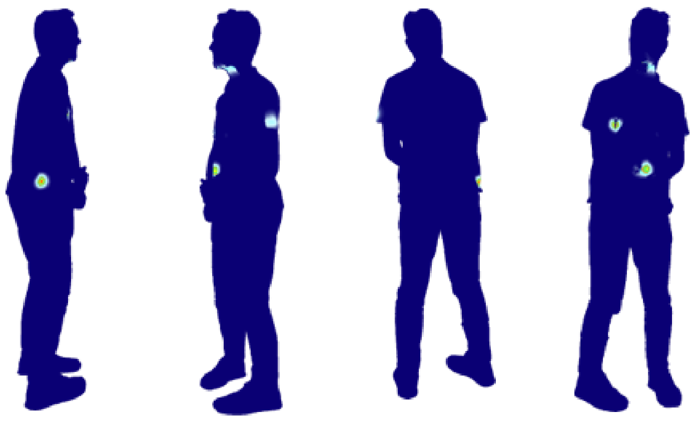


Figure 105: Fixation Heatmap of a BlueSpin frame in four views: Back, front, right, and left.

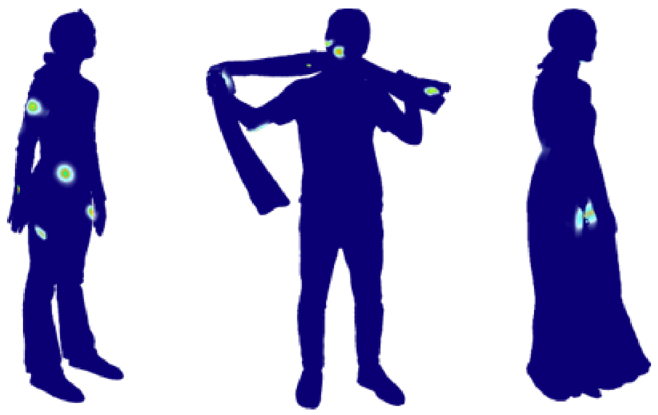


Figure 106: Fixation Heatmaps of the front view of CasualSquat, ReadyForwinter, and FlowerDance frames.

Funded by the Horizon Europe Framework Programme of the European Union

6.3.2 Eye-Tracking Results

From the eye-tracking study described earlier, we collected participants' gaze data in the form of their gaze origin and gaze direction for every frame. This data was utilised to compile visual fixation heatmaps for the participants, as described in [26]. We analysed this data further to obtain more insight into the overall behaviour of the participants with regard to head and gaze movement. This section discusses selected results from this analysis. The code for this analysis, along with more results not discussed here, can be found on GitHub: https://github.com/shivivats/PointCloudDataAnalysis.

6.3.2.1 Gaze Data Overview

Unity records the gaze origin and gaze direction as a set of *x*, *y*, and *z* values for every frame using the MRTK 2 library. The gaze origin is a 3-dimensional vector representing the participant's position offset from the scene origin in meters, as measured by Unity. The scene origin in our case was the point on the floor directly below the participant when they started the eye-tracking task. The gaze direction is represented as a 3-dimensional unit vector in degrees.

Since we allowed the participants to have six degrees of freedom during the tests, a change in the gaze origin can be brought about by either the participants moving their head in any way (tilting, shaking, or rolling) or by them moving their whole body (or just the head) to a new location. From now on, we will refer to the change in the gaze origin as *head movement*. A change in the gaze direction, referred to as *gaze movement*, results from the participants moving their irises, with or without moving their heads.

6.3.2.2 Preprocessing Gaze Data

It must be noted that, unlike processing the eye-tracking data to generate fixation maps [26], no pre-processing for accuracy is needed in this case since we only concern ourselves with the difference between a participant's gaze on a frame-to-frame basis, and removing the average angular error every frame would not impact these results.

The first 1.5 s were discarded to account for possibly erratic initial movements. After extracting the head movement and gaze movement change per frame, and consequently per second, the gaze data was filtered using the interquartile range (IQR) method [47] per user per point cloud. After filtering, we were left with 2857 data points of gaze and head movement per second.

6.3.2.3 Analysing Head and Gaze Movement

The analysis of the filtered data yielded the following results. The average head movement per participant was found to be 0.478 m/s, with a standard deviation of 0.385 m/s. The average gaze movement per participant was 65.728 °/s with a standard deviation of 57.381 °/s.

We classified the participants into three movement categories based on these movement velocities. Participants in the lowest and the highest 25 percentiles based on the respective velocity were placed in the *Low* and *High* categories, respectively, while the rest were placed in the *Medium* category.

The thresholds obtained for low and high head movement were determined to be 0.368 m/s and 0.630 m/s, respectively. Similarly, the thresholds for low and high gaze movements were set at 47.682 °/s and 85.913 °/s, respectively.

Funded by the Horizon Europe Framework Programme of the European Union

Interestingly, based on these thresholds, the same number of participants were classified into the three categories for both movement types. Namely, 11, 19, and 11 participants were classified into the *Low*, *Medium*, and *High* categories, respectively. However, only some participants belong to the same category for both movements. More specifically, only

- 4 out of 11 low head movement participants also have low gaze movement,
- 8 out of 19 medium head movement participants also have medium gaze movement,
- 7 out of 11 high head movement participants also have high gaze movement.

Figure 107 shows the average head and gaze movements of the participants in the three head movement categories across all videos, represented by the coloured dots. The figure also contains the head and gaze movement per point cloud, averaged across all participants. These values, represented by the red symbols, are as follows:

- *CasualSquat*: 0.542 m/s and 76.9 °/s
- *FlowerDance*: 0.478 m/s and 64.2 °/s
- *BlueSpin*: 0.450 m/s and 64.5 °/s
- *ReadyForWinter*: 0.445 m/s and 57.6 °/s.

CasualSquat is the DPC with the highest movement in both categories, which can be attributed to its DPC sequence having the most movement due to the squatting motion. Similarly, the *ReadyForWinter* object has the least movement in both categories since the sequence is simply a person tying and untying a scarf around their neck.

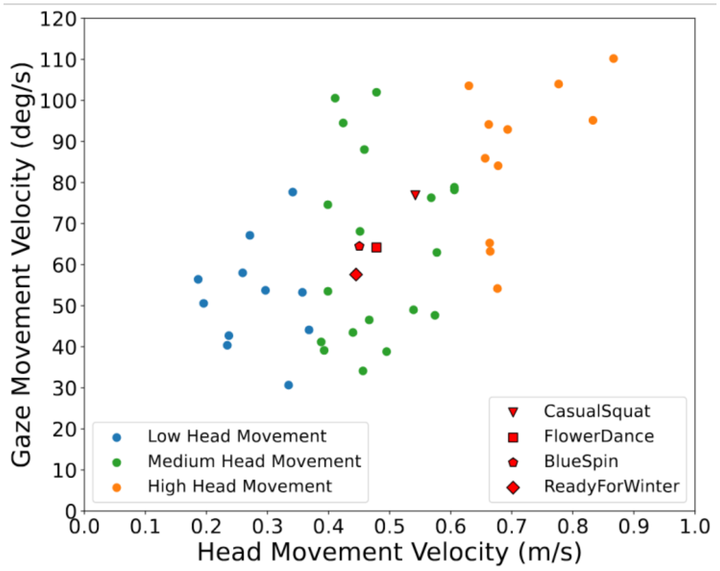
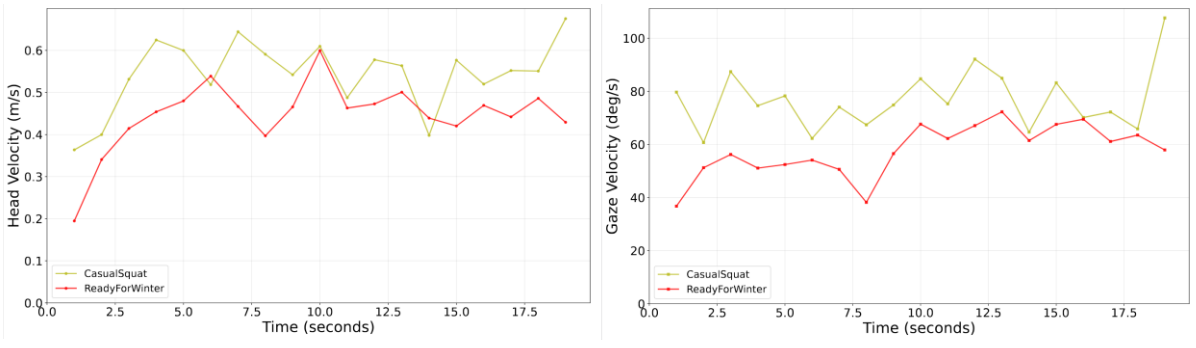


Figure 107: Average head vs. gaze movement velocities per participant for each movement category and overall for point clouds.

6.3.2.4 Head and Gaze Movement over Time

Figure 108 depicts the head and gaze movement over time for the *CasualSquat* and *ReadyForWinter* point clouds. The points on the graph represent the movement averaged over all participants for the previous one second. These DPC objects were chosen as they represent the most and least amount of movement, as discussed in the previous subsection. The difference in the participants' behaviour is evident in the graphs, where the participants are shown to have more movement for the *CasualSquat* object almost the whole time they are watching the DPC sequences.

Funded by the Horizon Europe Framework Programme of the European Union



(a) Head movement

(b) Gaze movement

Figure 108: Head and gaze movement over time for CasualSquat and ReadyForWinter

6.3.3 Subjective Study Results

Basic results of the ComPEQ-MR subjective study were already described in D4.2. Those results are briefly recapped in the first two subsections. The remainder of the section presents advanced results and discussions of the subjective study.

6.3.3.1 Opinion Scores

The raw opinion scores of the participants for all tested DPCs are shown in Figure 109. Some video sequences receive consistent scores from all participants. For example, the 18th and 41st videos (*i.e.*, *BlueSpin* and *CasualSquat* encoded by GPCC-Tri-RAHT at quality r01, respectively) have mostly all low scores (i.e., scores 1 or 2), while the 5th video gets high scores (*i.e.*, scores 4 and 5) from most of the participants. Regarding the participants, there are some participants who are not satisfied with the quality of the video sequences (*e.g.*, participants 2 and 30), while some others feel the opposite (*e.g.*, participant 17).

Mean Opinion Scores (MOS) were calculated according to ITU-R BT.500-15. 95% Confidence Intervals (CI) were also computed. In the subjective tests, no outliers were detected.

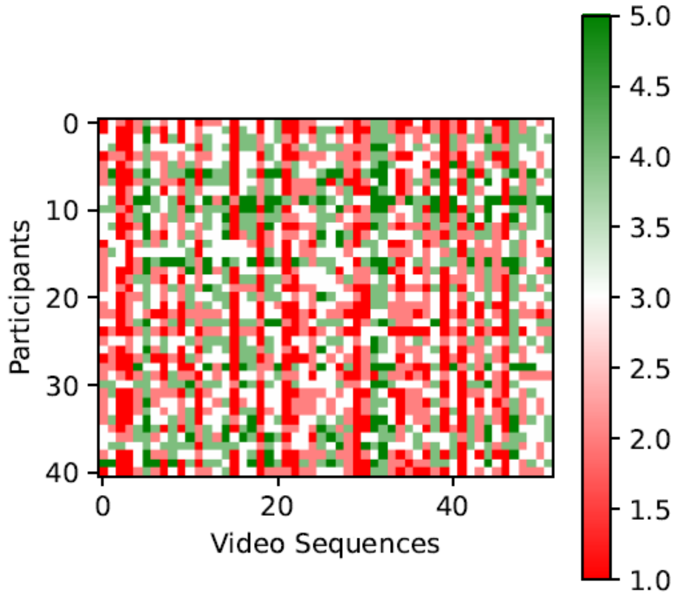


Figure 109 RAW OPINION SCORES

Funded by the Horizon Europe Framework Programme of the European Union

6.3.3.2 QoE Performance of the Compression Algorithms

Figure 110 compares the basic QoE (rate vs. MOS) performance of the compression algorithms (codecs) used in the study. When using – for the sake of fair comparison – the number of *bits per point* (bpp) for the rate metric, VPCC clearly achieves the best QoE (MOS) results, in accordance with findings of related work [48]. GPCC-Oct-Pred provides lower MOS than GPCC-Tri-RAHT, which is opposite to [48].

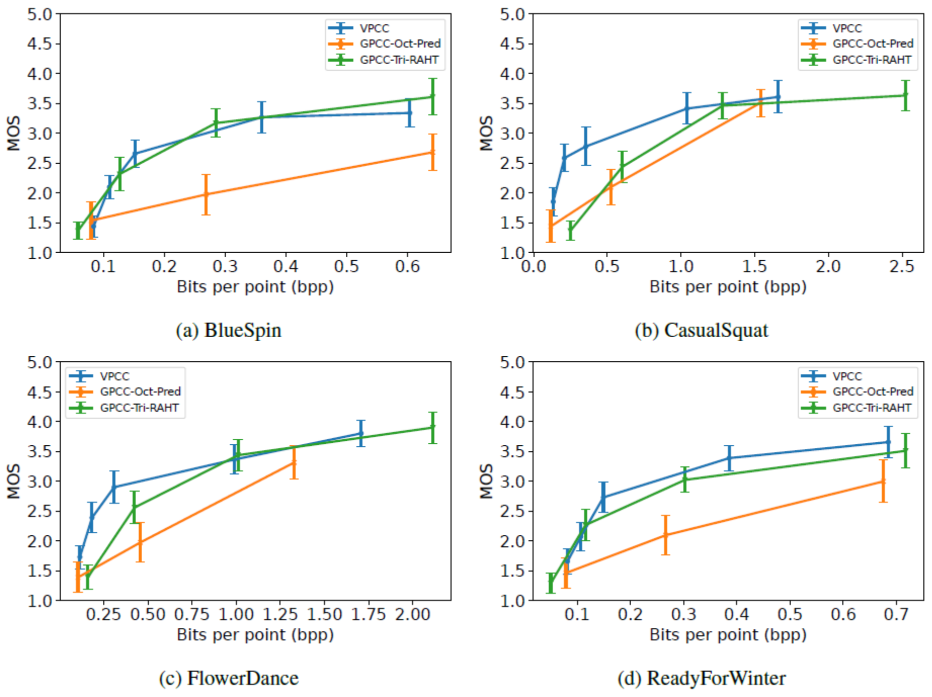


Figure 110: Bits per point vs. MOS for each video. The error bar is the 95% CI.

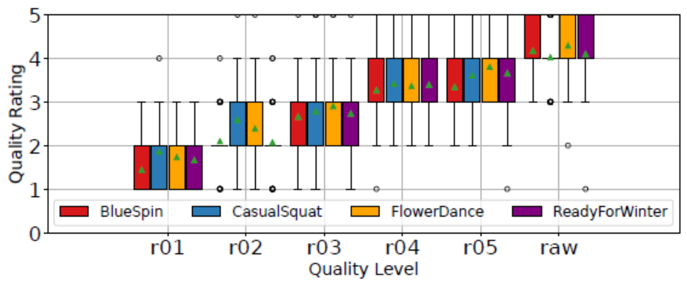
6.3.3.3 Impact Factors

First, we study 3 factors that may have a statistically significant impact on the quality ratings of the participants: compression algorithm (codec), encoding quality (quality level), and DPC content (characteristics). We utilize the three-way ANOVA (Analysis of Variance) test to determine the potential impact. The ANOVA results show that *codec* and *quality level* as well as the interaction between these two factors have a statistically significant effect (p < 0.001) on the ratings of participants. On the contrary, content and other interactions involving the content do *not* significantly contribute to explaining the variance in the ratings (p > 0.05).

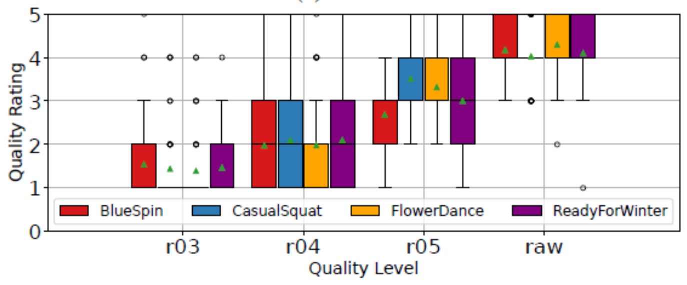
1) Impact of Quality Level

Figure 111 shows the quality ratings for various quality levels and codecs, including the raw content version. Generally, a higher quality level achieves better scores, regardless of the codecs and video content; and the raw/uncompressed DPCs always get the highest scores, close to 5. For the VPCC codec (Figure 111a), for example, the DPCs encoded at quality level *r*01 are rated on average less than 2 out of 5, that means a *poor* visual quality, whereas quality *r*05 can achieve up to 3.8, close to a *good* score.

Funded by the Horizon Europe Framework Programme of the European Union



(a) VPCC



(b) GPCC-Oct-Pred

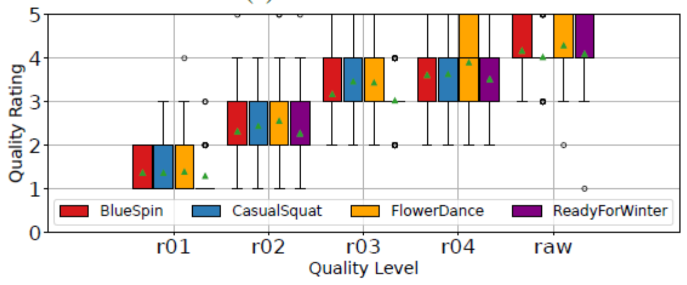


Figure 111: Quality ratings. Green triangles ▲ are average ratings.

Post-hoc pairwise, Tukey's HSD test (Honestly Significant Difference) reveals that quality ratings do *not* differ significantly ($p > 0.05$) between:

- For VPCC: *(i) r*04 and *r*05 for all contents, *(ii) r*02 and *r*03 for *CasualSquat*, *(iii) r*03 and *r*04 for *FlowerDance*, *(iv) r*01 and *r*02 for *ReadyForWinter*.

- For GPCC-Oct-Pred: *r*03 and *r*04 for *BlueSpin*.

- For GPCC-Tri-RAHT: *r*03 and *r*04 for *BlueSpin*, *CasualSquat*, and *FlowerDance*.

These findings are similar to [49] and indicate that, in specific cases, DPCs can be encoded into lower quality levels (with lower bitrates) to conserve bandwidth during transmission, without significantly compromising on the user's QoE.

2) Impact of Point Cloud Codec

As mentioned, there are significant differences in quality ratings across the codecs (VPCC, GPCC-Oct-Pred, and GPCC-Tri-RAHT). To further explore these discrepancies, a post-hoc analysis using Tukey's HSD test was performed. This analysis confirmed that the quality ratings for all pairs of encoders were significantly different from each other ($p < 0.05$). These findings imply that the choice of encoder has a notable impact on the quality ratings.

Funded by the Horizon Europe Framework Programme of the European Union

3) Impact of Prior Usage of an AR Headset

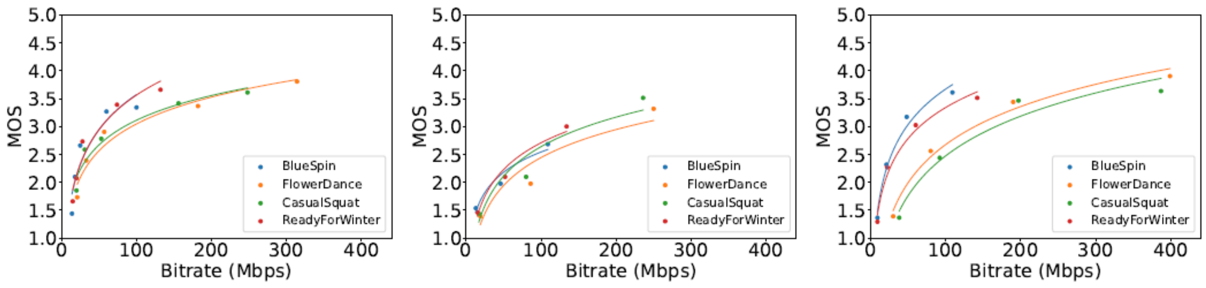
As initially indicated, there is quite some diversity in the participants' prior experience (or, frequency) of AR headset usage. Thus, ANOVA analysis and Tukey's HSD tests were performed as well. The results initially showed that this factor does have a statistically significant impact on the quality ratings. More precisely, the single frequent AR headset user ("more than 20 times") turned out to have rated differently. After removing that participant from the analyses, the results indicate that the prior usage frequency of an AR headset does *not* have a statistically significant impact on the quality ratings.

6.3.3.4 Further Analyses of the QoE Results

1) MOS vs. Bitrate

Figure 112 shows the relationship between bitrate and MOS for the three codecs. The curves are fitted using a logarithmic function.

MOS and bitrate correlate with each other well, except in cases of unusually high bitrates, as seen in Figure 112a and Figure 112c. This suggests that an increase in bitrate beyond a certain threshold does not lead to a corresponding improvement in perceived quality, confirming the finding stated earlier.



(a) VPCC (b) GPCC-Oct-Pred (c) GPCC-Tri-RAHT

Figure 112: MOS vs. bitrate

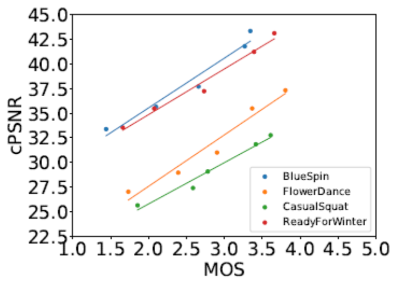
2) MOS vs. PSNR

PSNR metrics for the DPCs were computed using the MPEG *mpeg-pcc-dmetric* tool[20]. Figure 113 shows that higher PSNR values correspond with higher MOS, as would be expected. However, interesting conclusions can be derived from Figure 114, which shows the Spearman and Pearson correlation coefficients of colour PSNR (cPSNR) and geometry PSNR (gPSNR) with MOS for all three codecs.

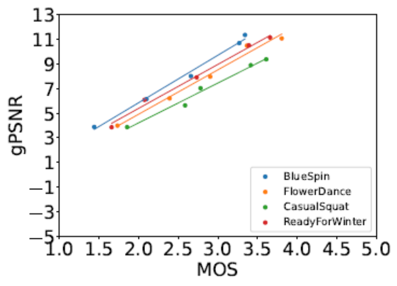
The correlation between cPSNR and MOS is significantly lower than the correlation between gPSNR and MOS, for both correlation coefficients. This suggests that, even though there is a general trend for the MOS values to increase as cPSNR increases, such an increase is not always guaranteed or proportional.

[20] https://github. com/MPEGGroup/mpeg-pcc-dmetric

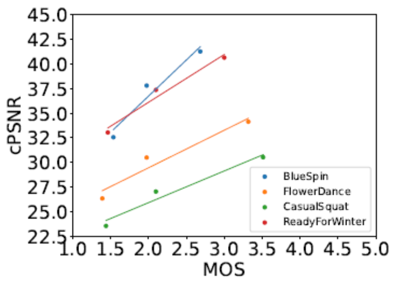
Funded by the Horizon Europe Framework Programme of the European Union



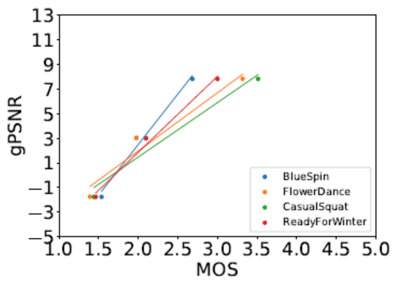
(a) VPCC: cPSNR



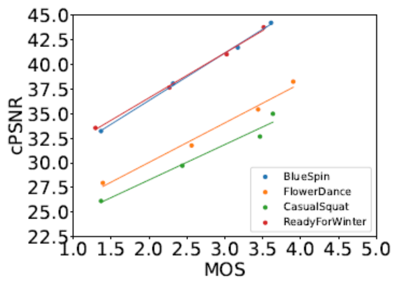
(b) VPCC: gPSNR



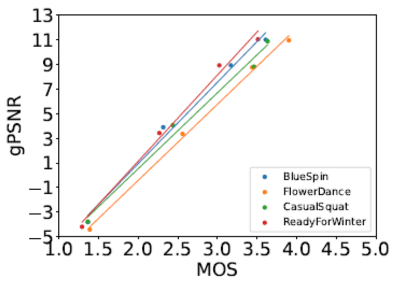
(c) GPCC-Oct-Pred: cPSNR



(d) GPCC-Oct-Pred: gPSNR



(e) GPCC-Tri-RAHT: cPSNR



(f) GPCC-Tri-RAHT: gPSNR

Figure 113: MOS vs. PSNR

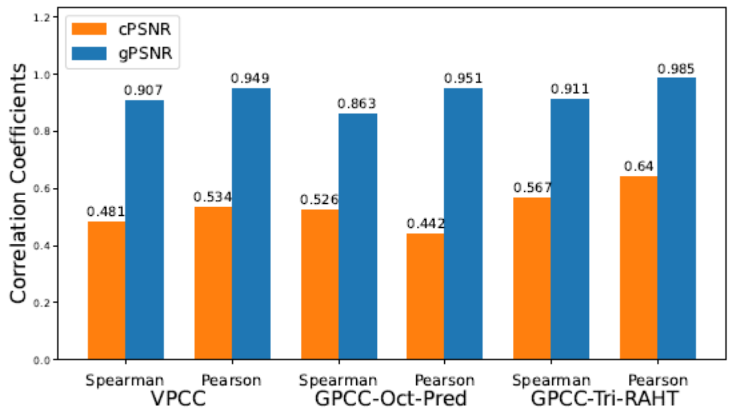


Figure 114: Correlation coefficients for MOS vs. PSNR

6.3.3.5 Participants' Feelings of Immersion and AR Sickness Symptoms

In the post-test questionnaire, the participants were asked: "*How would you describe the general experience?*", with the additional explanation whether or not they would agree to the following assertion: "*I felt the objects were part of the real environment.*" Figure 115 shows that 13 persons (strongly) disagree with that assertion, *i.e.*, do *not* feel that the MR scenes are immersive, 11 are neutral, whereas 17 participants (strongly) agree. Thus, although the DPCs presented were partially of *good* to *excellent* quality, the immersiveness of the MR presentations has quite some room for improvement.

Funded by the Horizon Europe Framework Programme of the European Union

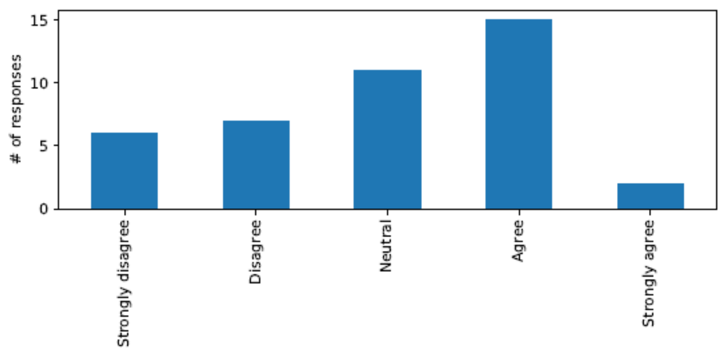


Figure 115: Feelings of immersion: (dis-)agreement with the assertion that "objects were part of the real environment"

The participants were also asked to report about potential AR sickness symptoms. Based on a recent questionnaire developed in [50], the participants provided their assessments on 9 AR sickness symptoms *before* and *after* the test (*general discomfort*, *fatigue*, *headache*, *eye strain*, *fullness of the head*, *dizziness (eyes open)*, *dizziness (eyes closed)*, *vertigo*, and *nausea*) on a 4-point Likert scale [0–none; 1–slight; 2–moderate; 3–severe]. Figure 116 depicts the results. In general, quite some symptoms emerged or were intensified by the test, yet mostly by only 1 level, if at all. *Eye strain* turned out to be the most common symptom. With one exception (a *severe* condition), the reported symptoms were mostly *slight*, with a few *moderate* post-test conditions. The results indicate that the participants could well manage the test with the HoloLens 2 headset.

We define the test's *effect* on the participants as the difference in the levels of AR sickness symptoms reported *after* and *before* the test, *i.e.*, post-test condition minus pre-test condition, for all 9 symptoms. To investigate whether the test's *effect* had a significant impact on the quality ratings, we split the participants into 3 groups: group 1 did not report any differences in symptoms (*no effect*, 18 participants); group 2 includes participants who reported that 1–4 conditions worsened by 1 level (*slight effect*, 12 participants); group 3 includes 1 person who reported that 8 conditions worsened by 1 level, and 10 persons who reported that at least 1 condition worsened by 2 levels (*moderate to severe effect*, 11 participants). (There were rare cases where a post-test condition was reported better than the pre-test condition; see Figure 116. These cases were counted as *no* differences in the analyses.)

ANOVA analysis and Tukey's HSD tests were performed. The results indicate that the group 2 ratings significantly differed from the other groups' ratings. However, the group 1 ratings (*no effect* group) and the group 3 ratings (*moderate to severe effect* group) did not significantly differ from each other. Thus, the potential hypothesis that the test's *effect* significantly impacted the quality ratings can be rejected.

Funded by the Horizon Europe Framework Programme of the European Union

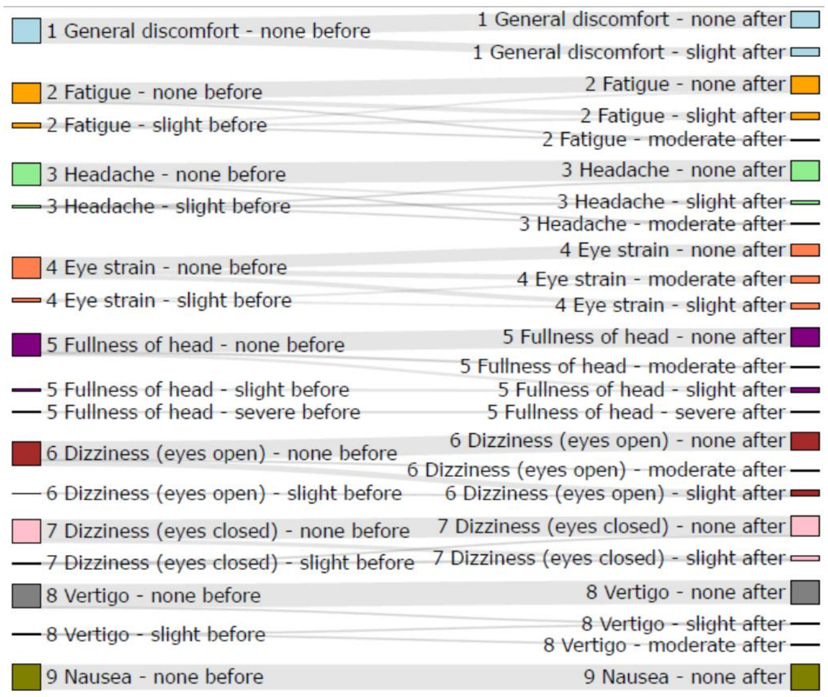


Figure 116: AR sickness symptoms before and after the test

6.3.4 Training and Evaluation of Objective Models for QoE Prediction

Previous studies [49], including our former work [28], have found that machine learning (ML) models can be effective in predicting the QoE of DPCs. Similar to our previous work, we have trained ML models on the ComPEQ-MR dataset, and we evaluate their performance here. Furthermore, we have also fine-tuned the Mode 0 of the P.1203 model [24] from ITU-T using the ComPEQ-MR dataset. We consider both approaches to objective QoE prediction and present the results in this section. The Python code for these analyses and the detailed results can be found here: https://github.com/ shivivats/PointCloudDataAnalysis.

6.3.4.1 Machine Learning Models

1) Data Preparation

As described earlier, we presented the subjective test participants with 52 DPC sequences, obtained from 4 PC objects and encoded using VPCC, GPCC-Oct-Pred, and GPCC-Tri- RAHT. After initial testing, we discovered that training one model for the three codecs would not be viable due to the differences in the encoding parameters used for each codec. Additionally, we learned that combining the ComPEQ-MR dataset with our first dataset (Section 0) would not lead to viable results due to the different natures of the parameters used during both subjective testing rounds.

Thus, we decided to use only the ComPEQ-MR dataset and train ML models separately for each codec using the content characteristics and specific encoding parameters. The former can be represented by the bitrates of the encoded bitstreams of the DPC sequences. The latter can be represented by the quantisation parameters (QPs) utilized by each codec. These are as follows (see also Table 19):

Funded by the Horizon Europe Framework Programme of the European Union

- VPCC: Geometry QP (G-QP) and Texture QP (T-QP);
- GPCC-Oct-Pred: QP and Depth;
- GPCC-Tri-RAHT: QP and Level.

From the 2132 data points we obtained, we removed the ones where the participants were watching the raw (uncompressed) sequences, as these did not have any quantisation parameters associated with them and thus could not be used to train the models. We separated the remaining 1968 responses by codec and removed outliers using the interquartile range (IQR) method [47]. After filtering, we were left with 776, 464, and 622 data points for VPCC, GPCC-Oct-Pred, and GPCC-Tri-RAHT, respectively.

In an effort to obtain unbiased results, we utilised leave-one-out cross-validation. We split the input data into 48 groups ($k = 48$), using $k - 1$ groups for training and the remaining one for validation. The process is repeated a total of k times, ensuring that each group is used for validation once.

2) Evaluation Results

We trained a total of 9 classification and 13 regression models from the *scikit-learn* library for Python on the participants' ratings. The mean squared error (MSE) and the R2 (R- squared) score were used to evaluate the results. The results of the 5 best-performing models per codec are presented in Table 21.

Funded by the Horizon Europe Framework Programme of the European Union

Table 21: Performance of ML models in predicting MOS of DPC sequences in MR environments

(a) VPCC

Model Name	R2 Score ↑	MSE ↓
Hist Gradient Boosting Classifier	0.9590	0.0224
Bagging Classifier	0.9541	0.0251
Gradient Boosting Classifier	0.9499	0.0274
Hist Gradient Boosting Regressor	0.9472	0.0289
Logistic Regression	0.9466	0.0292

(b) GPCC-Oct-Pred

Model Name	R2 Score ↑	MSE ↓
Random Forest Regressor	0.9298	0.0489
Bagging Regressor	0.9240	0.0529
Bagging Classifier	0.9215	0.0546
Lasso	0.9209	0.0551
ElasticNet	0.9209	0.0551

(c) GPCC-Tri-RAHT

Model Name	R2 Score ↑	MSE ↓
Polynomial Regression (Degree 2)	0.9757	0.0214
Bagging Classifier	0.9672	0.0289
Decision Tree Classifier	0.9647	0.0312
Decision Tree Regressor	0.9647	0.0312
Hist Gradient Boosting Regressor	0.9637	0.0320

Hist Gradient Boosting Classifier gives the best results for VPCC, but only marginally outperforms the regular Gradient Boosting Classifier and its Regressor variant. These results are in contrast with our previous analysis of ML models [28] with VPCC-encoded DPC sequences, where Gradient Boosting *Regressor* was the best performing model, with an R2 score and MSE of 0.8582 and 0.2874, respectively, and outperformed the other models by larger margins than seen here.

For GPCC-Oct-Pred, the regressor models outperformed the classifiers, with only one of the 5 best performing models belonging to the latter category. Finally, GPCC-Tri-RAHT is a mixed bag of results, with the Polynomial Regression performing the best, but closely followed by the Bagging Classifier and Decision Tree Classifier models. Notably, the Bagging Classifier model performs well regardless of the codec. This performance can be attributed to how the Bagging Classifier splits the data and trains it using an ensemble of models, leading to smoother predictions and the final result less prone to instability.

3) Feature Importance

Figure 117 presents feature importance for a chosen model for each codec, with a higher score indicating more importance when predicting the MOS. Since the *scikit-learn* library does not support feature importance for every model, we analysed the best-performing model for each

Funded by the Horizon Europe Framework Programme of the European Union

codec with the feature importance available: *Gradient Boosting Classifier* for VPCC, *Random Forest Regressor* for GPCC-Oct-Pred, and *Decision Tree Classifier* for GPCC-Tri-RAHT.

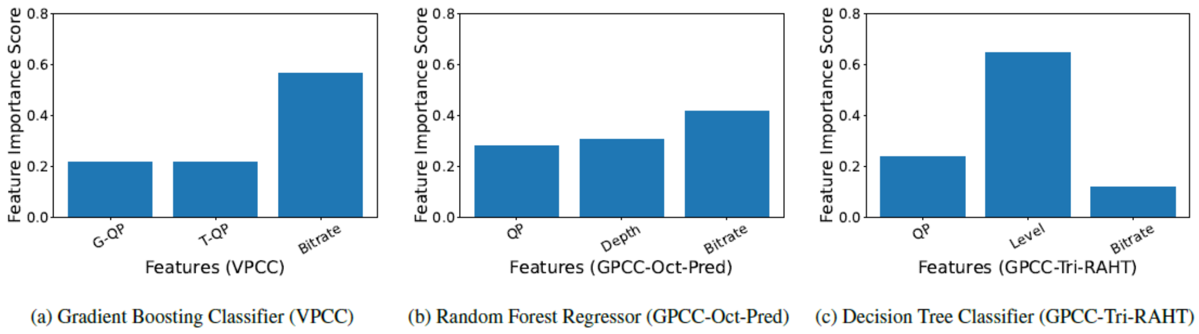


Figure 117: Feature importance scores for input parameters per model and codec

With a score of 0.56, the bitrate plays the most crucial role for predicting QoE for VPCC by a large margin, which is again in contrast to our previous findings where bitrate was shown to have the least amount of influence on predicting the QoE (importance score of just 0.05). This can be explained by the fact that for the current tests, we did not introduce other factors such as a fixed viewing distance or a quality switch during playback. Since we let the participants move around the DPCs and examine them from various angles, we can assume that the bitrate (and the overall quality) of the DPCs have a higher impact on the participants' final ratings.

While the bitrate is again the most influential feature for GPCC-Oct-Pred, all three parameters impact the QoE fairly evenly, with QP, depth, and bitrate having importance scores of 0.28, 0.30, and 0.42, respectively. For GPCC-Tri-RAHT, however, the level of detail when encoding the point clouds impacts the QoE the most (influence score 0.64), with bitrate having the least influence (0.12).

It can be observed that the three encoding schemes do not impact the perception of the encoded DPC sequences in the same way, and different parameters need to be considered when utilising these codecs for DPC transmission with the aim of maximizing QoE.

6.3.4.2 P.1203 Model

The P.1203 Model by ITU-T [24] can predict the QoE of traditional (2D) videos using its various modes. Mode 0 is the most basic as it only considers the video's metadata (*i.e.*, bitrate, frame rate, and resolution) as factors to predict the QoE. In previous work [43], we fine-tuned the P.1203 mode 0 using the data from our first subjective study [28]. We expanded on this work and re-trained the P.1203 model again, using the data from our second round of subjective testing [26] and a combination of the two datasets to potentially obtain a more versatile model.

1) Methodology

Along with a brute force approach, we employed the simplicial homology global optimization [51] and the dual annealing (a modified version of simulated annealing available in *scipy*[21]) algorithms to search for the combination of mode 0 coefficients that lead to the lowest RMSE.

[21] https://docs.scipy.org/doc/scipy/index.html

Funded by the Horizon Europe Framework Programme of the European Union

The data was split into training and validation sets for each case based on the DPC objects. For the first dataset, the *LongDress* and *Loot* objects were used for training, whereas the *RedandBlack* and *Soldier* objects were used for validation. Similarly, the ComPEQ-MR dataset was also split into pairs of *BlueSpin* and *CasualSquat* for training and *ReadyForWinter* and *FlowerDance* for validation. Notably, we only considered VPCC-encoded objects from the ComPEQ-MR dataset to have comparable results with our first study, which only used VPCC as the encoder.

2) Results

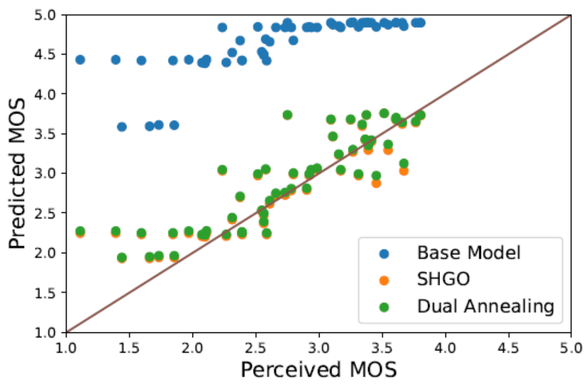
Table 22 contains the fine-tuned results, with the RMSE, Pearson Linear Correlation Coefficient (PLCC), and Spearman's Rank Correlation Coefficient (SRCC) being presented for each case. Comparing the fine-tuned results using both datasets with the original model and our previous work, we note that the fine-tuned models achieve lower RMSE but in most cases fail to achieve higher PLCC and SRCC values. This can be attributed to the two datasets differing too much in their nature for objective models to be able to use their data together in an effective manner. We noted this in the case of the ML models in Subsection 6.3.4.1 as well.

Table 22: Performance of base (untuned) and fine-tuned P.1203 models for various training and validation sets. Lower RMSE and higher PLCC and SRCC are better.

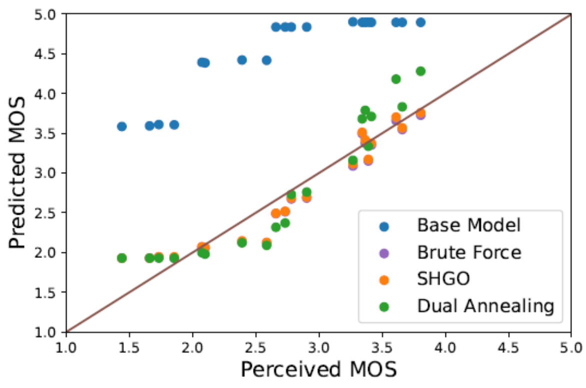
	Training			Validation		
	RMSE ↓	PLCC ↑	SRCC ↑	RMSE ↓	PLCC ↑	SRCC ↑
ITU-T P.1203 Base Model	0.887	0.785	0.766	1.032	0.829	0.918
Fine-tuned on First Dataset [21]	0.813	0.953	0.919	0.955	0.828	0.958
Both Datasets						
Base Model	2.005	0.904	0.847	2.132	0.854	0.742
Fine-tuned (SHGO)	0.788	0.912	0.912	0.831	0.809	0.853
Fine-tuned (Dual Annealing)	0.787	0.912	0.922	0.836	0.809	0.854
ComPEQ-MR Dataset						
Base Model	1.994	0.911	0.904	1.922	0.935	0.892
Fine-tuned (SHGO)	0.714	0.988	0.948	0.673	0.988	0.983
Fine-tuned (Dual Annealing)	0.744	0.997	0.923	0.691	0.985	0.961
Fine-tuned (Brute Force)	0.714	0.985	0.984	0.674	0.988	0.983

However, when fine-tuning the model using the ComPEQ-MR dataset, the new techniques lead to lower RMSE and higher correlation scores in all cases when compared to the base (untuned) model. The correlations of the compared models are visualized in Figure 118.

Funded by the Horizon Europe Framework Programme of the European Union



(a) For both datasets.



(b) For only the ComPEQ-MR dataset.

Figure 118: Perceived vs. predicted MOS using the untuned (base) and fine-tuned ITU-T P.1203 models

6.3.5 Conclusions and Future Work

Based on a subjective study of the QoE of 41 participants when presented 4 point cloud videos at 13 different quality levels in an MR setting and on the resulting ComPEQ-MR dataset, this section provided three main contributions: *(i)* eye-tracking data and an analysis thereof (visual attention maps, head and gaze movement); *(ii)* subjective quality results (rating scores), confirming that the choice of compression algorithm (VPCC, GPCC-Oct-Pred, GPCC-Tri-RAHT under investigation) and the quality level significantly impact user perception of visual quality, but also that there are scenarios where reducing the quality level (and, thus, bandwidth consumption) does not degrade QoE; and *(iii)* training and validation of machine learning models (classification and regression) and fine-tuned variants of the ITU-T P.1203 model to predict/estimate the QoE of compressed point cloud videos in the MR setting. The results suggest future work on utilising visual attention data for viewport prediction, on quality-adaptive point cloud video streaming and presentation algorithms, and on further QoE prediction models for MR experiences.

Funded by the Horizon Europe Framework Programme of the European Union

6.4 QOE EVALUATIONS FOR PROJECT-DESIGNATED USE CASES

To apply the QoE study and modelling results to the SPIRIT use cases, UNI-KLU estimated the QoE of the use cases. The following subsections detail the approaches devised and utilised for the various use cases.

6.4.1 Estimating QoE using the ITU-T P.1203 Model

6.4.1.1 Methodology

Initially, the use case owners captured representative regular 2D videos from their use cases (as presented to users and captured from their screens). Subsequently, UNI-KLU estimated the QoE based on these videos, using an objective 2D video metric/model. We used the ITU-T P.1203 model [24] to determine the objective QoE scores of the captured 2D use case videos, estimating what a user would experience. Both the regular and the fine-tuned P.1203 models were deployed. The limitations of this approach are discussed in the next sub-section.

The P.1203 model can calculate the overall QoE score of a video in four modes:

- Mode 0 considers only the metadata of the video such as bitrate, framerate, and resolution.
- Mode 1 considers some frame information along with all the data from mode 0.
- Mode 2 considers 2% of the bitstream along with the data from mode 1.
- Mode 3 considers the whole media stream information along with data from mode 1.

The model scores the videos up to 5 points, with a higher score indicating better QoE.

Table 23: Overall Qoe results from the regular P.1203 model

Use Case	Video	ITU-T P.1203 Overall QoE Score		
		Mode 0	Mode 1	Modes 2 and 3
Robot	Robot1	4.82	4.74	4.48
Robot	Robot2	4.82	4.72	4.47
Robot	Robot3	4.82	4.72	4.57
Avatar	Avatar1	4.81	4.57	4.78
Avatar	Avatar2	4.57	3.87	4.31
Avatar	Avatar3	4.39	3.94	4.46
Avatar	Avatar4	4.38	4.01	4.47

Table 24: Overall qoe results from our fine-tuned P.1203 model

Use Case	Video	Fine-Tuned ITU-T P.1203 Overall QoE Score		
		Mode 0	Mode 1	Modes 2 and 3
Robot	Robot1	3.18	2.15	1.94
Robot	Robot2	3.18	2.12	1.94
Robot	Robot3	3.18	2.11	1.96
Avatar	Avatar1	3.12	1.63	1.71
Avatar	Avatar2	3.07	1.54	2.32
Avatar	Avatar3	3.67	2.21	2.15
Avatar	Avatar4	3.08	1.48	2.33

Funded by the Horizon Europe Framework Programme of the European Union

It was determined that estimating the QoE using the P.1203 model would only be possible for the Robot and Avatar use cases, since the other use cases consider lossless compression methods for real-time communication and our datasets do not contain such information.

Table 23 and Table 24 contain the overall QoE scores of the videos recorded by the use case partners from the regular and fine-tuned P.1203 model, respectively. These videos estimate what a user of the use case would visually experience. The videos from the Robot and Avatar use cases are of 480p, 720p, and 1440p resolutions, respectively.

Notably, the Robot use case providers did not have a way to encode the recorded video directly to H.264/AVC, and the P.1203 tool does not support any codec other than H.264/AVC. Thus, the videos from the Robot use case were transcoded from VP9 to H.264/AVC using the following command:

ffmpeg -i inVideo.webm -c:v libx264 -c:a flac outVideo.mp4

The choice to not use a custom CRF value[22] was made to not influence the quality of the transcoded video beyond any default H.264/AVC parameters.

6.4.1.2 Interpreting Results

A few conclusions can be made from observing the results of the models. The regular version of the model gives good to very good results for all the videos. For mode 0, this is due to the fact that the videos are inherently of good (or rather, "good enough") quality based on just their metadata.

Upon considering more video data in mode 1, the results for some of the videos become worse. This is expected behaviour, as upon viewing the videos, it is clear that these videos are not deserving of a near-perfect score as mode 0 gave.

Mode 3 is even harsher, as the bitstream data provides more insight into the picture quality of the video and the results from mode 3 show this plainly.

For the fine-tuned model, the results are worse in general compared to the regular model. This can be explained due to the fine-tuned model being "harsher" with the videos based on the quality of the point clouds present in them. For instance, the videos from the Avatar use case are also given lower scores, but not too low, as the avatar present in the video is of good quality.

Notably, only the results from mode 0 of the fine-tuned model should be worth considering here, since we only fine-tuned the model for the said mode. In this process, some, but not all, weights for modes 1, 2, and 3 were also modified. Thus, the results for those modes are also different from the regular model, but in the same vein, are not fully accurate. They have been included here for the sake of completion.

Additionally, the videos from the Robot use case are "simply" 2D videos without any point cloud data. These videos are evaluated more accurately with the regular version of the model.

Similarly, the videos from the Avatar use case do not contain a point cloud, but rather a mesh avatar. The fine-tuned model should be more accurate here, since the avatar is visually not

[22] https://trac.ffmpeg.org/wiki/Encode/H.264

dissimilar to a (good) point cloud of a person, at least when presented in a head-mounted display.

6.4.1.3 Limitations of the P.1203 Model

There are a few limitations of the P.1203 model that should be noted here. First, and perhaps the biggest one, is that the P.1203 model is built for 2D videos only. Thus, it does not understand what a point cloud is and, in its default state, will fail to take the quality of the point cloud into account when determining the QoE score. We get around this limitation by using the results of our subjective tests to fine-tune the mode 0 of the model, making it more accurately assess 2D videos with point cloud data, as described in the previous sub-section.

Moreover, the P.1203 model cannot take into account the user's interactions (changing viewpoint and viewport as well as moving and acting) and feeling of immersion into the environment; the model purely relies on immutable 2D visual information. It must be emphasised that the model's scores are therefore *QoE estimates* only. Getting fully sound QoE scores for the use cases would require well-designed, tedious, and time-consuming subjective QoE studies as reported in the previous sections. This was regarded as infeasible by the SPIRIT partners.

The final limitation of this model is that mode 2 is currently implemented to be the same as mode 3, as the tool developer could not come up with a feasible way to only utilise 2% of the video's bitstream for mode 2 (which 2% to use?). This issue has been discussed briefly in a GitHub issues post[23]. However, we believe this deficiency is tolerable since mode 3 encapsulates mode 2 already by utilising the whole of the video's bitstream.

6.4.2 Estimating QoE of the Live Holographic Teleportation System

The Live Holographic Teleportation system, detailed in [52], is the predecessor of the "Multi-Source" use case in the SPIRIT project. We performed an analysis of the system described in the paper, since it presents concrete numbers and the specific presentation of the use case to the user has not changed since then. Methodology QoE estimation for this system was also done using a fine-tuned ITU-T P.1203 model. However, the whole video file was unavailable, and we only utilized video metadata, and consequently, only mode 0 of the P.1203 model, to calculate the QoE here.

The bitrate after compression using zstd[24] was considered, and the estimated QoE was calculated for both an "ideal" scenario with 30 FPS and for the evaluated scenario detailed in the associated study [52] with an average of approximately 25 and 18 FPS for HD and FHD resolutions, respectively.

[23] https://github.com/itu-p1203/itu-p1203/issues/39

[24] https://github.com/facebook/zstd

Funded by the Horizon Europe Framework Programme of the European Union

6.4.2.1 Results and Discussion

Table 25: QoE of the live holographic teleportation system

Resolution	**HD (1280x720)**		**FHD (1920x1080)**	
Case	Ideal	Evaluated	Ideal	Evaluated
FPS	30	25	30	18
Bitrate (Mbit/s)	150.48	125.40	338.40	203.04
QoE	3.06	3.02	3.77	3.48

Table 25 shows the estimated QoE (range 1 – 5) as calculated using the fine-tuned P.1203 model's mode 0 for the live holographic teleportation system. Similar to the Avatar use case, the model scores the videos between 3 and 4. This is also in-line with a subjective visual assessment of the use case videos by us. Furthermore, the QoE is rated higher in the ideal case, which is to be expected due to the higher FPS and bitrate.

6.4.3 Estimating QoE of the Holographic Communication Use Case

6.4.3.1 Methodology

The holographic communication system utilises Draco for real-time lossless encoding. Since our subjective testing did not involve such a system, we employed a third-party dataset containing user MOS on Draco videos [49]. Out of the various parameters available in the dataset, we trained machine learning models to determine QoE using the following parameters: frame rate, Draco quantisation parameter (QP)[25], frame size, and bits per point.

Since the use case utilises meshes instead of point clouds, we interpreted the bits per point metric to be bits per mesh vertex, and the resulting value was plausible for us to utilise. Based on this information, and the trained ML models, we calculated the QoE of the use case.

[25] https://github.com/google/draco

Funded by the Horizon Europe Framework Programme of the European Union

6.4.3.2 Results and Discussion

Table 26: QoE of the holographic communication use case

Resolution	**424x240p**	**640x360p**	**1280x720p**
Size in bytes	54512	153837	428490
Num. vertices	19543	65220	184033
Num. triangles	38174	128478	362470
Bits per vertex	22.31	18.87	18.63
Bits per triangle	11.42	9.58	9.46
QoE using bits per vertex	3.9235	3.8997	3.8994
QoE using bits per triangle	3.9267	3.8222	3.8141

The frame rate is 30 FPS and the Draco QP is 14 for this use case.

Table 26 contains the estimated QoE values of the use case, determined using the frame size, and the mesh characteristics provided to us by the use case owners. The QoE values are in a range of 1 to 5, with 1 being the worst and 5 being the best. We have included both bits per vertex and bits per triangle as the metrics here that were used instead of bits per point. Even though both metrics lead to similar results, as mentioned earlier, we would recommend using bits per vertex instead of bits per triangle.

It is also interesting to note that the resolution of the video does not impact the QoE. This is backed by the data as well (Pearson correlation of -0.0207 between the resolution and QoE). We assume here that the resolution increase is paired with a viewport increase or increase in the viewing distance. In both scenarios, the effective resolution of the mesh for the viewer would be quite similar, leading to similar QoE values.

Funded by the Horizon Europe Framework Programme of the European Union

7 CONCLUSIONS

The initial work consisted of integrating the partner components into a common testbed environment to enable third parties to use the various partner components and the testbeds for their use cases.

In the second phase, the two testbeds in Surrey and Berlin were connected so that the use cases and application frameworks can be deployed and validated across the testbeds. This interconnection was already successfully tested with the use cases Avatar: “Real-Time Animation and Streaming of Realistic Avatars” and “Holographic Human-to-Human Communication”. Several additional components were identified, developed and presented in D3.1 [8] and D3.2 [3], which were then integrated into the final version of the SPIRIT platform and into the platform testbeds.

While the project team was working on the second iteration of the SPIRIT platform, the first Open Call was successfully launched inviting third parties to realise their telepresence use cases experimentally. Similarly, while working on the third and final version of the platform, the second Open Call was launched. The large number of applicants in both Open Calls shows that the Scalable Platform for Innovation is of great interest to the Real-time Immersive Telepresence community. The results of these experiments have been used by the project to improve the existing partner components during the remainder of the project.

Funded by the Horizon Europe Framework Programme of the European Union

8 REFERENCES

[1] SPIRIT Consortium, “SPIRIT Platform (First Version),” Project "Scalable Platform for Innovations on Real-time Immersive Telepresence", EC grant agreement 101070672, 2023.

[2] SPIRIT Consortium, “Use Case Requirements, System Architecture and Interface Definition (Final Version),” Project "Scalable Platform for Innovations on Real-time Immersive Telepresence", EC grant agreement 101070672, 2024.

[3] SPIRIT Consortium, “Innovation Platform Enablers (Final Version),” Project "Scalable Platform for Innovations on Real-time Immersive Telepresence", EC grant agreement 101070672, 2024.

[4] SPIRIT Consortium, “SPIRIT Platform (Second Version),” Project "Scalable Platform for Innovations on Real-time Immersive Telepresence", EC grant agreement 101070672, 2024.

[5] S. Anmulwar, N. Wang, H. H. Vu San, S. Bryant, J. Yang and R. Tafazolli, “HoloSync: Frame Synchronisation for Multi-Source Holographic Teleportation Applications,” *IEEE Transactions on Multimedia,* pp. 1-14, 2022.

[6] M. De Fré, J. van der Hooft, T. Wauters and F. De Turck, Scalable MDC-Based WebRTC for Real-Time One-to-Many Volumetric Video Conferencing, ACM Transactions on Multimedia Computing, Communications, and Applications, 2025.

[7] J. Son, S. Gul, G. S. Bhullar, G. Hege, W. Morgenstern, A. Hilsmann, T. Ebner, S. Bliedung, P. Eisert and T. Schierl, “Split rendering for mixed reality: Interactive volumetric video in action,” *SIGGRAPH Asia 2020 XR,* pp. 1-3, 2020.

[8] SPIRIT Consortium, “Innovation Platform Enablers (First Version),” Project "Scalable Platform for Innovations on Real-time Immersive Telepresence", EC grant agreement 101070672, 2023.

[9] J. Bottomley, “Building Encrypted Images for Confidential Computing,” [Online]. Available: https://blog.hansenpartnership.com/building-encrypted-images-for-confidential-computing/. [Accessed 13 August 2023].

[10] J. Bottomley, “Deploying Encrypted Images for Confidential Computing,” [Online]. Available: https://blog.hansenpartnership.com/deploying-encrypted-images-for-confidential-computing/. [Accessed 13 August 2023].

Funded by the Horizon Europe Framework Programme of the European Union

[11] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer and R. Zimmermann, “A Survey on Bitrate Adaptation Schemes for Streaming Media over HTTP,” *IEEE Communications Surveys & Tutorials,* pp. 562-585, 2018.

[12] G. Cofano, L. De Cicco, T. Zinner, A. Nguyen-Ngoc, P. Tran-Gia and S. Mascolo, “Design and Experimental Evaluation of Network-assisted Strategies for HTTP Adaptive Streaming,” in *7th ACM International Conference on Multimedia Systems*, 2016.

[13] B. Taraghi, M. Nguyen, H. Amirpour and C. Timmerer, “INTENSE: In-Depth Studies on Stall Events and Quality Switches and Their Impact on the Quality of Experience in HTTP Adaptive Streaming,” *IEEE Access,* vol. 9, pp. 118087-118098, 2021.

[14] M. Krivokuća, M. Koroteev and P. A. Chou, “A Volumetric Approach to Point Cloud Compression,” *arXiv preprint arXiv:1810.00484,* 2018.

[15] R. L. de Queiroz and P. A. Chou, “Compression of 3D Point Clouds Using a Region-Adaptive Hierarchical Transform,” *IEEE Transactions on Image Processing,* vol. 25, no. 8, pp. 3947-3956, 2016.

[16] Y. Huang, J. Peng, C.-C. J. Kuo and M. Gopi, “A Generic Scheme for Progressive Point Cloud Coding,” *IEEE Transactions on Visualization and Computer Graphics,* vol. 14, no. 2, pp. 440-453, 2008.

[17] S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen and others, “Emerging MPEG Standards for Point Cloud Compression,” *IEEE Journal on Emerging and Selected Topics in Circuits and Systems,* vol. 9, no. 1, pp. 133-148, 2018.

[18] G. J. Sullivan, J.-R. Ohm, W.-J. Han and T. Wiegand, “Overview of the High Efficiency Video Coding (HEVC) Standard,” *IEEE Transactions on circuits and systems for video technology,* vol. 22, no. 12, pp. 1649-1668, 2012.

[19] J. van der Hooft, M. Torres Vega, C. Timmerer, A. C. Begen, F. De Turck and R. Schatz, “Objective and Subjective QoE Evaluation for Adaptive Point Cloud Streaming,” in *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, 2020.

[20] E. Alexiou, N. Yang and T. Ebrahimi, “PointXR: A Toolbox for Visualization and Subjective Evaluation of Point Clouds in Virtual Reality,” in *2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX)*, 2020.

[21] M. K. Bekele, R. Pierdicca, E. Frontoni, E. S. Malinverni and J. Gain, “A Survey of Augmented, Virtual, and Mixed Reality for Cultural Heritage,” *Journal on Computing and Cultural Heritage (JOCCH),* vol. 11, no. 2, pp. 1-36, 2018.

Funded by the Horizon Europe Framework Programme of the European Union

[22] S. Petrangeli, J. Famaey, M. Claeys, S. Latre and F. De Turck, "QoE-driven rate adaptation heuristic for fair adaptive video streaming," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM),* vol. 12, no. 2, pp. 1-24, 2015.

[23] H. T. Tran, D. Nguyen and T. C. Thang, "An open software for bitstream-based quality prediction in adaptive video streaming," in *the 11th ACM Multimedia Systems Conference*, 225-230, 2020.

[24] ITU-T, "P.1203.3: Parametric bitstream-based quality assessment of progressive download and adaptive audiovisual streaming services over reliable transport-quality integration module," International Telecommunication Union, 2019.

[25] J. Li, X. Wang, Z. Liu and Q. Li, "A QoE Model in Point Cloud Video Streaming," in *arXiv preprint arXiv:2111.02985*, 2021.

[26] M. Nguyen, S. Vats, X. Zhou, I. Viola, P. Cesar, C. Timmerer and H. Hellwagner, "ComPEQ-MR: Compressed Point Cloud Dataset with Eye Tracking and Quality Assessment in Mixed Reality," in *15th ACM Multimedia Systems Conference*, 2024.

[27] M. Nguyen, S. Vats, S. Van Damme, J. Van Der Hooft, M. T. Vega, T. Wauters, C. Timmerer and H. Hellwagner, "Impact of Quality and Distance on the Perception of Point Clouds in Mixed Reality," in *2023 15th International Conference on Quality of Multimedia Experience (QoMEX)*, 2023.

[28] M. Nguyen, S. Vats, S. Van Damme, J. Van der Hooft, M. T. Vega, T. Wauters, F. De Turck, C. Timmerer and H. Hellwagner, "Characterization of the Quality of Experience and Immersion of Point Cloud Videos in Augmented Reality through a Subjective Study," *IEEE Access,* 2023.

[29] S. Vats, M. Nguyen, S. Van Damme, J. van der Hooft, M. T. Vega, T. Wauters, C. Timmerer and H. Hellwagner, "A Platform for Subjective Quality Assessment in Mixed Reality Environments," in *2023 15th International Conference on Quality of Multimedia Experience (QoMEX)*, Ghent, 2023.

[30] S. Vats, C. Timmerer and H. Hellwagner, "STEP-MR: A Subjective Testing and Eye-Tracking Platform for Dynamic Point Clouds in Mixed Reality," in *22nd EuroXR International Conference (EuroXR 2025)*, Zurich, Switzerland, 2025.

[31] I. B. Adhanom, S. C. Lee, E. Folmer and P. MacNeilage, "GazeMetrics: An Open-Source Tool for Measuring the Data Quality of HMD-based Eye Trackers," in *ACM Symposium on Eye Tracking Research and Applications*, 2020.

[32] X. Zhou, I. Viola, A. Evangelos, J. Jack and C. Pablo, "QAVA-DPC: Eye-Tracking Based Quality Assessment and Visual Attention Dataset for Dynamic Point Cloud in 6 DoF," in

Funded by the Horizon Europe Framework Programme of the European Union

IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Sydney, Australia, 2023.

[33] D. D. Salvucci and J. H. Goldberg, “Identifying fixations and saccades in eye-tracking protocols,” in *ETRA '00: Proceedings of the 2000 symposium on Eye tracking research & applicatio*, Palm Beach Gardens, Florida, USA, 2000.

[34] E. d’Eon, B. Harrison, T. Myers and P. A. Chou, “8i Voxelized Full Bodies, version 2 – A Voxelized Point Cloud Dataset,” ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) Input Document M40059/M74006, 2017.

[35] MPEG 3DG, “Common Test Conditions for Point Cloud Compression,” ISO/IEC JTC1/SC29/WG11 Coding of Moving Pictures and Audio Meeting Proceedings, 2017.

[36] J. Birch, “Efficiency of the Ishihara Test for Identifying Red-Green Colour Deficiency,” *Ophthalmic and Physiological Optics,* vol. 17, no. 5, pp. 403-408, 1997.

[37] “ITU-T P.919: Subjective test methodologies for 360º video on head-mounted displays,” ITU, 2020.

[38] L. Stahle and S. Wold, “Analysis of Variance (ANOVA),” *Chemometrics and Intelligent Laboratory Systems,* vol. 6, no. 4, pp. 259-272, 1989.

[39] H. Abdi and L. J. Williams, “Tukey’s Honestly Significant Difference (HSD) Test,” *Encyclopedia of Research Design,* vol. 3, no. 1, pp. 1-5, 2010.

[40] T. K. Kim, “T Test as a Parametric Statistic,” *Korean Journal of Anesthesiology,* vol. 68, no. 6, pp. 540-546, 2015.

[41] H. Amirpour, R. Schatz, C. Timmerer and M. Ghanbari, “On the Impact of Viewing Distance on Perceived Video Quality,” in *2021 International Conference on Visual Communications and Image Processing (VCIP)*, 2021.

[42] H. T. Tran, N. P. Ngoc, C. T. Pham, Y. J. Jung and T. C. Thang, “A Subjective Study on QoE of 360 Video for VR Communication,” in *2017 IEEE 19th international workshop on multimedia signal processing (MMSP)*, 2017.

[43] M. Nguyen, S. Vats and H. Hellwagner, “No-Reference Quality of Experience Model for Dynamic Point Clouds in Augmented Reality,” in *3rd Mile-High Video Conference*, 2024.

[44] G. Gautier, A. Mercat, L. Fréneau, M. Pitkänen and J. Vanne, “UVG-VPC: Voxelized Point Cloud Dataset for Visual Volumetric,” in *15th International Conference on Quality of Multimedia Experience (QoMEX)*, 2023.

Funded by the Horizon Europe Framework Programme of the European Union

[45] MPEG, “Common Test Conditions for G-PCC,” in *ISO/IEC JTC1/SC29/WG7 N722*, 2021.

[46] M. 3DG, “JPEG Pleno Point Cloud Coding Common Test Conditions,” in *ISO/IEC JTC1/SC29/WG1 N91058*, 2021.

[47] D. Whaley, The Interquartile Range: Theory and Estimation. PhD thesis, East Tennessee State University, 2005.

[48] A. Ak, E. Zerman, M. Quach, A. Chetouani and A. Smolic, “BASICS: Broad Quality Assessment,” in *arXiv preprint arXiv:2302.04796*, 2023.

[49] J. Weil, Y. Alkhalili, A. Tahir, T. Gruczyk, T. Meuser, A. Mauthe, M. Mu and H. Koeppl, "Modeling Quality of Experience for Compressed Point Cloud Sequences based on a Subjective Study," in *15th International Conference on Quality of Experience (QoMEX)*, 2023.

[50] M. Hussain, J. Park and H. Kim, “Augmented Reality Sickness Questionnaire (ARSQ): A Refined Questionnaire for Augmented Reality Environment,” *International Journal of Industrial Ergonomics,* p. 97:103495, 2023.

[51] S. Endres, C. Sandrock and W. Focke, "A simplicial homology algorithm for Lipschitz optimisation," *Journal of Global Optimization,* p. 181–217, 72(2) 2018.

[52] P. Qian, V. S. H. Huynh, N. Wang, S. Anmulwar, D. Mi and R. R. Tafazolli, “Remote Production for Live Holographic Teleportation Applications in 5G Networks,” *IEEE Transactions on Broadcasting,* vol. 68, no. 2, 2022.

[53] C. Timmerer and A. C. Begen, “A Journey Towards Fully Immersive Media Access,” in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019.

[54] SPIRIT Consortium, “Use Case Requirements, System Architecture and Interface Definition (First Version),” Project "Scalable Platform for Innovations on Real-time Immersive Telepresence", EC grant agreement 101070672, 2023.

[55] M. Kowalski, Naruniec and M. Daniluk, “Livescan3D: A Fast and Inexpensive 3D Data Acquisition System for Multiple Kinect v2 Sensors,” in *IEEE International Conference on 3D Vision*, 2015.

[56] D. Novick and A. E. Rodriguez, “Comparative Study of Conversational Proxemics for Virtual Agents,” in *13th International Conference on Virtual, Augmented and Mixed Reality*, 2021.

[57] ITU-T, “P.919: Subjective Test Methodologies for 360-degree video on Head-Mounted Displays,” International Telecommunication Union, 2020.

Funded by the Horizon Europe
Framework Programme of the European Union

[58] R. Martin-Brualla, R. Pandey, S. Yang, P. Pidlypenskyi, J. Taylor, J. Valentin, S. Khamis, P. Davidson, A. Tkach and P. Lincoln, “Lookingood: Enhancing performance capture with real-time neural re-rendering,” *arXiv preprint arXiv:1811.05029,* 2018.

[59] L. Wang, C. Li, W. Dai, J. Zou and H. Xiong, “QoE-driven and Tile-based Adaptive Streaming for Point Clouds,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

Funded by the Horizon Europe Framework Programme of the European Union

APPENDIX A: SECURITY

A.1 ESTABLISHING THE VPN CONNECTION WITH THE LAB INFRASTRUCTURE

The lab infrastructure is protected by a Cisco ASA Firewall and VPN endpoint. This requires the use of the Cisco AnyConnect protocol to establish a VPN connection.

If you want to connect using the open source openconnect software issue the following command:

```
openconnect cloud-security-lab.de --servercert "pin-sha256:EpPc6Du8rBMM3B3KRe3QTlSk+rQ9zbL4KV1/zFAVnHE=" -u cloudtest --authgroup cloud_vpn
```

Note on openconnect: We have found that the latest openconnect 9 software has a bug when connecting to our ASA with latest software version. Please use openconnect 8.20 to work around that problem.

Note 2: We are using a Let's encrypt certificate for our Cisco ASA VPN system. The "servercert" option of the openconnect command contains the fingerprint of this certificate and must be updated regularly – the given value is only an example.

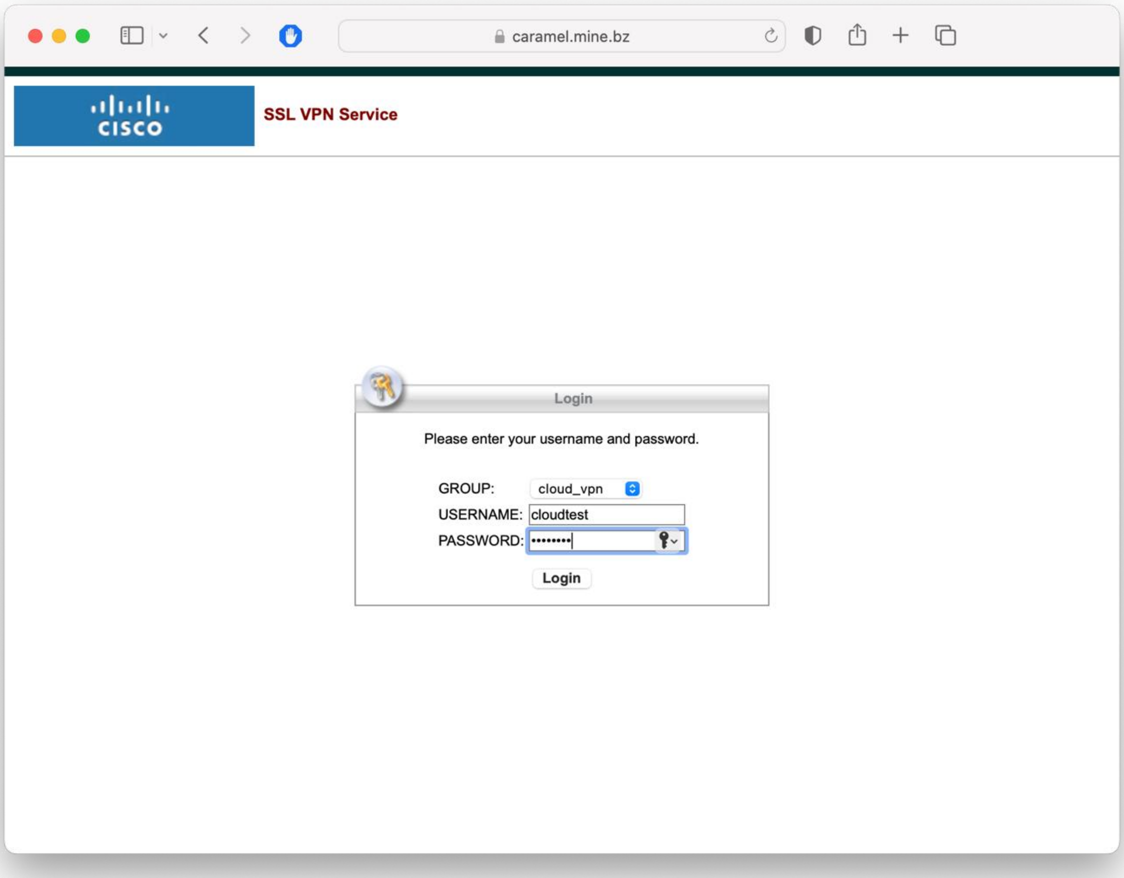
Request a username (instead of the sample "cloudtest") and password from the lab admin.

Alternatively, if you want to use the graphical Cisco Secure Client software (available for Linux, MacOS, and Windows) open the ASA home page in web browser with the URL

➲ https://cloud-security-lab.de

Since the certificate is self-signed, you must accept the untrusted web site.

Funded by the Horizon Europe Framework Programme of the European Union


caramel.mine.bz

SSL VPN Service

Login

Please enter your username and password.

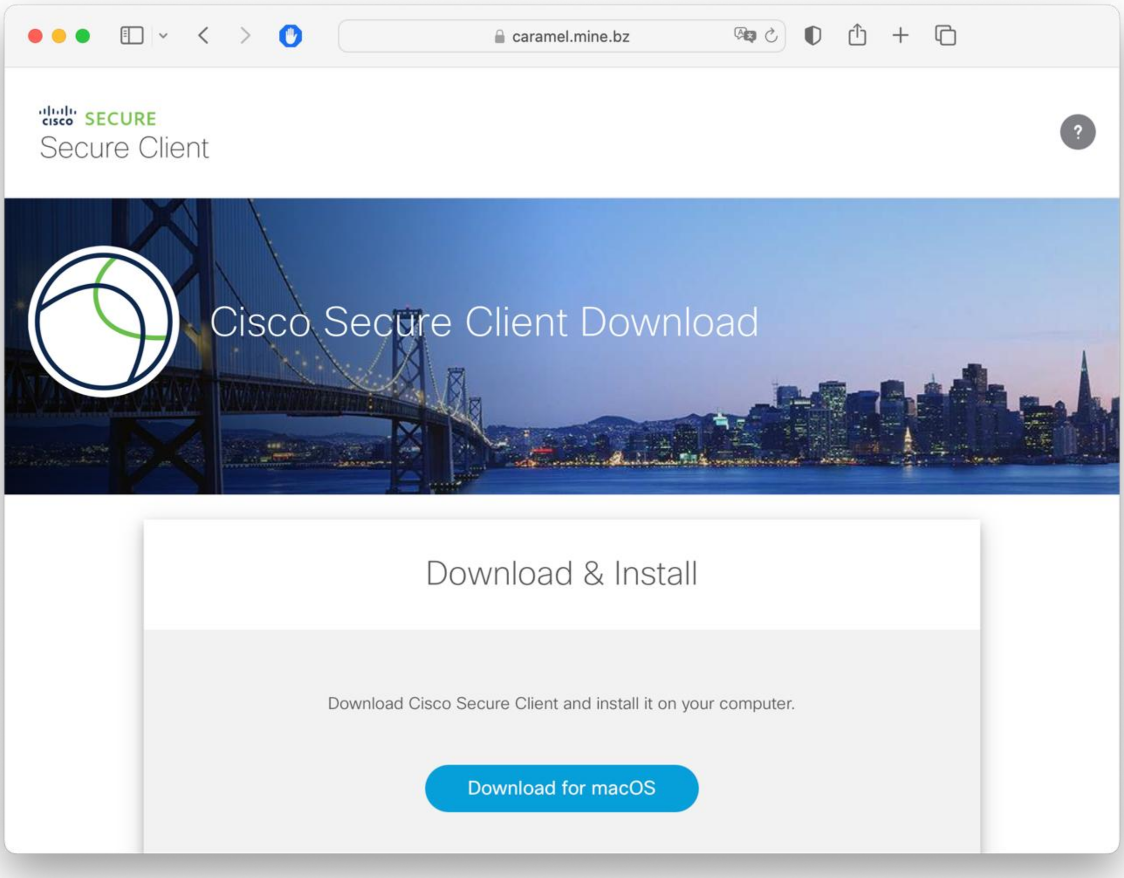
GROUP: cloud_vpn
USERNAME: cloudtest
PASSWORD: ••••••••

Login

Figure 119: Cisco ASA Login Screen

Figure 119 shows the Cisco ASA login screen: Login using your credentials as communicated by us.

Funded by the Horizon Europe Framework Programme of the European Union

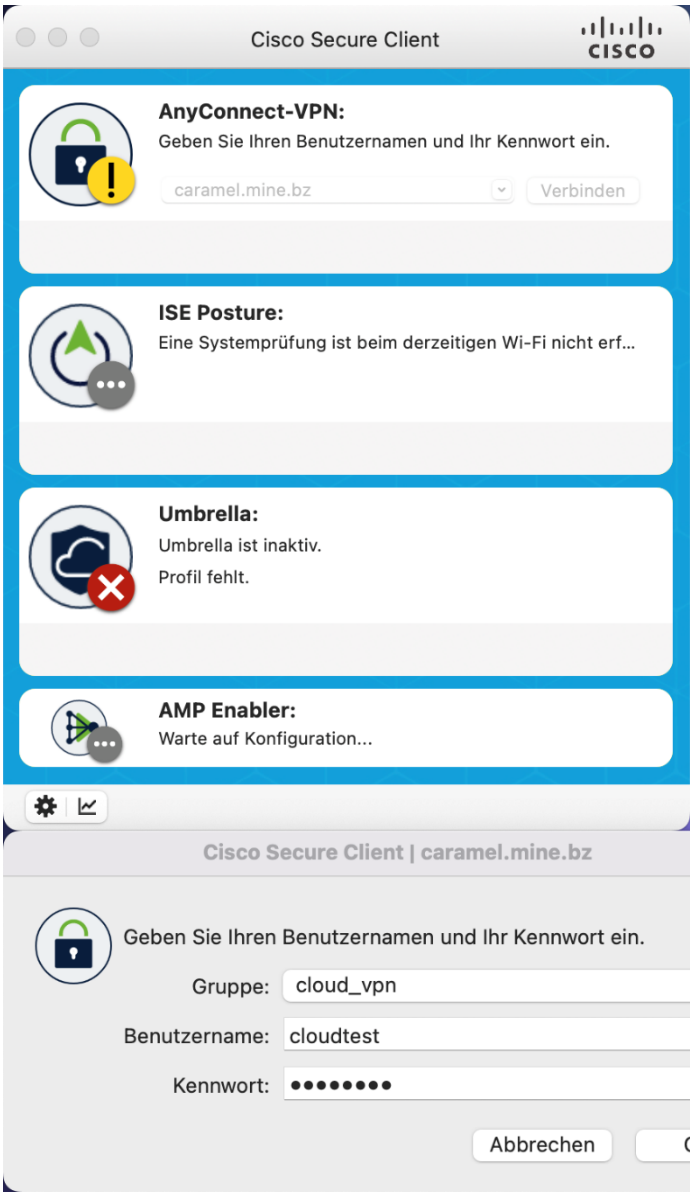


Download Cisco Secure Client and install it on your computer.

Figure 120: Cisco Secure Client Download Page

After login, you are brought to the Cisco Secure Client download screen (see Figure 120). Download the Secure Client for your operating system and install. Using the installed Cisco Client, you can establish the VPN connection (Figure 121).

Funded by the Horizon Europe Framework Programme of the European Union



AnyConnect-VPN:
Geben Sie Ihren Benutzernamen und Ihr Kennwort ein.

ISE Posture:
Eine Systemprüfung ist beim derzeitigen Wi-Fi nicht erf...

Umbrella:
Umbrella ist inaktiv.
Profil fehlt.

AMP Enabler:
Warte auf Konfiguration...

Geben Sie Ihren Benutzernamen und Ihr Kennwort ein.

Gruppe:	cloud_vpn
Benutzername:	cloudtest
Kennwort:	••••••••

Figure 121: VPN Connection with Cisco Secure Client

Funded by the Horizon Europe Framework Programme of the European Union

Notes on the Cisco AnyConnect VPN service in our lab environment:

- The VPN is required for download and upload to our NAS system.
- The VPN is also required for remote management of guest VMs. For this use case it is probably easiest to use the openconnect command line tool for automatic VPN setup.
- You will receive a dynamic IP address in the 10.0.12.0/24 subnet.
- All other traffic is routed normally to the Internet.
- The Cisco AnyConnect protocol also works behind firewalls through web proxies.
- UserIDs and passwords for VPN, NAS, and remote management (generally all the same if not otherwise requested) must be requested from a lab admin.
- We are able to open ports on our outside firewall interface (both IPv4 and IPv6) for application-specific purposes.

A.2

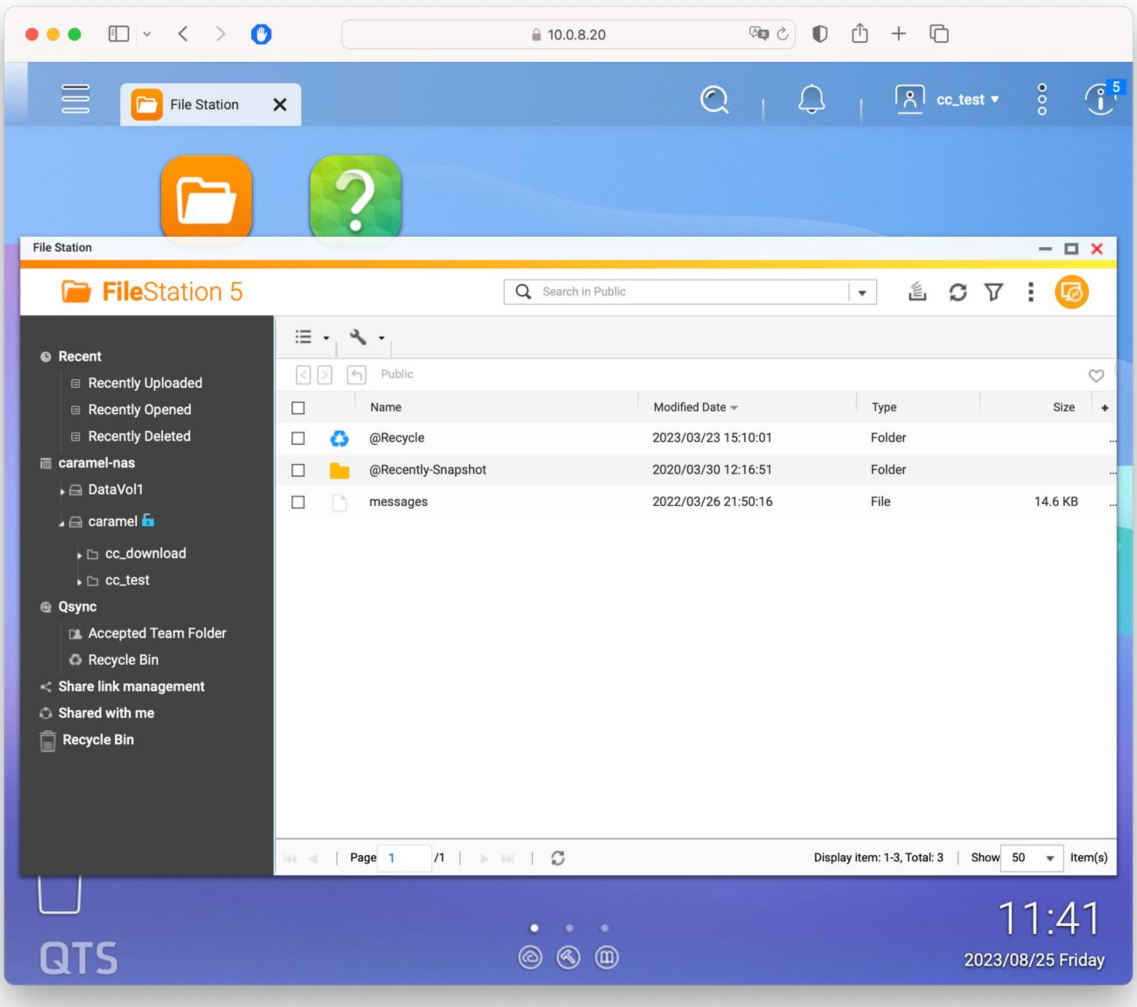
Funded by the Horizon Europe Framework Programme of the European Union

ACCESS NAS DEVICE FOR SOFTWARE DOWNLOAD AND VM UP-LOAD

The NAS (network attached storage) device is accessible over the VPN tunnel under the URL:

➲ `https://10.0.12.20:444/cgi-bin/`

Login with your lab credentials (user ID and password).



Name	Modified Date	Type	Size
@Recycle	2023/03/23 15:10:01	Folder	
@Recently-Snapshot	2020/03/30 12:16:51	Folder	
messages	2022/03/26 21:50:16	File	14.6 KB

Figure 122: NAS GUI with FileStation application

Figure 122 shows the NAS GUI in the web browser after clicking on the “FileStation” icon. Logged in with the sample user “cc_test”, the following two folders are of interest:

- cc_test (will be different for your user): This folder is writable. Upload your completed and encrypted guest VM images here.
- cc_download: This folder contains the tool downloads:
    - Readme.md: (optional) Markdown file with latest information on using the tool set that may not be in the latest version of this document.

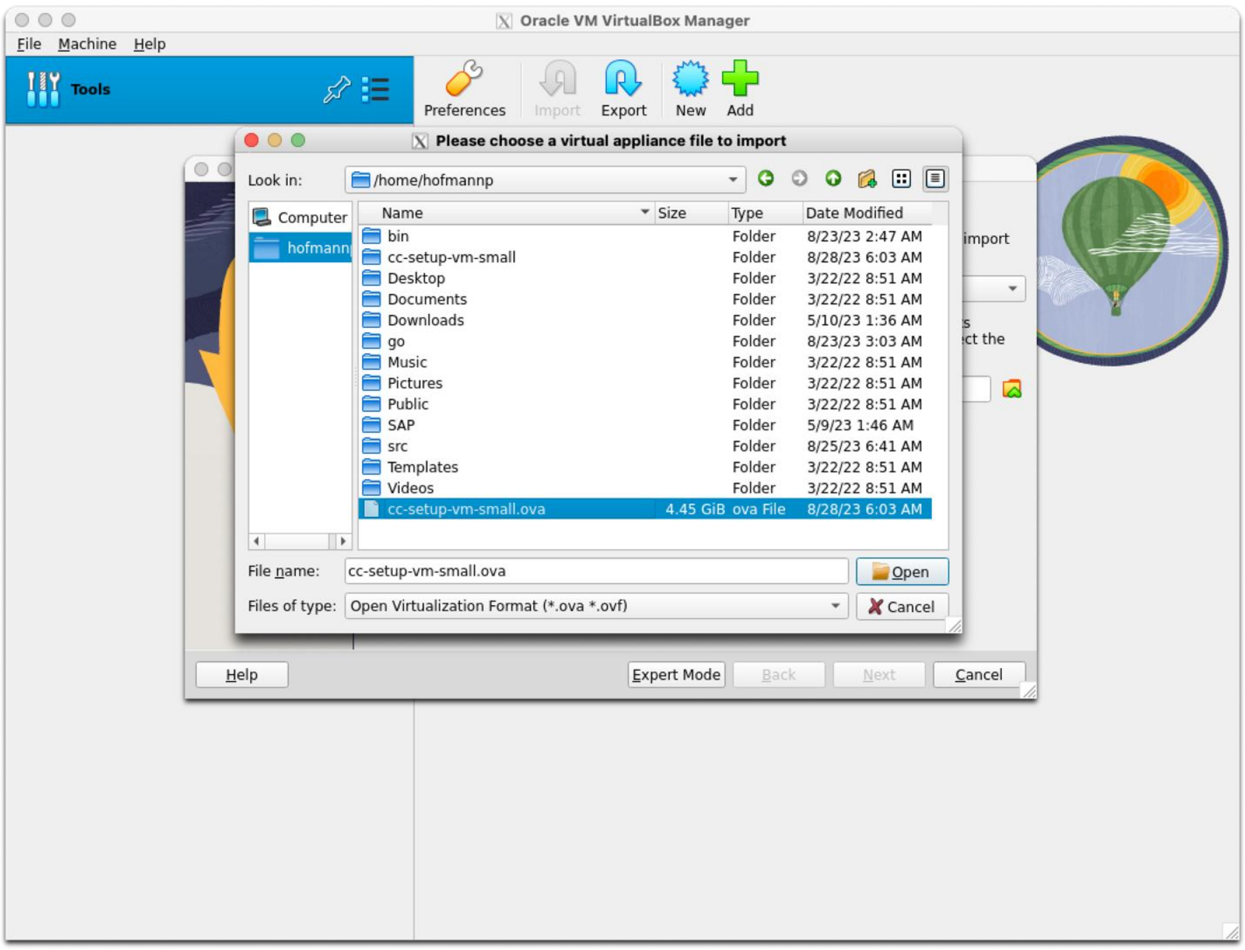
Funded by the Horizon Europe Framework Programme of the European Union

- remote-management.tar: TAR file of the exported Docker image
- remote-management-workdir.tar: Working directory to be used in conjunction with the Docker image
- cc-setup-vm.ova: OVF/OVA file with the tool VM for import into your hypervisor.
- cc-setup-workdir.tar.gz: TAR file with the workdir directory structure for setup tool VM.

A.3 USING THE SUPPLIED VM OVA FILE

Partners will be given the Guest VM Setup tool VM's image in OVA (`cc-setup-vm.ova`) format to import in their VMM (Virtual Machine Manager) of choice. The following steps show how to import the OVA file into the Virtual VMM on Linux.

Note that these steps have been successful on our test system. The specific steps might vary depending on your host operating system or hypervisor software. Please contact our support team if there are any difficulties installing or accessing the tool VM.



Look in: /home/hofmannp

Name	Size	Type	Date Modified
bin		Folder	8/23/23 2:47 AM
cc-setup-vm-small		Folder	8/28/23 6:03 AM
Desktop		Folder	3/22/22 8:51 AM
Documents		Folder	3/22/22 8:51 AM
Downloads		Folder	5/10/23 1:36 AM
go		Folder	8/23/23 3:03 AM
Music		Folder	3/22/22 8:51 AM
Pictures		Folder	3/22/22 8:51 AM
Public		Folder	3/22/22 8:51 AM
SAP		Folder	5/9/23 1:46 AM
src		Folder	8/25/23 6:41 AM
Templates		Folder	3/22/22 8:51 AM
Videos		Folder	3/22/22 8:51 AM
cc-setup-vm-small.ova	4.45 GiB	ova File	8/28/23 6:03 AM

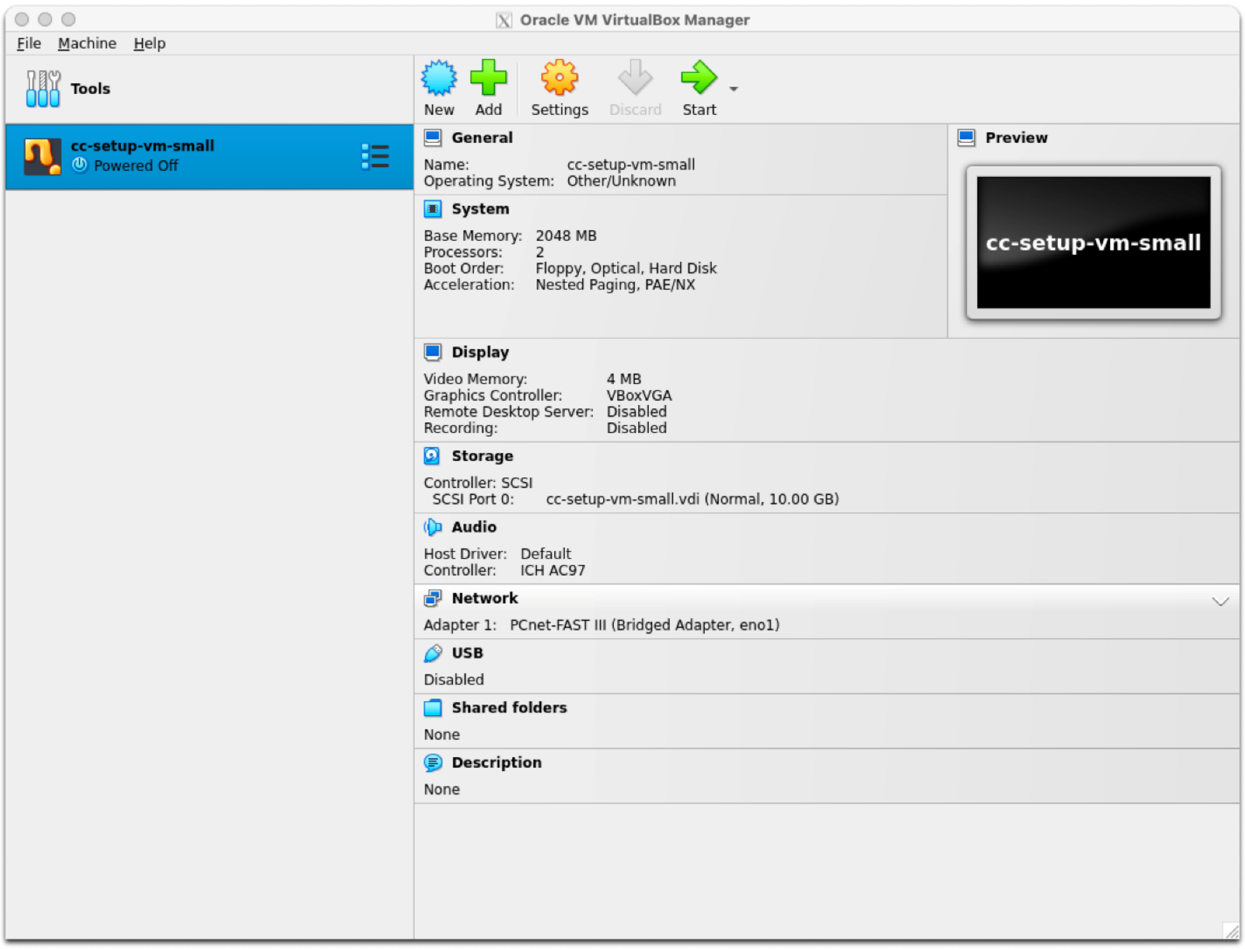
File name: cc-setup-vm-small.ova

Files of type: Open Virtualization Format (*.ova *.ovf)

Figure 123: Import of OVA file

Figure 123 shows how to select and import the `cc-setup-vm.ova` file into VirtualBox.

Funded by the Horizon Europe Framework Programme of the European Union



cc-setup-vm-small
Powered Off

General
Name: cc-setup-vm-small
Operating System: Other/Unknown

System
Base Memory: 2048 MB
Processors: 2
Boot Order: Floppy, Optical, Hard Disk
Acceleration: Nested Paging, PAE/NX

Display
Video Memory: 4 MB
Graphics Controller: VBoxVGA
Remote Desktop Server: Disabled
Recording: Disabled

Storage
Controller: SCSI
SCSI Port 0: cc-setup-vm-small.vdi (Normal, 10.00 GB)

Audio
Host Driver: Default
Controller: ICH AC97

Network
Adapter 1: PCnet-FAST III (Bridged Adapter, eno1)

USB
Disabled

Shared folders
None

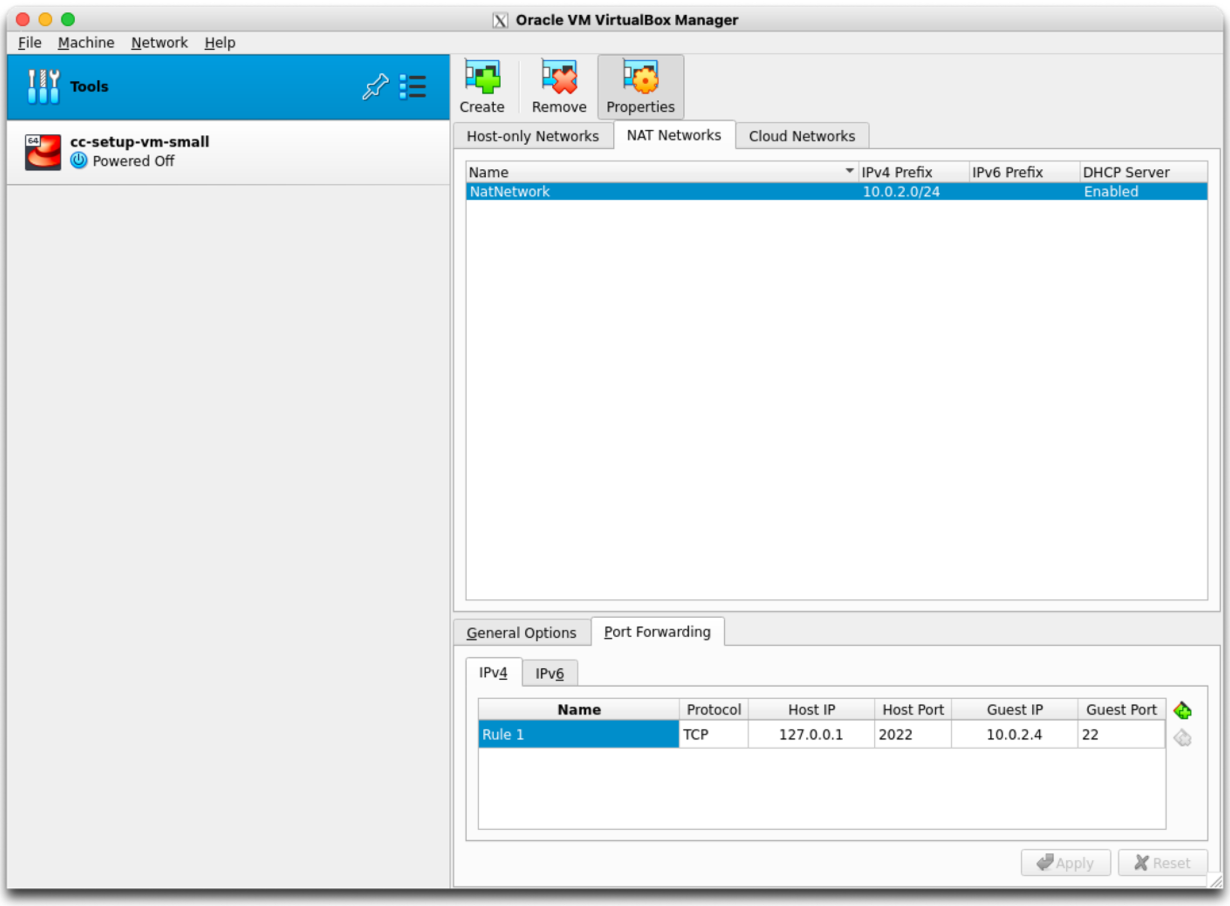
Description
None

Figure 124: Result of OVA import

Figure 124 shows the resulting screen in VirtualBox after the import.

Then “Tools” on the left and create a NAT network for VirtualBox (see Figure 125).

Funded by the Horizon Europe
Framework Programme of the European Union



Name	IPv4 Prefix	IPv6 Prefix	DHCP Server
NatNetwork	10.0.2.0/24		Enabled

Name	Protocol	Host IP	Host Port	Guest IP	Guest Port
Rule 1	TCP	127.0.0.1	2022	10.0.2.4	22

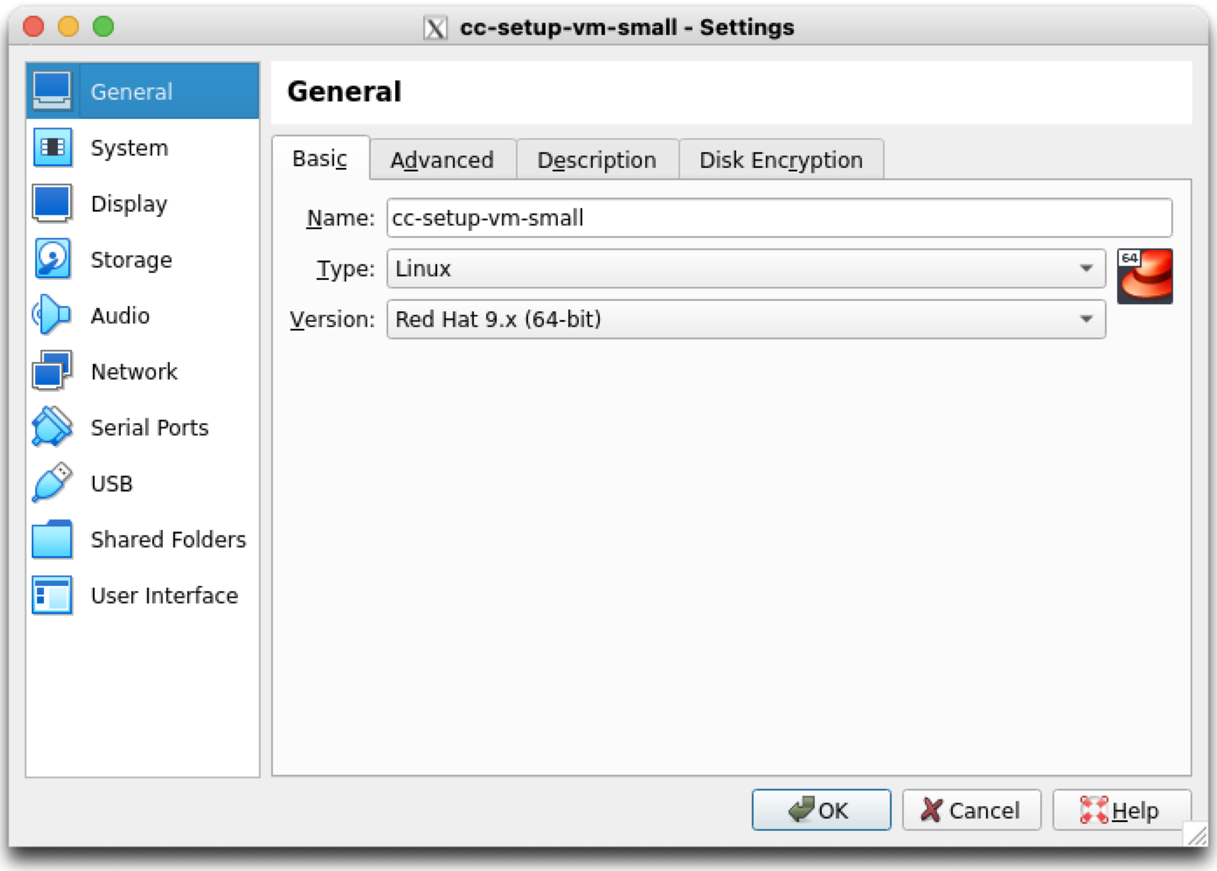
Figure 125: NAT network in VirtualBox

The NAT network should have the following parameters:

- Prefix 10.0.2.0/24
- DHCP enabled
- A port forwarding from host (127.0.0.1) port 2022 to port 22 on the guest system (10.0.2.4, IP address might vary in your setup – look into the VM once booted).

Now select the VM “cc-setup-vm-small” on the left and press “Settings”.

Funded by the Horizon Europe Framework Programme of the European Union



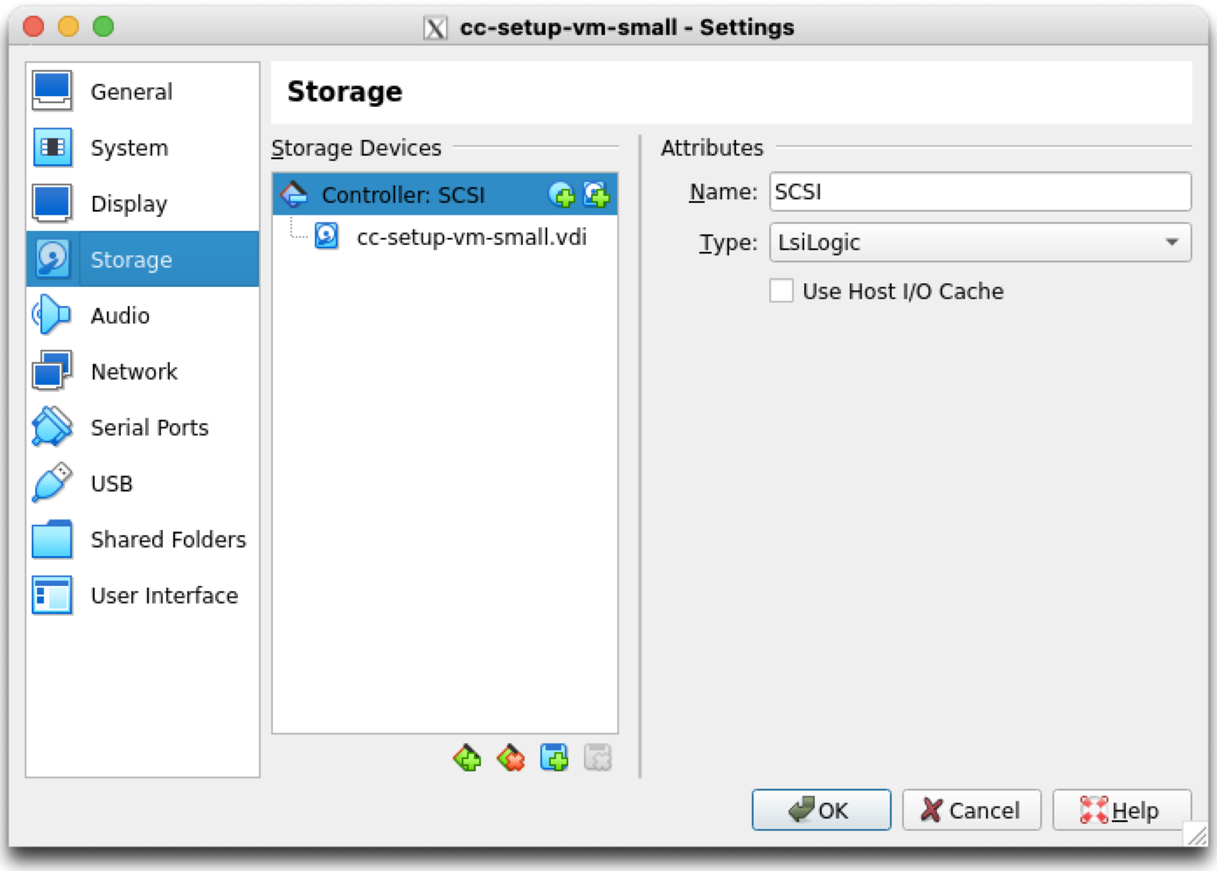
Name: cc-setup-vm-small
Type: Linux
Version: Red Hat 9.x (64-bit)

Figure 126: Change VM OS and version

Figure 126 shows how to change the OS to “Linux” and the OS version to “Red Hat 9.x (64-bit)”. This is necessary because our tool VM is a 64-bit Centos Linux system.

Now go to the Storage tab to create a large disk for the workdir.

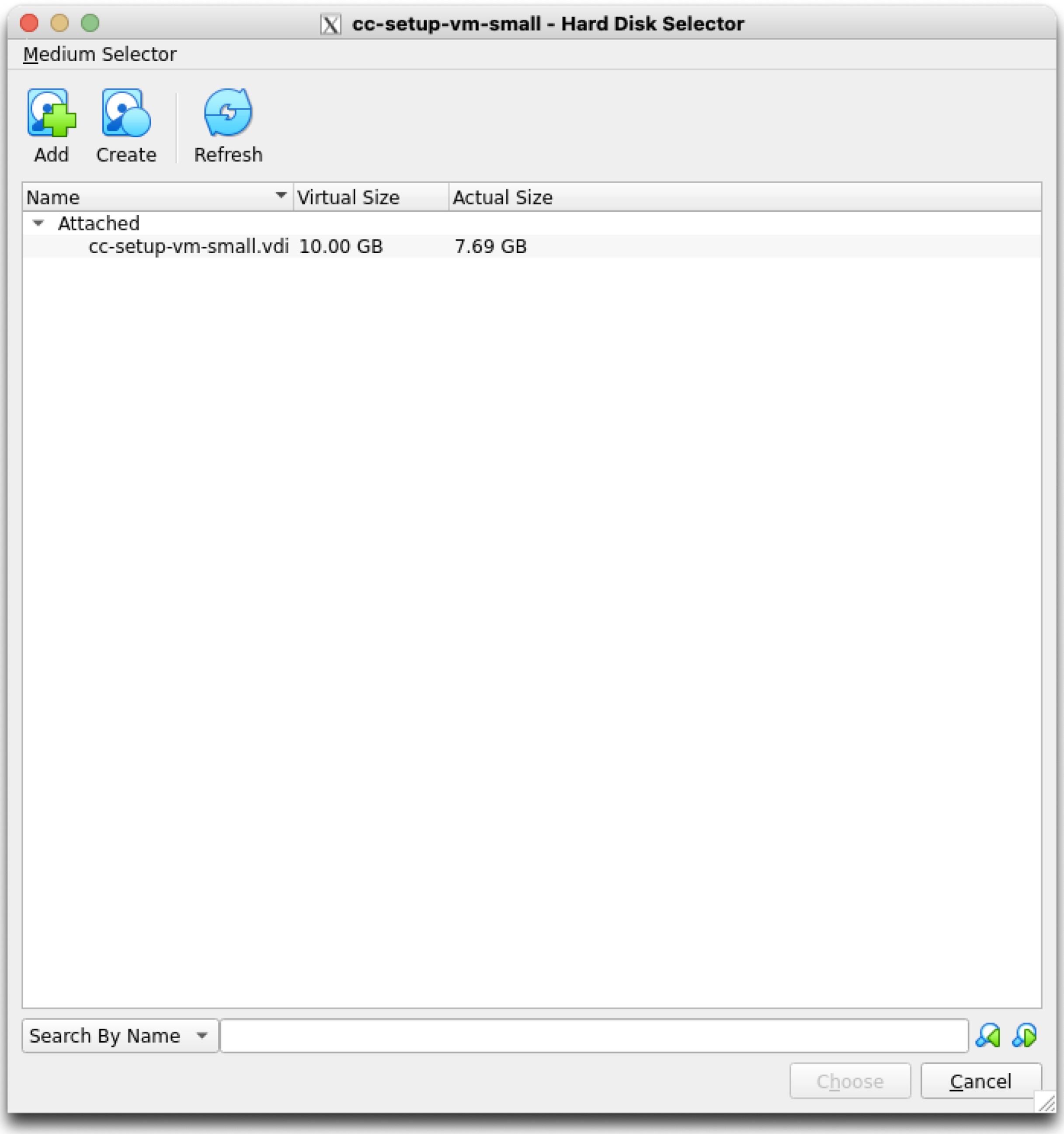
Funded by the Horizon Europe Framework Programme of the European Union



Name: SCSI
Type: LsiLogic
Use Host I/O Cache: ☐

Figure 127: Storage tab in VM settings

Funded by the Horizon Europe Framework Programme of the European Union

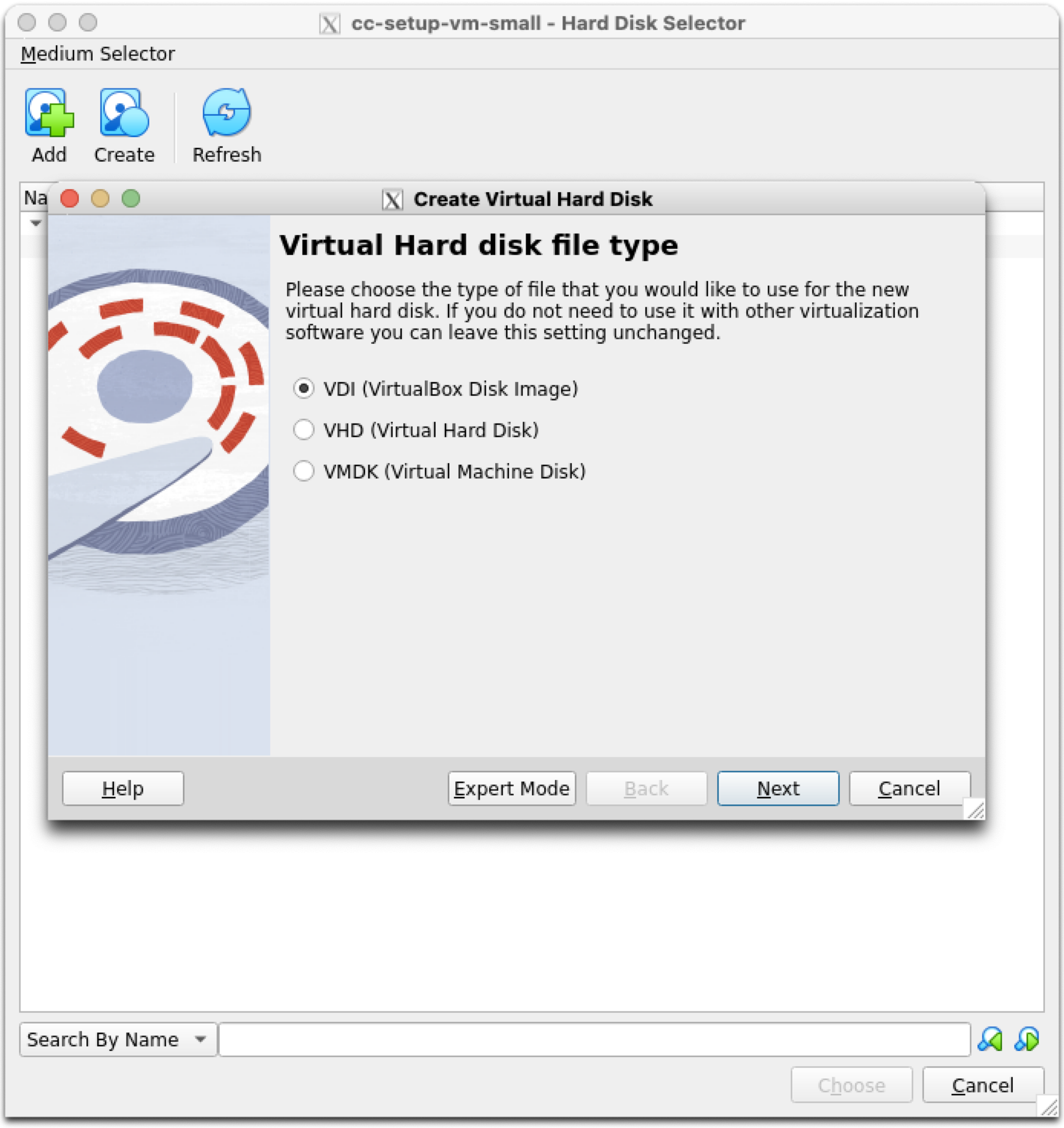


Name	Virtual Size	Actual Size
Attached		
cc-setup-vm-small.vdi	10.00 GB	7.69 GB

Figure 128: Hard disk selection in VirtualBox

Figure 128 shows the hard disk selection dialog after pressing “Add disk”. Here, press “Create” to create a new disk that is later used as the working directory for creating new guest images.

Funded by the Horizon Europe Framework Programme of the European Union



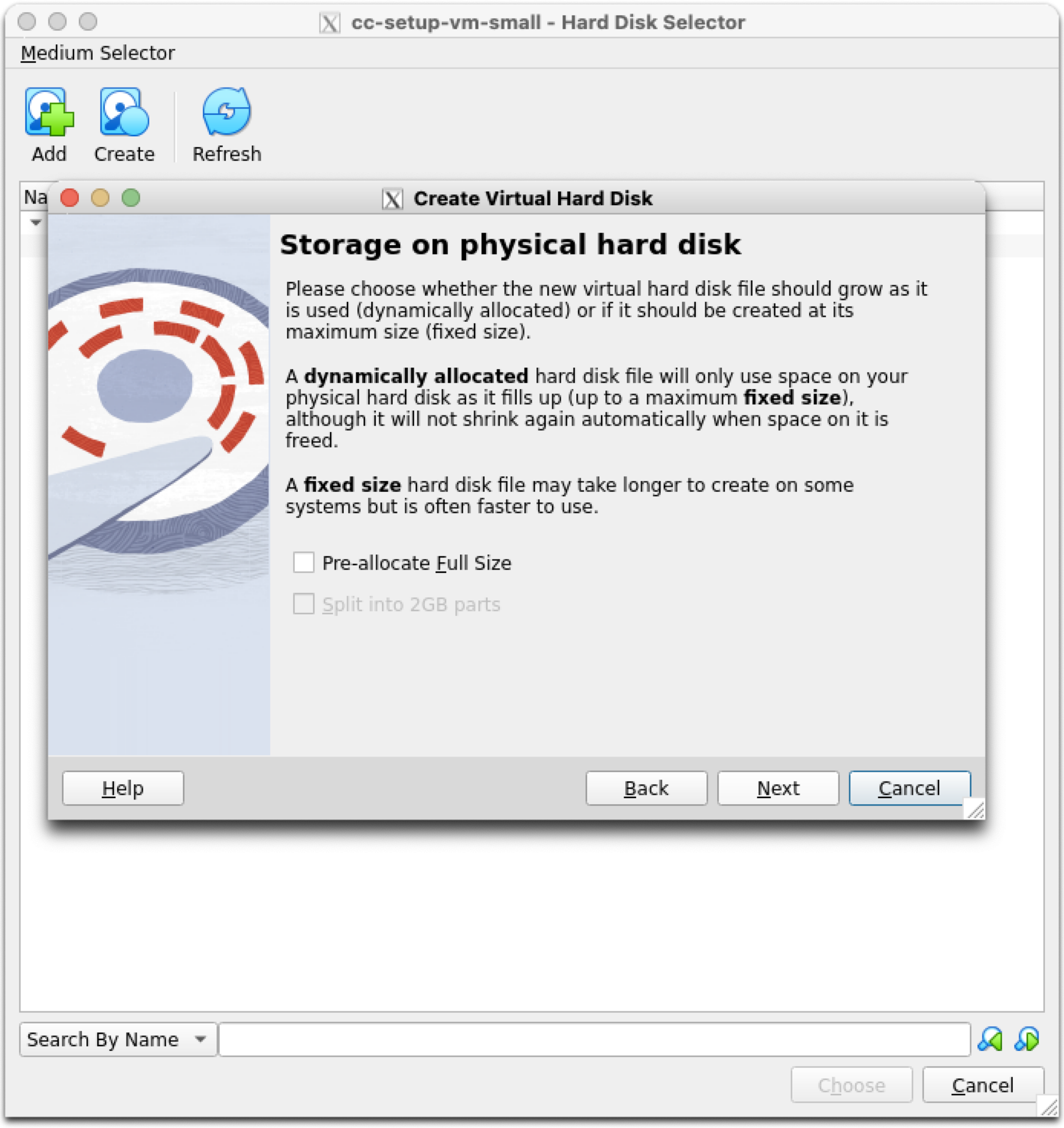
Virtual Hard disk file type

Please choose the type of file that you would like to use for the new virtual hard disk. If you do not need to use it with other virtualization software you can leave this setting unchanged.

[✓ VDI (VirtualBox Disk Image)]
VHD (Virtual Hard Disk)
VMDK (Virtual Machine Disk)

Figure 129: Virtual disk file type dialog

Funded by the Horizon Europe Framework Programme of the European Union



Storage on physical hard disk

Please choose whether the new virtual hard disk file should grow as it is used (dynamically allocated) or if it should be created at its maximum size (fixed size).

A **dynamically allocated** hard disk file will only use space on your physical hard disk as it fills up (up to a maximum **fixed size**), although it will not shrink again automatically when space on it is freed.

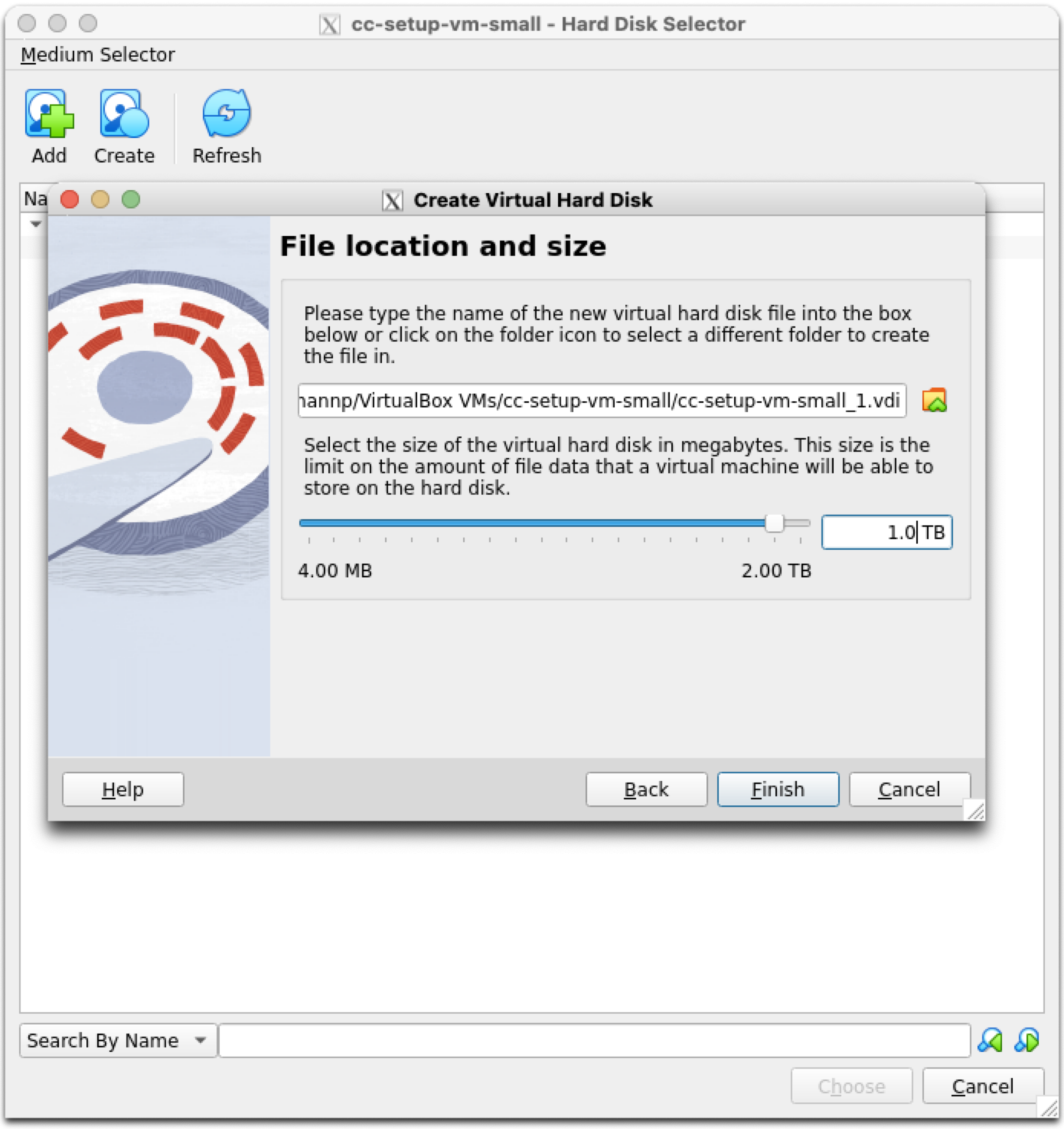
A **fixed size** hard disk file may take longer to create on some systems but is often faster to use.

☐ Pre-allocate Full Size

☐ Split into 2GB parts

Figure 130: Virtual disk allocation dialog

Funded by the Horizon Europe Framework Programme of the European Union



File location and size

Please type the name of the new virtual hard disk file into the box below or click on the folder icon to select a different folder to create the file in.

ıannp/VirtualBox VMs/cc-setup-vm-small/cc-setup-vm-small_1.vdi

Select the size of the virtual hard disk in megabytes. This size is the limit on the amount of file data that a virtual machine will be able to store on the hard disk.

1.0 TB

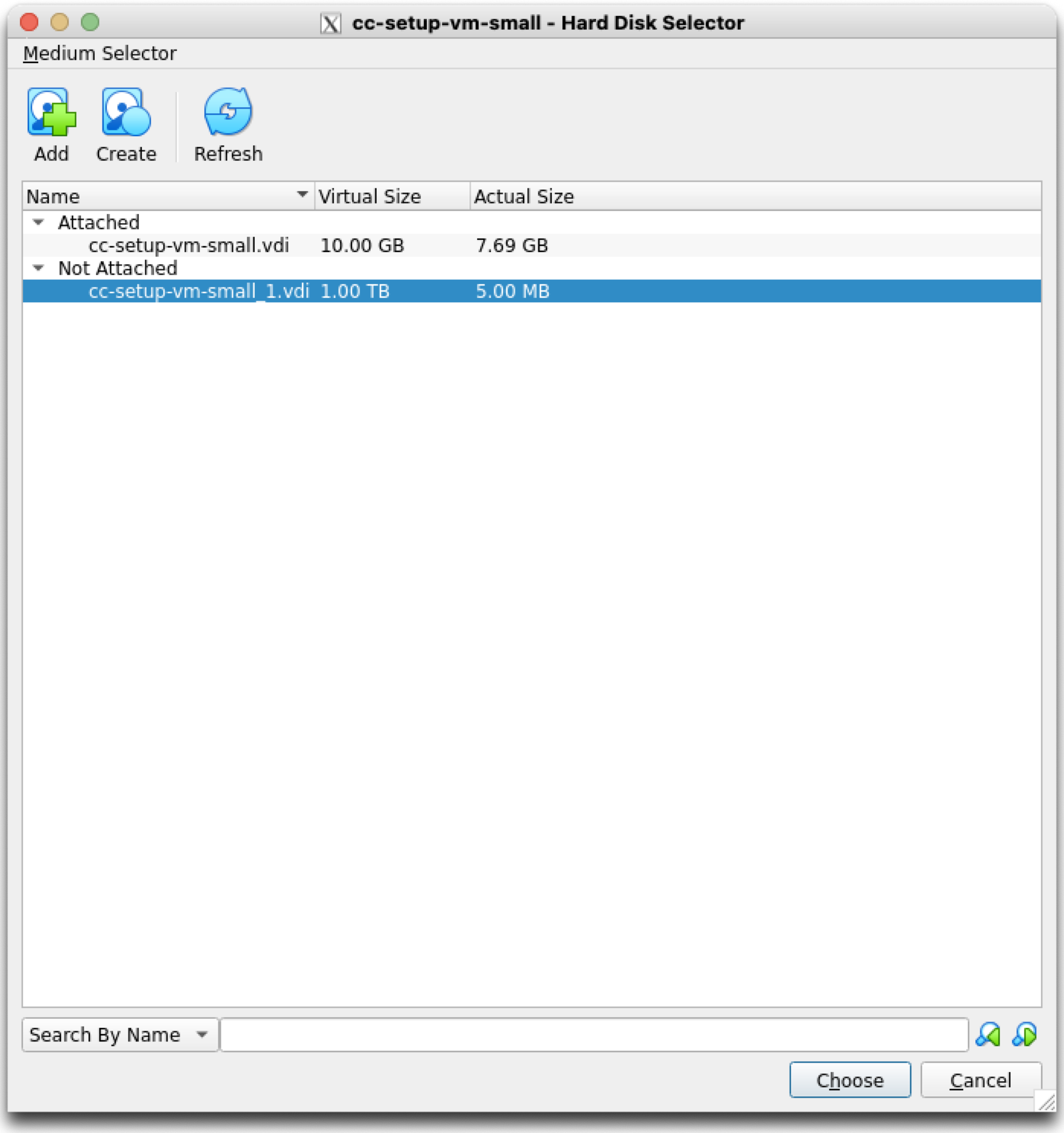
4.00 MB 2.00 TB

Figure 131: Virtual disk size dialog

Funded by the Horizon Europe Framework Programme of the European Union

Figure 129, Figure 130, and Figure 131 show the different steps for the new virtual disk creation:

- Use the type „VDI"
- Do not pre-allocate all storage (saves disk space by using sparse files)
- Define a size large enough to hold the templates as well as any number of guest images you might want to create (in this case 1 TB).

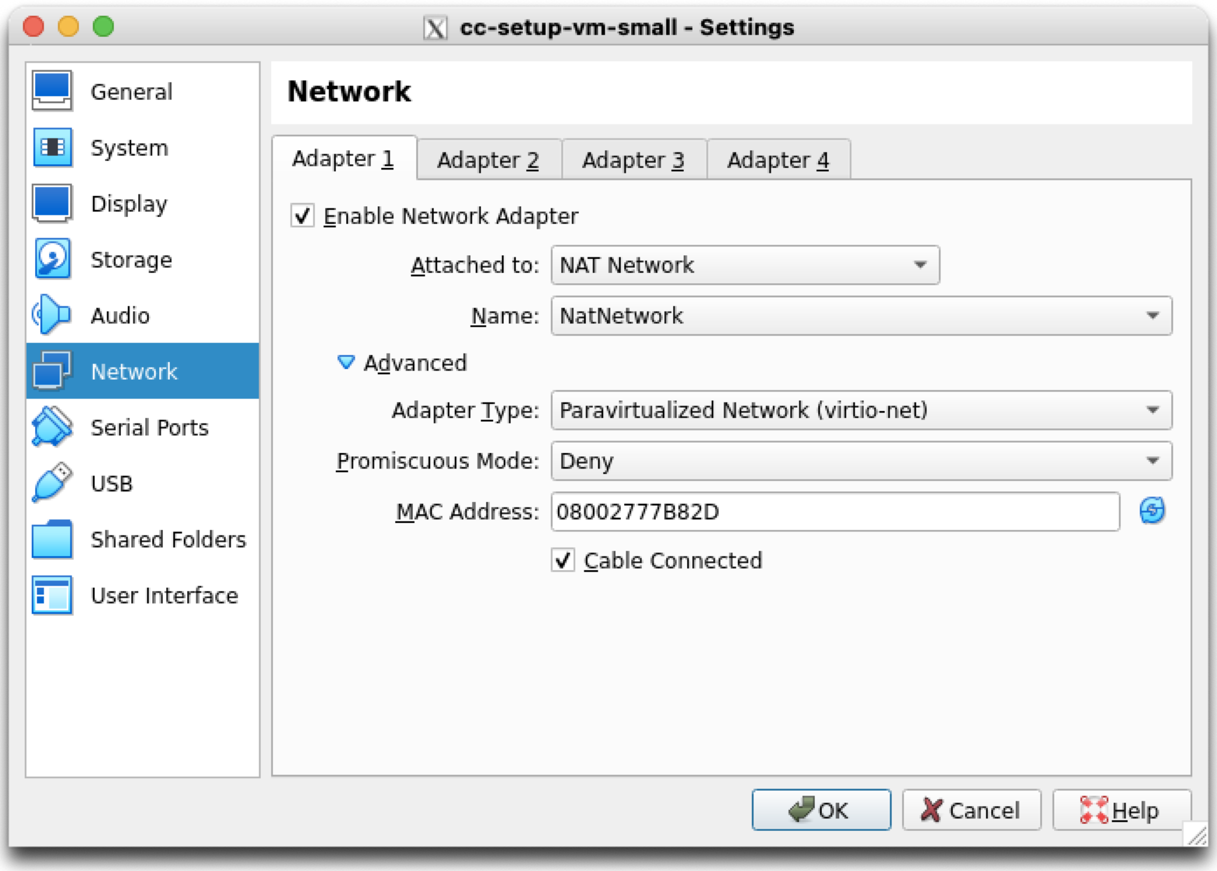


Name	Virtual Size	Actual Size
Attached		
cc-setup-vm-small.vdi	10.00 GB	7.69 GB
Not Attached		
cc-setup-vm-small_1.vdi	1.00 TB	5.00 MB

Figure 132: Result of virtual disk creation

Figure 132 shows the result of the disk creation: A new 1TB disk has been created, that only uses 5 MB of disk space at the moment.

Funded by the Horizon Europe Framework Programme of the European Union



☑ Enable Network Adapter
Attached to: NAT Network
Name: NatNetwork
Advanced
Adapter Type: Paravirtualized Network (virtio-net)
Promiscuous Mode: Deny
MAC Address: 08002777B82D
☑ Cable Connected

Figure 133: VirtualBox network configuration

Figure 133 show the final configuration of the network interface in the VirtualBox appliance:

- Select the NAT network „NatNetwork“ define previously.
- Choose adapter type “virtio-net”
- Select “Cable Connected”

Now startup the virtual machine. Because we have created a port forwarding, you can log into the virtual machine like this:

```
ssh -p2022 -Y user@127.0.0.1
su -
```

Note that the password for user and root is “udel1718”.

As root, check that the /dev/sdb is the new 1TB virtual disk:

```
$ fdisk -l /dev/sdb
Disk /dev/sdb: 1 TiB, 1099511627776 bytes, 2147483648 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
```

Funded by the Horizon Europe Framework Programme of the European Union

```
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

Then, create a file system on this disk:

```
$ fdisk /dev/sdb

Welcome to fdisk (util-linux 2.37.4).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0xe9d0e0fa.

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-2147483647, default 2048):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-2147483647, default 2147483647):

Created a new partition 1 of type 'Linux' and of size 1024 GiB.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

$ mkfs.xfs /dev/sdb1
meta-data=/dev/sdb1              isize=512    agcount=4, agsize=67108800 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1    bigtime=1 inobtcount=1 nrext64=0
data     =                       bsize=4096   blocks=268435200, imaxpct=25
         =                       sunit=0      swidth=0 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=131071, version=2
         =                       sectsz=512   sunit=0 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0

$ mkdir /data
```

Funded by the Horizon Europe Framework Programme of the European Union

Add the new filesystem to /etc/fstab:

```
$ vi /etc/fstab
#
# /etc/fstab
# Created by anaconda on Mon Aug 28 09:24:49 2023
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
/dev/sda1 /                                 xfs     defaults        0 0
/dev/sdb1 /data                                 xfs     defaults        0 0
```

Mount the disk:

```
$ /bin/mount -a
mount: (hint) your fstab has been modified, but systemd still uses
       the old version; use 'systemctl daemon-reload' to reload.
$ systemctl daemon-reload
```

Unpack the working dir tar (obtained from the NAS):

```
$ cd /data
$ tar xvf cc-setup-workdir.tar.gz
```

And link the the new working dir to /home/user/workdir:

```
$  cd /home/user/
$ rm -rf workdir
$ ln -s /data/cc-setup-workdir workdir
```

The system is now ready to create guest images as described in main part of this document.

Note: Alternatively, guest owner can request a KVM-compatible QCOW2 image from us that can directly be used with the Linux KVM hypervisor. This file has already 1 TB of free space on the home partition.

Funded by the Horizon Europe Framework Programme of the European Union

A.4 END-TO-END EXAMPLE

In this section we describe a complete end-to-end procedure for obtaining the necessary tools, preparing a Centos 9-based guest VM, and finally deploying it on our lab system. This section is meant to be a step-for-step guide for project partners for preparing their own workloads as protected images to run on our platform.

Obtaining and Installing the Software

Download the following files from the NAS device in our lab (from the cc_download shared folder, refer to the appendix):

- remote-management.tar: TAR file of the exported Docker image
- remote-management-workdir.tar: Working directory to be used in conjunction with the Docker image
- cc-setup-vm.ova: OVF/OVA file with the tool VM for import into your hypervisor.
- cc-setup-workdir.tar: Tool VM workdir with the prepared Centos 9 and Debian 11 templates (rename/move the included directory to /home/user/workdir).

Import the Docker image into your local repository and unpack the workdir for the Docker image:

```
docker load --input remote-management.tar
tar xvf remote-management-workdir.tar
```

The import of the OVF/OVA file cc-setup-vm.ova is described in the appendix.

Preparing the Guest VM

The guest VM must be prepared in the tool VM provided by us. Start the guest VM in your hypervisor (eg. VirtualBox) and login as “root” with password “udel1718”. Alternatively, you can use the user “user” with password “udel1718”, but the commands must be run as root because they require special privileges.

Prepare your guest VM from the Centos 9 template with this command (please choose your own recovery password):

```
prepare --centos centos-test Luci-0815-deli
```

The prepare script outputs the following:

```
Verifications...
Verifications OK
Password management...
33+0 records in
33+0 records out
33 bytes copied, 0.00022163 s, 149 kB/s
Password management OK
Partition is /dev/nbd9
```

Funded by the Horizon Europe Framework Programme of the European Union

```
Connected qemu-nbd device
Encrypting partition p2...
Added user password
Added high-entropy  password
User password and high entropy password correctly added, deleting standard password...
AMD files...
PDH EP384 D256 60f50e13747224658c3d84bebc21a2e888525649185f62ea19d314d7bf7eb72b
 ⬑ PEK EP384 E256 0a8e2bfd971103857c117e2b3b37bf8865a75441d945cbd9e46d7db764c5ef20
   •⬑                      OCA                       EP384                       E256
aed15dc56800252cc82114964cac98f47f42c068f19feaa189d14a2321800b06
    ⬑                      CEK                       EP384                       E256
537e69f2709c213aa69be03ae8be5307cf7963708de56a7a8658aed29117ab07
       ⬑                     ASK                       R4096                       R384
95cba79ba3c77daea79f741bade8156a50b1c59f6d6fda104d16dd264729f5ee8989522f3711fc7c847
19921ceb31bc0
         •⬑                    ARK                       R4096                      R384
569da618dfe64015c343db6d975e77b72fdeacd16edd02d9d09b889b8f0f1d91ffa5dfbd86f7ac574a1
a7883b7a1e737


 • = self signed, ⬑ = signs, •̷ = invalid self sign, ⬑̸ = invalid signs
AMD files OK
Disabling Debian LVM...
umount: /dev/mapper/debian--vg-root: no mount point specified.
Unmounting partitions...
umount: /dev/mapper/encrypted-image: no mount point specified.
umount: /dev/mapper/encrypted-boot: no mount point specified.
Device encrypted-image is not active.
Device encrypted-boot is not active.
Disconnecting nbd devices...
/dev/nbd0 disconnected
/dev/nbd1 disconnected
/dev/nbd10 disconnected
/dev/nbd11 disconnected
/dev/nbd12 disconnected
/dev/nbd13 disconnected
/dev/nbd14 disconnected
/dev/nbd15 disconnected
/dev/nbd2 disconnected
/dev/nbd3 disconnected
/dev/nbd4 disconnected
/dev/nbd5 disconnected
/dev/nbd6 disconnected
/dev/nbd7 disconnected
```

Funded by the Horizon Europe Framework Programme of the European Union

```
/dev/nbd8 disconnected
/dev/nbd9 disconnected
qemu-nbd: Cannot open /dev/nbd9p1: No such file or directory
qemu-nbd: Cannot open /dev/nbd9p2: No such file or directory
Prepare : DONE
```

Please check the output for any errors or problems and contact our support team if there are any issues. Sometimes the prepare process only terminates successfully after trying a second time.

As a result of this process, the transfer directory contains a file centos_centos-test.qcow2:

```
[root@localhost workdir]# ls -lsk transfer/
total 51350352
51350352 -rw-r--r--. 1 root root 1099679662080 Aug 25 08:28 centos_centos-test.qcow2
```

Note that this is a sparse file, i.e. while the file size is shown as 1099679662080 (around 1TB), the actual number of 1024 sectors is only 51350352.

Additionally, this process created two key files in the workdir/keys subdirectory:

- `recovery-password_centos-test.txt`: This file contains the low-entropy key (ie. short password) that was given on the command line: “Luci-0815-deli”.
- `high-entropy-password_centos-test.txt`: This file contains a high-entropy (i.e. long password) that was generated during the prepare process. Because it is a high-entropy password, it was added with few PBKDF2 iterations in order to speed up the boot process of the guest VM later on.

Note: these two passwords must never be divulged to the cloud operator (our lab admin). If, in the course of debugging the process, we gain insight into these passwords, the guest owner should re-create the guest VM image with different passwords in private afterwards.

In a next step the guest owner might want to perform a number of customization steps, such as:

- Updating the underlying Linux distribution using apt or dnf. **Warning: a blanket “dnf update” or “apt upgrade” might lead to a re-build of the initramfs filesystem which breaks the boot process because the initramfs is built on a different kernel than the final guest VM will boot. Any dnf or apt actions that might lead to a**

Funded by the Horizon Europe Framework Programme of the European Union

rebuild of the initramfs must be avoided. If unsure, it is better to postpone any dnf/apt commands until the VM is running in the lab environment.

- Installing additional software from the distribution or other repositories.
- Installation of own binaries or data needed for the application.
- Re-configuration of the Linux operating system.

Issue

```
mount centos-test
```

to invoke a chroot environment with all necessary directories of the guest VM mounted. This environment can be used almost like a VM actually running on the target system. For example, try to install a sample hello world application:

```
[root@localhost /]# echo '#!/bin/bash'  > /usr/local/bin/hello_world.sh
[root@localhost /]# echo 'echo "Hello World!"'  >> /usr/local/bin/hello_world.sh
[root@localhost /]# chmod +x /usr/local/bin/hello_world.sh
[root@localhost /]#
```

Now you should perform further steps to personalize the guest VM:

- Change the root and user password from the default “Udel1718” to a secure password only known to you (“passwd root” and “passwd user” commands).
- Regenerate the host keys and make a copy of the public keys so that you have unique, trusted host keys for later SSH login and administration (see the appendix for details for Debian and Centos).

Follow this procedure to re-generate the host keys:

Delete the old host keys:

```
rm -f /etc/ssh/ssh_host*
```

Regenerate the host keys:

```
ssh-keygen -f /etc/ssh/ssh_host_rsa_key     -N '' -q -t rsa

ssh-keygen -f /etc/ssh/ssh_host_ecdsa_key   -N '' -q -t ecdsa

ssh-keygen -f /etc/ssh/ssh_host_ed25519_key -N '' -q -t ed25519
```

Make a copy of the .pub files and enter them into your local .ssh/known_hosts file.

Exit the chroot environment by typing “exit” or CTRL-D. The mounted directories from the guest image must be unmounted using:

```
unmount-images
```

Now the guest image is ready for transfer to the lab environment. In order to preserve the sparse file properties, use TAR to compress the image before transfer:

```
cd ~/workdir/transfer
tar -S -zcvf centos_centos-test.tar.gz centos_centos-test.qcow2
```

The resulting TAR file should then be uploaded to the designated directory on the lab NAS system (refer to the appendix for details).

As a final step, the launch bundle for the guest VM must prepared and some parameters be transmitted to the host system. This requires a working VPN connection to our lab environment.

Now prepare a new subdirectory for our new VM `centos_centos-test`:

```
mkdir workdir/centos_centos-test
```

Then create the launch bundle and send it to our host system:

```
$ docker run -v /home/hofmannp/bare_metal_cc/workdir:/workdir:z remote-management create-launch-bundle /workdir/centos_centos-test
Writing to file: /workdir/centos_centos-test/godh.cert
Writing to file: /workdir/centos_centos-test/tmp_tk.bin
Writing to file: /workdir/centos_centos-test/launch_blob.bin

Command Successful
16+0 records in
16+0 records out
16 bytes copied, 0.000166162 s, 96.3 kB/s
16+0 records in
16+0 records out
16 bytes copied, 0.000161821 s, 98.9 kB/s

$ docker run -v /home/hofmannp/bare_metal_cc/workdir:/workdir:z remote-management send http://10.0.12.96:5000 centos_centos-test user /workdir/passwords/user_pwd.txt /workdir/centos_centos-test
Files were successfully sent
[hofmannp@r370-2 bare_metal_cc]$
```

Note: This command requires that the user “user” (sample user name used for documentation purposes) is registered for the guest owner and the password has been written into the workdir/passwords/user_pwd.txt file. Of course, instead of “user” you would need to use your VPN user ID and password for this.

Note on SE Linux: If the host system is Linux and implements SE Linux mandatory access control, the Docker container might not be able to access the files in the host’s working directory. You might need to issue commands like “`chcon unconfined_u:object_r:container_file_t:s0 workdir/passwords/user_pwd.txt`” to adapt SE Linux file labels.

Now the lab admin must be notified about the upload of the TAR.GZ file and the sending of the certificates. After confirmation, the VM can be started using the remote management commands described in the next section.

Funded by the Horizon Europe Framework Programme of the European Union

Running the Guest VM

The following command starts the remote VM in paused mode:

```
$ docker run -ti --network="host" -v /home/hofmannp/bare_metal_cc/workdir:/workdir:z
remote-management start http://10.0.12.96:5000 centos_centos-test user /workdir/pass-
words/user_pwd.txt
```

```
VM centos_centos-test was successfully started in paused mode
```

Now that the VM is started in paused mode, we must perform the remote attestation/measurement and inject the high entropy disk key/password:

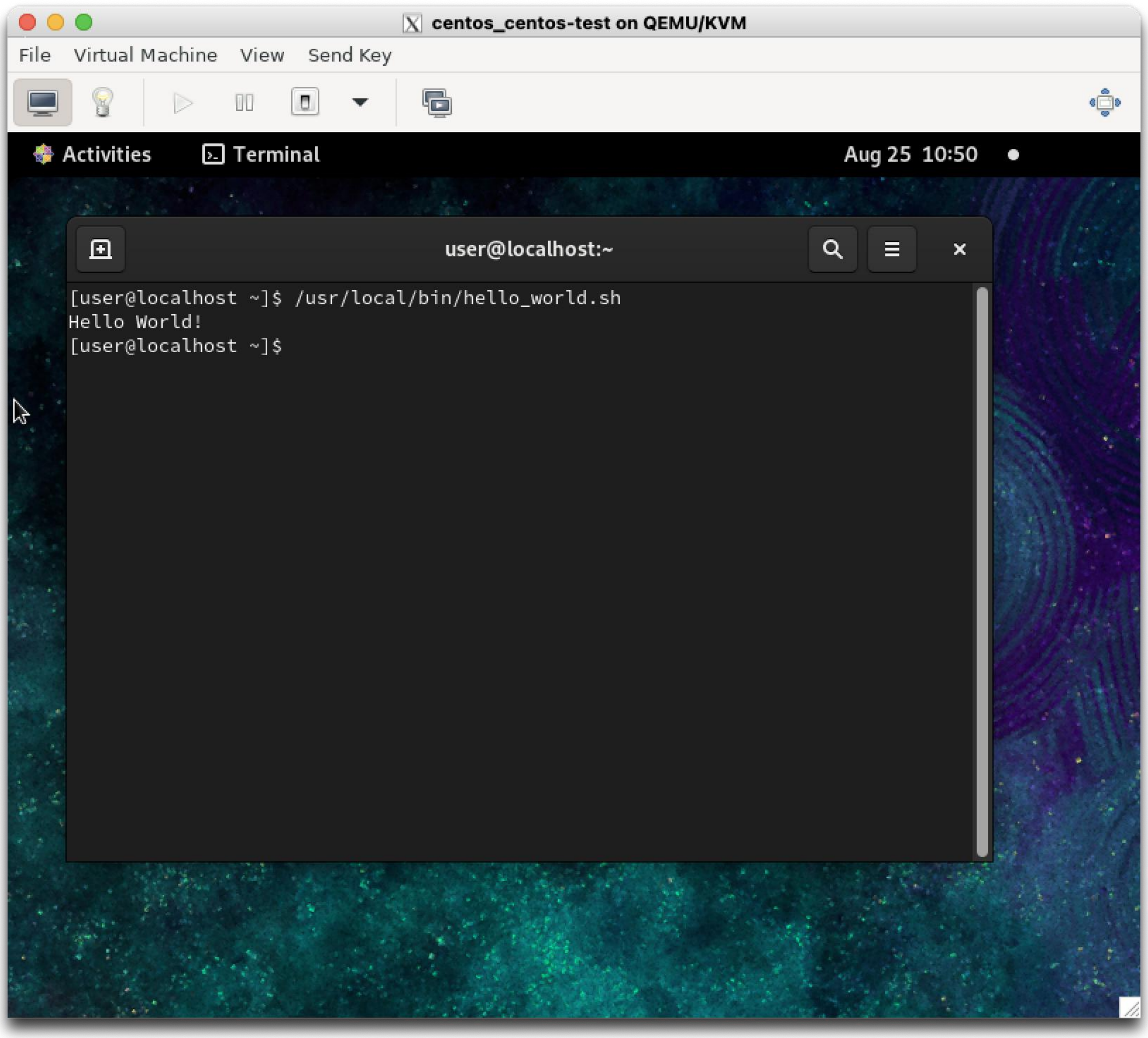
```
$ docker run -v /home/hofmannp/bare_metal_cc/workdir:/workdir:z remote-management
launch /workdir/centos_centos-test/high-entropy-password_centos-test.txt /work-
dir/centos_centos-test 10.0.12.96:5551
```

```
SEV query found API=1.53 build=5 policy=3


Getting Launch Measurement

Measure:        9a311c81d8e8078485f005c207a7519d2e971c9288dd906658dfc25269d1fc5a

Measure (b64):  b'mjEcgdjoB4SF8AXCB6dRnS6XHJKI3ZBmWN/CUmnR/Fo='

should be:      9a311c81d8e8078485f005c207a7519d2e971c9288dd906658dfc25269d1fc5a


Measurement matches, Injecting Secret


Secret Injection Successful, starting VM
```

Since the measurements match, the VM is started. Note the QMP protocol host and port “10.0.12.96:5551”. While the host IP address is fixed in our test environment, the QMP port 5551 must be defined for each VM. The port number is allocated by the lab admin during setup and communicated via email to the guest owner.

When looking at the GUI screen in our test environment, we can see that the hello word application is present (Figure 134):

Funded by the Horizon Europe
Framework Programme of the European Union



```
[user@localhost ~]$ /usr/local/bin/hello_world.sh
Hello World!
[user@localhost ~]$
```

Figure 134: Hello word application in the trusted guest VM

In order to stop the VM issue the shutdown and destroy commands:

```
$ docker run -v /home/hofmannp/bare_metal_cc/workdir:/workdir:z remote-management shutdown http://10.0.12.96:5000 centos_centos-test user /workdir/pass-words/user_pwd.txt
Command was sent successfully but VM encountered a problem. Current VM status is {"status":"running"}

$ sleep 30

$ docker run -v /home/hofmannp/bare_metal_cc/workdir:/workdir:z remote-management de-stroy http://10.0.12.96:5000 centos_centos-test user /workdir/passwords/user_pwd.txt
VM centos_centos-test was successfully shutdown
```

Funded by the Horizon Europe Framework Programme of the European Union

Security Checklist

To make sure that your VM really is really protected when running in the host environment, make sure:

- To not divulge the recovery and high-entropy passwords to anyone (not even the DT support team) and store them on your guest system in a safe location.
- Re-generate the SSH host keys and import them into your local SSH client configuration.
- Do not connect to a remote VM if the host keys seem to have changed!
- Change the default user and root passwords to something long and unique. The default passwords are mentioned in this document are not secure!

For added security, you could check the integrity of the template VMs by comparing files to the original Centos 9 and Debian distributions. You could also re-compile the OVMF/grub bundle and provide your own OVMF image to us.

A.5 FORMAL DEFINITION OF REMOTE MANAGEMENT WEB SERVICE (YAML)

For reference purposes we reproduce the YAML file for remote management REST service here:

```
openapi: 3.0.0
info:
  title: API for remote VM management
  version: 1.0.0
paths:
  /vm/start:
    post:
      summary: Start a VM in pause mode
      operationId: "vm.start"
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                vm_name:
                  type: string
                userid:
                  type: string
                password:
```

Funded by the Horizon Europe Framework Programme of the European Union

```
                  type: string
              required:
                - vm_name
                - userid
                - password
      responses:
        '200':
          description: OK
        '500':
          description: Internal server error
  /vm/destroy:
    post:
      summary: Destroy VM
      operationId: "vm.destroy"
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                vm_name:
                  type: string
                userid:
                  type: string
                password:
                  type: string
              required:
                - vm_name
                - userid
                - password
      responses:
        '200':
          description: OK
        '500':
          description: Internal server error
  /vm/shutdown:
    post:
      summary: Shutdown VM
      operationId: "vm.shutdown"
      requestBody:
        required: true
```

Funded by the Horizon Europe Framework Programme of the European Union

```
        content:
          application/json:
            schema:
              type: object
              properties:
                vm_name:
                  type: string
                userid:
                  type: string
                password:
                  type: string
              required:
                - vm_name
                - userid
                - password
      responses:
        '200':
          description: OK
        '500':
          description: Internal server error
  /vm/status:
    post:
      summary: Return status of the host
      operationId: "vm.status"
      requestBody:
        required: true
        content:
          application/json:
            schema:
              type: object
              properties:
                vm_name:
                  type: string
                userid:
                  type: string
                password:
                  type: string
              required:
                - vm_name
                - userid
                - password
      responses:
```

Funded by the Horizon Europe Framework Programme of the European Union

```
        '200':
          description: OK
        '500':
          description: Internal server error
```

A.6 UPDATING THE TEMPLATE FILES

The Debian 11 and Centos 9 templates must be regularly updated to fix bugs and security problems. This activity needs to be carried out by the provider of the confidential computing infrastructure on a regular basis, so this section is not relevant to partners wanting to use the provided templates to create guest VMs.

Before the update, the template images should be saved to have a fallback position if there are problems with the updated images:

```
tar --hole-detection=seek -S -cvf template-backup.tar centos9-en-
crypted-template.qcow2 debian11-encrypted-template.qcow2
```

This command preserves the sparse file properties of the templates. The resulting template-backup.tar can also be used to transfer the updated images later to the tool VM.

After saving the images, the Debian 11 and Centos 9 images should be started in turn.

On Debian 11, as root, the following commands update the packages:

```
apt update

apt upgrade
```

For Debian 11, the following lines have been added to `/etc/apt/apt.conf.d/50unat-tended-upgrades` to avoid kernel upgrades:

```
Unattended-Upgrade::Package-Blacklist {
"linux-generic";
"linux-image-generic";
"linux-headers-generic";
};
```

On Centos 9, as root, the following command updates the packages:

```
dnf update
```

For Centos 9, the following line has been added to `/etc/dnf/dnf.conf` to avoid kernel upgrades:

```
exclude=kernel* redhat-release* kmod-kvdo
```

Funded by the Horizon Europe Framework Programme of the European Union

After both templates are checked to still work with measured start and basic OS operation is possible, re-create the TAR file as outlined above and use it to update the templates in the tool VM.

Note on Kernel upgrades: Kernel upgrades require manual changes to the grub.cfg because both templates have been modified to be "Confidential Computing-enlightened". The procedures for kernel upgrades will be documented in later versions of this document.

In the tool VM, the TAR file must be unpacked. The template QCOW2 files must be renamed to `centos9-template.qcow2` and `debian-template.qcow2`, respectively. Afterwards, issue the following commands as root to reclaim space for optimal download size of the tool VM:

```
rm -f /home/user/workdir/keys/* /home/user/workdir/transfer/*

fstrim -av
```

The tool VM must then be exported as OVF/OVA file using the following command:

```
$ sudo /home/hofmannp/go/bin/ovf-export --list
UUID                     Name
---------------------------------------------------------
808a59a6d53b4f9ebfa3cb8df75ef8f3  cc-setup-vm
$ mkdir cc-setup-vm
$          sudo          /home/hofmannp/go/bin/ovf-export          -id
808a59a6d53b4f9ebfa3cb8df75ef8f3 -output ~/cc-setup-vm -ova
```

The ovf-export is available as open source [OVFEXPORT]. The following changes were necessary to make the software run on our system:

```
diff -ruw ovf-export.orig/qemu-utils/qemu-img-convert.go ovf-export/qemu-utils/qemu-
img-convert.go

--- ovf-export.orig/qemu-utils/qemu-img-convert.go   2023-08-25 06:40:51.604966171 -
0400

+++ ovf-export/qemu-utils/qemu-img-convert.go  2023-08-25 06:41:21.531023780 -0400

@@ -3,7 +3,7 @@

 import (

        "fmt"

        "os/exec"

-       "runtime"

+//     "runtime"

        "strconv"
```

Funded by the Horizon Europe Framework Programme of the European Union

```
        "strings"
 )
@@ -44,7 +44,7 @@
                vmdkopts = append(vmdkopts, "zeroed_grain")
        }

-       var args = []string{"convert", "-m", strconv.Itoa(runtime.NumCPU()), "-O", "vmdk"}
+       var args = []string{"convert", "-m", "16", "-O", "vmdk"}

        if len(vmdkopts) > 0 {
                args = append(args, "-o", strings.Join(vmdkopts, ","))
```

The software must be recompiled with

```
go install -a ./cmd/ovf-export
```

A.7 SETUP ON THE HOST SYSTEM FOR NEW GUEST VMS

The host admin (i.e. the infrastructure or cloud provider) must perform the following steps for setting up a new guest VM after having been notified by the guest owner that a new VM has been uploaded:

1. Copy any existing .xml file from an existing example (for Centos or Debian)
2. In the .xml file, rename the domain (global replace of name, should be 3 times)
3. In the .xml file, change the UUID so that it is unique
4. Adjust the path to the .qcow2 file
5. Copy the /var/lib/libvirt/qemu/nvram/old_template_VARS.fd file to /var/lib/libvirt/qemu/nvram/new-name_VARS.fd.
6. Change the QMP port to some unique value (will be used later for measured launch).
7. Open new QMP port on host firewall: `firewall-cmd --zone=public --add-port=XXXX/tcp –permanent; systemctl reload firewalld`
8. Make the domain known to libvirt: `virsh define new-name.xml`

After the Guest Owner provided the certificates for measured launch:

Funded by the Horizon Europe Framework Programme of the European Union

1. Paste the contents of the user-provided “session.b64” file into the <session> clause of the .xml file.
2. Paste the contents of the user-provided “godh.b64” file into the <dhCert> clause of the .xml file.
3. Test that the new domain can be started in paused mode: `virsh start --paused new-name`.

A.8 CONFIGURE THE REST SERVER AS SYSTEMD SERVICE

This activity needs to be carried out by the cloud or infrastructure provider to establish the remote management service for guest owners.

Create a file `/etc/systemd/system/remote-management.service` with the following contents:

```
[Unit]
Description=remote management daemon
After=network.target

[Service]
Type=notify
# the specific user that our service will run as
User=root
Group=root
WorkingDirectory=/usr/local/remote-management/rest_server
ExecStart=/usr/local/remote-management/env/bin/gunicorn -b 0.0.0.0:5000 app:app
ExecReload=/bin/kill -s HUP $MAINPID
KillMode=mixed
TimeoutStopSec=5
PrivateTmp=true

[Install]
WantedBy=multi-user.target
```

This requires the following:

Funded by the Horizon Europe Framework Programme of the European Union

- A chroot environment with the necessary python modules and the gunicorn server under `/usr/local/remote-management/env`.
- The prepared rest server installed under `/usr/local/remote-management/rest_server`.
- If there are problems with SE linux permissions issue:
    - `restorecon /usr/local/remote-management`
    - `chcon unconfined_u:object_r:bin_t:s0 /usr/local/remote-management/env/bin/*`

Enable and start the service like this:

```
systemctl enable remote-management

systemctl start remote-management
```

If any changes to the user.py configuration are made, restart the service:

```
systemctl restart remote-management
```

Funded by the Horizon Europe Framework Programme of the European Union

A.9 API DESCRIPTION

Guest owners need a remote access API to manage their VMs themselves. The interface features:

- Endpoints to execute the basic libvirt-based commands on a specified VM (start, shutoff, status...).
- Basic authentication for each endpoint: guest owner ID and password (assigned by DT), assigned VMs names.
- QMP protocol for measured launch.

The API is made using Python Flask framework for REST APIs (second most used Python framework for REST APIs and compatible with OpenAPI Generator). It needs to be run as root in order to execute libvirt commands.

The API structure is as follows:

rest_server/

├── app.py (main file containing the API code)

├── certificates.json (file containing certificates contents for each VM, required for measured launch)

├── flask_api.yaml (file to generate API structure using OpenAPI Generator)

├── users.py (file defining guest owner's usernames, password and assigned VMs names)

├──... files generated by OpenAPI Generator (not necessary)

Authentication

The API uses the HTTPBasicAuth library from Flask to authenticate users. Users' id, password and their assigned VMs names are defined in the "users.py" file. If the user doesn't have the right IDs or the VM is not attributed to them when attempting to access to an endpoint, the server will send an error 401 Unauthorized.

Generate API structure from YAML

Base structure of the REST API can be generated from the "flask_api.yaml" file using OpenAPI Generator. Endpoints and authentication are defined in the yaml file.

```
openapi-generator-cli generate -i home/cp/rest_api/flask_api.yaml -g python-flask -o rest_server
```

/!\ This only generates the API base structure, not the code for each endpoint

Funded by the Horizon Europe Framework Programme of the European Union

Launch API

```
cd /home/cp/rest-server/

flask run
```

/!\ It is recommended to use a python virtual env to use Flask (`source /home/cp/env/bin/activate`) + and start the API as root.

Endpoints specifications

Every endpoint can be accessed with HTTP requests following this format:

- POST method
- "Authorization" header with user:password (can be done with the --user option of curl command)
- "Content-Type" header : 'application/json'
- VM name in request's body as json : {"vm_name": "<vm_name>"}

Template using `curl` command:

```
curl -X POST <host>/vm/status --user "<username>:<password>" -H 'Content-Type: application/json' -d "{"vm_name": <vm_name>"}"
```

Available Endpoints:

- `/vm/start:` start VM in paused mode (uses `virsh start` command).
- `/vm/shutdown:` shutdown VM (uses `virsh shutdown` command).
- `/vm/destroy:` destroy VM (uses `virsh destroy` command).
- `/vm/status:` return VM status as `{"<vm_name>": "status" }` (uses `virsh dominfo` command).
- `/vm/certificates:` print given certificates content in the "certificate.json" file.

Obtaining and using the Docker Image for VM Operations

The Docker image is available from our download server as a TAR file "remote-management.tar". It can be imported into a local Docker or Podman installation like this:

```
docker load –input remote-management.tar
```

Funded by the Horizon Europe Framework Programme of the European Union

Guest owners can use the provided Docker image for remote management to manage their VM. The Docker container runs on CentOS 9 and has all the required software to run the personalized commands (described below). Most commands require username and password of the guest to authenticate to API.

Necessary files (password file for API authentication, OVMF file...) for measured launch procedure are contained in the directory provided by DT. Please unpack the remote-management-workdir.tar in an appropriate location. The resulting workdir directory needs to be mounted to the Docker container as "/workdir" volume (using: `docker -v local_dir:/workdir …`).

The following commands are available on the container (in the "`/usr/local/bin`" directory):

```
create-launch-bundle <launch-bundle-dir>
```

This command is used to create a launch bundle containing the required files for measured start of the encrypted VM. Requires the presence of the pdh.cert file provided by the remote host/cloud provider. It works as follow:

- Arguments:
  - `<launch-bundle-dir>`: the path to the directory where the files will be generated (directory will be created if not existing).
- Calls the `sevtool –generate-launch-blob` which "generate launch_blob.bin" and "godh.cert" certificates in the launch-bundle-dir. The files are then converted to "session.b64" and "godh.b64" files.

```
start <host> <vmname> <userid> <passwordfile>
```

This command is used to start the specified VM in paused mode (state required for measured launch). It works as follow:

- Arguments:
  - `<host>`: the address of the host.
  - `<vmname>:` the name of the VM to start in paused mode.
  - `<userid>:` the username used to connect to DT API.
  - `<passwordfile>:` the path to the passwordfile used to connect to DT API.
- Sends a call to API's endpoint "/vm/start" with VM name in the request's body.

```
launch <passwordfile> <create-launch-bundle directory> <qmp-socket>
```

This command is used to do a measured launch of the specified VM. It requires that the VM has been previously started in paused mode, and certificates were sent to the provider. It works as follow:

- Arguments:
    - `<passwordfile>`: A file containing the high-entropy password of the encrypted VM, as a String.
    - `<create-launch-bundle directory>`: the path to the launch-bundle-directory previously created.
    - `<qmp-socket>`: the qmp-socket port of the VM (given by DT).
- Does a measured launch of the VM through QMP protocol (does not call the REST API).

```
    send <host> <vmname> <userid> <passwordfile> <create-launch-bundle-directory>
```

This command is used to send the content of the documents necessary to do a measured launch to the host (godh.b24 and session.b64 file). It works as follow:

- Arguments:
    - `<host>`: the address of the host.
    - `<vmname>`: the name of the VM to start in paused mode.
    - `<userid>`: the username used to connect to DT API.
    - `<passwordfile>`: the path to the passwordfile used to connect to DT API.
- Sends a call to API's endpoint /vm/certificates with VM name and files' content in the request's body.

```
destroy <host> <vmname> <userid> <passwordfile>
```

This command is used to destroy the specified VM (forceful shutdown). It works as follow:

- Arguments:
    - `<host>`: the address of the host.
    - `<vmname>`: the name of the VM to start in paused mode.
    - `<userid>`: the username used to connect to DT API.
    - `<passwordfile>`: the path to the passwordfile used to connect to DT API.
- Sends a call to API's endpoint `/vm/destroy` with VM name in the request's body.

```
shutdown <host> <vmname> <userid> <passwordfile>
```

This command is used to shut down the specified VM. It works as follow:

Funded by the Horizon Europe Framework Programme of the European Union

- Arguments:
  - `<host>`: the address of the host.
  - `<vmname>:` the name of the VM to start in paused mode.
  - `<userid>:` the username used to connect to DT API.
  - `<passwordfile>:` the path to the passwordfile used to connect to DT API.
- Sends a call to API’s endpoint /vm/destroy with VM name in the request’s body.

```
status <host> <vmname> <userid> <passwordfile>
```

This command is used to retrieve the current status of the specified VM. It works as follow:

- Arguments:
  - `<host>`: the address of the host.
  - `<vmname>:` the name of the VM to retrieve status from.
  - `<userid>:` the username used to connect to DT API.
  - `<passwordfile>:` the path to the password file used to connect to DT API.
- Sends a call to API’s endpoint /vm/status with VM name in the request’s body.

Funded by the Horizon Europe Framework Programme of the European Union

APPENDIX B: SUBJECTIVE TEST PLATFORM

B.1 STEP-MR PLATFORM INSTALLATION

The platform can be found at: https://github.com/shivivats/MR-Subjective-Testing-Platform.

The initial (older) version can be found at: https://github.com/shivivats/MR-Subjective-Testing-Platform/tree/old-version.

The following subsections detail how to utilize both versions of the platforms, focusing primarily on the newer version, also known as STEP-MR, and noting where there are differences in the procedure when utilizing the older version.

B.1.1 System Requirements

This platform was tested on a laptop with an i7-12700H, 32 GB of DDR5-4800 MHz memory and an NVIDIA RTX A2000 8GB GPU. This is a mid-to-high-end system (as of mid-2025), but the single most important requirement here is the memory size: at least 32 GB of main memory is recommended. Having a GPU with ≥ 8 GB of memory will also be beneficial.

B.1.2 Software Pre-requisites

The project has been tested with Unity version 2021.3.19f1. Please only use the same, as using newer versions can introduce bugs.

The project uses MRTK2 (ver. 2.8.3) to work with the HoloLens 2. Check the tools needed to use MRTK 2 here[26] and install them.

B.1.3 Preparing the Data

The point clouds PLY files need to be in binary little-endian format. This platform uses "Pcx - Point Cloud Importer/Renderer for Unity"[27], and it only works with that format.

If using the old platform version with meshes, the meshes need to be generated offline. Ensure the meshes have the correct materials associated with them in Unity! The platform was developed and tested with OBJ files and works well with them.

One PLY or OBJ file per frame is utilised to animate the object on the screen. Thus, the point cloud folders detailed in the following subsection should contain one file per intended point cloud frame.

[26] https://learn.microsoft.com/en-us/windows/mixed-reality/develop/install-the-tools?tabs=unity. Accessed 02 November 2023.

[27] https://github.com/keijiro/Pcx. Accessed 02 November 2023.

Funded by the Horizon Europe Framework Programme of the European Union

B.1.4 Setting Up the Project

1. Download this repository and extract it.
2. Place the prepared point cloud files in the `Assets\Resources\PointClouds\<Name_of_point_cloud>\<Name_of_Codec>\<Name_of_quality>\PointClouds`. If using the old platform, the codec folder is omitted.
3. If using the old platform with meshes, place the prepared mesh files in the `Assets\Resources\PointClouds\<Name_of_point_cloud>\<Name_of_quality>\Mesh`.
4. Select to `PointClouds->Update Point Clouds From Assets` in the menu bar to let the project configure itself using the added objects.
5. MRTK should already be set up correctly but double-check the XR settings according to the MRTK documentation.
6. Enable/disable Holographic Remoting[28] as you desire.

B.1.5 Point Clouds Loader

For all functionalities of these platforms, there is a `Manager` object in their respective scene. It contains a `Point Clouds Loader (Script)` component. It is responsible for loading the point clouds/meshes into memory. Make new ScriptableObjects with the data of the point clouds and add these to the `Pc Objects` array in this component. Only the objects and qualities mentioned here will be loaded into memory and are the only ones that can be used/interacted with.

The `Load Meshes` toggle in the old version controls whether the mesh files for the specified objects will be loaded.

Other values in this script should not be changed.

B.2 PLATFORM USAGE

B.2.1 Point Clouds Preview

The Point Cloud Preview scene can be found in `Assets\_ConfigurationScene\ConfigurationScene.unity`.

1. Configure the `Point Clouds Loader` as desired.
2. The distance slider min and max values can be changed by updating the associated values in the `Distance Changer (Script)` component.
3. The maximum number of allowed point clouds can be changed in the `Configuration Scene Manager` component. It is recommended to keep this value less than 4 (default is 2).
4. Run the scene.

[28] https://learn.microsoft.com/en-gb/windows/mixed-reality/mrtk-unity/mrtk2/features/tools/holographic-remoting?view=mrtkunity-2022-05. Accessed 02 November 2023.

Funded by the Horizon Europe Framework Programme of the European Union

The user can see and control the objects appearing on the screen. Up to 4 objects can be displayed and configured individually. The animation and interaction can be toggled.

B.2.2 Subjective Testing

The Subjective Testing scene can be found in `Assets\_SubjectiveTesting\SubjectiveTestingScene.unity`.

Configure the `Point Clouds Loader` as desired.

B.2.2.1 Configuring the Test

The `SubjectiveTest` object in the scene has an `ST Manager (Script)` component. It is used to configure the tasks.

Open the `Tasks` array in the component and add task elements. Select the desired point clouds, and the quality representations (codec + quality) for each task. If using the old version, select the desired point clouds, representations, distances and qualities for each task.

The `Randomise Tasks` toggle controls whether the tasks are randomised. The sequences within the tasks are always randomised.

The `Use Fixed Y Offset For Distance` will add a height offset equal to `Y Offset` (in meters) to all objects displayed. You can use this to make your objects appear as if they are standing on the ground.

B.2.2.2 Running the Test

A `Start` button will be displayed before the start of each task. After every sequence, the test participant will be asked to give feedback between 1 and 5 using a set of buttons, or between 1 and 10 using an immersive slider if using the old version. The feedback is stored in the `Assets\CSV\ratings.csv` file.

B.2.3 Eye Tracking Testing

The Eye Tracking Testing scene can be found in `Assets\ _EyeTracking\ _EyeTrackingScene.unity`.

The point cloud configuration is the same as described in the previous subsection and should be done in the components of the respective `Manager` object.

B.2.3.1 Configuring the Test

The duration of the error markers shown to the participant at the beginning of each object's playback is also configurable via the `Error Profiling Manager` component.

B.2.3.2 Running the Test

A `Start` button will be displayed before the playback of each point cloud sequence. First, the error markers will be shown and the participant should look at them. Using another button, the participant can start the point cloud sequence playback. After the sequence playback, the participant will be asked to go back to the initial position and start again with the error markers.

Funded by the Horizon Europe Framework Programme of the European Union

The participant's error data will be stored as JSON files in the `ErrorCSV` folder, and the participants gaze data will be stored as JSON files in the `JSON` folder.

B.2.3.3 Generating Fixation Data

The functionality for generating the fixation data is present in the `Assets\ _EyeTracking\ _EyeProcessScene.unity.`

The `Gaze Data Error Read` component and the `Eye Tracking Heatmap Process` component will read the error and the gaze data from the configured folders to identify fixations. The parameters for the I-DT algorithm[29] used for identifying fixations can also be configured.

MATLAB scripts present in the `matlab` directory at root of the repository further process the data using the DBSCAN[30] algorithm and generate fixation maps in the form of weights for each point per user, and summed up across all users.

B.2.3.4Generating Heatmap Images

The `Heatmap Visualizer` component in the `_EyeProcessScene.unity` contains functionality to view and save screenshots of the heatmaps using the data generated in the previous step.

[29] https://dl.acm.org/doi/abs/10.1145/355017.355028

[30] https://en.wikipedia.org/wiki/DBSCAN

Funded by the Horizon Europe Framework Programme of the European Union