IGMR Institute of Mechanism Theory, Machine Dynamics and Robotics

RWTH AACHEN UNIVERSITY

This thesis was submitted to the Institute of Mechanism Theory, Machine Dynamics and Robotics

Conceptualization, Development and User Validation of a Mixed Reality User Interface for the Teleoperation of Robots

Master's Thesis

by:

Lukas Johannes Windstetter, B.Sc.

Student number: 432 549

supervised by:

Elodie Hüsing, M.Sc. (RWTH)
Luisa Mayershofer, M.Sc. (DLR)

Examiner:

Univ.-Prof. Dr.-Ing. Dr. h. c. Burkhard Corves

Prof. Dr.-Ing. Mathias Hüsing

Aachen, 5 February 2026

Master's Thesis

by Lukas Johannes Windstetter, B.Sc.
Student number: 432 549

Conceptualization, Development and User Validation of a Mixed Reality User Interface for the Teleoperation of Robots

Im Rahmen des DLR-Projekts Surface Avatar wird ein System zur Teleoperation von Robotern entwickelt, das die Teleoperation eines gemischten Roboterteams vom Erdorbit zur Erdoberfläche ermöglicht. Die bisherige Mensch-Roboter-Schnittstelle basiert auf einer zweidimensionalen grafischen Benutzeroberfläche (2D-GUI), über die ein Operator einzelne Roboter auswählen kann, um ihnen automatisierte Aufgaben zu erteilen oder sie direkt zu steuern. Diese 2D-GUI besteht aus mehreren Fensterbausteinen, die jeweils einen Teil der benötigten Informationen liefern, sodass der Operator ein vollständiges Verständnis des Roboters und seiner Umgebung aufbauen kann. Die Hauptinformationsquellen sind dabei ein Kamerastream mit Augmented-Reality-Overlay bekannter Objekte sowie eine 3D-Darstellung der Roboterkonfiguration (Gelenkzustände) anhand eines virtuellen Modells. Als Eingabegeräte für die direkte Teleoperation humanoider Greifarme kommen ein Joystick sowie ein Force-Feedback-Gerät (Sigma.7) zum Einsatz, während die 2D-GUI dem Operator visuelles Feedback liefert.

In mehreren Testläufen wurde unter anderem der humanoide Roboter Rollin' Justin von der Internationalen Raumstation (ISS) aus auf der Erde teleoperiert, um verschiedene Interaktionsaufgaben durchzuführen. Die Tests zeigten, dass es dem Operator in mehreren Fällen schwerfällt, die dreidimensionale Umgebung des Roboters korrekt einzuschätzen. Besonders bei hochpräzisen Aufgaben wie Greifvorgängen spielt die Tiefenwahrnehmung eine entscheidende Rolle, die jedoch nur unzureichend über einen zweidimensionalen Farbbildstream vermittelt werden kann. Es hat sich gezeigt, dass es erheblichen Trainingsaufwand erfordert, aus den einzelnen Datenströmen ein korrektes mentales Gesamtbild zu rekonstruieren. Dadurch bleibt die Wahrscheinlichkeit von Fehlverhalten oder sicherheitsrelevanten Vorfällen infolge fehlerhafter Tiefeneinschätzung sowie die Effizienz der Aufgabenbewältigung ein Problem.

Die Motivation der vorliegenden Arbeit ist daher, die bestehende Mensch-Roboter-Schnittstelle um ein Virtual-Reality-Interface (VR) zu erweitern. Basierend auf einer Literaturrecherche relevanter wissenschaftlicher Arbeiten zeigt sich, dass VR das Potenzial besitzt, die Tiefenwahrnehmung und damit die Situational Awareness zu verbessern. Ziel dieser Arbeit ist die Entwicklung und Evaluation des Prototyps einer neuen Mensch-Roboter-Schnittstelle in Mixed Reality (XR), die eine intuitivere Interaktion und Steuerung von Robotern ermöglichen soll. Die immersive Darstellung in XR soll die Tiefenwahrnehmung verbessern, die Effizienz bei der Ausführung von Aufgaben steigern und den Trainingsaufwand für

Operatoren reduzieren.
Das Vorgehen umfasst die Entwicklung einer Anwendung mit der Game-Engine Unity für die XR-Brille Meta Quest 3. Dazu wird eine Datenstream-Pipeline zur Brille entwickelt sowie die Verarbeitung und Visualisierung dieser Daten in einer grafischen Benutzeroberfläche (GUI) umgesetzt. Die nötigen Datenströme des Roboters werden hierzu zunächst auf Basis eines virtuellen Modells generiert. Anschließend validiert eine Nutzerstudie die entwickelte XR-Schnittstelle hinsichtlich Tiefenwahrnehmung, Aufgabeneffizienz und Usability im Vergleich zu einer äquivalenten 2D-GUI.

Supervisor: Elodie Hüsing, M.Sc. (RWTH)
Luisa Mayershofer, M.Sc. (DLR)

Acknowledgements

I would like to express my sincere gratitude to my external supervisor, Luisa Mayershofer, from the German Aerospace Center (DLR), for her continuous support, valuable guidance, and constructive feedback throughout this thesis.

I also thank my internal supervisor, Elodie Hüsing, at RWTH Aachen University for her valuable support, insightful comments, and academic guidance during the course of this work.

Special thanks go to Ajithkumar Manaparampil for the hands-on support with the system integration to the robotic platform, which was essential for the successful realization of this thesis.

Furthermore, I would like to thank all participants who took part in the user study for their time and valuable contributions.

Contents

Formula symbols and indices

Lower case latin letters as formula symbols

d	planar target offset
d_z	Cohen's effect size for paired samples
e	Euclidean target error
h	release height
i	participant index
n_g	number of grasp attempts
n	number of paired observations
p	probability value
$\mathbf{p}$	position vector
t	test statistic of the paired-samples t-test
$t_{\textbf{end}}$	final timestamp of a run
w_e	weighting factor for release height
s_{TE}	task efficiency scaling factor

Upper case latin letters as formula symbols

A	target accuracy
A_i	score of condition A for participant i
B_i	score of condition B for participant i
C	cost term
G	grasp effectiveness score
L	end-effector path length

O	obstacle avoidance indicator
S	success term
T	task duration
TE	task efficiency

Lower case greek letters as formula symbols

α	alpha transparency value
α	significance level

Upper case greek letters as formula symbols

Δ	within-subject difference operator

List of abbreviations

2D two-dimensional

3D three-dimensional

API application programming interface

AR augmented reality

CPU central processing unit

CSV comma-separated values

DLR German Aerospace Center

DPQ depth perception questionnaire

DV dependent variable

ESA European Space Agency

EVA extra-vehicular activity

GO game object

GPU graphics processing unit

GUI Graphical User Interface

HCI Human-Computer Interaction

HMD head-mounted display

HRI Human-Robot Interaction

ID identification

ISS International Space Station

IV independent variable

LN links and nodes

LWR lightweight robotic arm

MR mixed reality

NASA National Aeronautics and Space Administration

PC personal computer

PC-VE point cloud visualization element

RGBA red-green-blue and alpha

RGB red-green-blue

RGB-D red-green-blue and depth

RM-VE robot model visualization element

ROS robot operating system

SDK software development kit

SD standard deviation

SEQ system evaluation questionnaire

SO scriptable object

SUS system usability scale

TE task efficiency

TLX task load index

ToF Time-of-Flight

UI User Interface

URDF unified robot description format

USB universal serial bus

VE visualization element

VFX visual effects

VID-VE video screen visualization element

VR virtual reality

XR extended reality

1. Introduction

Robotic systems enable the access to locations that are remote, dangerous or inaccessible to the human body. Teleoperation is one of the key technologies used to accomplish tasks remotely by human operators, which allows them to be physically separate from the environment at hand. Although autonomous robotics are capable of achieving many different tasks on their own, human operator approaches remain crucial for reasons like higher adaptability through more general intelligence. This is why many space-related activities use telerobotic systems, keeping a human in the loop even in autonomous scenarios, such as exploration activities.

The Surface Avatar teleoperated technology demonstration is developed and tested at the German Aerospace Center (DLR) in Oberpfaffenhofen. It aims to explore multi-modal telecommand of a heterogeneous robotic team on the ground operated from the International Space Station (ISS). A graphical user interface (GUI) was developed with a scalable autonomy approach, so operators can select from multiple control modes depending on the task, including direct teleoperation and supervised autonomy. Within the most recent orbit-to-ground experiment in 2025, major feedback from astronaut Jonny Kim was the lack of depth information in both control modes, especially in direct teleoperation. This showed that depth perception is crucial for effective teleoperation. [MKM26, p. 7]

This finding motivates the extension of the UI from Surface Avatar to improve depth perception and spatial understanding for the operator. While [MKM26] focuses on improvements during knowledge-driven supervised autonomy control, this work aims to enhance the user interface (UI) regarding depth perception during direct teleoperation. It is hypothesised that immersive technology combined with depth sensor data reconstruction can improve task performance during direct teleoperation. The objective of this thesis is to develop and implement a novice Human-Robot Interface (HRI) with improved depth feedback that combines mixed reality technology with the visual representation of depth data as a point cloud in addition to already proven interfaces like 3D virtual robot configuration and a video-based interface. The effects of depth perception during an unknown object manipulation task are then tested within a user study.

The structure of this work is as follows: Chapter 2 provides the necessary background from general robotic systems for teleoperation to Surface Avatar. Chapter 3 introduces the new system concept and methodology for improving the Surface Avatar system, followed by Chapter 4 demonstrating the implementation. Chapter 5 describes the conducted user study for hypotheses validation, after which Chapter 6 shows the study results. Finally

Chapter 7 discusses the findings, limitations and implications for future work, before concluding this work with Chapter 8.

2. Background and State of the Art

This chapter provides an overview of the state of the art in human-robot interaction (HRI) in teleoperation robotics and serves as a technical foundation for later chapters. It starts with general teleoperation definitions and technologies, then focuses on space applications and finally describes specific projects within the field. This chapter concludes with an overview of the Surface Avatar project and how this work is positioned within it.

2.1. Teleoperation Fundamentals

Telerobotics, commonly used synonymously for teleoperation, is one of the earliest application domains of robotics. It defines robotic systems with a human operator, the so-called human-in-the-loop concept. The robotic system is primarily responsible for execution of commands, while a human operator is responsible high-level control decisions. The word *Tele* originates from the Greek meaning *far*, referring to the distance between the remote robot and the controlling human operator. Figure 2.1 shows the system spanning from operator site to remote site. The data flow is visualized by the brown arrow loop, connecting operator input to the remote robot and receiving feedback to be displayed to the operator. [SK16, pp. 1085–1087]

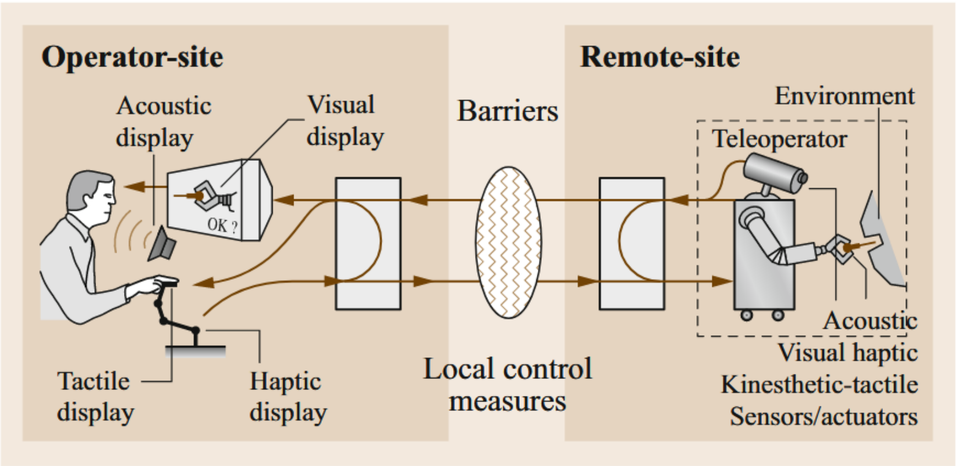


Figure 2.1.: Telerobotic system overview showing the separation between operator-site with the UI system (left), and the remote-site with the robotic platform (right) [SK16, p. 1086]

Control Architectures

Control architectures for telerobotics can be viewed as a spectrum ranging from so-called direct or manual control to supervisory control. In direct control, the user operates the robot explicitly through input commands without automated support, whereas in supervisory control only abstract, high-level commands are issued and executed autonomously by the robot. Supervisory control architectures therefore require the robot to have much more sophisticated autonomy capabilities. A mix between the two extremes is called shared control and supports direct inputs by some level of autonomy. An overview of the described telerobotic control architectures can be seen in Figure 2.2. [SK16, pp. 1090–1093]

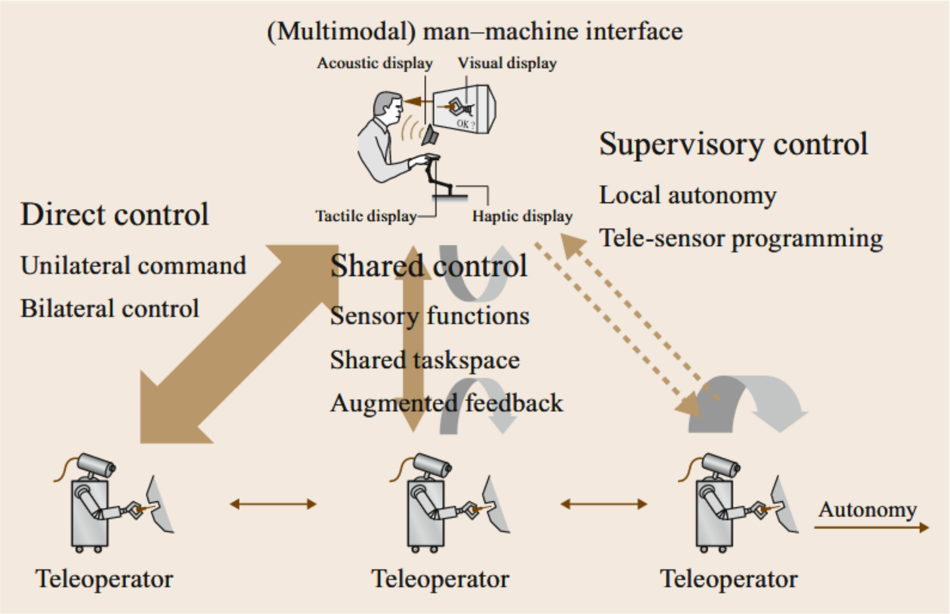


Figure 2.2.: Telerobotic control architectures showing the levels of autonomy from direct control (no autonomy) over shared control (limited autonomy) to supervisory control (local autonomy) [SK16, p. 1091]

Interaction Concepts

A common human-in-the-loop teleoperation approach is the master–slave paradigm, in which a human-operated master device directly controls a remote slave robot that reproduces the commanded motion. More advanced bilateral control schemes extend this concept by incorporating feedback from the slave to the master, enabling the operator to perceive interaction forces and improving situational awareness. [SK16, pp. 1091–1100]

An alternative interaction concept is based on specifying the desired position and orientation of the robot's end-effector rather than directly controlling joint motions. In such approaches, inverse kinematics is used to determine suitable joint configurations, allowing intuitive control using simplified input devices for translational and rotational commands. [SK16, pp. 29–33, 905–911]

2.2. Visual and Depth Sensing Technologies for Teleoperation

In teleoperation scenarios, sensor-based perception is particularly critical, as the human operator relies on sensor data to understand the remote environment and to effectively control the robot. Unlike autonomous systems, teleoperation requires sensor information to be presented in a form that supports human perception and decision-making. Consequently, sensor technologies play a central role in enabling situational awareness and effective human-in-the-loop operation. The following sections introduce and classify the visual and depth sensing technologies relevant to this work.

2.2.1. Visual Sensors (RGB Cameras)

One of the most used sensors for capturing visual information is red–green–blue (RGB) cameras. Digital color images typically use the RGB format, representing the three primary color channels red, green and blue. RGB cameras capture color images in a defined resolution on a discrete grid of pixels, where every pixel contains the corresponding red, green and blue intensities resembling its color when mixed together. As passive sensors they rely on ambient illumination, meaning no signal is actively emitted by the camera to capture the environment. In teleoperation systems, RGB cameras are commonly used to provide a continuous video stream of a remote environment to the human operator. This form of visual feedback is intuitive and easy to interpret, as it closely resembles natural human vision. This results in fast perception of the remote scene to form a solid foundation for situational awareness in many teleoperation systems. [Sze22, Ch. 2]

Alternatively, a stereo camera setup arranged in a human eye–like configuration can be used to provide stereoscopic visual feedback by presenting separate images to each eye, thereby creating depth perception through binocular vision. While approaches like this can enhance spatial perception compared to single video, depth perception remains dependent on human visual interpretation and is constrained by the fixed viewpoint of the stereo camera setup. Consequently, the operator's perception of spatial structure is limited to the perspective provided by the camera configuration. [CLY24]

Purely 2D color-based feedback does not provide explicitly measured depth information, making it difficult for operators to accurately estimate distances and understand spatial composition. These limitations can negatively affect task performance, particularly in manipulation tasks, and motivate the use of depth sensing technologies to complement a color-based video stream.

2.2.2. Depth Sensors

To address the limitations of purely image-based visual feedback, various depth sensing technologies are commonly applied in teleoperation systems. These technologies differ in how depth information is acquired and represented.

Depth sensing approaches can be broadly categorized based on their acquisition principles. Active sensing methods, like Time-of-Flight (ToF) depth cameras, emit signals into the environment and directly measure distances from the returned reflections [FAT11]. On the other hand, passive approaches like stereo vision rely on ambient light to estimate depth indirectly from visual information [SK16, pp. 789–793].

Consequently, depth information can either be provided as explicit metric distance values or derived implicitly from visual cues like perspective, shading and stereo disparity. While implicit depth cues can be effectively interpreted by human operators, computer-based perception and control work more reliably on explicit depth representations to ensure spatial understanding in teleoperation systems.

Stereo Vision

Stereo vision systems consist of two or more spatially separated RGB cameras that capture the same scene from different viewpoints. By comparing the disparity between corresponding image points, stereo vision allows the reconstruction of three-dimensional (3D) scene geometry using stereo matching algorithms. Compared to monocular RGB cameras, this approach provides additional spatial information without the need for active illumination. [SK16, pp. 789–793]

Depth estimates derived from stereo vision rely exclusively on implicit depth cues and do not provide direct ground-truth depth measurements. The quality of depth estimates obtained from stereo vision strongly depends on the ability to establish reliable correspondences between the input images. Therefore, performance is sensitive to scene texture, lighting conditions and occlusions. In addition, stereo-based depth accuracy decreases with increasing distance and is strongly influenced by image resolution. Furthermore, stereo matching algorithms can be computationally demanding, which creates challenges for real-time operation in robotic teleoperation systems. [SK16, pp. 789–793][GG25]

Structured Light Sensors

Structured light sensors estimate depth by projecting a known light pattern into the scene and analyzing the observed deformation through a camera. From the distortion of the pattern, depth information is reconstructed through geometric triangulation. Structured light enables accurate depth measurements at close distances with dense depth maps and high spatial resolution. However, its performance is sensitive to ambient light and reflective surfaces, which limits its use for outdoor or well-lit environments. This makes it unsuitable for common teleoperation scenarios. [SFP10]

Time-of-Flight (ToF) Sensors

Time-of-Flight (ToF) sensors calculate depth by measuring the travel time of emitted infrared light reflected from the scene. This allows for active distance measurement per pixel across the sensor's field of view. Many ToF cameras emit modulated light and estimate distance from the phase shift of the reflected signal. Dense depth maps at high frame rates can be captured that way, making ToF sensors suitable for real-time robotic perception. However, depth accuracy is limited by multiple factors such as sensor noise, interference effects and hardware-related constraints, which restrict measurement precision as well as usable range. [FAT11]

RGB-D Cameras

RGB-D cameras combine a conventional RGB camera with a depth sensor, most commonly based on ToF technology, within a single device. By providing synchronized color images and per-pixel depth measurements, RGB-D sensors enable the joint representation of visual appearance and spatial structure. Depth maps, colored point clouds, and 3D scene reconstructions can be generated from this data and are well suited for visualization and interaction tasks. [TTP22; RLS19]

Due to their compact design and integrated sensing capabilities, RGB-D cameras offer a practical solution for teleoperation systems, where camera-based visual feedback is already an essential component. Having explicit depth information in addition to RGB images improves the operator's understanding of the spatial structure of the remote environment and enables more intuitive user interfaces for human–robot interaction. [VQG]

2.3. User Interfaces for Teleoperation

Research on field and disaster robotics shows that a substantial portion of robot failures happen due to human interaction and operator errors, rather than purely technical faults. This highlights the importance of HRI [CM05]. HRI is multidimensional and addresses topics like ergonomics, system design, situational awareness and training. Unlike Human-Computer Interaction (HCI) used with only virtual environments, HRI mediates an active relationship between a physical robot and a human operator. This implies that incorrect inputs can lead to irreversible consequences with potentially catastrophic outcomes. [Tad19, pp. 507–510]

This highlights the role of user interfaces (UIs) as a key component in supporting situational awareness. This includes graphical user interfaces (GUIs), input devices, interaction logic and feedback mechanisms [SK16, p. 1596]. Since GUIs and three-dimensional visual feedback are central to this study, common GUI concepts within teleoperation are presented next.

Video-Based Visualization Elements

Video-based visual feedback is one of the most widely used visualization concepts in teleoperation systems. A continuous color video stream provides an intuitive two-dimensional representation of the remote scene, allowing the human operator to directly observe the environment from the robot's perspective. As human situational awareness is largely driven by perceptual processes, with visual perception playing a central role [End95], video-based GUIs convey a rich set of information and therefore form the foundation of feedback in many teleoperation systems. [WSC25; SK16]

Depth-Based Visualization Elements

Depth-based visualizations represent spatial information by projecting measured depth data into a virtual 3D space. One straightforward representation is the point cloud, which visualizes the environment as a set of points, where depth is shown through each points 3D position in space. Point clouds provide a direct deterministic representation of sensor depth data and are commonly used for interaction tasks in robotics due to their reliability, simplicity and flexibility. [RC11; SK16]

Alternatively, mesh-based representations combine neighboring points into continuous surfaces forming connected triangles. Compared to point clouds, meshes provide smoother and more visually coherent representations of object geometry, which can improve spatial perception. However, generating the mesh requires additional processing and works less reliably when confronted with discontinuous surfaces. The fidelity of representation can therefore be reduced by low-resolution data and noise, leading to incorrect connections and confusing visual artifacts. [SCW16]

Model-Based Visualization Elements

Another important visualization modality in teleoperation systems is the model-based representation of the robot itself. In this approach, a virtual three-dimensional model of the robot is used to display the robot's current pose as defined by its joint configuration. The virtual robot acts as a digital twin by updating all joints according to telemetry data received from the remote system, thereby reflecting the current robot configuration in real time.

While model-based visualizations do not provide information about the surrounding environment, they provide a clear representation of the robots current configuration state. This supports monitoring the joint limits, preventing self-collisions and understanding robot orientation in teleoperation interfaces. [SK16, pp. 1085–1104]

Display Platforms

In addition to the visualization element itself, the display platform on which it is presented plays an important role in the UI. Desktop monitors are commonly used for both HCI and HRI. They allow for 2D GUI elements to be shown to the user on a screen surface and form the technical standard for traditional computer interaction. Advances in visualization technology have led to the development of immersive platforms like virtual reality (VR) interfaces. They are typically designed as head-mounted displays (HMDs) rendering one screen for each eye and thereby mimicking a 3D presentation of a virtual world. Newer platforms called mixed reality (MR) interfaces incorporate cameras on the outside of the HMD to capture the real environment around the user. This allows virtual elements to be shown as augmented overlays on top of the captured real surroundings. The virtuality continuum, a term popularized by Milgram and Kishino in 1994 [MK94], describes the spectrum between reality and virtuality and is illustrated in Figure 2.2. It shows the stepwise transition from reality (left) to virtuality (right) as one dimension to classify immersive technologies. A second dimension representing the degree of integration is added, ranging from virtual overlays on top of reality to full spatial integration connecting virtual and real environments in three dimensions. The terms virtual reality (VR), augmented reality (AR), mixed reality (MR) and extended reality (XR) are positioned accordingly to clearly distinguish between the approaches. [Wöl23, pp. 16–26]

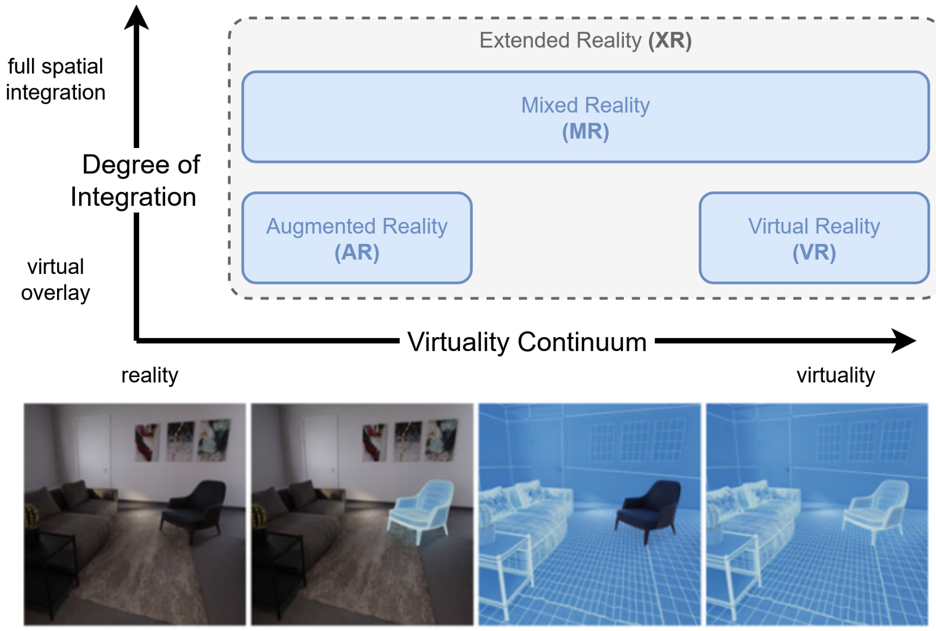


Figure 2.3.: Virtuality continuum by Milgram and Kishino (left reality, right virtuality) extended by a dimension for degree of spatial integration, adapted from [Wöl23, p. 16]

2.4. Teleoperation Use Cases

Teleoperation has been successfully applied in several demanding domains outside of space robotics, including medicine, the nuclear industry, and intervention in hazardous environments. These systems address challenges such as limited feedback, high precision requirements, and operator–system interaction under constrained conditions. Teleoperation projects from related domains serve as useful reference points for visualization and interaction principles that are directly relevant to space teleoperation.

Medical Teleoperation: Da Vinci Surgical System

The da Vinci Surgical System is a widely used teleoperation system used in medical robotics. It enables surgeons to perform minimally invasive procedures by remotely controlling robotic instruments with high precision through a master–slave interface. The system provides stereoscopic visual feedback and motion scaling to enhance accuracy while filtering hand tremors. [GSG23; DaV]

Key aspects of the da Vinci system are transferable to space teleoperation, including the physical separation of operator and robot, precise manipulation under constrained visual feedback, and the use of intuitive interfaces to support reliable execution of safety-critical tasks.

Teleoperation in Nuclear Decommissioning

Teleoperation systems are often used in nuclear facilities for maintenance, inspection, and decommissioning tasks in hazardous environments where direct human presence poses a significant safety risk. Remote handling systems enable the manipulation of tools and components under limited visibility and without direct physical access. [KLY25]

The challenges encountered in nuclear teleoperation closely resemble those in space robotics, including restricted sensory feedback, safety-critical operation, and reliable manipulation in complex and only partially observable environments. Transferable lessons include the design of fault-tolerant control architectures, the use of model-based and sensor-based visual feedback, and approaches for operating robustly under uncertainty. [KLY25]

2.5. Teleoperation in Space Applications

Robotic systems play a crucial role in space applications, as the extreme environmental conditions and remoteness of orbital and planetary missions severely limit direct human involvement. Teleoperated robots are therefore widely used for exploration, assembly, servicing, and maintenance tasks in environments that are inaccessible or hazardous to astronauts.

A defining constraint of space teleoperation is communication latency, which increases with distance and can significantly impact real-time human–robot interaction. Depending on the mission scenario, this necessitates either a high degree of onboard autonomy with the operator issuing only high-level commands, or positioning the human operator sufficiently close to the robot to allow near real-time control. Additional challenges include strict safety requirements, limited sensory feedback, limited bandwidth, and the impossibility of physical intervention in case of failure. Table 2.1 provides an overview of established space teleoperation systems that demonstrate how these constraints have been addressed in operational missions. [YWH16]

More recent developments continue to explore human-in-the-loop teleoperation for space-relevant manipulation tasks, particularly using humanoid robotic platforms and advanced

Table 2.1.: Key milestones in space teleoperation [SK16, pp. 1424–1437]

Period	System / Mission	Teleoperation characteristics
1970–1973	Lunokhod 1 & 2 (Moon)	First operational space teleoperation, direct ground-based driving, multi-second communication latency
1981–2011	Orbital manipulator systems	Long-term operational teleoperation, direct human-in-the-loop manipulation in Earth orbit
1993	ROTEX (Spacelab D2)	Ground-to-orbit teleoperation demonstrator, predictive visual feedback for delay compensation
1997–1999	ETS-VII (Japan)	Supervised teleoperation, rendezvous, capture and manipulation of a free-flying target
2011–today	Humanoid teleoperation (ISS)	Dexterous teleoperation, EVA-compatible tools, direct operator supervision
2020–today	Humanoid teleoperation research platforms	Ground-based teleoperation experiments, space-relevant humanoid manipulation, telepresence-focused control [LSL22; RSH15]

telepresence concepts. The following sections highlight representative projects in this context.

2.5.1. Valkyrie (NASA)

The NASA Valkyrie humanoid robot was developed as a high-dexterity research platform to study human-in-the-loop control of humanoid robots for space-relevant tasks. From early stages, the project focused on teleoperation and supervised control, allowing human operators to perform locomotion and manipulation in unstructured and potentially hazardous environments, with research progressing from mechanical design toward whole-body control, perception, and increasingly intuitive human–robot interfaces. [RSH15]

More recent work introduced immersive mixed reality (MR) interfaces based on virtual reality environments to provide intuitive visual feedback and control for locomotion and manipulation. These interfaces incorporate 3D scene representations, including point cloud visualizations, to convey spatial structure and robot state. VR-based cockpit interfaces enable operators to interact with Valkyrie using natural movements and perspectives, improving situational awareness and telepresence compared to conventional monitor-based interfaces. [JWP22]

This progression from classical teleoperation toward immersive MR-based control highlights a key trend in humanoid teleoperation research for space applications. Valkyrie is of particular relevance as it demonstrates how MR interfaces can extend traditional control concepts toward more intuitive human–robot interaction. Similar MR-based approaches are therefore being investigated for space-relevant humanoid systems such as the DLR Surface Avatar.

2.5.2. Surface Avatar (DLR)

For long-term human space exploration missions to celestial bodies, safe landing and reliable infrastructure on planetary surfaces must be established in advance. This requires a range of robotic systems for construction, scientific operations, and infrastructure maintenance prior to human presence. These challenges motivated a series of teleoperation experiments conducted by the DLR. [LSL22; SLK20; SBB23]

The Surface Avatar space telerobotics experiment explores human–robot interaction concepts for controlling teams of heterogeneous surface robots from an orbiting platform. Its objective is to assess the operational feasibility of such telerobotic systems for supporting future crewed missions to the Moon and Mars. [LSL22; SBB23]

Surface Avatar consists of multiple experiments carried out by the German Aerospace Center (DLR) in cooperation with the European Space Agency (ESA). The experiments aim to evaluate the use of heterogeneous robot teams for cooperative task execution on planetary surfaces, as well as to study multimodal user interfaces that enable human operators to teleoperate robots as intelligent partners with haptic coupling. [LSL22; SLK20; DLR26]

Knowledge Driven Approach

A knowledge-driven approach is used to implement action templates for high-level supervisory commands. Known objects as well as the robots themselves were integrated into the object-centered domain, creating an internal representation of a structured environment. [SBB23]

The following preliminary experiments were conducted to work towards this approach. The Multi-Purpose End-To-End Robotic Operation Network (METERON) was a collaborative space telerobotics project involving ESA, DLR, NASA, and Roscosmos. The project investigated key technologies and operational concepts for space telerobotics, with a particular focus on human-in-the-loop control, telepresence and supervision under realistic mission conditions. [SBB23]

A series of ISS-to-ground experiments were conducted to develop the knowledge-driven approach, as shown in Table 2.2. Preliminary experiments addressed specific aspects of this approach, focusing chronologically on force feedback input devices, semi-autonomy, task-level command with full autonomy, object detection, and open-loop teleoperation. [SBB23]

Table 2.2.: Overview of METERON experiments [SBB23])

Experiment	Focus
METERON HAPTICS	Telepresent command of ground robot, perception of force feedback, microgravity environment
METERON Interact	Semi-autonomous navigation, reduced mental effort by novel GUI
METERON SUPVIS-E & SUPVIS-M	Supervisory robot command, predefined task-level commands (autonomy scenarios), optimizing workload balance between astronaut and robot
METERON SUPVIS Justin	Supervised autonomy, humanoid robot as co-worker, structured environments, autonomy functions (object detection, context-specific action execution), untrained crew capabilities
METERON ANALOG-1	Open-loop teleoperation, haptic telemanipulation, novel Robot Command Terminal (Sigma.7 force feedback, joystick, GUI)

Scalable Autonomy

Combined findings from supervised autonomy (METERON SUPVIS Justin) and force-feedback telepresence (METERON ANALOG-1) inspired the extension of the knowledge-driven approach by scaling the autonomy level depending on the task. Each robot can switch between autonomous and direct operation modes, including discrete, open-loop and closed-loop force-reflection teleoperation. This scalable autonomy approach was tested in preliminary sessions of the Surface Avatar ISS-to-ground space telerobotics experiment. The UI system from ANALOG-1 was reused and extended using the knowledge-driven teleoperation concept. In addition, the GUI was augmented with a 3D rendering of the robot's current configuration. [SBB23]

Heterogenous Robotic Team

A central motivation for the Surface Avatar mission was to develop and investigate the technologies required for teleoperating a heterogeneous robotic team with scalable autonomy. The robots involved in collaborative operation during the latest ISS-to-ground experiments included Rollin' Justin (humanoid robot), Interact Rover (wheeled rover with gripper), Bert (compact quadruped scout robot), Spot (quadruped with gripper) and ELAFANT (lander arm system). [MKM26; LSL22]

Surface Avatar ISS-to-ground experiment sessions

During multiple ISS-to-ground sessions, different approaches and teleoperation modes have been studied, with astronauts on board the ISS teleoperating multiple robots located at the DLR site in Oberpfaffenhofen. Most relevant for this work are the phases with direct teleoperation, particularly tasks in unstructured environments, in which operators were required to manipulate unknown objects using a robotic arm. Visual feedback was primarily provided through a video stream and a robot model visualization. The following paragraphs highlight two of the direct control phases during those sessions in more detail. [MLB26; MKM26]

In the second prime session of Surface Avatar in July 2024, rock samples needed to be picked up from a handover station and placed into an analyzer on the ELAFANT lander. The task consisted of multiple phases with varying levels of autonomy, including a direct control phase. During this phase, the Interact rover was operated to search for rock samples scattered within the environment, grasp them using the attached robotic arm, and place them into the handover station for further manipulation. [MLB26]

In the most recent ISS session of Surface Avatar in July 2025, a similar setup was used, with an extended protocol. The humanoid robot Rollin' Justin collected sample containers from the handover station and placed them onto a platform attached to the ELAFANT lander. During the final part of the placement, the robotic arm was directly teleoperated, while previous phases included task-level commands. [MKM26]

Operator-UI

The Operator-UI is a multi-windowed GUI concept focusing on dynamic adaptability for the operator. The selection of visual elements and their position within the GUI can be changed depending on the necessity and preference. The core principle is to adapt the interface to the current autonomy level of the scalable autonomy approach. This allows a natural transition between overseeing the entire robotic team during high-level autonomous operation and focusing on an individual robot during direct teleoperation modes such as telepresence or supervision. [MLB26]

Figure 2.4 shows the Operator-UI's multi-window concept. The specific visual elements of the GUI include video streams, a robot model, communications, action templates of the selected robot, control mode indicators and a map view. The large central element shows the video stream from the currently selected robot (head-mounted camera of the humanoid Rollin' Justin). The top-right element shows a virtual model of the currently selected robot, displaying its joint configuration. Below this, in the bottom-right area, a list of currently available commands (supervisory action templates) is provided, allowing interaction with known objects, other robots, or navigation within the environment. At the bottom center, an interactive map presents a bird's-eye view of the structured environment, including the entire robotic team and known objects. In addition, a protocol brief window serves as a text-based communication channel with command centers, accompanied by a secondary, smaller video stream element that can be used to display an additional camera view when available. [LSL22]

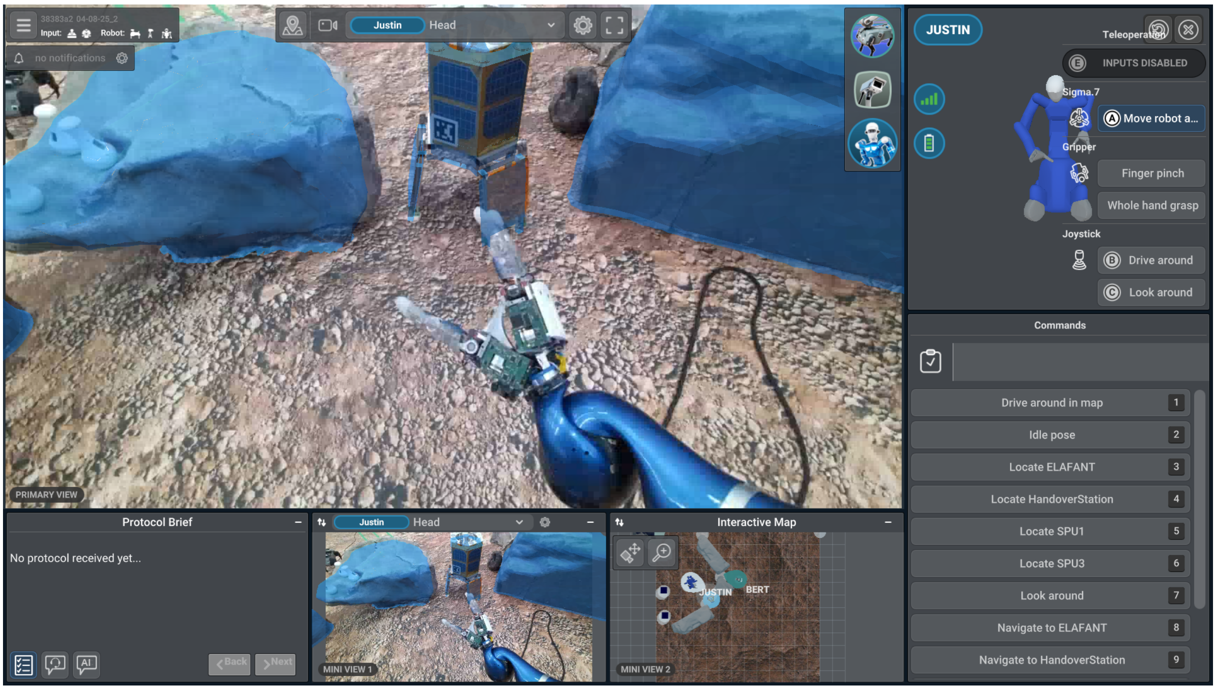


Figure 2.4.: Operator-UI: GUI for task-level commands of the knwoledge driven approach from Surface Avatar, picture taken in the user study of [LSL22]

Results of Studies and Limitations of Current Approaches

In the final session of the Surface Avatar ISS-to-ground experiments, one phase involved object grasping with haptic telepresence, during which the astronaut Jonny Kim communciated a lack of depth information in both control modes. This showed that depth perception is crucial for effective teleoperation and a clear potential to further enhance the Surface Avatar GUI concept during telepresence. [MKM26, pp. 6–7]

This work aims to demonstrate the feasibility of integrating mixed reality technologies with three-dimensional graphical user interfaces for representing unstructured environments during direct teleoperation of robotic grasping tasks. As a first step toward integration into existing approaches, a scaled-down experimental setup is proposed to conduct a user study evaluating both the potential improvement in depth perception and the overall technical feasibility of the approach.

3. System Concept and Methodology

This chapter serves as a bridge between the identified limitations of the current Surface Avatar approach, discussed in Section 2.5.2 of the previous chapter, and the new system concept. First, the conceptual continuation of the Surface Avatar approach is formulated, and research objectives are derived. Subsequently, the system requirements are described, followed by the overall system concept, the visualization concept, and the user study concept.

3.1. Conceptual Continuation of the Surface Avatar Approach

During the ISS-to-ground experiments, teleoperating astronauts experienced difficulties with depth perception and situational awareness while performing unknown object manipulation tasks with direct control. While the Surface Avatar Operator-UI improved spatial understanding of the robot's configuration by providing a 3D virtual rendering of the robot, it lacked an intuitive 3D representation of the surrounding environment beyond known objects. Unknown objects were only visible in the 2D video stream, without a 3D representation, which limited intuitive depth perception and spatial understanding during direct teleoperation. As a result, operators had to rely solely on 2D visual cues when interacting with unknown objects, which made precise depth estimation during direct teleoperation more challenging. In particular, object grasping often required additional corrective motions or repeated attempts, indicating reduced task efficiency in scenarios requiring accurate spatial understanding. This highlights the absence of a dedicated 3D depth visualization for unknown objects in the Operator-UI, and motivates the introduction of depth-based visualization concepts to better support precise object manipulation during direct teleoperation. [MKM26]

As depth perception and intuitive spatial understanding represent key gaps in the system for direct teleoperation, this is of major interest for further research. This work investigates the potential to improve depth perception, spatial understanding, and consequently task efficiency in unknown object manipulation scenarios by incorporating 3D environmental visualization methods and immersive display platforms, specifically point cloud representations combined with mixed reality technology.

3.2. Research Objective and Methodology

This work aims to incorporate the identified key aspects into a robotic setup that closely resembles the direct teleoperation scenarios investigated during the Surface Avatar experiments. The core novel contribution of this work is a newly developed UI system that integrates a 3D point cloud visualization, usable on both a desktop monitor and a mixed reality headset.

The system is intentionally designed as a first-step demonstrator. Its purpose is to provide a controlled basis for comparing different visualization approaches for tasks similar to those conducted within the Surface Avatar experiments. The focus is on grasping tasks involving unknown objects, as used in previous experiments. The demonstrator allows a comparison of different display platforms, specifically a desktop monitor and a mixed reality interface. It also enables a comparison of different visualization concepts, including 2D video feedback and 3D point cloud representations. The subsequent user study investigates how these combinations affect depth perception, spatial understanding, and task efficiency for the defined manipulation task.

The developed UI is based on a heavily reduced version of the existing Operator-UI, with only elements that are essential for direct teleoperation of the specified task, such as video feedback and a 3D robot model including the known static parts of its environment. Knowledge-driven approaches, including augmented overlays for known objects, are intentionally excluded, as the primary goal is to investigate unknown object manipulation tasks.

The UI is then extended by adding a point cloud–based environment visualization as a new window element. As some of the robotic platforms participating in the Surface Avatar experiments already incorporate RGB-D cameras, the point cloud representations are calculated by combining the depth and color data in a newly developed visualization pipeline.

For the user study, the UI performs in multiple visualization modes. These modes allow individual elements to be shown alone or in a specific combination to create controlled conditions for different study runs.

3.3. System Requirements and Assumptions

The following section defines the usability requirements and technical constraints guiding the system design to create the necessary user study conditions.

Usability Requirements

For usability, the system is required to intuitively visualize spatial information to the user while keeping cognitive load low. Therefore, a minimalistic GUI design and minimal distractions during operation are necessary. The system must enable users to successfully perform object manipulation tasks involving previously unknown objects, requiring fast learnability with only a short familiarization phase. Additionally, the GUI and interaction behaviour need to stay consistent across different display platforms, with switchable GUI modes to create controlled conditions during the user study.

Technical Constraints

Several technical constraints are limiting the system concept. The depth resolution of the RGB-D camera is limited, which directly affects the fidelity of the point cloud representation. In addition, potential latency from data acquisition and processing limits real-time teleoperation, which can influence perception and interaction. Furthermore, computational performance constraints limit processing capabilities on standalone mixed reality devices compared to desktop systems.

These constraints influence the choice of the visualization pipeline and motivate the use of a desktop system for computationally intensive processing, while the resulting visualization is streamed to a mixed reality device. The system design is also influenced by the use of existing hardware and software components for sensor integration, teleoperation control, and mixed reality output, including the available level of support for mixed reality devices on different operating systems. Finally, the integration with an existing robotic platform and its predefined command structure introduces additional design considerations, requiring the developed UI to be compatible with and connected to the existing control architecture.

3.4. System Concept Overview

Figure 3.1 presents a conceptual overview of the teleoperation system centered around the new UI. This provides a high-level understanding of the system's general structure, which forms the basis for more detailed system architecture representations.

The system builds upon the multi-windowed GUI approach from Surface Avatar, removing the autonomy features to focus exclusively on studying direct teleoperation modes. The system supports switching between different display platforms, specifically a desktop monitor and a mixed reality headset, as well as switching between different GUI visualization modes. The RGB-D sensor is positioned to view both the robotic system itself and the relevant parts of its surrounding environment. Its data is used in two different ways to visualize the robot and its environment. Firstly, the data is presented as a conventional 2D video stream, and secondly, it is processed into a colored 3D point cloud representation for spatial visualization. In addition, interaction with the GUI is implemented, including changing the placement of visualization elements and synchronously rotating all 3D representations to effectively modify the viewing angle. Separate input devices are used for GUI interaction and robot control, and an automated data collection system is employed for the user study.

The idea behind this setup is to create a scientific user study environment in which variables can be changed independently to gain insight into effects on depth perception and spatial understanding. A key consideration for the chosen depth representation approach is to keep data processing to a minimum in order to stay close to the ground truth and reduce potential mismatches between reality and representation. To minimize technical misrepresentations during the visualization process, the system is therefore designed around this core principle, relying exclusively on deterministic data processing techniques and compact processing pipelines that introduce minimal computational deviation.

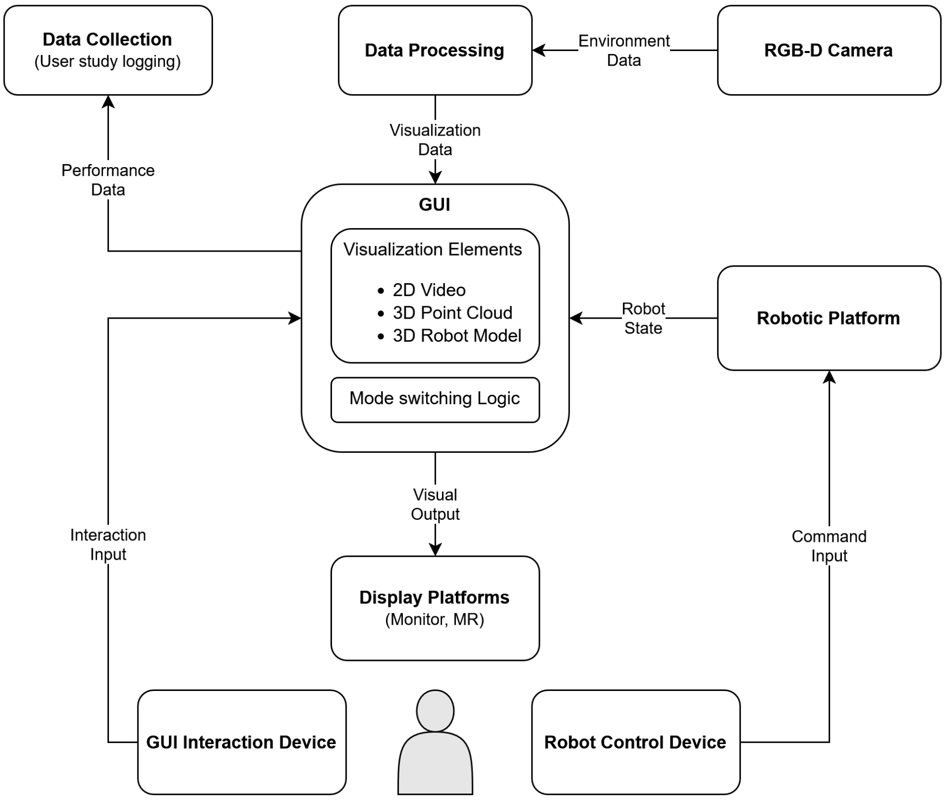


Figure 3.1.: Conceptual overview of the teleoperation system and user interface developed in this work

3.5. Visualization Concept

As previously described, the visualization concept of the GUI uses multiple windows, inspired by Surface Avatar. Each window can display a specific visualization element and can be rearranged and scaled differently. Each element represents distinct information content, interaction principles, and perceptual goals.

Video-based Visualization

A key visualization element in the GUI is a color video stream derived from RGB data. This video-based visualization allows the operator to quickly and intuitively understand the overall scene, providing essential context in unfamiliar environments. It serves as a perceptual baseline, as it provides an immediate overview without requiring any interactive view manipulation and conveys static 2D visual information.

Point Cloud Visualization

The central 3D visualization element is the point cloud representation. It combines RGB data with depth data to present a collection of colored points in virtual space. The underlying concept is to visualize the robot's environment three-dimensionally similar to how a real scene would be perceived by a human physically present, thereby conveying depth intuitively. Since this visualization is 3D, the element can be interacted with to rotate its view angle.

Robot Model Visualization

The third and final visualization element is the robot model representation. It uses the real robot's joint data to configure a similar digital model in the same joint state. This element therefore serves a purely supportive function in this work, as it does not process information about the unknown environment. Instead, it provides a static and abstract representation of the robot's surroundings for orientation purposes. However, the robot model is a 3D visualization and can be rotated synchronously with the other 3D element described above, namely the point cloud.

Mixed Reality Display

The system is designed to operate both on a conventional monitor and on a MR headset. For comparability within the study, the GUI is implemented to be functionally and visually consistent across both display platforms. The primary motivation for using immersive head-mounted displays is the true three-dimensional presentation of virtual elements, such as point clouds, achieved by rendering separate images for each eye to create an immersive viewing experience.

The main function of the system is teleoperation, therefore it is critical to have dedicated input devices that operate independently of the used GUI display platform. The standard VR controllers are closely coupled to the corresponding head-mounted display and are therefore impractical when operating from a desktop monitor. As the system's core design needs to allow the switch between display platforms while keeping the rest of the UI setup unchanged, independent input devices are placed on a table in front of the operator. This, however, introduces the requirement that these devices remain visible even when the operator is wearing a head-mounted display. For this reason, a mixed reality headset is used, which utilizes passthrough technology to display the real environment while overlaying virtual elements. This way, the immersive capabilities of virtual reality are combined with continuous awareness of the physical surroundings, allowing the operator to see and intuitively interact with the input devices during operation. Additionally, effects like motion sickness and disorientation commonly experienced with virtual reality technology are reduced.

3.6. Study Concept Overview

A key interest of the study is to clarify the correlation between immersive 3D visualizations of the robot's environment and the operator's ability to estimate depth and spatial structure of a remote scene during precise object manipulation tasks. The intention is to improve the user's task efficiency by creating a more intuitive teleoperation process with less human error. This work hypothesizes that higher levels of visual immersiveness support a more intuitive mental reconstruction of the 3D situation leading to improved depth perception during teleoperation. By presenting spatial information in a manner that closely resembles real-world perception, immersive visualization is expected to reduce the operator's cognitive load by emphasizing only the most relevant information. Point cloud representations are expected to provide a more intuitive visualization of depth information compared to conventional 2D displays and increased immersiveness through mixed reality technology is expected to further enhance this effect.

To evaluate these assumptions, objective performance metrics such as task completion time, error rate and end-effector trajectory are measured. In addition, subjective questionnaire-based metrics are collected to assess the operator's perceived workload and spatial understanding.

The evaluation is conducted on two distinct comparison levels. The first focuses on the visualization element used in the GUI, while the second addresses the display platform. Participants perform the task under multiple experimental conditions. Within each comparison level, identical run conditions ensure that performance differences can be directly linked to the modality under investigation.

4. System Design and Implementation

As discussed in the previous chapter, a testable environment for the planned user study required the development of a new GUI and a dedicated integration with a suitable robotic platform for teleoperation. For mixed reality support, it was necessary to redesign the UI system from the ground up, while still building on the core design principles of the Surface Avatar GUI. This chapter presents the resulting system implementation and describes its components and functionality in detail.

4.1. High-Level System Architecture

For context, a high-level overview of the entire architecture is first presented in Figure 4.1. The system consists of two workstations, two input devices, an RGB-D sensor, two display platforms, and a robotic platform interacting with the physical environment. The workstations act as physically and functionally separate processing stations, one dedicated to GUI visualization and the other to teleoperation control. Overall, they connect the operator space with the robot space serving as a central data hub for communication and processing. In conclusion, the system enables GUI interaction and teleoperation inputs to command the robot and displays visualizations from captured sensor data.

The core element is the **Visualization Workstation** responsible for receiving, processing, and visualizing data from both the robot and the RGB-D sensor. For data exchange, a communication module and a sensor interface module are implemented. Data is first stored, then processed and finally visualized as a virtual representation embedded within a coherent GUI. Manual GUI configuration is possible via dedicated GUI interaction input devices. The GUI is rendered and output to one of the available display platforms, namely a desktop monitor or a MR headset. The visualization workstation is also responsible for connecting to the RGB-D sensor to control data acquisition and receive the sensor stream.

The second central element is the **Teleoperation Control Station**, which is responsible for receiving teleoperation inputs from the user, processes the data, and sending the corresponding commands to the robotic platform for execution. On the robotic platform, actuators execute the received commands, while sensors measure the resulting robot configuration. This information is then sent back to the control station. From there, the data is then transmitted to the visualization workstation via the communication module.

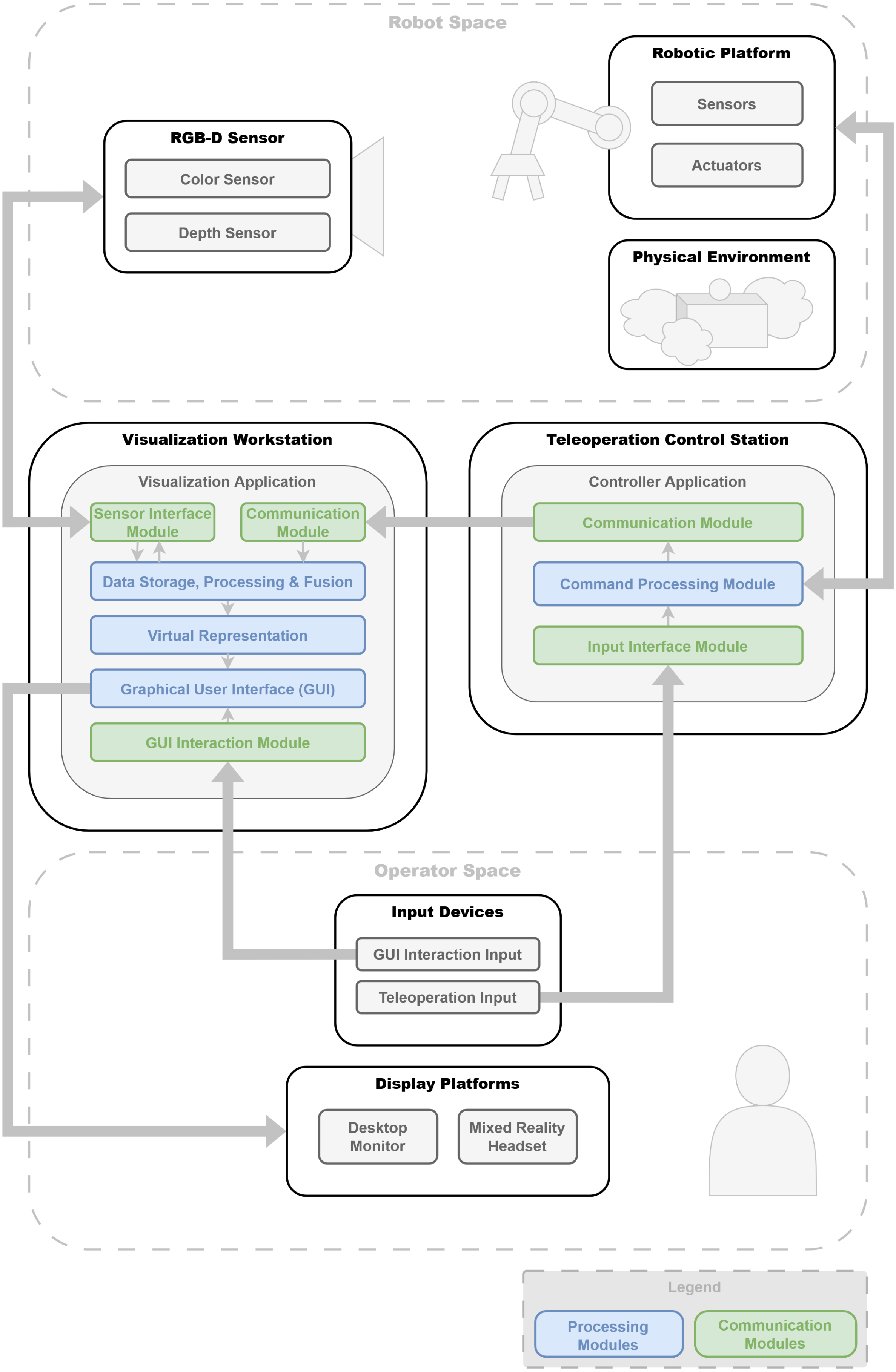


Figure 4.1.: High-level system architecture of the teleoperation system: operator-space (bottom), robot-space (top), and central data hub (center) connecting the two

4.2. Hardware Architecture

Figure 4.2 shows the implemented hardware architecture, boxes represent components, while lines show the connection types in between. The system overview structure from Figure 4.1 was adapted to illustrate how the hardware components correspond to each system element. For clarity, the most relevant components are shown using photographs taken from the real system. The Orbbec Femto Mega is used as the RGB-D sensor, and the Meta Quest 3 serves as the mixed reality headset for the system. The robotic platform for teleoperation consists of a KUKA Lightweight Robotic Arm (LWR) mounted on a fixed testbed and equipped with a DLR CLASH end-effector developed for object grasping. [FR20; AHO07; BKS10]

The two core stations in the center are each an individual desktop personal computer (PC), one for visualization, the other for teleoperation control. They act as central computing units and junctions, from which every main connection runs to the other hardware components. Some components are connected via a local area network (LAN), while others use Universal Serial Bus (USB) connections.

The peripheral hardware components can be grouped functionally and locally into operator space and robot space, as previously explained in Section 4.1. The operator space comprises all UI devices, including the two input devices and the display platforms used by the operator to interact with the system. A standard computer keyboard is connected to the visualization desktop PC and serves as the input for GUI interaction. A three-axis input device (SpaceMouse) is connected to the teleoperation control desktop PC and is used for robot control. The visualization desktop PC also connects to a desktop monitor via a video connection and to the Meta Quest 3 using an MR-capable link cable.

At the top of Figure 4.2, the two components located in the robot space are shown. The first component is the robot itself, a KUKA LWR equipped with a DLR CLASH hand, forming an arm-like manipulation setup capable of advanced and precise object manipulation. It is connected via network to the teleoperation control desktop PC. The second component is the Orbbec Femto Mega RGB-D camera, which is capable of capturing synchronized color and depth data from a single device and is connected via USB to the visualization desktop PC.

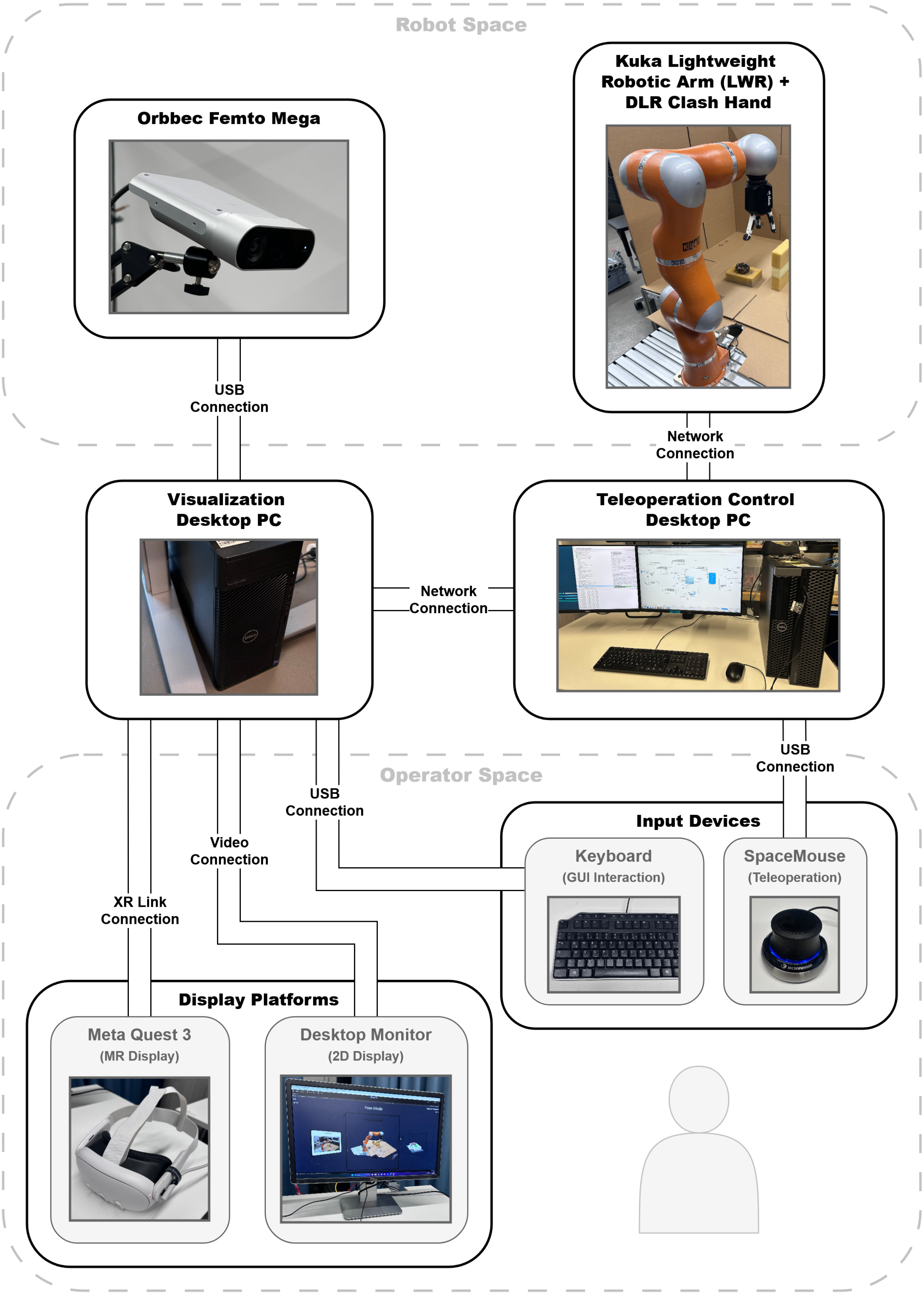


Figure 4.2.: Hardware architecture of the teleoperation system: photographs of the individual components and connections between them

4.3. Software Architecture

Figure 4.3 shows the two software applications running in the system. For this work, the visualization application was built from the ground up, while the teleoperation application was largely based on an existing system, with the additional integration of SpaceMouse control input handling as well as a WebSocket-based connection back to the visualization application.

4.3.1. Visualization Application (Unity)

On the left in Figure 4.3, the visualization application is shown. It is based on the Unity game engine and is responsible for data acquisition and communication, data pipeline and storage, data processing, visualization and GUI handling, as well as display and output. The modules form a linear processing chain that transforms incoming sensor and telemetry data into visual feedback for the operator.

More specifically, data acquisition and communication connect to the Orbbec Femto Mega (RGB-D sensor) via an SDK-based interface and to the teleoperation application via a WebSocket-based connection. The data pipeline defines provider-to-stream routing, specifying a dedicated storage container for the acquired data, using Scriptable Objects (SOs) as Unity-specific data buffers. While the pipeline is designed this way to support multiple RGB-D sensors operating in parallel, this work deliberately uses a single RGB-D camera to more closely reflect the sensor capabilities of the mobile robotic platforms used in the Surface Avatar project.

Data processing takes the stored data and transforms it into displayable concepts, such as a point cloud particle system derived from depth data. In addition, joint state data is mapped to a virtual robot model. The processed data is then integrated into a GUI concept, where the calculated elements are presented and can be interacted with. Finally, the GUI is rendered and output as a video stream, depending on the target display either as a conventional 2D rendering or as a mixed reality rendering.

4.3.2. Teleoperation Application

On the right side of Figure 4.3, the second software application is shown. It runs on the teleoperation control desktop PC and functions as a robot control hub implemented in the Manager of Links-and-Nodes (LN) [SB14], a modular middleware developed by DLR that is conceptually similar to the Robot Operating System (ROS) [QCG09]. It hosts

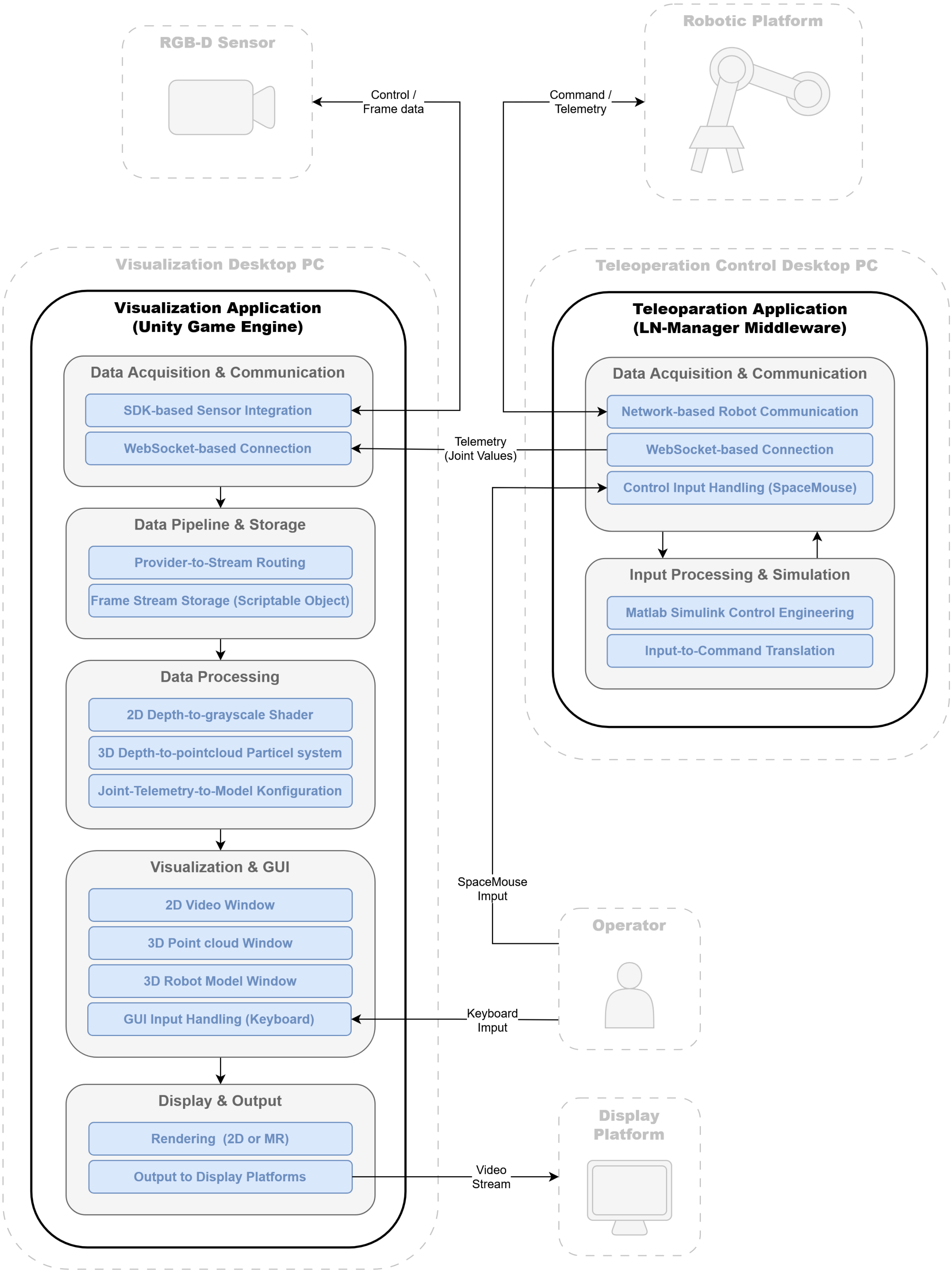

RGB-D Sensor
Control / Frame data
Command / Telemetry
Robotic Platform
Visualization Desktop PC
Visualization Application (Unity Game Engine)
Data Acquisition & Communication
SDK-based Sensor Integration
WebSocket-based Connection
Data Pipeline & Storage
Provider-to-Stream Routing
Frame Stream Storage (Scriptable Object)
Data Processing
2D Depth-to-grayscale Shader
3D Depth-to-pointcloud Particel system
Joint-Telemetry-to-Model Konfiguration
Visualization & GUI
2D Video Window
3D Point cloud Window
3D Robot Model Window
GUI Input Handling (Keyboard)
Display & Output
Rendering (2D or MR)
Output to Display Platforms
Teleoperation Control Desktop PC
Teleoparation Application (LN-Manager Middleware)
Data Acquisition & Communication
Network-based Robot Communication
WebSocket-based Connection
Control Input Handling (SpaceMouse)
Input Processing & Simulation
Matlab Simulink Control Engineering
Input-to-Command Translation
Telemetry (Joint Values)
SpaceMouse Imput
Operator
Keyboard Imput
Display Platform
Video Stream

Figure 4.3.: Software architecture of the teleoperation system: visualization application to process the GUI in Unity (left) and teleoperation application to process robot commands (right)

all processing software between the robot control inputs and the robotic platform. Input data is first acquired and communicated, then processed within simulation environments such as MATLAB Simulink for control engineering, and finally translated into executable commands that are sent to the robotic platform via network-based communication. In addition, the application receives telemetry data from the robot via network-based communication. This data is used as feedback for the internal simulation and is forwarded as joint value updates to the visualization application via a WebSocket-based connection.

4.4. Data Pipeline Design

Since the RGB-D stream serves as the primary data source for both point cloud and video-based visualization elements, a newly implemented modular data pipeline was developed to support the integration of additional visualization elements or RGB-D sensors in the future.

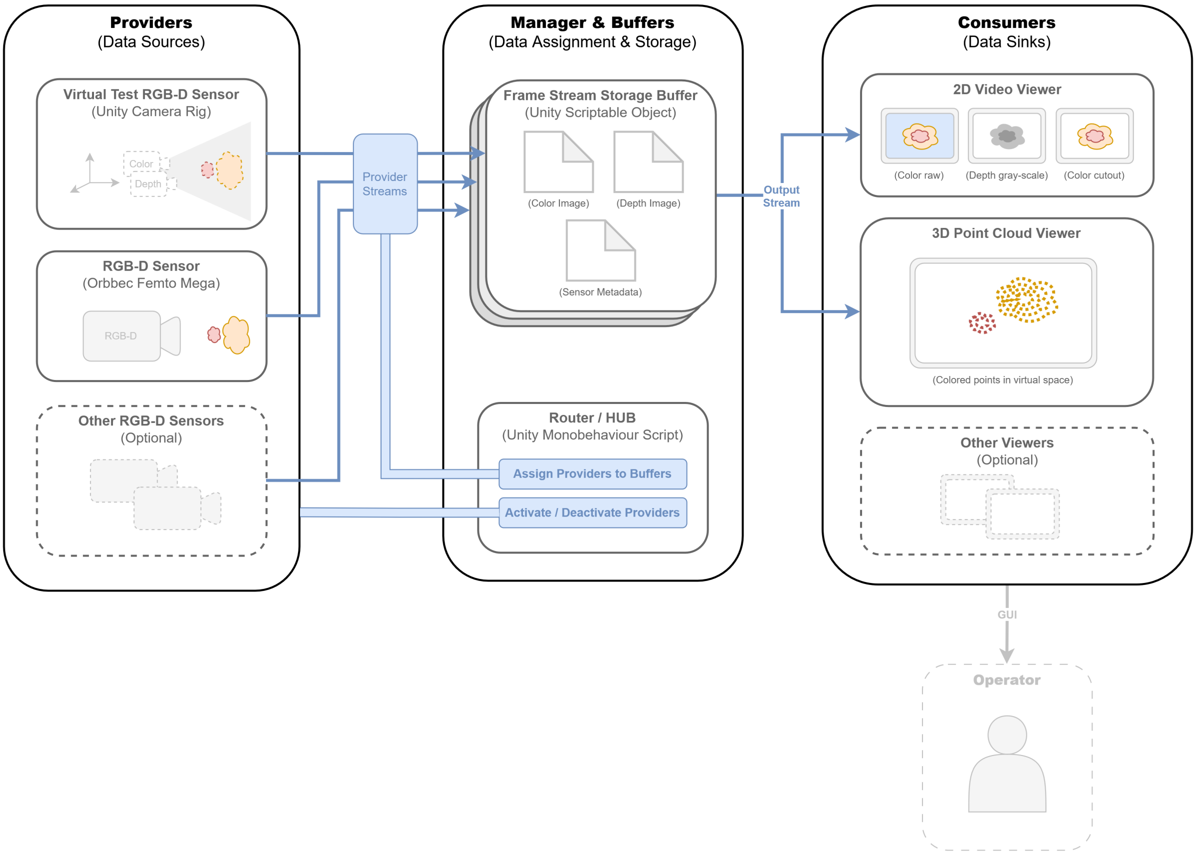


Figure 4.4.: Data Pipeline Design of the teleoperation system: providers (left), data buffers with provider manager (center), and consumers (right)

As shown in Figure 4.4, this data pipeline design distinguishes between data sources (providers), data sinks (consumers), and the mediating manager and buffer for data assignment and storage. The idea behind this system is modularity and scalability, as the design allows multiple providers to each be assigned a distinct frame stream storage buffer and multiple consumers to read from any one of those buffers to process and visualize the data. While the exact types of providers and consumers can vary, the frame stream buffer connecting them follows a consistent design. It stores the most recently received frame data, consisting of a color image (RGB), depth image, and additional sensor metadata such as camera intrinsics for spatial reconstruction.

As already introduced, the provider used in this work is a single RGB-D camera (Orbbec Femto Mega). Two different consumer types are implemented, namely a 2D video viewer and a 3D point cloud viewer. Although this work therefore requires only a single pipeline stream between provider and buffer, this concept supports the integration of additional pipelines, enabling the use of multiple sensors to provide complementary views of the same scene.

4.5. Visualization Implementations

As the visualization application serves as the core contribution of this work, the following section is dedicated to describing the implementation of the visualization concept already presented in Section 3.5 of the previous chapter. The entire application is built within the Unity environment, extended by packages and imported software development kits (SDKs). The main processing logic in Unity is implemented as MonoBehaviour scripts, written in the `C#` programming language.

4.5.1. Sensor and Stream Integration

This section focuses on the integration of the RGB-D sensor and the abstraction of sensor-specific data streams into a unified internal data representation used by the visualization pipeline.

First, the implementation of the data pipeline design introduced in the previous section is described. The RGB-D sensor used in this work (Orbbec Femto Mega) connects via an official Unity-specific Orbbec SDK, which provides an application programming interface (API) for connecting Unity to the sensor and offers high-level control functionality implemented in `C#`. This includes methods like finding an externally connected Orbbec device, specifying the data parameters, and starting the data stream pipeline from the device into Unity. It also handles synchronization between the color and depth images, so corresponding pixels are correctly aligned. This is especially necessary, since the Orbbec

Femto Mega captures color images at a much higher resolution than depth images, and the sensors for each modality are slightly offset from each other. In addition, the capture timing may be slightly misaligned. This requires the depth image to be upscaled and spatially aligned to match the color image exactly, which is handled by the SDK. Temporal synchronization with the corresponding color frame is performed by the sensor software itself.

The Orbbec SDK also integrates its own visualization concept, however the goal of the system described here is to operate independently of any specific sensor SDK. Therefore, the frame stream buffer concept was introduced in Section 4.4. A so-called Scriptable Object (SO) represents this buffer in Unity, functioning as a storage container for the most recently received frame data. It contains variables for a color texture, a depth texture, camera intrinsics, and other sensor metadata. Even though the specific system used in this work is configured as described earlier, it can be replaced or complemented by any other sensor that provides a Unity-compatible interface. To enable this clear modular separation, the Orbbec SDK transfers its SDK-specific internal representation of the captured frame data into the generic Unity SO–based frame stream buffer. From this point onward, the sensor frame data is stored internally within the Unity environment and can be accessed by any Unity-based consumer.

4.5.2. Video Visualization

The first consumer type integrated into the system is a 2D video visualization of the frame stream data. It is implemented as a Unity Raw Image UI element, which displays a texture rendered onto a quad that can be freely positioned in 3D virtual space. This becomes particularly important for the MR presentation described later. In computer graphics, any mesh is composed of triangles for efficient processing. A quad is then simply represented as two triangles forming a rectangular surface, onto which a rectangular image texture can be rendered.

Since the texture is read directly from the frame stream buffer, the image is updated whenever the buffer texture is replaced, which occurs once per incoming sensor frame. Therefore, the resulting visual output is a sequence of image frames displayed on the UI element, which effectively acts as a virtual 2D video screen.

A raw RGB color texture can be displayed directly in this manner, displaying the depth texture however is slightly more complex, as the values stored per pixel do not represent color but physical distance. This requires the distance values to be pre-processed into color representations to make them displayable, which is accomplished using shaders. A shader is a small program that runs entirely on the graphics processing unit (GPU) and uses parallel processing for efficient graphical image manipulation. It typically consists of two stages: a vertex shader and a fragment shader. The vertex shader operates on mesh vertices and can modify their positions in space, while the fragment shader computes the color of each pixel. For video visualization purposes, only the fragment shader is required.

Raw Color Visualization

As previously mentioned, the RGB texture from the frame stream can be displayed directly on the Raw Image UI element without any additional processing, as it already provides the correct format. Although not activated for the user study, the system supports the use of a shader to manipulate the raw color image prior to display if desired. The following sections describe two implemented shader-based techniques to visualize depth on a 2D image. Figure 4.5 shows the raw color visualization of a virtual test object, which serves as the basis for the depth visualization techniques described in the following sections.

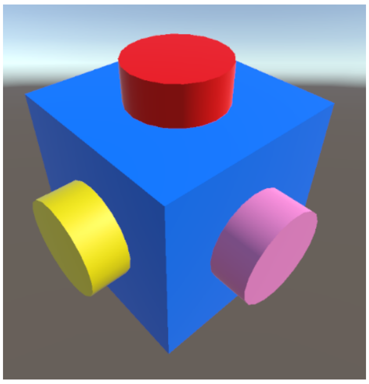
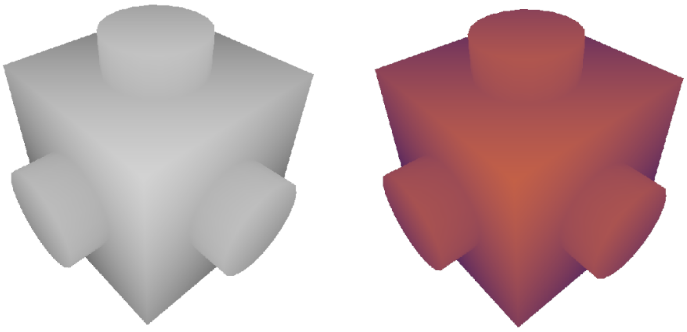


Figure 4.5.: Raw color image of a virtual test object as reference for following 2D depth visualization techniques

2D Depth Visualization

The simplest approach to visualizing depth information in 2D is a grayscale representation. In this method, raw depth values provided by the sensor are first clamped to be within a defined minimum and maximum depth range. Then depth values close to the minimum or maximum range are rendered fully transparent, creating a stencil-like cutout of valid surfaces within range. The remaining visible values are then normalized and mapped to grayscale intensities, where nearer points appear darker and farther points brighter.

As an alternative to grayscale, the same concept can be used to produce a depth heatmap by mapping depth values to a predefined color gradient, where different depth ranges are distinguishable by color instead of intensity. Figure 4.6 shows the visual outcome of the two visualization techniques derived from the unprocessed raw color image.



(a) Depth as grayscale **(b)** Depth as heatmap

Figure 4.6.: 2D depth visualizations techniques: grayscale (a) and heatmap (b)

Color Cutout Visualization

The stencil-like cutout, as described in the last section about 2D depth visualization, can also be used on the raw color image directly. Instead of mapping the depth values to grayscale or a heatmap, the transparency defined by the depth range is combined with the original RGB texture. As a result, only surfaces within the specified depth range remain visible, while all pixels outside this range are rendered fully transparent. This creates a cutout of valid surfaces in real color, effectively removing the background environment from the color image.

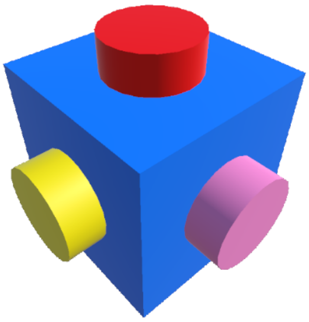


Figure 4.7.: Depth-based 2D color cutout from the virtual test object image in Figure 4.5

4.5.3. Point Cloud Visualization

The central visualization element in this work, the point cloud, is calculated by a Unity particle system called VFX Graph. Systems like this allow the creation and manipulation of millions of particles in real time by utilizing parallel processing on the GPU. VFX Graph has a graphical user interface that allows logical node operators to be connected to process input data and create particles from it.

At runtime, the node-based graph is compiled into shader language code that is executed on the GPU. The logical operations visually defined in the VFX Graph are evaluated once per Unity tick for each individual particle. As a result, these operations remain computationally simple to be processed in parallel on the GPU, creating a real-time dynamic particle system.

VFX Graph Overview

The node-based interface of the VFX Graph provides an intuitive representation of the per-particle logic, illustrating how particles are generated, processed, and updated each frame. The central structure of the particle system is shown conceptually in Figure 4.8. A photograph of the real point cloud particle system as implemented using the VFX Graph for this work is shown in Figure A.1.

A VFX Graph particle system has three areas, the base element being the particle system pipeline responsible for creating, updating and rendering the particles, as can be seen on the right side (orange) in Figure 4.8. On the opposite end, the system imports properties (parameters) from outside of the VFX Graph, shown on the left (blue) in Figure 4.8. In this work, the point cloud system imports the properties for color texture, depth texture, texture dimensions, camera intrinsics, particle size and a depth range to hide particles outside the specified distance limits. In between the two described areas the per particle data processing is happening by connecting mathematical operator nodes. VFX Graph also allows the creation of subgraphs as nodes, which encapsulate complex operations while exposing only a black box interface with defined inputs and outputs. This helps to keep the logic visually tidy and readable, while subdividing logical operations into chunks that can be viewed as simple node blocks in the global view of the VFX Graph. Figure 4.8 shows the processing nodes in the center (gray), where green boxes represent base functions in VFX Graph, such as mathematical operators or texture sampling, and black boxes represent subgraph nodes implementing the complex calculations shown within them. The central node in the point cloud particle system of this work is the backprojection, which is described in more detail in the following section.

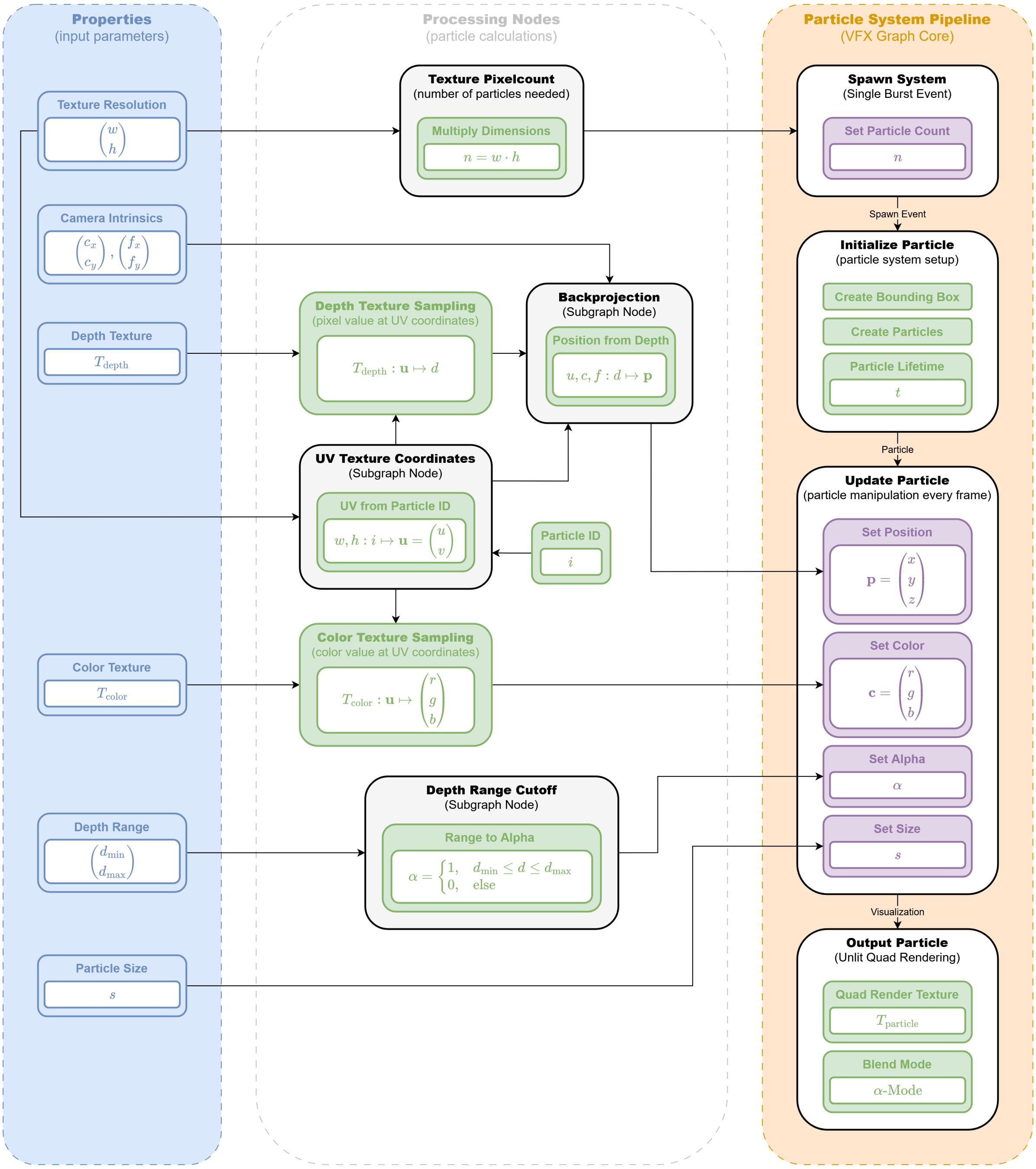


Figure 4.8.: Overview of the point cloud particle system implemented in VFX Graph showing input parameters (blue), basic operation nodes (green), subgraph nodes (gray) and particle instructions (purple)

Backprojection Subgraph

In order to use this system to create a point cloud from depth data, a backprojection technique is used. Conceptually, this works in the same way as depth sensing, but in reverse. Instead of capturing depth from 3D geometry, the depth texture is used to reconstruct 3D point positions for each pixel. The reconstruction requires the camera intrinsics used during depth acquisition, which define how 3D points are projected onto the 2D image plane, including focal length and principal point. This block represents the central logic, taking the depth texture and camera intrinsics as input and outputting particle positions. Figure 4.9 shows the node connections of the subgraph in more detail.

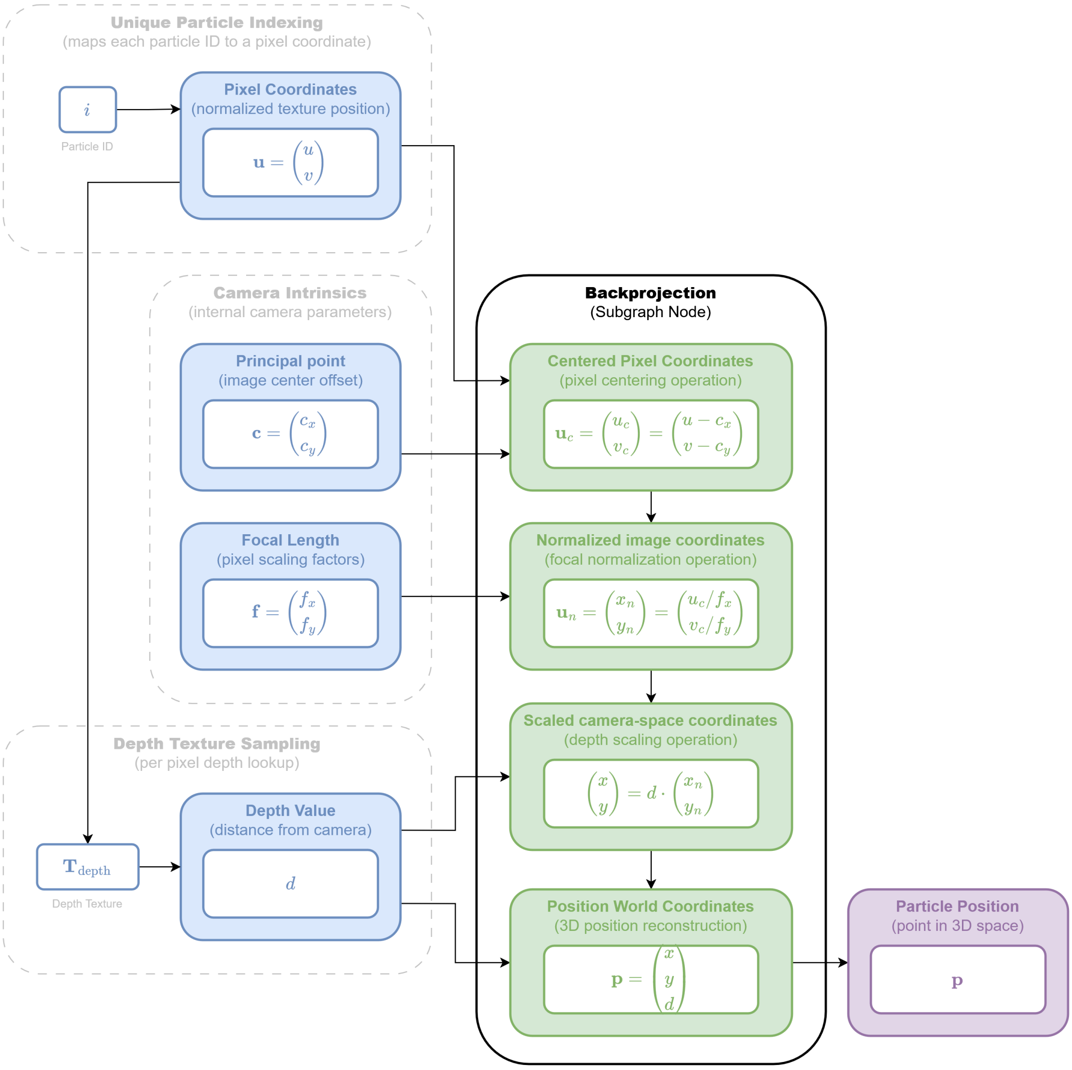

Unique Particle Indexing
(maps each particle ID to a pixel coordinate)
i
Particle ID
Pixel Coordinates
(normalized texture position)
$\mathbf{u} = \begin{pmatrix} u \\ v \end{pmatrix}$
Camera Intrinsics
(internal camera parameters)
Principal point
(image center offset)
$\mathbf{c} = \begin{pmatrix} c_x \\ c_y \end{pmatrix}$
Focal Length
(pixel scaling factors)
$\mathbf{f} = \begin{pmatrix} f_x \\ f_y \end{pmatrix}$
Depth Texture Sampling
(per pixel depth lookup)
$\mathbf{T}_{\text{depth}}$
Depth Texture
Depth Value
(distance from camera)
d
Backprojection
(Subgraph Node)
Centered Pixel Coordinates
(pixel centering operation)
$\mathbf{u}_c = \begin{pmatrix} u_c \\ v_c \end{pmatrix} = \begin{pmatrix} u - c_x \\ v - c_y \end{pmatrix}$
Normalized image coordinates
(focal normalization operation)
$\mathbf{u}_n = \begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} u_c/f_x \\ v_c/f_y \end{pmatrix}$
Scaled camera-space coordinates
(depth scaling operation)
$\begin{pmatrix} x \\ y \end{pmatrix} = d \cdot \begin{pmatrix} x_n \\ y_n \end{pmatrix}$
Position World Coordinates
(3D position reconstruction)
$\mathbf{p} = \begin{pmatrix} x \\ y \\ d \end{pmatrix}$
Particle Position
(point in 3D space)
$\mathbf{p}$

Figure 4.9.: Concept of the Backprojection Subgraph used for point cloud calculation in VFX Graph showing input parameters (blue), inner subgraph operations (green) and particle position output (purple)

The inner operations of the subgraph (green) reveal the mathematical steps used to transform depth data (blue) into 3D positions (purple). These steps include centering pixel coordinates, normalizing image coordinates, scaling camera-space coordinates, and finally reconstructing world coordinates that describe the particle position output. This describes how the Backprojection Subgraph, visualized as a black box in Figure 4.8, converts its inputs into particle positions.

Depth Range Cutoff Subgraph

Another important node is the Depth Range Cutoff Subgraph, which defines the minimum and maximum depth range within which particles remain visible. As shown in the overview diagram in Figure 4.8, this is achieved by setting the transparency value α to 1 when a particle is located within the specified depth range, and to 0 when it lies outside. The Set Alpha function in the particle system pipeline then applies the calculated α value. As a result, particles are rendered either fully visible or fully invisible, which creates a depth range cutoff effect without changing the fixed number of particles defined by the number of texture pixels. With this approach, particles only need to be created once during initialization and can be rendered invisible whenever they should not be displayed, instead of being constantly created and destroyed.

4.5.4. GUI Implementation and Interaction

The GUI structures and presents the visualization elements (VEs) introduced in the previous sections in a coherent manner. Since the primary focus of this work is a user study investigating the influence of the GUI on operational effectiveness, the study requirements directly influence the GUI design and interaction concept. As already stated, a multi-window GUI approach is implemented.

Visualization Elements Overview

First, Table 4.1 lists all VEs used in the GUI, defines an abbreviation for reference, and briefly describes each element. All VEs have been thoroughly introduced in previous sections, with the exception of the Robot Model VE, which is inspired by and adopted from the Surface Avatar Operator-UI and is not the primary focus of this work. The robot model is represented using the Unified Robot Description Format (URDF), specifying the robot's kinematic structure and joint metadata and enabling accurate pose reconstruction. In addition, user input allows a synchronized rotation of all 3D elements around their local axes, effectively changing the viewing angle.

Table 4.1.: Visualization Elements used in the GUI

Visualization Element (VE)	Abbreviation	Description
Video Screen VE	VID-VE	2D virtual video screen visualizing RGB or depth texture on mesh quad (see Section 4.5.2)
Point Cloud VE	PC-VE	3D particle system visualizing RGB and depth texture as colored point cloud (see Section 4.5.3)
Robot Model VE	RM-VE	3D URDF robot model for current pose (joint configuration)

GUI Modes

The interface supports three distinct operation modes, each imposing specific constraints on the use of VEs listed in Table 4.1. These modes are fixed and cannot be changed by the user, as they are intentionally designed to introduce controlled GUI limitations for the purpose of the study. Both the video screen and point cloud mode show only a single fixed VE centrally focused on the display, meaning the user cannot switch between VEs. In contrast the free focus mode implements all three VEs from Table 4.1 in a way, that lets the user switch through display configurations, effectivly choosing which one of the three VEs is prominently shown in the center. The other two will be visualized noticably smaller next to the central one.

Table 4.2.: GUI Modes: specific combinations of the VEs from Table 4.1

Mode	VEs	Interaction Concept
Video Screen Mode	VID-VE	static with no interaction
Point Cloud Mode	PC-VE	dynamic rotation by user inputs
Free Focus Mode	VID-VE, PC-VE, RM-VE	Switch through VEs by user inputs (one in central focus, other two smaller on the side), simultaneous dynamic rotation of PC-VE & RM-VE by user inputs

GUI Implementation

To realize the multi-window GUI and its described mode functionality, three Unity transforms are used as placeholders defining the position, rotation, and scale of the GUI windows. One placeholder is large and centrally positioned on the display, while two smaller placeholders are located on either side of the central window. In the single VE modes (video screen mode and point cloud mode), only the central placeholder is used, in the free focus mode however all three placeholders are active simultaneously.

The VEs are implemented as prefabs, which represent predefined combinations of Unity GameObjects (GOs) that can be instantiated within the scene. Each window includes a 2D frame border to clearly indicate its boundaries and to visually distinguish it from the surrounding environment, which is particularly beneficial when displaying 3D visualization elements. Figure 4.10 illustrates the concept of placeholder windows and their allocation of VE prefabs.

A distinction between 2D and 3D visualization elements (VEs) is needed, as they are implemented differently in Unity. 2D elements must be placed within a Canvas GO in Unity, which acts as a container for 2D content and can be positioned, rotated, and scaled within the 3D scene, effectively behaving like a virtual screen. In contrast, 3D visualization elements can be placed directly into the scene, as they do not require this intermediate dimensional mapping.

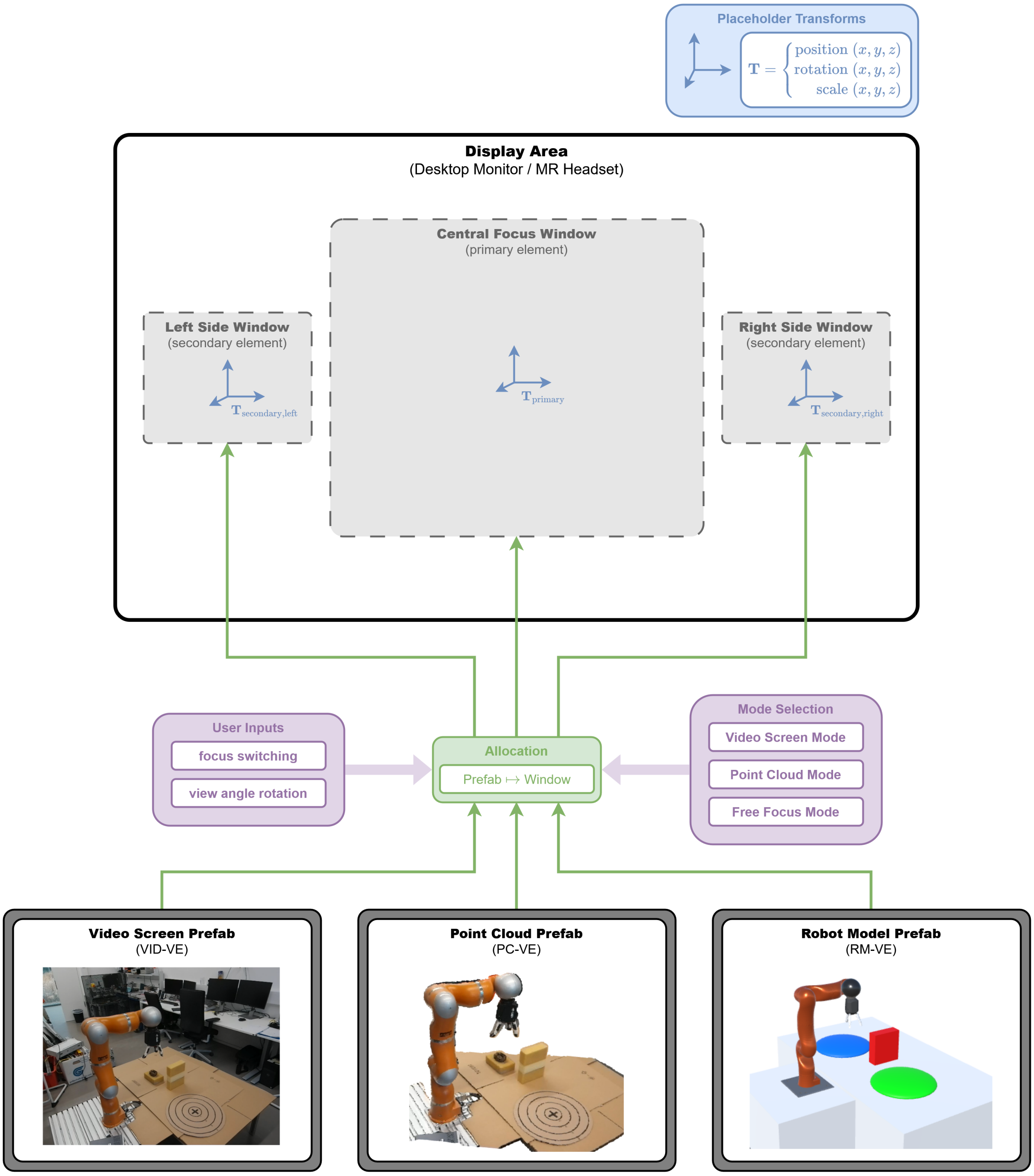


Figure 4.10.: GUI implementation overview of visual element positioning in the display area by allocating (green) VE prefabs to placeholder transforms (blue) depending on user inputs and mode selection (purple)

4.5.5. MR Rendering

Unity integrates the standardized OpenXR package to support XR interaction and rendering, including MR. As part of this framework, the Quest Feature Group allows connection to the Meta Quest 3 headset used in this work.

Since the GUI application runs on a Windows desktop computer, a streaming connection to the Meta Quest 3 is required. This connection is established using the Meta Quest Link application, which operates outside of Unity and enables a wired streaming link between the desktop computer and the headset.

During startup of the GUI application, Unity detects when a Link connection to the Meta Quest 3 is available, in which case the application runs in MR rendering mode and streams the output directly to the headset. If no Link connection is detected, Unity falls back to rendering the application on the desktop monitor without MR support. This behavior makes it possible to use the same application across both display platforms, which is beneficial for direct comparability between them.

4.5.6. Performance Optimizations

The core processing of graphics-related computations typically involves doing the same simple operations on a large number of data points, such as pixels in an image or particles in a point cloud. This can easily amount to millions of individual data points that must be recomputed every visual update in Unity (tick). Processing such workloads on a small number of complex processor cores, as found in a central processing unit (CPU), is inefficient and impractical.

Instead, modern computer systems include a graphics processing unit (GPU), which is specifically optimized for this type of workload. GPUs consist of a large number of simple cores that can execute identical instructions in parallel across many data points simultaneously. This is possible because the individual data points are independent of one another, allowing each computation to be performed in any order without dependencies between points. An overview comparison between the CPU and the GPU of a standard computer is shown in Figure A.2.

The GUI system implemented in this work involves two computationally demanding processing tasks. The first is the shader-based video visualization described in Section 4.5.2, which operates on every pixel of a texture during each update cycle. The second is the particle system introduced in Section 4.5.3, which computes particle positions and colors once per tick. If these computations were executed sequentially on CPU cores, the resulting frame rate would be too low to be perceived as a real-time video stream by a human user.

To mitigate this computational load and associated latency, both the shader and the particle system are dispatched once from the CPU to the GPU during initialization. After that, all computations run entirely on the GPU without the need for per-frame CPU involvement or readbacks.

The same principle applies to the transfer of Orbbec-specific texture data into Unity SOs, as described in the data pipeline design in Section 4.4. This transfer is handled using Unity's Graphics Blit function, which performs pixel-wise texture copying directly on the GPU, avoiding any CPU processing.

4.6. Data Flow

To clarify how data is transferred and processed between the system's main components, a standard high-level data flow diagram is shown in Figure 4.11.

During runtime, frame-wise RGB-D data is acquired from the depth sensor and transmitted to the visualization application via cable connection. In parallel, robot telemetry data (joint configuration) is acquired from the robotic platform and communicated from the teleoperation application to the visualization application via network connection. In addition, the operator provides input for GUI interaction and teleoperation control through multiple input devices connected via cable connection.

The incoming sensor and telemetry data are routed to dedicated internal data buffers. The frame stream buffer provides access to the most recent sensor frame, while the robot telemetry buffer stores the latest joint configuration. These buffering mechanisms decouple data acquisition from data consumption within the system.

The buffered data is then distributed to the visualization pipelines, where sensor-based data is combined with robot state information within the GUI using multiple visualization elements. This results in a unified data interface that is continuously updated based on the most recent data streams.

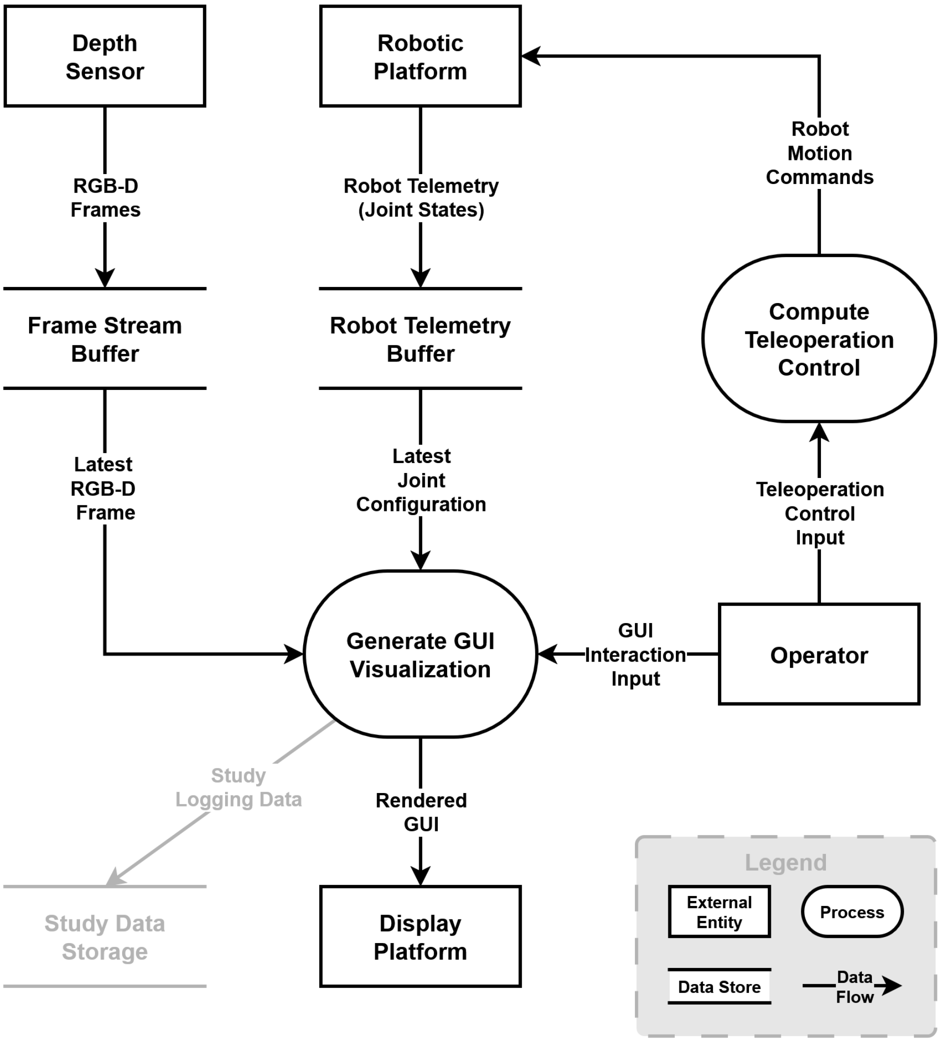


Figure 4.11.: Data flow diagramm of complete system

4.7. Backend Logging and Tracking

Additionally, a backend logging and tracking system for the user study is implemented, as already visualized in gray in the data flow diagram of Figure 4.11. This is not a necessary part of the teleoperation system to be operational, however it is responsible for capturing and measuring relevant study logging data and storing it in a standardized format. Since the Unity visualization application serves as a central hub through which all crucial system data passes in a synchronized manner, the logging is implemented there as well. An overview of the logging and tracking system architecture is shown in Figure 4.12.

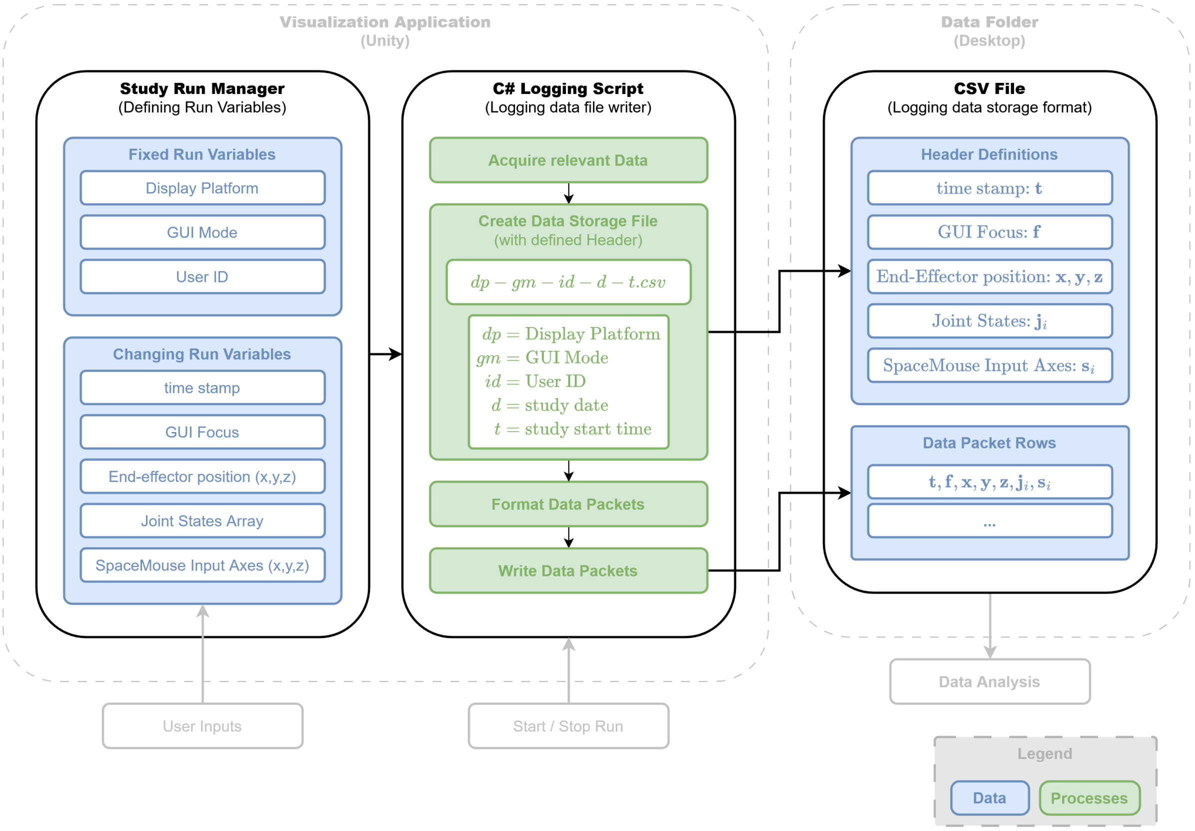


Figure 4.12.: Backend logging system for tracking and storing user study run data: Unity-internal run manager (left) and logging script (center) creating a CSV file (right) for Unity-external data storage

As illustrated, the Unity visualization application executes a `C#` script that handles data logging during a study run. A predefined key combination is used to start and stop a logging session. Once activated, the script acquires relevant data from the Unity environment, including both fixed and time-varying run variables defined and managed by a separate Unity script referred to as the Study Run Manager.

A comma-separated values (CSV) file is then created outside the Unity application and named according to a standardized scheme, as shown in the C# Logging Script block in Figure 4.12. The file headers follow a fixed structure, illustrated in the CSV File block on the right. During the logging session, data packets are generated at regular time intervals and appended as individual rows to the CSV file, resulting in continuous tracking of the relevant data values. The logging process is terminated using the same key combination as for initialization, which finalizes the file and produces a complete CSV record spanning from the initial timestamp to the end of the study run.

The logging concept and CSV format were chosen to provide a clear, structured, and well-labeled storage of run data that can be reliably traced back to the corresponding study conditions and participants. This allows straightforward interpretation and visualization of the results in tools such as Microsoft Excel, while also supporting automated large-scale data analysis.

4.8. Operational Workflow

To conclude the description of the system functionality, the following workflow illustrates the operational interaction of the system components.

In the first step of the operational workflow, the teleoperation application establishes a WebSocket connection to the visualization application and a network connection to the robotic platform, enabling actuator control and sensor feedback. In parallel, the Unity-based visualization application connects to the Orbbec Femto Mega depth sensor and to the selected display platform, either a desktop monitor or the Meta Quest 3 headset.

Once all communication channels are established, the graphical user interface is generated and rendered on the display. The depth sensor is positioned to cover both the robotic arm and its surrounding workspace, providing color and depth information for visualization.

After a GUI mode has been selected, the operator can interact with the system to teleoperate the robot. Keyboard inputs are used to interact with the GUI and visualization elements, while the SpaceMouse provides three-axis input for controlling the robot end-effector. Depending on the selected mode, the operator may switch the focused visualization element or adjust the view angle of the 3D visualizations.

Robot motion commands derived from user input are transmitted to the robotic platform, and the resulting sensor feedback is visualized in real time as current robot pose within the GUI. In addition, the depth sensor data are rendered using the video screen or point cloud visualization elements, providing a continuous visual representation of the robot and its surrounding environment as captured by the sensor. Throughout operation, a physical and visual separation between the operator and the robot workspace is maintained, reflecting the teleoperation concept of the system.

5. User Study

The following chapter describes the conducted user study in detail. First, the objectives and research questions are stated, followed by sections on user study design, procedure, and data collection. The chapter concludes with a description of the analysis methodology used.

5.1. Objectives and Research Questions

As outlined in the introduction, the goal of this work is to evaluate the effects of different immersion-level visualization techniques on task performance during direct teleoperation. Using the novice teleoperation system described in Chapter 4, the study investigates the relationship between MR technology, point cloud environment visualizations, and operators' depth perception and task performance in a controlled experimental setup. The study focuses on direct teleoperation of an unknown object manipulation task, evaluated under varying display platform and visualization conditions.

Research Questions (RQ)

Based on the described task and experimental scope, the following research questions are addressed:

- RQ1: How does the visualization element (2D video versus 3D point cloud) affect task performance and depth perception during direct teleoperation?
- RQ2: How does the display platform (desktop monitor versus MR headset) affect task performance and depth perception during direct teleoperation?
- RQ3: How does immersive display platform and three-dimensional visualization element jointly influence task performance and depth perception during direct teleoperation?

Based on these research questions, the following hypotheses are formulated to systematically evaluate performance-related and perception-related effects.

The hypotheses address performance and perception differences for the same unknown object manipulation task under varying display platforms and visualization conditions. The study is conducted within the context of the Surface Avatar project, in which a direct teleoperation mode represents a core application, and is replicating the experimental setup to evaluate the investigated teleoperation interfaces. The hypotheses are organized into performance-related and perception-related sub-hypotheses to reflect both objective task outcomes and subjective user experience.

Task performance was assessed using quantitative metrics, while subjective perception and user experience were evaluated using post-task questionnaires.

In the final comparison (H3), the desktop monitor with video-based visualization closely resembles the previous Surface Avatar setup, while the MR headset combined with point cloud visualization represents the newly proposed setup.

Hypotheses (H)

Based on the defined research questions, the following hypotheses are formulated:

- **H1 (Visualization Element):**
  - **H1a (Performance):** The use of a 3D point cloud visualization leads to improved task performance compared to a 2D video-based visualization during direct teleoperation.
  - **H1b (Perception):** The 3D point cloud visualization provides higher perceived depth perception, spatial understanding and is preferred by users compared to the 2D video-based visualization.
- **H2 (Display Platform):**
  - **H2a (Performance):** The use of an MR headset as display platform leads to improved task performance compared to a desktop monitor.
  - **H2b (Perception):** The use of an MR headset leads to higher perceived depth perception and spatial understanding, reduced perceived workload, improved system usability, and higher overall user preference compared to a desktop monitor.
- **H3 (Comparison to the previously used project setup):**
  - **H3a (Performance):** The combination of an MR headset and point cloud visualization leads to higher task efficiency compared to a desktop monitor and video-based visualization.

- **H3b (Perception):** Compared to a desktop monitor and video-based visualization setup, the combined use of an MR headset and point cloud visualization provides higher perceived depth perception, improved spatial understanding.

The hypotheses formulated in this study are derived from the challenges and motivations outlined in Chapters 1 and 3. Prior work has highlighted effective depth perception and situational awareness as critical factors for successful direct teleoperation, particularly in manipulation tasks involving unknown objects. Theoretical considerations and related work suggest that immersive display technologies and three-dimensional representations of the remote environment can support intuitive spatial understanding [JWP22; ES24]. Accordingly, the hypotheses systematically investigate the influence of display platform and visualization elements on task performance, depth perception, and perceived workload during the described tasks.

5.2. Study Design

In the following sections, the study design is described, starting with the experimental design. Subsequently, the independent and dependent variables are defined, leading to the specification of the experimental conditions.

5.2.1. Experimental Design

The study followed a within-subject experimental design, in which each participant experienced all relevant system conditions. To mitigate learning and order effects, the sequence of conditions was varied across participants using a controlled counterbalancing strategy. A separable comparison of two visualization concepts (video screen versus point cloud) and two display platforms (desktop monitor versus MR headset) was conducted. In addition to the individual visualization concepts, a combined visualization condition incorporating all visualization elements was included. Consequently, three visualization conditions were evaluated across two display platforms, resulting in six experimental conditions. Each participant completed one task execution per condition. The task itself remained constant and involved the manipulation of an unknown object, including grasping, moving, and releasing, with accuracy as the primary performance criterion.

5.2.2. Independent and Dependent Variables

During the experimental tasks, the Independent Variables (IVs) listed in Table 5.1 are systematically varied.

Table 5.1.: Independent variables and their respective modalities

Independent Variable	**Modalities**
Visualization Element	Video stream, Point cloud, Robot model
Display platform	Monitor, MR headset

This two-level separation enables a clear association of observed effects with the respective modality of each independent variable.

The Dependent Variables (DVs) capture both objective task performance metrics and subjective user experience measures as listed in Table 5.2. These are further complemented by the derived metric task efficiency.

(a) Objective metrics

Grasp attempts
Planar target offset
Release height
Obstacle avoidance
Path length (end-effector)
Task completion time

(b) Subjective measures

Perceived depth
Spatial understanding
Task workload
System usability
Preferred visualization element
Preferred display platform

(c) Derived metrics

Task efficiency

Table 5.2.: Dependent variables grouped by metric type

5.2.3. Experimental Conditions

To achieve the two-level separation described, every run condition is a specific combination of display platform and a given set of visualization elements defined by the GUI Modes as described in Table 4.2. This leads to the following experimental conditions being used for the user study runs:

The conditions C3 and C6 differ from the other conditions, as they let the user switch freely through all three Visual Elements in the Free Focus Mode, instead of limiting the GUI to one specific Visual Element.

Table 5.3.: Experimental Conditions defining the six runs (deliberately combined independent variable modalities using the previously defined GUI modes in Table 4.2)

Condition	Display Platform	GUI Mode
C1	Monitor	Video Screen Mode
C2	Monitor	Point Cloud Mode
C3	Monitor	Free Focus Mode
C4	MR Headset	Video Screen Mode
C5	MR Headset	Point Cloud Mode
C6	MR Headset	Free Focus Mode

5.3. Study Procedure

The overall user study procedure was the same for every participant, only the order randomization of experimental conditions differed between participants. First, the participant was introduced and briefed about the framework of the user study, followed by an explanation of the teleoperation system. After a short familiarization phase, the task instructions were presented and one run for every experimental condition from Table 5.3 was conducted. Questionnaires had to be answered at different stages of the study and every feedback of the participants was noted. Short breaks were provided throughout the study to reduce fatigue. All verbal instructions were standardized and read from a predefined script (see Appendix C). Overall, the study session lasted approximately 45 minutes per participant.

Introduction and Briefing

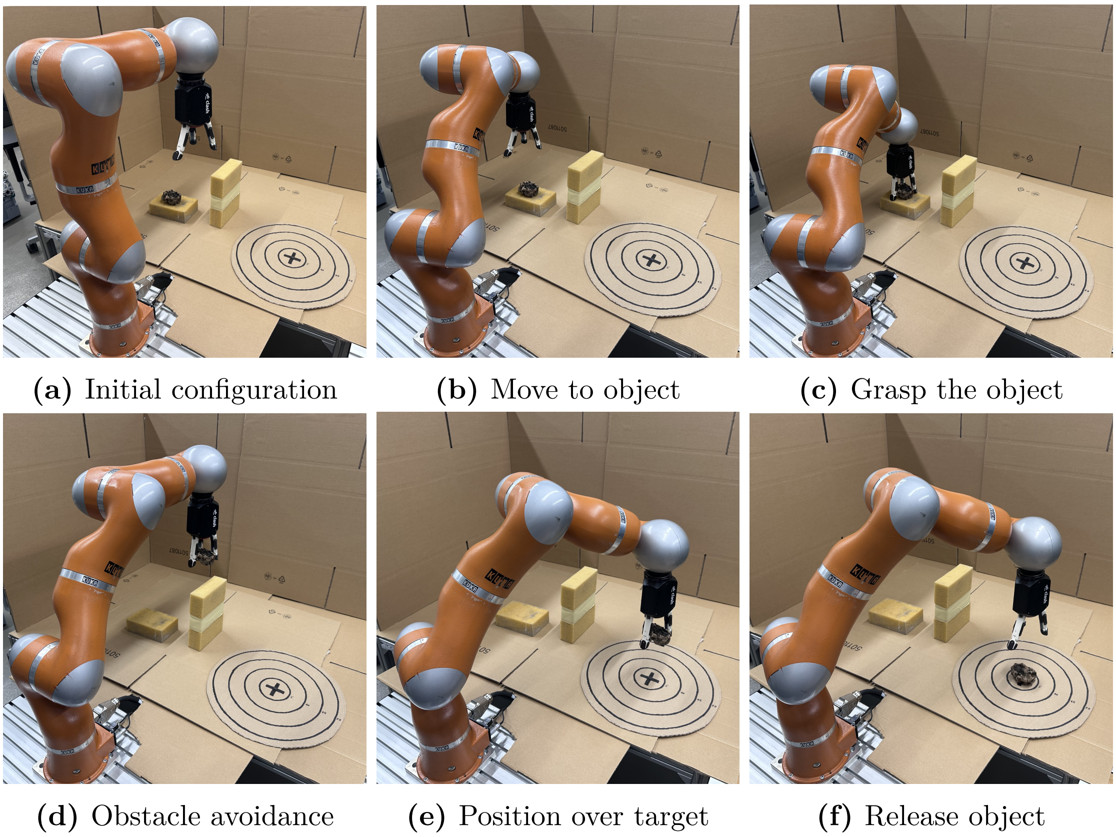
At the beginning of the study, participants were welcomed and informed about the general goal and concept of the experiment. An informed consent form was signed and basic participant information was collected on an observation sheet, consisting of age, gender and handedness. A participant identification (ID) was assigned to enable anonymous allocation of data from all collection sources to the corresponding participant. The study procedure was then explained without revealing any hypotheses.

Familiarization Phase

The system was explained in detail, focusing successively on GUI interaction, teleoperation input device and robotic platform. Any questions from the participant were answered or demonstrated. Once the system setup and interaction were understood, a few minutes of unevaluated familiarization with the dynamics of robot and GUI interaction took place. After feeling comfortable to operate the system, the actual task was introduced to the participant.

Task Instructions

A single task instruction was given which stayed the same for every experimental condition. The task consisted of a predefined object manipulation scenario. The participant was instructed to first move the robotic arm towards an object and grasp it using the gripper attached to the end of the arm. Then, the object needed to be transported over to a marked target location while avoiding obstacles along the path. Finally, the object had to be released as accurately as possible onto a marked target area. Figure 5.1 shows all task stages in sequential order using photos taken from the real experimental setup.



(a) Initial configuration **(b)** Move to object **(c)** Grasp the object

(d) Obstacle avoidance **(e)** Position over target **(f)** Release object

Figure 5.1.: All user study task stages in sequential order from **(a)** to **(f)**

After the task was explained and questions were answered, the participant was now ready for the evaluation runs.

Task Execution Runs

Six evaluation runs are conducted in total, one for every experimental condition from Table 5.3. They are divided into two phases, one for C1 to C3, the other for C4 to C6. Each phase uses one of the two display platform setups, either a desktop monitor or a mixed reality (MR) headset. The order of phases is deliberately varied per participant to mitigate learning effects influencing task performance. This means half of the participants first conducted the monitor phase followed by the mixed reality phase, and the other half in reverse.

Each phase therefore consists of three runs, one for every GUI mode described in Table 4.2. Here, deliberate order variation between participants was applied again for the reasons described above. At the end of each phase, questionnaires about the specific phase were filled out. They addressed depth perception, spatial understanding, perceived workload and usability.

Before each run, the robot was reset to its idle position, always creating the same initial conditions. Then the automatic Unity tracking backend was started, which marked the beginning of the run. From here, the participant operated the system to execute the task, during which additional data was recorded manually. The run concluded if the object release occurred successfully or the third grasping attempt had failed. Then the Unity backend tracking was stopped and the data saved.

After all runs had been completed, a final questionnaire captured the overall system evaluation, encompassing participant preferences regarding display platforms and GUI elements, as well as overall system feedback.

Order Randomization and Counterbalancing

As already stated, to mitigate learning effects, the order of display platform phases was counterbalanced across participants. Within each phase, the order of GUI Modes was also randomized. This ensured that no experimental condition consistently benefited from prior task experience.

Safety Measures

As this teleoperation system included a freely moving physical robotic arm, safety measures were crucial. One major concern was potential collisions between the robot and hard surfaces, which could damage both the robot and its surroundings. Therefore, the experimenter closely monitored the scene and was always ready to use an emergency button to instantly shut down the entire robotic system. Another issue to be avoided was the alignment of multiple robot joint axes, as such configurations are known as kinematic singularities and can result in unstable joint motions when using inverse kinematics. [SK16, pp. 32–33]

5.4. Participants

A total of 12 participants took part in the study, all of whom were male and right-handed. Participants' ages ranged from 22 to 49 years (mean ± standard deviation: 28.92 ± 7.32 years). Recruitment took place within the Robotics and Mechatronics Center (RMC) at the DLR site in Oberpfaffenhofen. The participants consisted of employees, researchers and students, primarily from engineering-related fields. Therefore, most participants exhibited a strong technical background and prior experience with robotic systems, including experience with video-based teleoperation in some cases. Demographic characteristics and prior experience with mixed reality or teleoperation varied. Participation was voluntary, and the ethics committee classified the study as safe.

Participants were required to meet the following criteria:

- Basic technical literacy
- No physical or sensory impairments affecting visual perception or interaction with the system input devices
- Normal or corrected-to-normal vision
- Ability to interpret 3D visualizations
- No prior involvement in the development of the system or close collaboration with the development process
- No prior knowledge about the exact intentions of the study
- No prior experience with the specific system
- Ability to understand and operate the system after a short familiarization phase

5.5. Data Collection and Metrics

Data collection during the user study was done in three different ways. All objective metrics that could be automatically and reliably captured were digitally recorded by the backend tracking system in a time-synchronized manner. Metrics that could not be digitally acquired were assessed manually through direct physical measurement or observation and documented on an observation sheet for each run, as shown in Figure A.3. All subjective measures were collected via questionnaires answered by the participants, complemented by experimenter notes on participants' verbal comments during each run.

Overview of DVs data collection

Table 5.4 shows the data collection overview of DVs during the user study, including measurement instrument, unit/scale and tolerance (measurement error margin) for every tracked DV separately.

Table 5.4.: Data Collection of DVs: measurement instruments with respective units/scales and tolarances

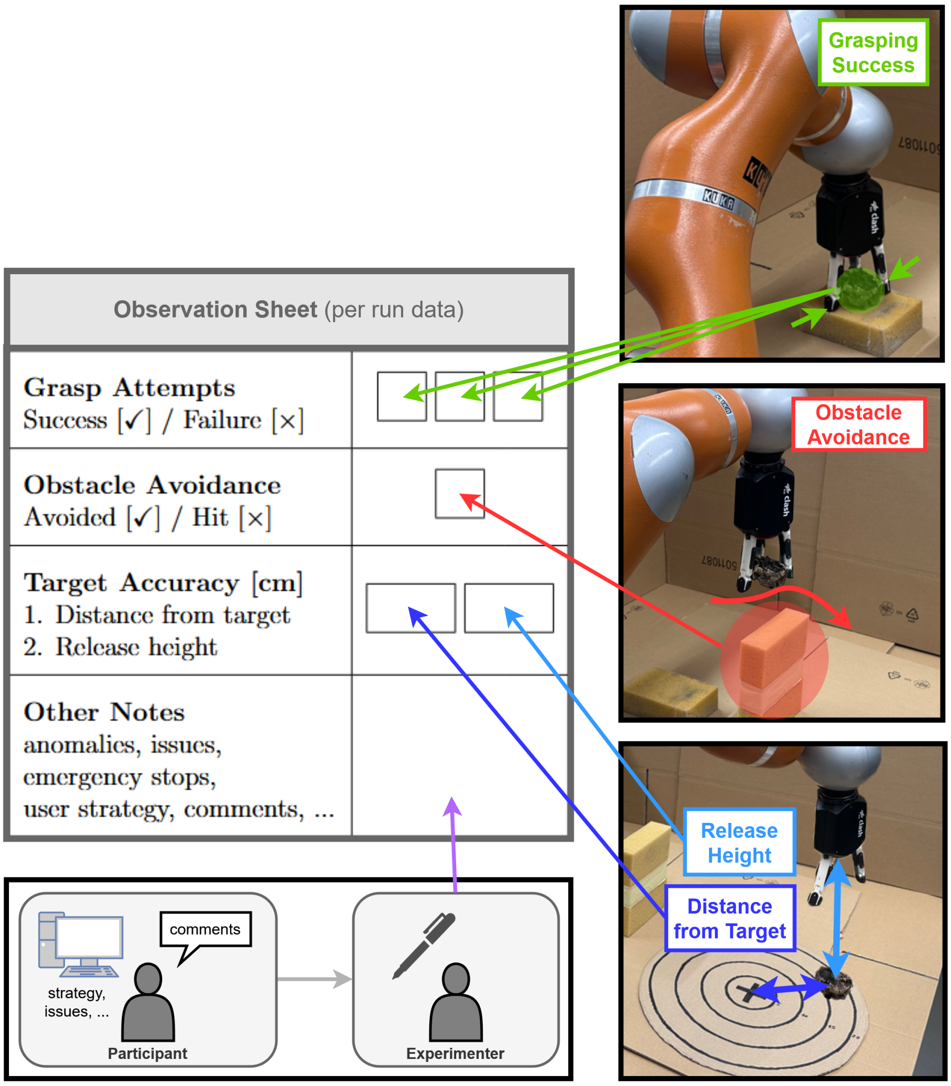
DVs	Instrument	Unit / Scale (± Tolerance)
Grasp attempts	Backend logging	Count (max. 3)
Obstacle avoidance	Observation	Binary (yes/no)
Planar target offset	Manual measurement	cm ±1 cm
Release height	Manual measurement	cm ±1 cm
Path length	Backend logging	m $\pm 10^{-9}$ m
Task completion time	Backend logging	s ±0.2 s
Perceived depth	Custom questionnaire (DPQ)	Likert-scale (1–7)
Spatial understanding	Custom questionnaire (DPQ)	Likert-scale (1–7)
Task workload	Standardized questionnaire (TLX)	Subjective rating (0–100)
System usability	Standardized questionnaire (SUS)	Likert-scale (1–5)
Preferred visualization element	Custom questionnaire (SEQ)	Categorical (nominal)
Preferred display platform	Custom questionnaire (SEQ)	Categorical (nominal)

5.5.1. Digitally Recorded Run Data

The automatic backend tracking system in Unity was started at the beginning of every run and stopped at the run's conclusion. It provided precise timestamps and continuous sampling at fixed intervals, enabling subsequent temporal analysis. The objective metrics captured this way included the central visualization element used, grasping/releasing events, end-effector position, and joint-values. Indirectly, the final timestamp also represents task completion time, in case a successful grasp has occurred. While the joint values and grasping/releasing events were captured from actual robot telemetry data, the end-effector position was recreated by a URDF robot model within Unity. This model used the joint value telemetry data to configure itself in the same pose as the real robot. The resulting end-effector position relative to the robot's base in virtual 3D space was then tracked by the system, with one virtual unit corresponding to one meter in reality.

5.5.2. Manually Recorded Run Data

For recording metrics directly from the real physical system, an observation sheet was completed by the experimenter during each run. Grasp attempt success, obstacle avoidance, and positioning accuracy were all captured this way. While the first two metrics were recorded as binary success or failure outcomes, the third metric was obtained through manual measurements. Specifically, the robot end-effector height above the target plane after object release and the planar distance between the object's contact point and the target center on the target plane were measured. Additionally, any anomalies, issues, or comments during the run were noted in the observation sheet. Figure 5.2 shows the observation sheet and its recorded metrics.



Observation Sheet (per run data)

Grasp Attempts Success [✓] / Failure [×]	□ □ □
Obstacle Avoidance Avoided [✓] / Hit [×]	□
Target Accuracy [cm] 1. Distance from target 2. Release height	□ □
Other Notes anomalies, issues, emergency stops, user strategy, comments, ...	

Figure 5.2.: Manually recorded run data in observation sheet: grasping successes, obstacle avoidance, target accuracy (planar target offset/distance from target, release height), and experimenter notes

5.5.3. Subjectively Reported Questionnaire Data

The subjective measures from Table 5.2b were captured by four different questionnaires. First, the NASA Task Load Index (TLX) and the System Usability Scale (SUS) were employed as well-established and widely used standard instruments for assessing perceived workload and system usability, respectively. They are described in detail by the following paragraphs. Second, two additional custom questionnaires were designed specifically for this study. One assessed depth perception and spatial understanding separately for each experimental condition, while the other provided a final system evaluation, including user preferences, likes and dislikes, as well as overall impressions and comments. All questionnaires were implemented in HTML and completed using a web browser. Once completely filled out, they were saved locally on the computer in JSON format, with all input data stored accordingly.

The **NASA Task Load Index** (TLX) is a commonly used questionnaire for self-reported subjective workload evaluation originally developed by NASA. It captures perceived task load across six dimensions: Mental Demand, Physical Demand, Temporal Demand, Performance, Effort, and Frustration. In this study, the unweighted Raw-TLX score was used, in which participants directly rated each workload dimension on a continuous scale between 0 and 100. Higher values generally indicate greater perceived workload. The collected ratings represent subjective workload scores for each participant and experimental condition. The NASA-TLX does not define normative thresholds or reference values for these scores, meaning workload ratings are interpreted in a relative manner, focusing on comparisons between experimental conditions and across workload dimensions. Figure A.6 shows the TLX questionnaire implemented for this study. [HS88; Har06]

The **System Usability Scale** (SUS) is a standardized questionnaire consisting of ten statements (items) to address perceived system usability. Each item is rated on a Likert scale ranging from 1 (*strongly disagree*) to 5 (*strongly agree*). Instead of interpreting per item, the SUS is designed to evaluate an aggregated score over all items, resulting in a single usability index per participant between 0 and 100. The SUS deliberately combines positively and negatively worded items. Positively formulated items (odd-numbered: $1, 3, 5, 7, 9$) cover favorable usability characteristics, where higher agreement indicates better usability. Negatively formulated items (even-numbered: $2, 4, 6, 8, 10$) capture difficulties with usability such as complexity or inconsistency, meaning higher agreement on these items indicates poorer usability. This alternation between positive and negative wording is a core design choice of the SUS to reduce response biases. It also always combines complementary aspects of system usability to form the final score. Figure A.7 shows the SUS questionnaire implemented for this study. [Bro95; BKM08]

The **Depth Perception Questionnaire** (DPQ) was specifically designed for the user study to assess subjective depth perception, 3D spatial understanding, and the effects of the different GUI modes on those measures. It used a Likert scale from 1 (*very poorly/not intuitive at all*) to 7 (*very well/very intuitive*). The items were arranged into three blocks with two questions each. Every block assessed one of the three GUI-Modes from Table 4.2, where the two single element modes (video screen and point cloud) had identical questions to allow comparison in the subsequent statistical analysis. Figure A.4 shows the questionnaire implementation used for the study. The exact questions for each block were:

1. Block: **Video Screen Mode** (C1 / C4)
    - How well were you able to perceive depth in the scene (for example which elements were closer or farther away)?
    - How intuitive was your understanding of the 3D spatial layout during execution?
2. Block: **Point Cloud Mode** (C2 / C5)
    - *questions identical to 1. Block*
3. Block: **Free Focus Mode** (C3 / C6)
    - How much did the ability to switch between different views help you complete the task?
    - Compared to using a single interface in the first part, how much did the additional information shown in the smaller side windows help you perform the task?

The **System Evaluation Questionnaire** (SEQ) was specifically designed for the user study to assess user preferences for display platform and visualization element. It also included overall system feedback, such as positive and negative aspects, as well as a free comments section. While the user preferences were dropdown elements to choose from predefined answers, the system feedback section consisted of text fields to freely enter their thoughts. Figure A.5 shows the questionnaire implementation used for the study. The section's questions were:

1. Block: **User Preferences**
    - Which visual element did you find most helpful for accomplishing the task? *(Video Stream, Point Cloud, Robot Model)*

- Which display platform did you prefer for performing the task?
  (Monitor, Mixed Reality)

- Would an overlaid view combining the point cloud and the 3D robot model have been more beneficial than showing them separately?
  (yes, no)

2. Block: **System Feedback**

   - What did you like about the system? Please list up to three positive aspects.
     (textbox)

   - What did you dislike about the system? Please list up to three negative aspects.
     (textbox)

   - Do you have any additional comments, suggestions, or feedback about the system?
     (textbox)

5.5.4. Derived Metrics: Task Efficiency

For this specific task case, a definition for task efficiency TE is constructed as S (numerator) over C (denominator), where $S = A \cdot O \cdot G$ represents the success term and $C = L \cdot T$ defines the cost term. It combines target accuracy, obstacle avoidance, grasp attempts, path length, and task duration in one derived metric for statistical analysis. To avoid excessively small values of TE close to zero, an additional linear scaling factor s_{TE} is applied.

$$\mathrm{TE} = s_{TE} \cdot \frac{S}{C} = s_{TE} \cdot \frac{A \cdot O \cdot G}{L \cdot T} \ , \quad s_{TE} = 10000 \tag{5.1}$$

A represents the target accuracy and increases TE. It uses a Euclidean distance error e by combining planar target offset d and release height h, as shown in Figure 5.2. The release height is weighted by $w_e < 1$ in order to shift the influential focus on d as primary success indicator.

$$A = \frac{1}{1+e} \ , \quad e = \sqrt{d^2 + (w_e \cdot h)^2} \ , \quad w_e = 0.5 \tag{5.2}$$

The obstacle avoidance was represented as O and is an absolute indicator for success or failure of the task, rendering $\mathrm{TE} = 0$ if the obstacle was not avoided.

$$O = \begin{cases} 1, & \text{if obstacle avoided,} \\ 0, & \text{if obstacle hit.} \end{cases} \tag{5.3}$$

G is the effectiveness score regarding attempts until successful grasp n_g. Lower attempts mean higher score and it is capped at a maximum of 3 attempts, after which the run is considered failed. Then G becomes zero.

$$G = \begin{cases} \dfrac{1}{n_g}, & n_g \leq 3, \\ 0, & \text{if third attempt failed,} \end{cases} \tag{5.4}$$

Finally, the cost is represented by the path length L, describing the total travel distance of the end-effector, and by the task duration T. Both quantities reduce the task efficiency TE as their values increase. The path length L is computed by summing the Euclidean distances between consecutive end-effector positions $\mathbf{p}$ over all recorded samples of a run. The task duration T is obtained directly from the final recorded timestamp t_{end}.

$$L = \sum \|\Delta \mathbf{p}\| \ , \quad T = t_{\text{end}} \tag{5.5}$$

5.6. Data Analysis Methodology

In this section the statistical analysis methods to evaluate metric data and the hypothesis testing methodology are presented. This includes analysis procedures for both the SUS an the TLX questionnaire as well as hypothesis testing methodology.

Descriptive Statistics

For statistical analysis, the arithmetic mean with standard deviation (M ± SD) is used to summarize central tendency and variability of relevant direct variables across participants. Bar plots with error bars are used to illustrate the data and provide a general overview of results. Measurements and Likert scale ratings are processed and visualized that way. Although Likert scale data is strictly ordinal, meaning without fixed or well-defined distances between the values, the arithmetic means and standard deviations enable straightforward comparability and interpretation of the data. All reported values are therefore intended to provide an overview of central tendency and variability rather than to imply interval-scale measurement.

TLX Analysis Procedure

The NASA-TLX is evaluated using the unweighted Raw-TLX approach, in which the overall workload score is computed as the arithmetic mean of the six dimension ratings per participant [Har06].

In this study, the performance dimension is specifically altered. While higher ratings on most TLX dimensions indicate higher perceived workload, the performance item in the official TLX document from [Div] is phrased in a way that higher values reflect better task performance and thus indicate lower workload. Therefore, to create overall consistency across all dimensions, the performance scale on the questionnaire is inverted for this study. All TLX analyses are conducted using these transformed performance values, meaning higher scores uniformly indicate higher perceived workload on every dimension.

For each display platform, aggregated TLX scores per participant are obtained by averaging all dimensions with equal weighting, as commonly used for calculating the Raw TLX score [Har06]. A paired-samples t-test is then used to compare the two display platforms for statistical significance, as described in more detail in the following section on hypothesis testing. In addition, mean dimension-level results are visualized to provide insights into specific sources of workload per display platform.

SUS Analysis Procedure

The System Usability Scale (SUS) is administered and scored following the standard procedure described by Brooke in [Bro95]. Raw item responses are transformed according to the SUS scoring rules, with positively and negatively worded items treated differently, and then summed and scaled. The resulting values are aggregated into a single SUS score per participant, resulting in a single composite usability score between 0 and 100. For better understanding a reference score of 68 is used, as this value reflects the average usability score reported in large-scale empirical analyses of SUS data. [BKM08]

For each display platform, SUS scores are computed per participant followed by a paired-samples t-test to compare the two display platforms for statistical significance. In addition, mean item-level results are visualized to provide an overview comparison between display platforms for every item, where positively and negatively worded items are visually distinct.

Hypothesis Testing

Paired-samples t-tests are conducted to analyze statistical differences between metrics. For all statistical tests, a significance level of $\alpha = 0.05$ is applied. Since each participant evaluated all conditions, the analysis is based on within-subject comparisons.

For each participant, scores from two specific conditions to compare are treated as paired observations. The null hypothesis assumes no difference between conditions, and the alternative hypothesis assumes a non-zero mean difference. Statistical significance is examined using the resulting p-values. In addition, Cohen's d_z is reported to describe effect sizes of observed differences independent of sample size.

When analyzing individual factors (IVs), the effects of display platform and GUI mode are considered separately. As each experimental run condition combines both factors, the recorded data per participant containes one entry for each display platform GUI mode combination. To test the effect of a single factor, the respective other factor is first averaged within each participant. This ensures that all levels of the secondary factor contribute equally and avoid bias due to unequal numbers of runs. Paired-samples t-tests are then applied to these aggregated values.

For comparisons between experimental conditions (specific factors combinations), no averaging is needed, as every participant experienced each experimental condition exactly once. In these cases, paired-samples t-tests are conducted directly on the corresponding conditions per participant. In case one or both conditions to compare contained no data for a participant (e.g. due to task failure), the sample size is simply reduced by one, as no within-subject comparison is possible for this participant.

To validate the hypotheses, paired differences are computed for each participant i by subtracting the two condition scores of interest to get within-subject deltas ($\Delta_i = A_i - B_i$). The corresponding Δ plots visualize the mean of those deltas across participants as bar plots with error bars indicating standard deviation. Based on the paired differences, statistical significance is evaluated by paired-samples t-tests. The number of paired observations (n), t-values, p-values, and effect sizes (Cohen's d_z) are reported directly in the plots.

Qualitative Data Processing

Participant feedback on specific subjectively perceived aspects during the study, as well as observations by the experimenter, are summarized as qualitative data. Participant feedback was collected through free-text sections in questionnaires and verbal comments from participants, which were noted by the experimenter during the user study. Participant feedback is divided into negative aspects of the system, positive aspects of the system, and comments and suggestions for improvement. In addition, experimenter observations are presented separately. The qualitative findings are grouped into conceptually similar statements or incidents and presented accordingly. The exact wording is not reproduced, only the essence of each statement is conveyed.

6. Results

This chapter presents the results of the user study with respect to the research questions and is structured into quantitative, questionnaire-based, and qualitative findings.

Quantitative results are visualized using diagrams that show the arithmetic mean ± standard deviation of the respective performance metrics, as described in Section 5.6. These plots illustrate general trends and differences between conditions.

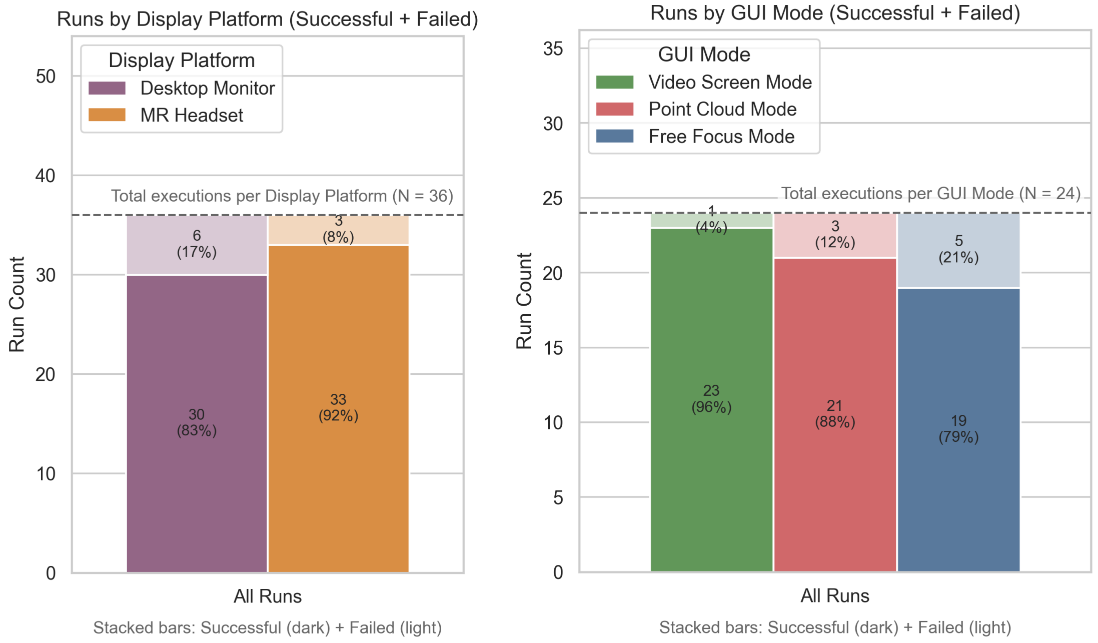
For hypothesis-driven analyses, Δ-plots are used to visualize paired condition differences in accordance with the hypothesis testing procedure described in Section 5.6. Each Δ-plot shows the mean paired difference as the bar height and the standard deviation as error bars. The plot also directly reports the number of paired observations (n), the associated t-value, p-value, and effect size (Cohen's d_z). This illustrates the direction and magnitude of condition effects both visually and numerically within the Δ-plot. Where appropriate, the numerical results are additionally mentioned in the corresponding paragraph.

A consistent global color scheme is used throughout all figures to visually distinguish between the three GUI modes and the two display platforms. The GUI-Modes are encoded using the following colors: Video Screen Mode (green), Point Cloud Mode (red), and Free Focus Mode (blue). Display platforms are differentiated using purple for the Desktop Monitor and orange for the MR Headset. Combined display platform and GUI mode conditions are visualized as striped patterns showing both corresponding colors at the same time.

6.1. Quantitative Results

Every participant completed three runs on each display platform, one for every GUI mode. This resulted in 36 runs per display platform or 24 runs per GUI mode. Out of the total 72 runs, 63 (87.5%) successfully executed the full task while 9 failed to grasp the object within the defined three attempts resulting in no data after grasping for those runs. Figure 6.1 shows the split into successful (darker) and failed (lighter) runs for the total execution counts by different conditions as stacked bars, display platform in (a) and GUI-Mode in (b).

The execution order of runs for each participant and its effect on task success is visualized as stacked bars in Figure 6.2.



(a) Total counts by display platform

(b) Total counts by GUI mode

Figure 6.1.: Overall execution success by run conditions: Successful (dark) + Failed (light)

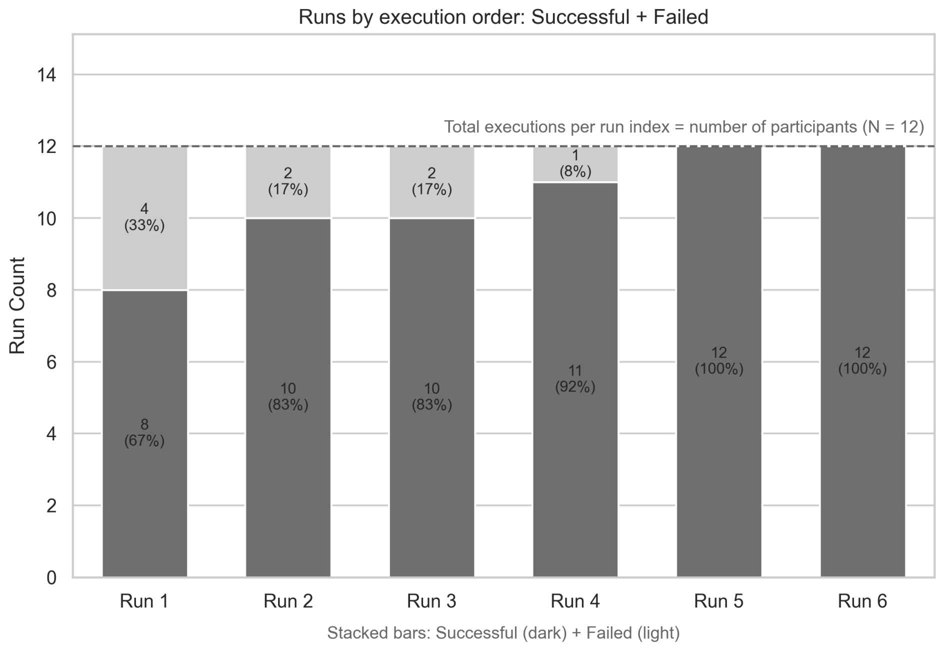


Figure 6.2.: Overall execution success by run order: Successful (dark), Failed (light)

Grasp Attempts

Results for the amount of grasp attempts until success or third failure are shown in the following section. For a holistic overview of the data, firstly the interaction between display platform and GUI mode is illustrated in Figure 6.3 and secondly the combined display platform with GUI mode conditions presented in Figure 6.4.

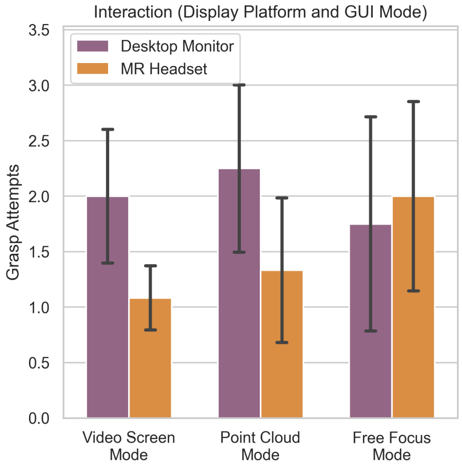


Figure 6.3.: Grasp Attempts: Interaction between display platform and GUI mode

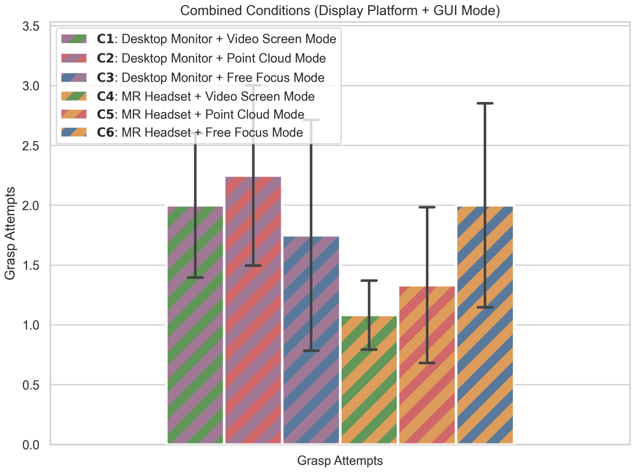
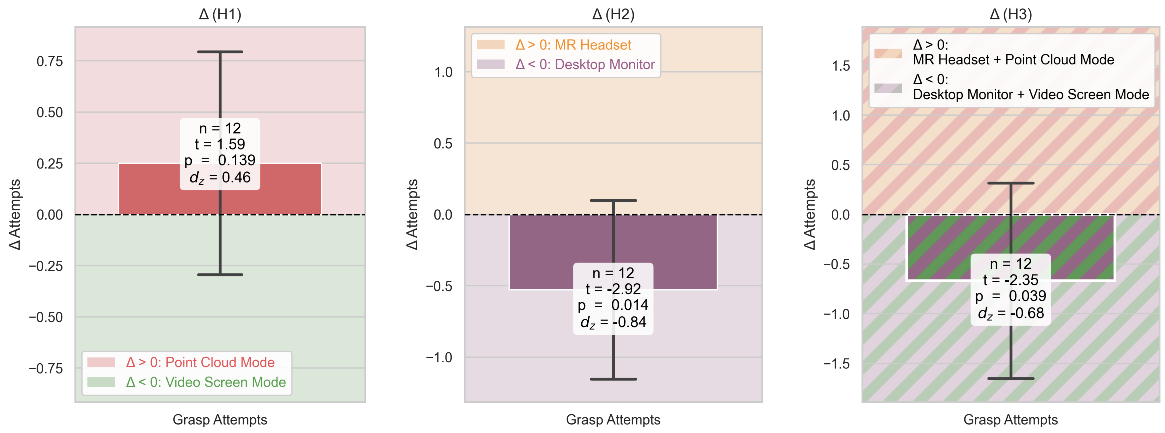


Figure 6.4.: Grasp Attempts: Combined conditions of display platform + GUI mode

Figure 6.5a shows the paired Δ comparison for hypothesis **H1**a (GUI modes), comparing the Video Screen mode to the Point Cloud mode averaged across display platforms. The comparison revealed no statistically significant difference in the number of grasp attempts.

Figure 6.5b presents the paired Δ comparison for hypothesis **H2**a (display platforms), comparing the MR Headset to the Desktop Monitor averaged across GUI modes. The comparison revealed a statistically significant difference in the amount of grasp attempts, with a large effect size, indicating fewer grasp attempts for the MR Headset [$t(11) = -2.92$, $p = 0.014$, $d_z = -0.84$].

Figure 6.5c illustrates the paired Δ comparison for hypothesis **H3**a (combined conditions), comparing MR Headset with Point Cloud mode to the Desktop Monitor with Video Screen mode. The comparison revealed a statistically significant difference in the amount of grasp attempts, with a medium effect size, indicating fewer grasp attempts for the MR Headset with Point Cloud mode combination [$t(11) = -2.35$, $p = 0.039$, $d_z = -0.68$].



(a) **H1**a (Point Cloud vs. Video Screen; display platform averaged per participant) (b) **H2**a (MR Headset vs. Desktop Monitor; GUI mode averaged per participant) (c) **H3**a (MR Headset + Point Cloud vs. Desktop Monitor + Video Screen)

Figure 6.5.: Grasp Attempts: paired mean differences (Δ) ± SD across participants in H1 (a), H2 (b), and H3 (c)

Target Accuracy

Target accuracy is used here as an umbrella term for the two spatial performance measures planar target offset and release height. To provide an initial overview of the data, Figure 6.6 illustrates the interaction between display platform and GUI mode for both measures. Figure 6.7 further presents the corresponding combined display platform with GUI mode conditions, allowing direct comparison across all experimental configurations.

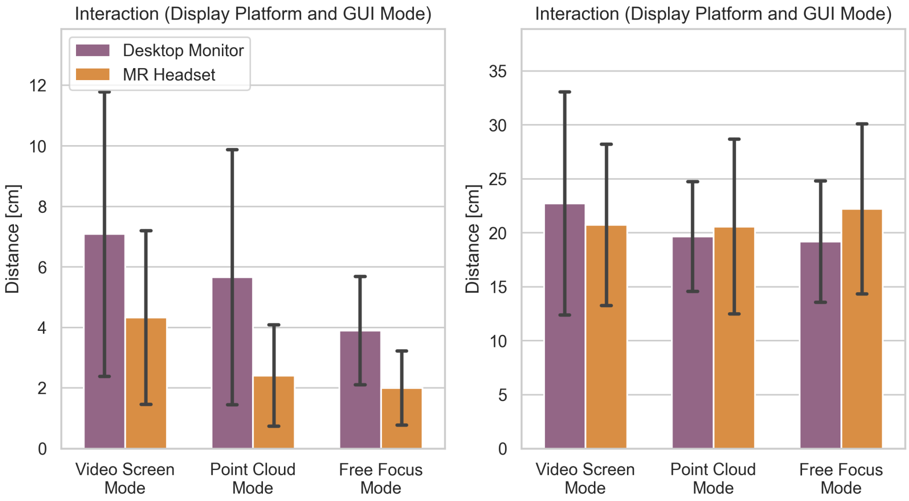


Figure 6.6.: Target Accuracy: Interaction between display platform and GUI mode (left: Planar target offset, right: Release height)

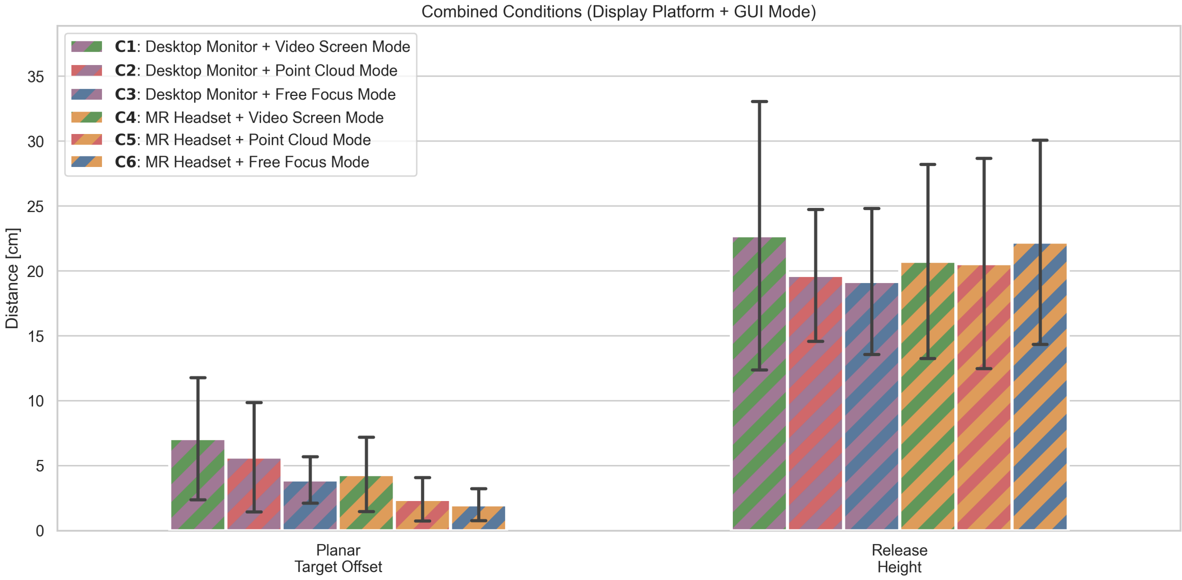
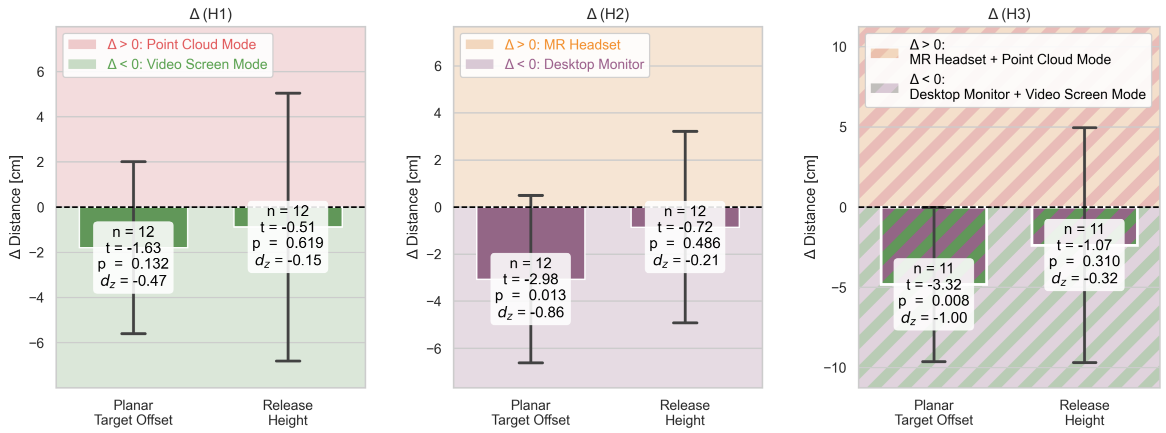


Figure 6.7.: Target Accuracy: Combined conditions of display platform + GUI mode (left: Planar target offset, right: Release height)

Figure 6.8a shows the paired Δ comparison for hypothesis **H1**a (GUI modes), comparing the Video Screen mode to the Point Cloud mode averaged across display platforms. The comparison revealed no statistically significant differences in both planar target offset and release height.

Figure 6.8b presents the paired Δ comparison for hypothesis **H2**a (display platforms), comparing the MR Headset to the Desktop Monitor averaged across GUI modes. The comparison revealed a statistically significant difference in planar target offset, with a large effect size, indicating less offset for the MR Headset [$t(11) = -2.98$, $p = 0.013$, $d_z = -0.86$]. No statistically significant difference was observed for release height.

Figure 6.8c illustrates the paired Δ comparison for hypothesis **H3**a (combined conditions), comparing MR Headset with Point Cloud mode to the Desktop Monitor with Video Screen mode. The comparison revealed a statistically significant difference in planar target offset, with a large effect size, indicating a smaller offset for the MR Headset with Point Cloud combination [$t(10) = -3.32$, $p = 0.008$, $d_z = -1.00$]. No statistically significant difference was observed for release height.



(a) **H1**a (Point Cloud vs. Video Screen; display platform averaged per participant)

(b) **H2**a (MR Headset vs. Desktop Monitor; GUI mode averaged per participant)

(c) **H3**a (MR Headset + Point Cloud vs. Desktop Monitor + Video Screen)

Figure 6.8.: Planar target offset and release height: paired mean differences (Δ) ± SD across participants in H1 (a), H2 (b), and H3 (c)

Task Efficiency

Results of the derived metric task efficiency, as defined in Section 5.5, are presented next. For a general overview, the interaction between display platform and GUI mode is shown in Figure 6.9, while Figure 6.10 presents the combined conditions (display platform with GUI mode).

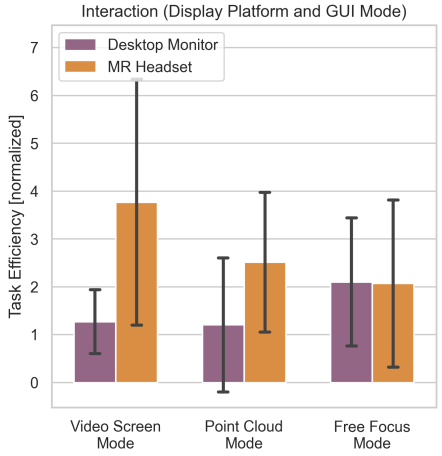


Figure 6.9.: Task Efficiency: Interaction between display platform and GUI mode

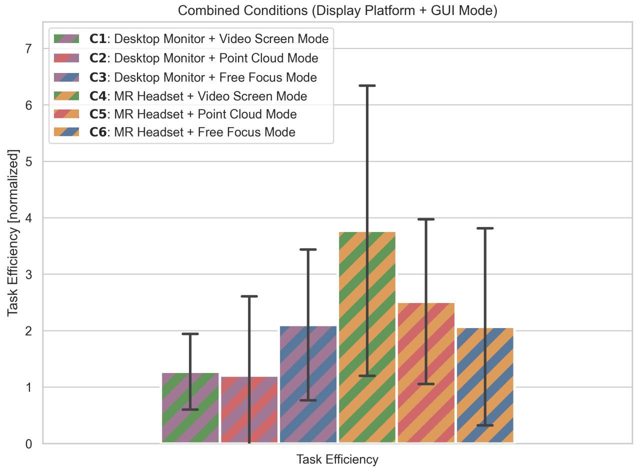
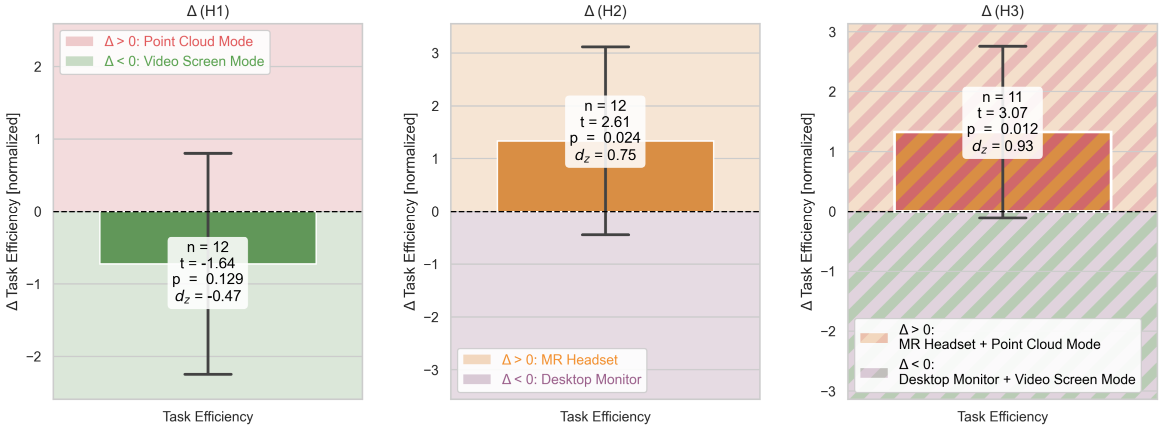


Figure 6.10.: Task Efficiency: Combined conditions of display platform + GUI mode

Figure 6.11a shows the paired Δ comparison for hypothesis **H1**a (GUI modes), comparing the Video Screen mode to the Point Cloud mode averaged across display platforms. The comparison revealed no statistically significant difference in task efficiency.

Figure 6.11b presents the paired Δ comparison for hypothesis **H2**a (display platforms), comparing the MR Headset to the Desktop Monitor averaged across GUI modes. The comparison revealed a statistically significant difference in task efficiency, with a medium effect size in favor of the MR Headset [$t(11) = 2.61$, $p = 0.024$, $d_z = 0.75$].

Figure 6.11c illustrates the paired Δ comparison for hypothesis **H3**a (combined conditions), comparing MR Headset with Point Cloud mode to the Desktop Monitor with Video Screen mode. The comparison revealed a statistically significant difference in task efficiency, with a large effect size in favor of the MR Headset with Point Cloud mode combination [$t(10) = 3.07$, $p = 0.012$, $d_z = 0.93$].



(a) **H1**a (Point Cloud vs. Video Screen; display platform averaged per participant)

(b) **H2**a (MR Headset vs. Desktop Monitor; GUI mode averaged per participant)

(c) **H3**a (MR Headset + Point Cloud vs. Desktop Monitor + Video Screen)

Figure 6.11.: Task efficiency: paired mean differences (Δ) ± SD across participants in H1 (a), H2 (b), and H3 (c)

Obstacle Avoidance

The obstacle avoidance results showed that no collisions occurred under any condition. Accordingly, all participants successfully navigated around the obstacle in every run, resulting in a 100% obstacle avoidance rate.

Summary of quantitative metric results

To simplify interpretation for the subsequent discussion chapter, Table 6.1 provides a summarized overview of the statistically significant findings ($p < \alpha = 0.05$) from the quantitative metrics analysis, sorted by hypotheses. For each metric, the table lists the associated hypothesis, exact p-value, and standardized effect size (d_z). Positive effect sizes ($d_z > 0$) indicate higher metric values for the first condition of the respective comparison, as defined for each hypothesis below:

- **H1**: GUI mode (point cloud versus video screen)
- **H2**: Display platform (MR headset versus desktop monitor)
- **H3**: Combined condition (MR headset + point cloud versus desktop monitor + video screen)

Table 6.1.: Summary of only the statistically significant quantitative findings for the hypotheses

Hypothesis	**Metric**	**p-value**	**Effect Size (d_z)**
H2a	Grasp Attempts	$p = 0.014$	$d_z = -0.84$ (large)
H2a	Planar Target Offset	$p = 0.013$	$d_z = -0.86$ (large)
H2a	Task Efficiency	$p = 0.024$	$d_z = 0.75$ (medium)
H3a	Grasp Attempts	$p = 0.039$	$d_z = -0.68$ (medium)
H3a	Planar Target Offset	$p = 0.008$	$d_z = -1.00$ (large)
H3a	Task Efficiency	$p = 0.012$	$d_z = 0.93$ (large)

6.2. Questionnaire Results

For the conducted questionnaires, each of the 12 participants filled the DPQ, TLX and SUS exactly once for each of the two display platforms. This results in a total of 24 data points for each of the mentioned questionnaires, 12 under every platform condition.

Depth Perception Questionnaire (DPQ)

The DPQ results on depth perception and spatial understanding are presented in the following section. The first two diagrams show mean Likert-scale ratings from 1 to 7 for each of the six questions across the data set. Different shades of the base colors defined at the beginning of this chapter are used to indicate which GUI mode each of the six questions targets. Figure 6.12 shows mean ratings by display platform while indicating the GUI mode allocation by color.

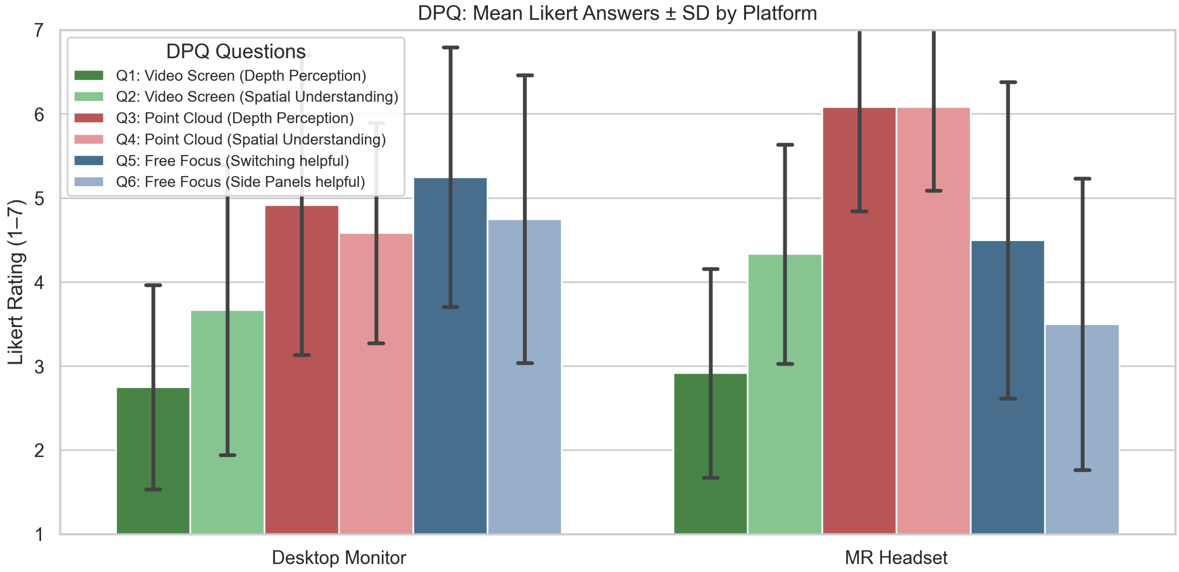


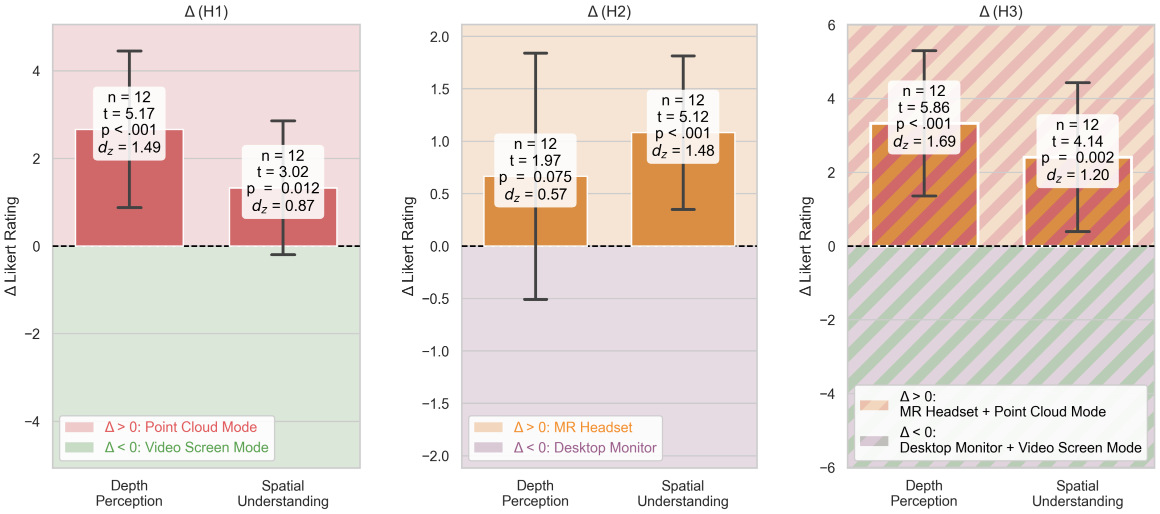
Figure 6.12.: DPQ: Mean Likert answers by display platform with color-coded GUI mode

The Likert-scale rating differences in self-reported depth perception and spatial understanding under the hypotheses conditions are presented in Figure 6.13.

Figure 6.13a shows the paired Δ comparison for hypothesis **H1**b (GUI modes), comparing the Video Screen mode to the Point Cloud mode averaged across display platforms. The comparison revealed a statistically significant difference in both depth perception and spatial understanding, with a large effect size for both in favor of the point cloud mode $[t(11) = 5.17,\ p < 0.001,\ d_z = 1.49]$, $[t(11) = 3.02,\ p = 0.012,\ d_z = 0.87]$.

Figure 6.13b presents the paired Δ comparison for hypothesis **H2**b (display platforms), comparing the MR headset to the desktop monitor averaged across GUI modes. The comparison revealed no statistically significant difference in depth perception but a statistically significant difference in spatial understanding, with a large effect size in favor of the MR headset $[t(11) = 5.12,\ p < 0.001,\ d_z = 1.48]$.

Figure 6.13c illustrates the paired Δ comparison for hypothesis **H3**b (combined conditions), comparing MR Headset with Point Cloud mode to the Desktop Monitor with Video Screen mode. The comparison revealed a statistically significant difference in both depth perception and spatial understanding, with a large effect size for both in favor of MR headset + point cloud mode combination [$t(11) = 5.86$, $p < 0.001$, $d_z = 1.69$], [$t(11) = 4.14$, $p = 0.002$, $d_z = 1.20$].



(a) **H1**b (Point Cloud vs. Video Screen; display platform averaged per participant)

(b) **H2**b (MR Headset vs. Desktop Monitor; GUI mode averaged per participant)

(c) **H3**b (MR Headset + Point Cloud vs. Desktop Monitor + Video Screen)

Figure 6.13.: Depth perception and spatial understanding: paired mean differences (Δ) ± SD across participants in H1 (a), H2 (b), and H3 (c)

System Usability Scale (SUS)

Results of the SUS questionnaire are presented in this section. Figure 6.14 shows the mean Likert-scale rating from 1 to 5 split by display platform. Positively formulated questions (odd-numbered) are represented in a darker shade while the negatively formulated questions (even-numbered) are shown in a lighter shade of the corresponing display platform.

The mean SUS score across participants is calculated to be 67.1 for desktop monitor and 82.3 for MR headset. The paired Δ comparison for hypothesis **H2**b (display platforms) revealed a statistically significant difference in system usability, with a large effect size in favor of the MR headset [$t(11) = 3.79$, $p = 0.003$, $d_z = 1.09$]. Both results are visualized in Figure 6.15, the reference line at SUS = 68 indicates the average usability score reported in large-scale empirical analyses of SUS data [BKM08].

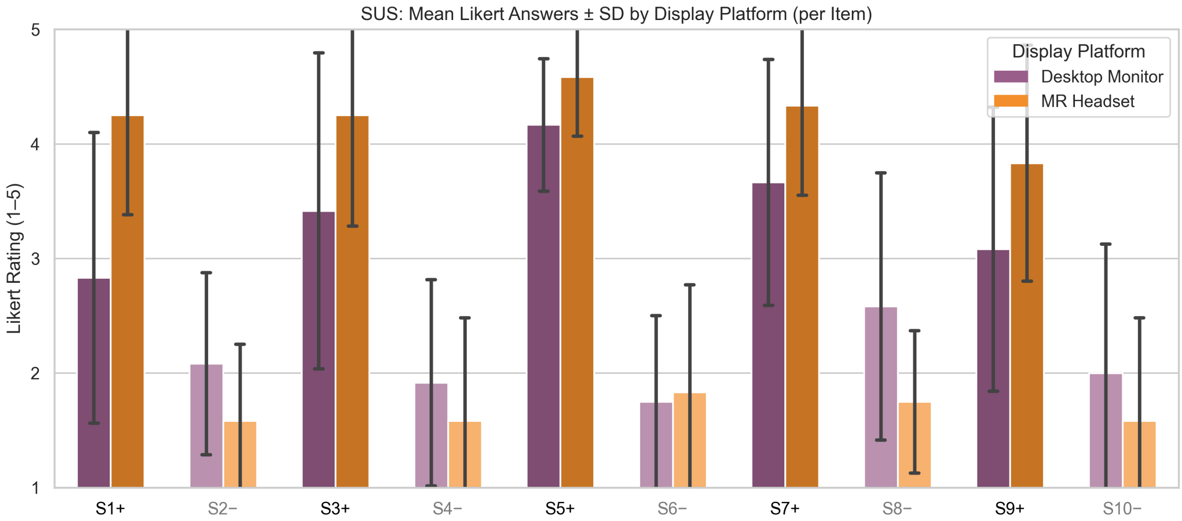
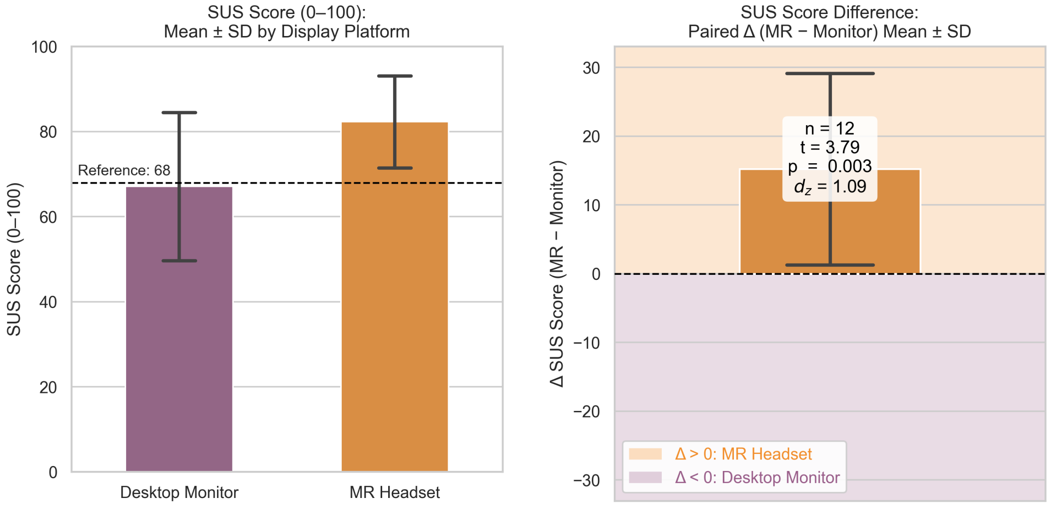


Figure 6.14.: SUS: mean Likert answers by display platforms with positively formulated questions (darker) and negatively formulated questions (lighter))



(a) SUS Score (0-100) by display platform

(b) SUS Score difference (Δ) ± SD across participants for **H2**b (display platforms)

Figure 6.15.: Usability: overview of the calculated SUS Score data

Task Load Index (TLX)

The raw TLX questionnaire results are presented in this section. Figure 6.16 shows the mean rating between 0 and 100 split by display platform.

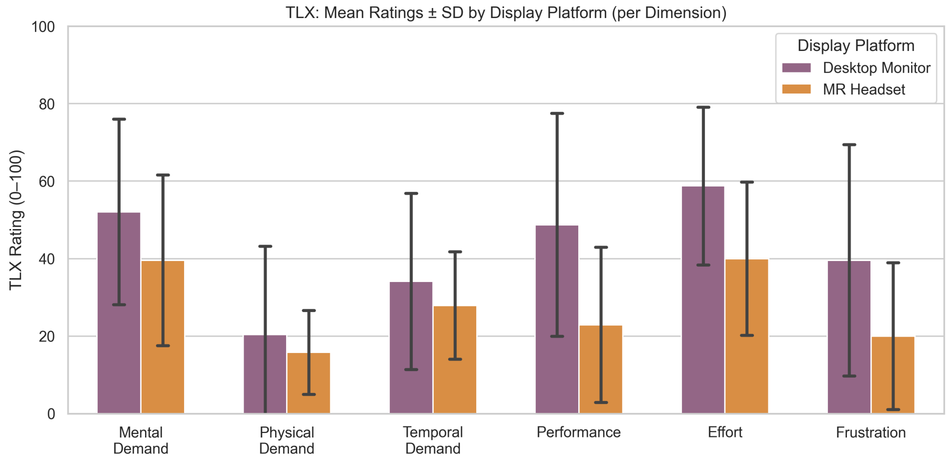
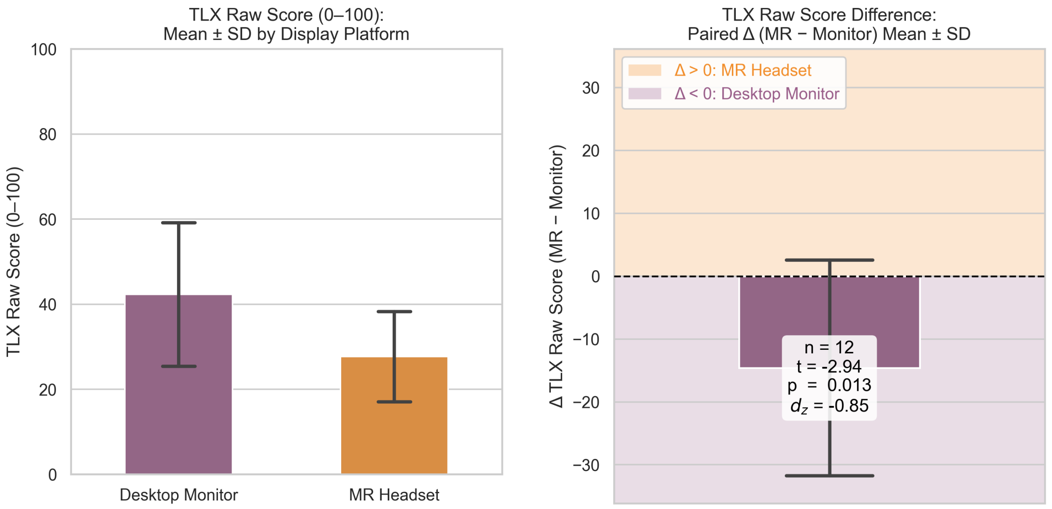


Figure 6.16.: TLX: mean ratings for every taskload dimension by display platforms

The mean Raw TLX score across participants is calculated to be 42.3 for desktop monitor and 27.7 for MR headset. The paired Δ comparison for hypothesis **H2**b (display platforms) revealed a statistically significant difference in workload, with a large effect size, indicating lower workload for the MR headset [$t(11) = -2.94$, $p = 0.013$, $d_z = -0.85$]. Both results are visualized in Figure 6.17.



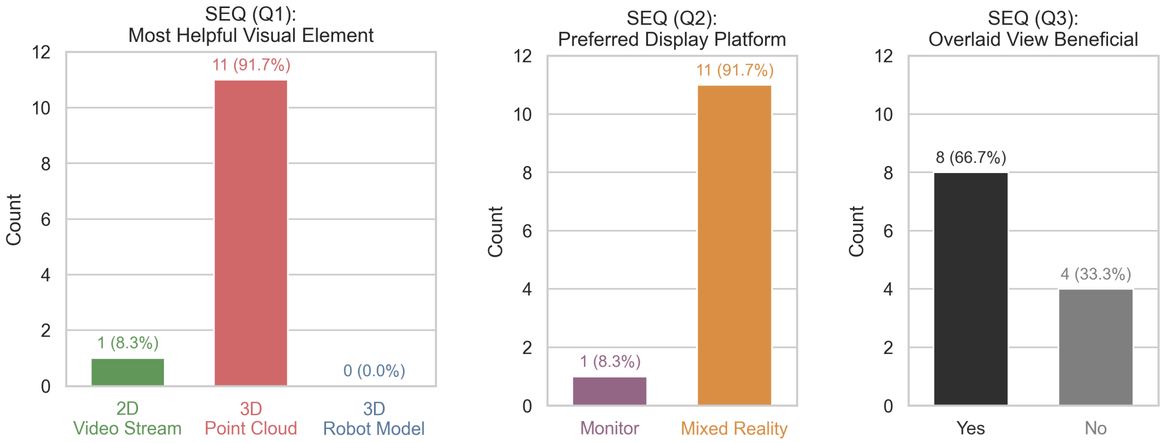
(a) Raw TLX Score (0-100) by display platform

(b) Raw TLX Score difference (Δ) ± SD across participants for **H2**b (display platforms)

Figure 6.17.: Workload: overview of the calculated Raw TLX Score data

System Evaluation Questionnaire (SEQ)

For the final system evaluation, the results of the quantitative parts of the SEQ questionnaire are presented in this section. Figure 6.18 shows summed counts (with percentages in parentheses) of chosen preferences for visual element (a), display platform (b), and potential benefit of combining the robot model with the point cloud visualization (c).



(a) Most helpful visual element **(b)** Preferred display platform **(c)** Benefit point ploud overlay on robot model

Figure 6.18.: SEQ quantitative data: (a) Question 1, (b) Question 2, (c) Question 3

Figure 6.18a shows, 11 out of the 12 participants (91.7%) chose 3D Point Cloud to be the most helpful visual element, 1 chose 2D Video Screen and 0 chose 3D Model View. Figure 6.18b indicates that 11 out of 12 participants (91.7%) chose Mixed Reality to be the preferred display platform for using the system. Figure 6.18c shows that 8 participants (66.7%) answered "yes" to the question of whether a point cloud overlay on the robot model would have been beneficial.

Summary of Questionnaire results

To simplify interpretation for the subsequent discussion chapter, Table 6.2 provides a summarized overview of the statistically significant findings ($p < \alpha = 0.05$) from the questionnaire ratings analysis, organized by hypotheses. For each rating dimension, the table lists the associated hypothesis, exact p-value, and standardized effect size (d_z). Positive effect sizes ($d_z > 0$) indicate higher rating values for the first condition of the respective hypothesis comparison, as already stated in Section 6.1.

Table 6.2.: Summary of only the statistically significant questionnaire findings for the hypotheses

Hypothesis	**Rating Dimension**	p**-value**	**Effect Size** (d_z)
H1b	Depth Perception	$p < 0.001$	$d_z = 1.49$ (large)
H1b	Spatial Understanding	$p = 0.012$	$d_z = 0.87$ (large)
H2b	Spatial Understanding	$p < 0.001$	$d_z = 1.48$ (large)
H2b	Usability (SUS-Score)	$p = 0.003$	$d_z = 1.09$ (large)
H2b	Workload (TLX-Score)	$p = 0.013$	$d_z = -0.85$ (large)
H3b	Depth Perception	$p < 0.001$	$d_z = 1.69$ (large)
H3b	Spatial Understanding	$p = 0.002$	$d_z = 1.20$ (large)

6.3. Qualitative Feedback

This section summarizes qualitative feedback by the participants targeted by the last three questions of the SEQ (final questionnaire) divided in positives, negatives and free comments about the system. Additionally, verbal comments by participants, as well as behavior and incidents observed by the experimenter during the user study are presented.

Positives about the System

Participants reported that mixed reality was immersive and it was easy to operate the robot with it. They repeatedly said leaning into the visualization elements helped a lot to focus on certain details. With mixed reality, depth information was much clearer, especially in combination with the point cloud, which was called the most useful visual element in combination with mixed reality for 3D perception. However, the clearest system overview and therefore the highest operative confidence were reported while using the free focus mode in combination with mixed reality, as switching during different phases was very helpful to accomplish the task to gain better spatial understanding of the full situation. The additional smaller side windows in free focus mode helped, acting as additional information channels to support the main focused element, especially during object identification.

For assessment during grasping, the 2D video element helped with its high resolution to be an intuitive feedback with clear visual cues like shadows. Also, some participants had previous experiences with such systems for teleoperation from which they benefited. The

3D point cloud element was positively assessed with regards to navigating in 3D coordinates, height assessment above ground, depth perception and overall 3D scene overview. The robot model element was useful for understanding robot configuration because of its high resolution.

The GUI system's ability to rotate 3D elements was called easy to use and highly beneficial to change perspective in order to confirm and inspect details from different view angles, especially helpful during grasping and target releasing. Overall, the interface was called simple and intuitive to use and the minimalistic lean design was rated positively in terms of mental load. The input devices were called intuitive to use, as inputs gave direct visual feedback and the GUI interaction input (left hand) was clearly separated from the robot control input (right hand).

Negatives about the System

The robot control input device (SpaceMouse) was repeatedly reported to be too sensitive, as the robot stuttered and reversed its movement axis frequently when applying too much force on the device, taking a while to get used to. The fact that translational robot control axes were fixed in space while the 3D visualization elements could be rotated confused participants. Additionally, the GUI rotation was reported to have felt inverted for some participants.

Using the desktop monitor was highly demanding, while the GUI was called too small with lack of zoom capabilities. When rotating the point cloud on the desktop monitor, losing track of orientation was reported to be frustrating. While wearing the mixed reality headset, some participants felt discomfort due to minor motion sickness and the weight of the head-mounted display pushing on the head, especially during prolonged time periods of uninterrupted wearing.

Depth perception was reportedly weak when using only the video element. The point cloud visualization had low resolution with sparse and large points, also disturbing flickering and noisy areas, specifically at the robot fingertips, were addressed. In addition, view occlusion affected the point cloud when the robot hand covered the object from view or the backside of the robot arm was constantly missing (sensor shadow). This made operation with the point cloud element reportedly challenging and frustrating. The robot model element had issues with overall relevance for the task, as the object was not tracked and the scene mock-up was not precisely placed. Therefore the robot model element was called not very useful overall, as the robot configuration was already sufficiently visualized by the other visualization elements.

Comments and Suggestions for the System

Participant comments included that lighting conditions during study helped a lot when using the video element, as sharp cast shadows indicated the height over the ground, therefore increasing task precision without real depth understanding of the scene. The multi-windowed GUI created a complementary holistic overview, as different visual elements helped best during specific task phases. The video gave little information about precise target release, especially when high over the ground. Mixed reality had a much larger visible area and was perceived as bigger GUI compared to the desktop monitor. A general enthusiasm about the used system was also expressed.

Participants suggested using multiple depth cameras for better coverage of the backside of the robot to complete the missing shadow areas in the point cloud. Also higher resolution of the point cloud as well as presets for top-, side- and front-view to reset to after free rotation were proposed ideas. With those improvements, the previous benefits of the video element was imagined to be entirely replaceable by the point cloud with no more drawbacks. Another suggestion was to integrate a virtual projection of the robot's hand position onto the xy-plane on the ground, indicating height and positioning over the ground intuitively. Finally it was addressed that an alignment of the input devices control axes with the current view angle of the 3D GUI elements and another less sensitive input device would have been beneficial for effective robot control.

Observations by Experimenter

More general observations by the experimenter during task executions showed clearly visible learning effects between first and last runs despite a preceding familiarization phase. This included getting used to the sensitivity of the SpaceMouse as well as to the GUI interaction in the actual task setup, resulting in an ineffective robot oscillation movement in the beginning. However there was a high variety between participants in adaptation speed and intuition for the input device.

During the video screen mode runs, frequent collisions with the object or the foam platform underneath were observed, which created a visible movement cue indicating proximity to the object during approach for grasping. Additionally different strategy trends were observed for video compared to point cloud. With only video, participants tried to position the hand in front of the object before approaching in line of sight straight towards the object, whereas with only point cloud the common strategy was an approach from above lowering down onto the object. A commonly observed occurrence also was misjudgment of height above the ground with video visualization often resulting in the necessity to warn participants about the proximity, which happened not even once during point cloud runs. Accuracy during both grasping and target releasing was observed to be consistently off the central point by a few centimeters towards the depth camera when using point cloud and in addition participants often positioned the hand too high above the object. During free focus mode runs it was observed that task duration was generally higher, as participants took longer breaks to evaluate current position more thoroughly by switching through multiple visual elements. This mode also changed behavior toward more methodical approaches.

7. Discussion

In this chapter, the findings from the results Chapter 6 are discussed and interpreted. The research questions will be answered before validating the hypotheses from Section 5.1. Subsequently the findings and tendencies of this work are compared to related work, before discussing the limitations within this work.

7.1. Interpretation of Results

Overall, a high task success rate was achieved across all conditions, as shown in Figure 6.1. In addition, task success rates increased across consecutive runs, indicating a clear practice effect, as Figure 6.2 shows. Due to learning effects, performance was lowest during the first run and increased before stabilizing between the third and fourth run. Therefore, varying the order of display platforms and GUI modes proved to be an appropriate measure to mitigate learning effects. However, as the only task failure criterion was a third unsuccessful grasp attempt, this effect provides limited insight into actual task effectiveness beyond basic task completion.

The number of necessary grasp attempts until a successful grasp (or third failed attempt) tended to be higher while using the point cloud GUI than while using the video GUI. This was not the expected outcome, however, participant comments and experimenter observations revealed that specifically for grasping using the point cloud GUI proved to be harder than using the video screen GUI. The reasons indicated for this effect are the point cloud's low resolution and noisy depth fluctuations due to depth sensor limitations and inaccuracies. Also the depth sensor positioning on the opposite side of the robot arm with respect to the grasp area maximized the effect of self shadowing, where the robot arm would cover the small object to grasp from view. This might have shrunk the information enough to render the point cloud depth less useful than the higher resolution RGB Video. Participant comments indicate difficulty with precise assessment of position once the robot hand was close to the ground due to covering line of sight missing crucial depth information when most needed. Additionally, the depth appearing noisier at the tips of the robot hands fingers further complicating precise spatial assessment while being close to the ground.

This also resulted in another effect while using the point cloud, as a consistent spatial offset in the perceived object position was observed, characterized by a systematic displacement toward the depth sensor and above the actual object location by a few centimeters. This effect was both captured during object grasping as well as releasing at target, resulting also in negatively shifted measurements for target accuracy. This indicates a measurable mismatch between where the robot hands center point was thought to be and where it truly was. Participants suggested that additional depth sensors covering the other side of the robot would have presented a more holistic overview of the missing backside information, which might have greatly increased grasping confidence and success using the point cloud.

Furthermore, the observed strategies varied substantially between using video and point cloud visualizations. While being limited to only the video stream, participants strongly tended to first move the robot end-effector into the line of sight in front of the object before approaching, while staying roughly on the line of sight between camera and object. Comments suggest that this proved to be a well performing strategy, however providing very little actual understanding for depth and spatial distances. The success also significantly depended on visual cues to assess proximity, like hitting and therefore moving the object or the raised platform on which the object rested. Also the almost vertical light conditions in the task setup created a distinct shadow cast by the robot, visually pinpointing the current position over the ground plane while the robot hand was close to it. This effect also influenced target accuracy, resulting in no significant difference in planar target offset and subsequently task efficiency when comparing point cloud with video GUI. The most common strategy for grasping when using the point cloud was to first move over the object while staying high above the ground, and then moving down towards the ground while constantly adjusting to stay roughly over the object until it is reached. While this strategy indicates a more methodical and precise 3D approach for grasping, it also increases the path length and task duration by not choosing the most efficient route in that regard. This also had an effect on task efficiency, as path length and duration took part in its calculation, resulting in limited information value about actual task performance in this specific comparison.

As both the questionnaire and comment data suggest, the 3D point cloud was clearly preferred as visual element while providing significantly better depth perception as well as spatial understanding compared to the 2D video screen. The majority of participants also indicated that a combination of the 3D robot model and the 3D point cloud as an overlay would have been beneficial for the task. As the robot model missed crucial information about the robot's unknown environment, with object position being the most important, this visual element alone did not prove viable to accomplish the task successfully. Overlaying the point cloud onto the robot model would combine a clear 3D representation of the precise robot configuration with the 3D data of the unknown environment through the point cloud, creating a single holistic visual element.

The obstacle was avoided successfully in every run, suggesting that every combination of conditions was able to provide sufficient feedback to move around a simple obstacle. However, the exact size and positioning of the obstacle used for the study might also not have been ideal, resulting in no additional insights about depth perception effects.

The planar target offset, as the main metric to measure accuracy capability, indicated a clear trend of higher precision during object release when using the point cloud rather than video, while both proved to be the ideal setup. Using the MR Headset improved precision across all GUI combinations even further, resulting especially in a significant difference with high effect size between mixed reality with point cloud and desktop monitor with video.

Similar improvements were observed for the task efficiency, however, path length and task duration did not clearly indicate significant effects on their own. This is thought to be the case because the route strategy differences described earlier were likely compensated by movement confidence and therefore adjustment pace during precise positioning.

The general comparison between the two display platforms showed with strong significance that the MR headset outperformed the desktop monitor on multiple metrics. Especially successful was the combination of point cloud or free focus (including all visual elements) displayed on the MR headset. This was not only the preferred participant combination shown in questionnaires and comments, but also statistically one of the most significant (highest effect sizes) across multiple metrics when compared to the video and desktop monitor combination. This suggests clear improvements compared to the video-based monitor setup from previous work in direct teleoperation of unknown object manipulation tasks, such as the Operator-UI used in Surface Avatar.

Questionnaire data comparison also shows that both subjective depth perception and spatial understanding were much better while using the MR headset, especially when displaying the point cloud element. Comments and observations clearly indicated, that the 3D immersion of mixed reality further improved the benefit of 3D GUI elements like the point cloud. Being able to lean forward into the GUI created a zoom-like effect for focusing on specific details and further improved immersion. This reportedly helped for both the video and the point cloud GUIs when using mixed reality, as some suggested the GUI was small in some cases and zoom capabilities would have been beneficial when using the desktop monitor.

Furthermore, the TLX and SUS results showed an overall higher workload and lower usability for the desktop monitor setup compared to the MR headset. This indicates that mixed reality proved to be less demanding and more user-friendly to use. The fact that mixed reality showed the user's real surroundings overlaid by virtual GUI elements is thought to be crucial for workload and usability, as the physical input devices could still be seen this way and the user keeps a sense for base reality, important for both interaction and safety in the operator space. Negative effects, however, also decreased the benefit of mixed reality, for one the head-mounted display gets heavy to wear for prolonged times, and secondly motion sickness appears to be an issue for some users especially with flickering and rotating GUI elements.

As an additional GUI combination, the free focus mode showed effects of using the major GUI elements used for direct teleoperation in surface avatar (Video and Robot Model) by adding the new point cloud to the mix and to see effects that mixed reality has. When using the free focus mode, both path length and duration were higher than during each of the single visual elements (Video, Point Cloud). This effect was also very inconsistent across participants, resulting in the highest standard deviation across all conditions. Observations and comments likely suggest this happened due to the increase in workload while having to interpret more information and also interact with the GUI to switch between visual elements during the run. However, no quantitative data was measured to verify this claim, as the TLX was only conducted to compare the two display platforms. Even though the free focus mode adding additional information to the GUI (compared to the single visual elements), it took more grasp attempts on average, indicating more practice would have been necessary to use the free focus mode effectively. Precision accuracy proved to be highly increased when using the free focus compared to singe visual elements, showing the benefits of having multiple GUI elements, each conveying a certain aspect of feedback most optimally.

7.2. Validation of Hypotheses

In the following section, the established hypotheses are validated individually based on the experimental results. Each hypothesis is considered in its respective subparts and evaluated accordingly.

H1 (Visualization Element)

Performance findings did not support H1a, indicating no statistically significant differences for grasp attempts, target accuracy and task efficiency when comparing the video interface with the point cloud interface [Table 6.1].

H1b however was supported by the results, suggesting a strong tendency for improved depth perception and spatial understanding when using the point cloud compared to the video element. In addition, participants showed a clear preference for the point cloud visualization compared to the video-based GUI, indicating advantages in terms of perceived user experience [Table 6.2].

In conclusion, H1 is partially supported, suggesting that while the point cloud visualization on its own does not lead to measurable performance improvements, it still provides clear perceptual and experiential benefits over a purely video-based interface.

H2 (Display Platform)

The results strongly support H2a, indicating that the use of a MR headset leads to improved task performance compared to a desktop monitor. Task efficiency was significantly higher for the mixed reality conditions, accompanied by a medium effect size. Both Planar target offset and grasp attempts were significantly lower with a large effect size when using an MR heaset. This suggests, that the observed performance improvement is not only statistically reliable but also practically relevant [Table 6.1].

In addition, H2b was supported even stronger by the findings, as lower workload, higher system usability and user preference indicated a high tendency that a MR headset outperforms a desktop monitor setup for the described teleoperation applications. In addition the highest effect size for all conducted comparisons on spatial understanding highly favored mixed reality, further strengthening this claim [Table 6.2].

The magnitude of effects for H2 in general indicates, that immersive display technologies like mixed reality can substantially enhance task execution during direct teleoperation of unknown object manipulation tasks.

H3 (Comparison to the previously used project setup):

The combined condition comparison between MR headset with point cloud interface and desktop monitor with video interface supports H3a with an even larger effect size for task efficiency than the display platform alone did in H2a. Aditionally, planar target offset and grasp attempts both showed significantly lower values with medium to large effect sizes for the combination of MR headset with point cloud interface, strongly showing fundamental task performance improvements [Table 6.1].

H3b is strongly supported, with the largest observed effect size in depth perception and a large effect size in spatial understanding, substantially favoring the MR headset with point cloud interface when compared to the desktop monitor with video interface [Table 6.2].

H3 is therefore fully and strongly supported, suggestinging significant improvements of the newly proposed UI concept compared to the previous setup.

7.3. Answering the Research Questions

Based on the previous discussion of results and the validation of the hypotheses, this section answers the research questions that motivated the study. The research questions are addressed sequentially, covering visualization elements, display platforms, and their combined effects.

RQ1: Visualization Element

The 3D point cloud visualization on its own did not result in consistent performance improvements over a 2D video-based interface. Nevertheless, it significantly enhanced perceived depth perception and was clearly preferred by users.

RQ2: Display Platform

The use of a mixed reality headset improved subjective depth perception, spatial understanding, and overall user experience compared to a desktop monitor. In addition, performance-related metrics showed a clear positive tendency in favor of the mixed reality display platform.

RQ3: Combined Effects

The combined use of a mixed reality headset and a point cloud visualization provided the highest levels of perceived immersiveness and spatial understanding across the conducted comparisons. While this setup led to significant performance improvements, the strongest effects were observed in subjective user experience.

7.4. Comparison with Related Work

The findings of this work aligned well with the state of the art in related work. Especially the tendency for improvements in depth perception, spatial understanding and therefore situational awareness when using more immersive technology like mixed reality was supported by related work, such as the conclusion on the usage of mixed reality in combination with point cloud visualizations in NASA Valkyrie [JWP22, p. 6]. Using a 3D visualization element like a point cloud representation has shown to improve better understanding of a remote environment for operators. Related work investigating teleoperation of robots in mixed reality based on user gestures and concluded improvements in depth perception and spatial understanding through 3D object visualization, and free viewpoint changes as well as fast adaptation to dynamic task environments through the use of mixed reality [ES24, pp. 21–22].

Novel contributions from this work include a better understanding of the mentioned effects in specific teleoperation setups, as they occurred during the Surface Avatar experiments. As this use case also predefined specific telerobotic platforms operating on remote planetary surfaces, sensor capabilites and placement were a key limitation to work with. The findings of this work provided novel insights into the viability of point cloud reconstruction from a single fixed depth camera visualized in mixed reality for Surface Avatar. While performance improvements were limited to specific aspects, the results indicated clear benefits in terms of perceived depth perception, spatial understanding, system usability, and overall user experience. In addition, task load effects showed the importance for such systems to be minimalistic in GUI presentation and interaction, as single-windowed GUIs proved to be more task efficient than a dynamic multi-windowed counterpart.

7.5. Limitations

The teleoperation system used in this work was limited in its hardware and sensor infrastructure. As this system was created specifically for the user study, no actual remote capabilities were considered in its design, as the entire system was located in a single room. This allowed for simple cable connections between the depth sensor and the GUI hardware, same for connecting the input device directly by cable to the robot's processing hardware. In actual remote teleoperation this would not be possible, so this system did not address performance when confronted with signal latency and bandwidth limitations. Environment effects like changing lighting conditions were therefore also not considered, as the robotic system was located indoors. The GUI was primarily designed to track variables and separate effects during the user study, not to be an optimal teleoperation interface.

The user study design also limited the information value about real teleoperation applications. For one thing, the participants did see the robot and its scene prior to the study, meaning it was not representing a fully unknown scenario. Secondly, the operator and the teleoperated robot were located in the same room, although facing in the opposite direction granting no direct visual contact to the robotic scene, some aditional feedback like sound of the robot's actuators or collisions reached the operator, therefore not representing actual remoteness. Also the study was not conducted from space, as previous user studies in Surface Avatar were (ISS-to-ground), which limits the comparability with findings from Surface Avatar in that regard.

8. Conclusion and Future Work

The focus of this work was to design, implement, and evaluate a novel Unity-based graphical user interface for robotic teleoperation, supporting both conventional desktop displays and mixed reality head-mounted displays.

To achieve this, the system was developed in close alignment with object manipulation scenarios encountered in the Surface Avatar experiments, with the goal of enabling direct teleoperation under comparable spatial and perceptual constraints. Mixed reality visualization was combined with real-time depth sensor data and robot control interfaces. A dedicated visual processing pipeline was implemented to capture, store, and convert RGB images and raw depth measurements into different visualization elements, including 2D video streams and 3D point cloud representations. For that, a custom particle system was implemented to process depth and color textures into a coherent colored point cloud representation. The GUI was integrated with an existing robot control framework running on a separate machine via WebSocket-based communication, allowing synchronized real-time visualization of robot state and sensor data during teleoperation. In addition, a backend tracking and logging system automatically records all study-relevant variables, including run condition, participant index, timestamps, GUI mode, as well as end-effector poses and joint states, which are exported in a structured CSV format for subsequent analysis.

Then, the developed system was evaluated in a user study to compare effects on depth perception and spatial understanding by measuring different precision metrics, task success indicators, and a derived task efficiency, as well as self-reported data by participants. As a result, an empirical comparison of immersive and non-immersive teleoperation interfaces for unknown object manipulation tasks was achieved, distinguishing between objective performance metrics and subjective user experience.

In summary, a clear tendency towards improved depth perception was detected when comparing mixed reality to a desktop monitor setup. This effect was further enhanced when comparing the mixed reality point cloud combination with the baseline desktop monitor video setup, revealing lower perceived workload during task execution as well as a much higher measured precision during object manipulation, indicating improved spatial understanding of the robot environment. However, experimental evidence showed that improved perceived depth perception and spatial understanding do not necessarily translate into measurable performance gains in direct teleoperation tasks.

The key findings of this work highlight the potential of three-dimensional visualizations presented in mixed reality to improve teleoperation interface design in space robotics, especially for scenarios in which operators interact with unknown or partially known environments under limited sensory feedback. Overall, immersive display platforms and three-dimensional visualizations can substantially enhance perceived depth perception and spatial understanding, which are essential for situational awareness during remote manipulation tasks.

At the same time, the observed discrepancy between subjective experience and objective task performance demonstrates how important it is to balance immersive visualization techniques with the task-specific performance requirements. Rather than replacing traditional video-based interfaces entirely, hybrid approaches that combine two-dimensional and three-dimensional visual elements can offer a more reliable approach for complex teleoperation scenarios.

Beyond general teleoperation scenarios, these findings are especially relevant to the Surface Avatar project, which supports scalable autonomy ranging from high-level autonomous execution all the way to full direct teleoperation. The results demonstrated a clear improvement potential for the currently used UI system for direct control, especially during manipulation tasks involving unknown objects, where operators must rely solely on visual feedback to estimate depth and spatial relationships. In scenarios where the knowledge-driven approach from Surface Avatar is confronted with unknown objects, so augmented reality overlays for high-level command execution were not available, immersive visualization techniques and three-dimensional environment representations can substantially support spatial understanding and operator confidence.

Future work is recommended to build upon the presented user study-driven evaluation platform, which serves as an initial step towards immersive teleoperation interface development, to further develop it into a technology demonstrator suitable for deployment within the Surface Avatar project. In particular, the system can be adapted for realistic space robotics scenarios by enabling remote operation capabilities. This would require replacing the current cable-based depth sensor connection with a network-based data stream to support distributed system architectures and long-distance communication.

In addition, performance evaluations on less powerful hardware platforms can be conducted. While the presented system was evaluated on a high-performance desktop workstation with no relevant computational constraints, future studies are expected to assess the feasibility of mixed reality visualization with simultaneous point cloud processing on hardware more representative of space-deployed systems. This would allow for a more realistic assessment of computational limitations and rendering performance in operational environments.

Moreover, future studies can further investigate the effects that communication latency and bandwidth limitations can have on immersive teleoperation with mixed reality interfaces. Particularly interesting for space robotics is the impact of delayed or degraded depth data on spatial perception and task performance.

In addition, the proposed system could also be evaluated on mobile robotic platforms beyond the stationary manipulation setup used in this work. This specifically refers to the platforms used within the Surface Avatar experiments, including humanoid robots such as Rollin' Justin (DLR) and smaller quadrupedal robots like Bert (DLR). It could demonstrate how well the presented visualization approach transfers to different robot types and task scenarios.

Finally, future work can integrate this visualization system into the broader scalable autonomy approach of the Surface Avatar project. This would show how more immersive teleoperation interfaces could improve the coordinated control of a mixed robotic team, operating on different autonomy levels.

Bibliography

[AHO07] Albu-Schäffer, A. O.; Haddadin, S.; Ott, C.; Stemmer, A.; Wimböck, T.; Hirzinger, G.
The DLR Lightweight Robot – Design and Control Concepts for Robots in Human Environments
In: INDUSTRIAL ROBOT-AN INTERNATIONAL JOURNAL, 34 (Mar. 2007) 5, https://elib.dlr.de/51178/ (visited on 02/03/2026), pp. 376–385.

[BKM08] Bangor, A.; Kortum, P. T.; Miller, J. T.
An Empirical Evaluation of the System Usability Scale
In: International Journal of Human–Computer Interaction, 24 (July 29, 2008) 6, DOI 10.1080/10447310802205776, https://doi.org/10.1080/10447310802205776 (visited on 01/29/2026), pp. 574–594.

[BKS10] Bischoff, R.; Kurth, J.; Schreiber, G.; Koeppe, R.; Albu-Schaeffer, A.; Beyer, A.; Eiberger, O.; Haddadin, S.; Stemmer, A.; Grunwald, G.; Hirzinger, G.
The KUKA-DLR Lightweight Robot Arm - a New Reference Platform for Robotics Research and Manufacturing
In: ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics), pp. 1–8.

[Bro95] Brooke, J.
SUS: A Quick and Dirty Usability Scale
In: Usability Eval. Ind. 189 (Nov. 30, 1995).

[CLY24] Cheng, X.; Li, J.; Yang, S.; Yang, G.; Wang, X.
Open-TeleVision: Teleoperation with Immersive Active Visual Feedback
http://arxiv.org/abs/2407.01512 (visited on 01/09/2026).

[CM05] Carlson, J.; Murphy, R.
How UGVs Physically Fail in the Field
In: IEEE Transactions on Robotics, 21 (June 2005) 3, DOI 10.1109/TRO.2004.838027, https://ieeexplore.ieee.org/document/1435486 (visited on 02/03/2026), pp. 423–437.

[DaV] DaVinci-Website
Da Vinci Robotic Surgical Systems | Intuitive
https://www.intuitive.com/en-us/products-and-services/da-vinci (visited on 01/27/2026).

[Div] Division, N. A. H. S. I.
NASA-TLX
NASA Ames Research Center, https://humansystems.arc.nasa.gov/groups/tlx/downloads/TLXScale.pdf (visited on 01/29/2026).

[DLR26] DLR-Website
Surface Avatar
https://www.dlr.de/en/rm/research/projects/completed-projects/surface-avatar (visited on 01/28/2026).

[End95] Endsley, M. R.
Toward a Theory of Situation Awareness in Dynamic Systems
In: Human Factors, 37 (Mar. 1, 1995) 1, DOI 10.1518/001872095779049543, https://doi.org/10.1518/001872095779049543 (visited on 01/28/2026), pp. 32–64.

[ES24] Esaki, H.; Sekiyama, K.
Immersive Robot Teleoperation Based on User Gestures in Mixed Reality Space
In: Sensors, 24 (Jan. 2024) 15, DOI 10.3390/s24155073, https://www.mdpi.com/1424-8220/24/15/5073 (visited on 09/01/2025), p. 5073.

[FAT11] Foix, S.; Alenya, G.; Torras, C.
Lock-in Time-of-Flight (ToF) Cameras: A Survey
In: IEEE Sensors Journal, 11 (Sept. 2011) 9, DOI 10.1109/JSEN.2010.2101060, https://ieeexplore.ieee.org/document/5686908 (visited on 01/28/2026), pp. 1917–1926.

[FR20] Friedl, W.; Roa Garzon, M. A.
CLASH - A Compliant Sensorized Hand for Handling Delicate Objects
In: Frontiers in Robotics and AI (, Jan. 17, 2020), https://www.frontiersin.org/article/10.3389/frobt.2019.00138 (visited on 02/02/2026).

[GG25] Ghosh, S.; Gallego, G.
Event-Based Stereo Depth Estimation: A Survey
In: IEEE Transactions on Pattern Analysis and Machine Intelligence, 47 (Oct. 2025) 10, DOI 10.1109/TPAMI.2025.3586559, arXiv: 2409.17680 [cs], http://arxiv.org/abs/2409.17680 (visited on 01/28/2026), pp. 9130–9149.

[GSG23] Gawaikar, A.; Sorte, S.; Ghutke, P.; Patil, W.; Joshi, S.
Review of Da Vinci Surgical System and Haptic Feedback Device for Telesurgery
In: 2023 8th International Conference on Communication and Electronics Systems (ICCES), pp. 1596–1600.

[Har06] Hart, S. G.
Nasa-Task Load Index (NASA-TLX); 20 Years Later
In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 50 (Oct. 1, 2006) 9, DOI 10.1177/154193120605000909, https://doi.org/10.1177/154193120605000909 (visited on 01/29/2026), pp. 904–908.

[HS88] Hart, S. G.; Staveland, L. E.
Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research
In: Advances in Psychology, North-Holland, Jan. 1, 1988, pp. 139–183.

[JWP22] Jorgensen, S. J.; Wonsick, M.; Paterson, M.; Watson, A.; Chase, I.; Mehling, J. S.
Cockpit Interface for Locomotion and Manipulation Control of the NASA Valkyrie Humanoid in Virtual Reality (VR)
June 1, 2022, https://ntrs.nasa.gov/citations/20220007587 (visited on 08/30/2025).

[KLY25] Khedr, M.; Le, Q. D.; Yang, E.
Teleoperation Control for Robotic Systems in Hazardous Environments: Overview and Challenges
In: 2025 30th International Conference on Automation and Computing (ICAC), pp. 1–6.

[LSL22] Lii, N. Y.-S.; Schmaus, P.; Leidner, D.; Krueger, T.; Grenouilleau, J.; Pereira, A.; Giuliano, A.; Bauer, A. S.; Köpken, A.; Lay, F. S.; Sewtz, M.; Bechtel, N.; Bustamante Gomez, S.; Denninger, M.; Friedl, W.; Butterfass, J.; Ferreira, E.; Gherghescu, A.; Chupin, T.; den Exter, E.; Gerdes, L.; Panzirsch, M.; Singh, H.; Balachandrand, R.; Hulin, T.; Gumpert, T.; Schmidt, A.; Seidel, D.; Hermann, M.; Maier, M.; Burger, R.; Schmidt, F.; Weber, B.; Bayer, R.; Pleintinger, B.; Holderried, R.; Pavelski, P. H.; Wedler, A.; von Dombrowski, S.; Maurer, H.; Görner, M.; Wüsthoff, T.; Bertone, S.; Müller, T.; Söllner, G.; Ehrhardt, C.; Brunetti, L.; Holl, L.; Bévan, M.; Muehlbauer, R.; Visentin, G.; Albu-Schäffer, A. O.
Introduction to Surface Avatar: The First Heterogeneous Robotic Team to Be Commanded with Scalable Autonomy from the ISS
In: 73rd International Astronautical Congress (IAC).

[MK94] Milgram, P.; Kishino, F.
A Taxonomy of Mixed Reality Visual Displays
In: IEICE Transactions on Information, E77-D (Dec. 25, 1994) 12, pp. 1321–1329.

[MKM26] Manaparampil, A. N.; Koepken, A.; Mayershofer, L.; Batti, N.; Singh, H.; Panzirsch, M.; Lay, F. S.; Luo, X.; Knestel, P.; Bauer, A.; Schmaus, P.; Leidner, D.; Luz, R.; Kruger, T.; Lii, N. Y.
Enhancing Scalable Autonomy Space Teleoperation with User Intervention during Task Execution
In: 2026 IEEE Aerospace Conference (Accepted for Publication).

[MLB26] Mayershofer, L.; Lay, F. S.; Batti, N.; Brinkman, S.; Butterfaß, J.; Ehlert, T.; Exter, E. D.; Friedl, W.; Gumpert, T.; Kopken, A.; Luo, X.; Manaparampil, A. N.; Raffin, A.; Schmidt, A.; Schmidt, F.; Seidel, D.; Luz, R.; Bauer, A. S.; Schmaus, P.; Leidner, D.
Toward Improving Task-Level Commanding in Space Robotics Teleoperation through Shared Mental Models
In: 2026 IEEE Aerospace Conference (Accepted for Publication).

[QCG09] Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A. Y.
ROS: An Open-Source Robot Operating System
In: ICRA Workshop on Open Source Software.

[RC11] Rusu, R. B.; Cousins, S.
3D Is Here: Point Cloud Library (PCL)
In: 2011 IEEE International Conference on Robotics and Automation, pp. 1–4.

[RLS19] *RGB-D Image Analysis and Processing*
Cham: Springer International Publishing, 2019, ISBN 978-3-030-28602-6 978-3-030-28603-3, DOI 10.1007/978-3-030-28603-3, http://link.springer.com/10.1007/978-3-030-28603-3 (visited on 12/27/2025).

[RSH15] Radford, N. A.; Strawser, P.; Hambuchen, K.; Mehling, J. S.; Verdeyen, W. K.; Donnan, A. S.; Holley, J.; Sanchez, J.; Nguyen, V.; Bridgwater, L.; Berka, R.; Ambrose, R.; Myles Markee, M.; Fraser-Chanpong, N. J.; McQuin, C.; Yamokoski, J. D.; Hart, S.; Guo, R.; Parsons, A.; Wightman, B.; Dinh, P.; Ames, B.; Blakely, C.; Edmondson, C.; Sommers, B.; Rea, R.; Tobler, C.; Bibby, H.; Howard, B.; Niu, L.; Lee, A.; Conover, M.; Truong, L.; Reed, R.; Chesney, D.; Platt Jr, R.; Johnson, G.; Fok, C.-L.; Paine, N.; Sentis, L.; Cousineau, E.; Sinnet, R.; Lack, J.; Powell, M.; Morris, B.; Ames, A.; Akinyode, J.
Valkyrie: NASA's First Bipedal Humanoid Robot
In: Journal of Field Robotics, 32 (2015) 3, DOI 10.1002/rob.21560, https://

onlinelibrary.wiley.com/doi/abs/10.1002/rob.21560 (visited on 01/27/2026), pp. 397–419.

[SB14] Schmidt, F.; Burger, R.
How We Deal with Software Complexity in Robotics:'Links and Nodes' and the 'Robotkernel'
In: 14th IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2014.

[SBB23] Schmaus, P.; Bauer, A.; Bechtel, N.; Denninger, M.; Köpken, A.; Lay, F.; Schmidt, F.; Sewtz, M.; Krüger, T.; Leidner, D.; Pereira, A.; Lii, N. Y.
Extending the Knowledge Driven Approach for Scalable Autonomy Teleoperation of a Robotic Avatar
In: 2023 IEEE Aerospace Conference, pp. 1–10.

[SCW16] San, W. Y. K.; Chen, S.; Wiliem, A.; Di, B.; Lovell, B. C.
How Do You Develop a Face Detector for the Unconstrained Environment?
In: 2016 International Conference on Image and Vision Computing New Zealand (IVCNZ), pp. 1–6.

[SFP10] Salvi, J.; Fernandez, S.; Pribanic, T.; Llado, X.
A State of the Art in Structured Light Patterns for Surface Profilometry
In: Pattern Recognition, 43 (Aug. 1, 2010) 8, DOI 10.1016/j.patcog.2010.03.004, https://www.sciencedirect.com/science/article/pii/S003132031000124X (visited on 01/26/2026), pp. 2666–2680.

[SK16] *Springer Handbook of Robotics*
Cham: Springer International Publishing, 2016, ISBN 978-3-319-32550-7 978-3-319-32552-1, DOI 10.1007/978-3-319-32552-1, https://link.springer.com/10.1007/978-3-319-32552-1 (visited on 12/27/2025).

[SLK20] Schmaus, P.; Leidner, D.; Krüger, T.; Bayer, R.; Pleintinger, B.; Schiele, A.; Lii, N. Y.
Knowledge Driven Orbit-to-Ground Teleoperation of a Robot Coworker
In: IEEE Robotics and Automation Letters, 5 (Jan. 2020) 1, DOI 10.1109/LRA.2019.2948128, https://ieeexplore.ieee.org/document/8873599 (visited on 01/28/2026), pp. 143–150.

[Sze22] Szeliski, R.
Computer Vision: Algorithms and Applications
Cham: Springer International Publishing, 2022, ISBN 978-3-030-34371-2 978-3-030-34372-9, DOI 10.1007/978-3-030-34372-9, https://link.springer.com/10.1007/978-3-030-34372-9 (visited on 01/26/2026).

[Tad19] *Disaster Robotics: Results from the ImPACT Tough Robotics Challenge*
Cham: Springer International Publishing, 2019, ISBN 978-3-030-05320-8 978-3-030-05321-5, DOI 10.1007/978-3-030-05321-5, http://link.springer.com/10.1007/978-3-030-05321-5 (visited on 12/27/2025).

[TTP22] Tychola, K. A.; Tsimperidis, I.; Papakostas, G. A.
On 3D Reconstruction Using RGB-D Cameras
In: Digital, 2 (Sept. 2022) 3, DOI 10.3390/digital2030022, https://www.mdpi.com/2673-6470/2/3/22 (visited on 01/26/2026), pp. 401–421.

[VQG] Vuong, Q.; Qin, Y.; Guo, R.; Wang, X.; Su, H.; Christensen, H.
Single RGB-D Camera Teleoperation for General Robotic Manipulation
https://ar5iv.labs.arxiv.org/html/2106.14396 (visited on 01/28/2026).

[Wöl23] Wölfel, M.
Immersive Virtuelle Realität: Grundlagen, Technologien, Anwendungen
Berlin, Heidelberg: Springer, 2023, ISBN 978-3-662-66907-5 978-3-662-66908-2, DOI 10.1007/978-3-662-66908-2, https://link.springer.com/10.1007/978-3-662-66908-2 (visited on 12/27/2025).

[WSC25] Wolf, M.-M.; Schmidt, H.; Christl, M.; Fank, J.; Diermeyer, F.
A User-Centered Teleoperation GUI for Automated Vehicles: Identifying and Evaluating Information Requirements for Remote Driving and Assistance
http://arxiv.org/abs/2504.21563 (visited on 01/09/2026).

[YWH16] Yoshida, K.; Wilcox, B.; Hirzinger, G.; Lampariello, R.
Space Robotics
In: Siciliano, B.; Khatib, O. (eds.), Springer Handbook of Robotics, Cham: Springer International Publishing, 2016, ISBN 978-3-319-32552-1, pp. 1423–1462.

List of Tables

List of Figures

A. Abbildungen

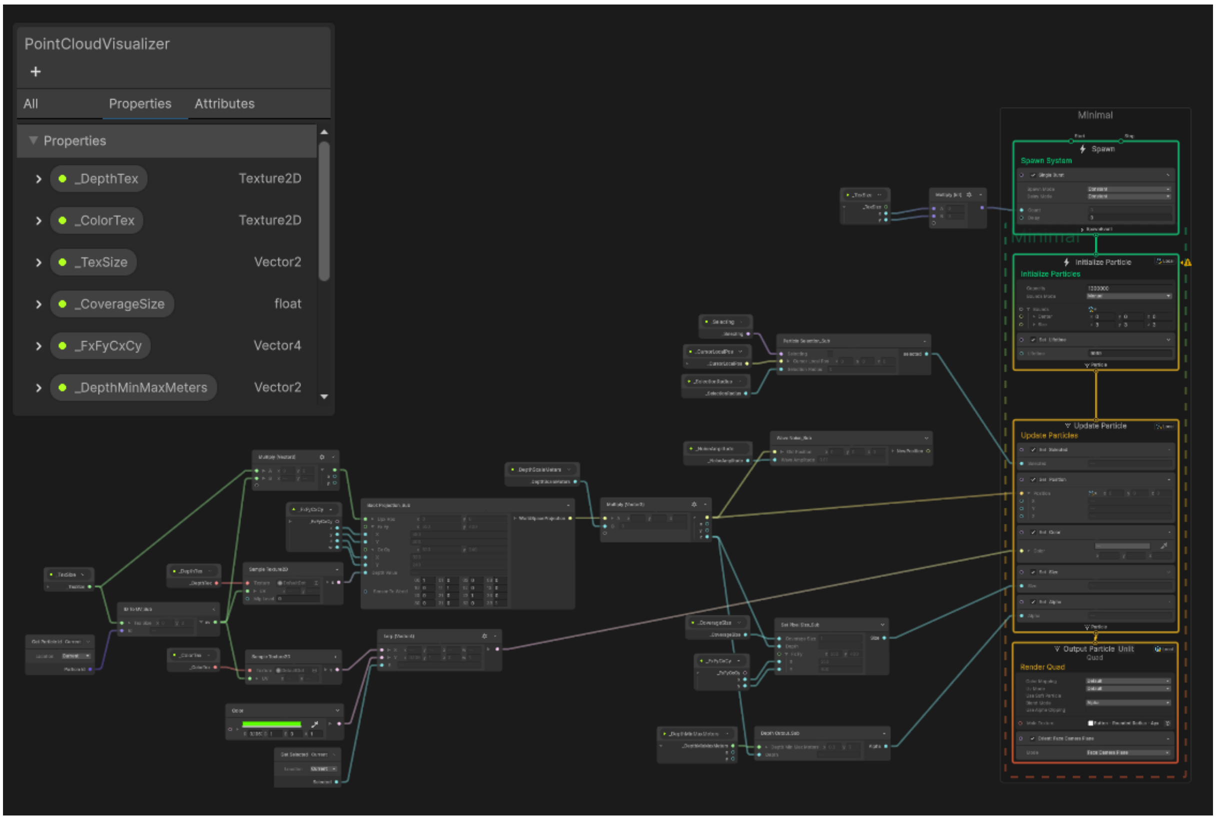


Figure A.1.: Screenshot of the actual point cloud particle system implemented in VFX Graph in Unity for this work

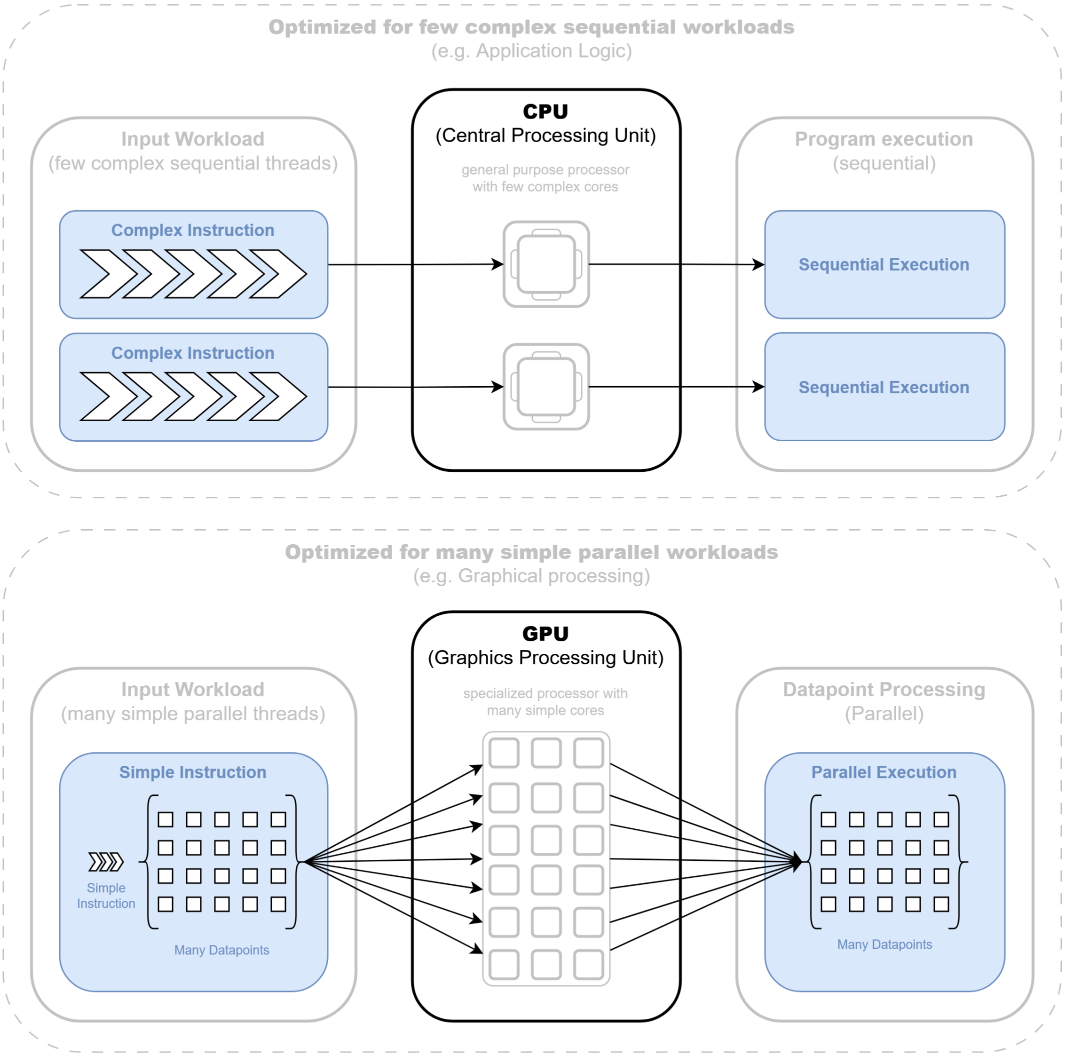
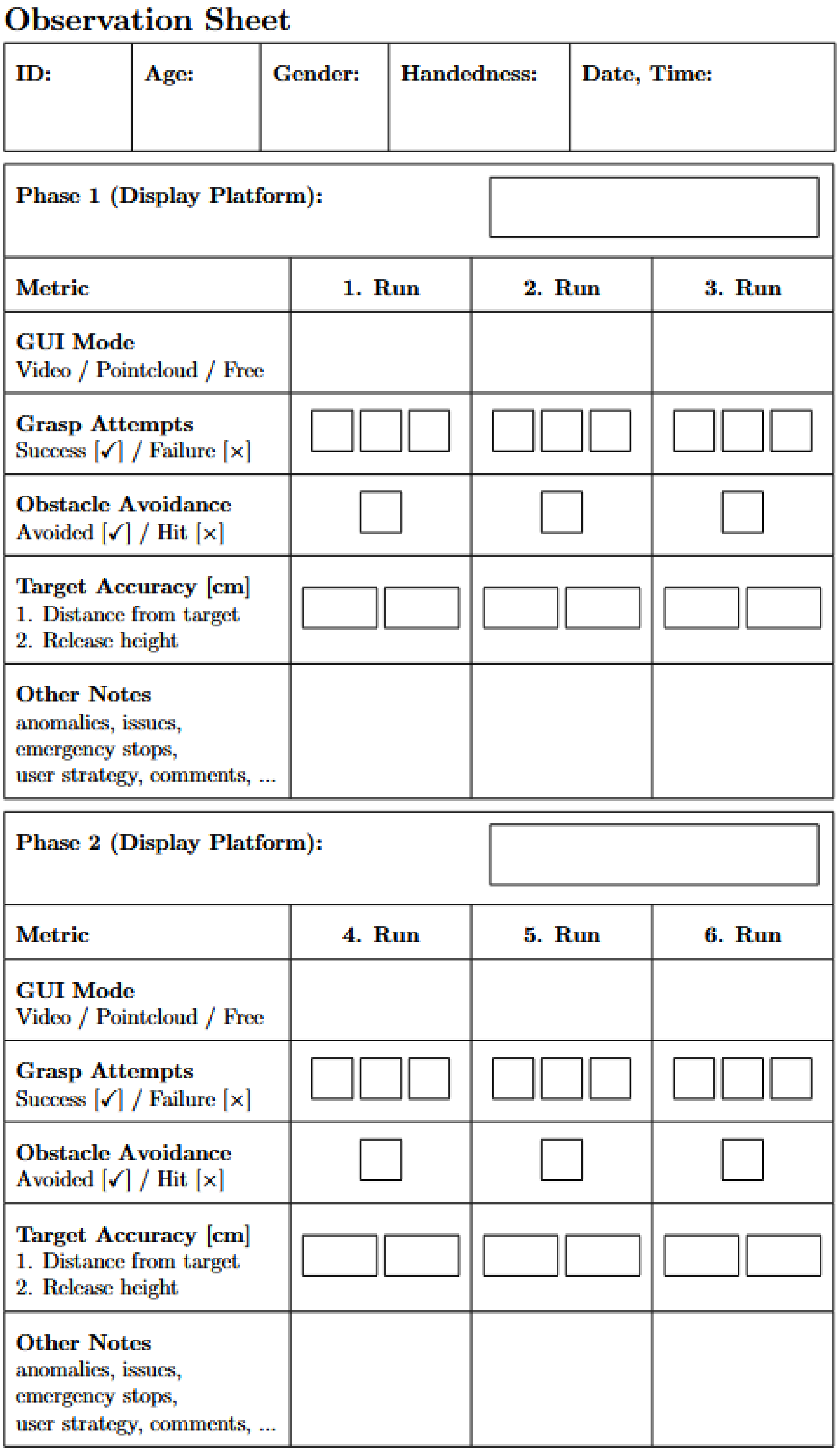


Figure A.2.: CPU and GPU comparison, optimized for different data processing tasks



Observation Sheet

ID:	**Age:**	**Gender:**	**Handedness:**	**Date, Time:**

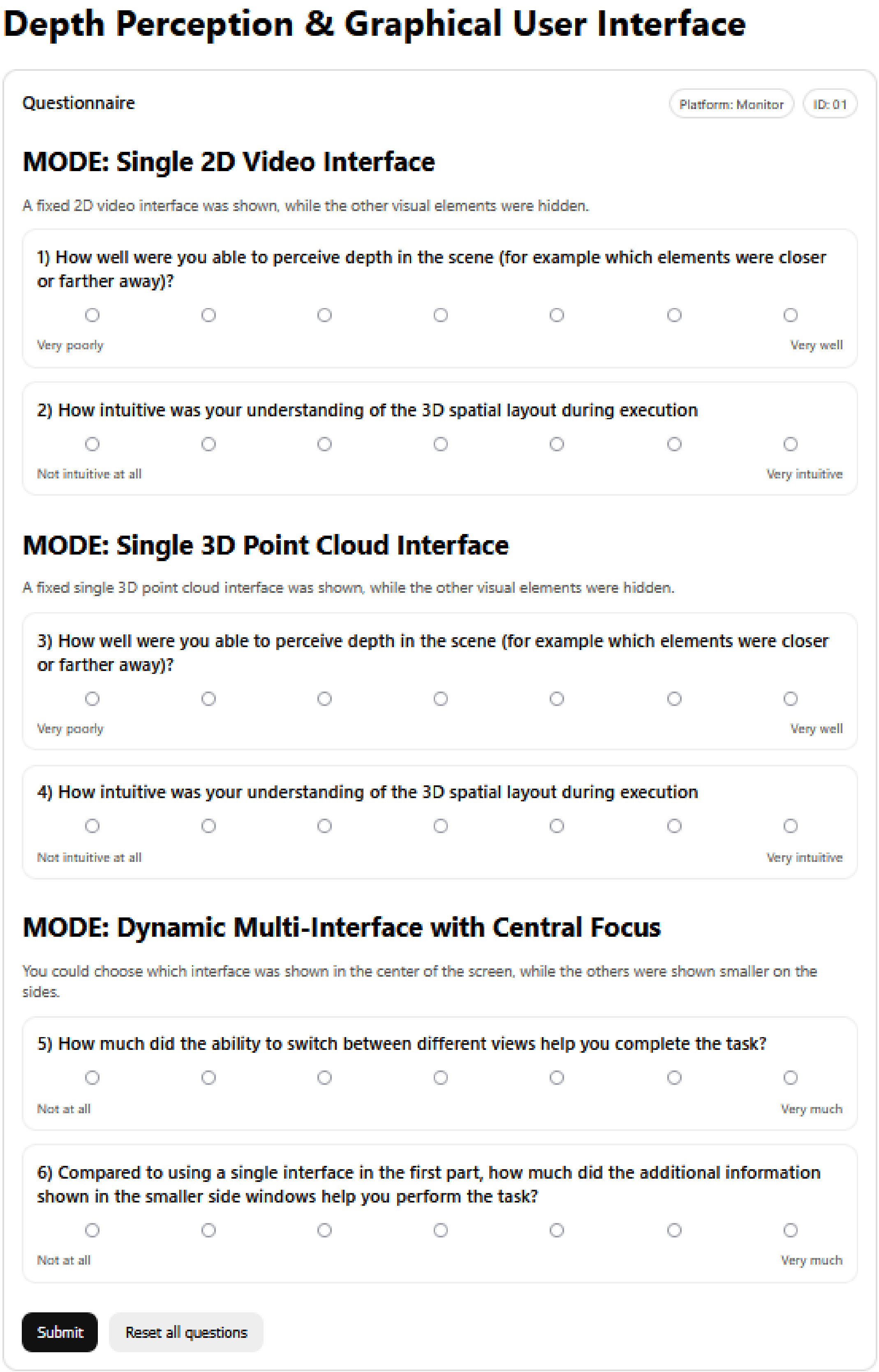
Phase 1 (Display Platform): □

Metric	**1. Run**	**2. Run**	**3. Run**
GUI Mode Video / Pointcloud / Free			
Grasp Attempts Success [✓] / Failure [×]	□ □ □	□ □ □	□ □ □
Obstacle Avoidance Avoided [✓] / Hit [×]	□	□	□
Target Accuracy [cm] 1. Distance from target 2. Release height	□ □	□ □	□ □
Other Notes anomalies, issues, emergency stops, user strategy, comments, ...			

Phase 2 (Display Platform): □

Metric	**4. Run**	**5. Run**	**6. Run**
GUI Mode Video / Pointcloud / Free			
Grasp Attempts Success [✓] / Failure [×]	□ □ □	□ □ □	□ □ □
Obstacle Avoidance Avoided [✓] / Hit [×]	□	□	□
Target Accuracy [cm] 1. Distance from target 2. Release height	□ □	□ □	□ □
Other Notes anomalies, issues, emergency stops, user strategy, comments, ...			

Figure A.3.: Obersvation sheet used to track manually recorded run data during the user study



Depth Perception & Graphical User Interface

Questionnaire

Platform: Monitor ID: 01

MODE: Single 2D Video Interface

A fixed 2D video interface was shown, while the other visual elements were hidden.

1) How well were you able to perceive depth in the scene (for example which elements were closer or farther away)?

○ ○ ○ ○ ○ ○ ○

Very poorly Very well

2) How intuitive was your understanding of the 3D spatial layout during execution

○ ○ ○ ○ ○ ○ ○

Not intuitive at all Very intuitive

MODE: Single 3D Point Cloud Interface

A fixed single 3D point cloud interface was shown, while the other visual elements were hidden.

3) How well were you able to perceive depth in the scene (for example which elements were closer or farther away)?

○ ○ ○ ○ ○ ○ ○

Very poorly Very well

4) How intuitive was your understanding of the 3D spatial layout during execution

○ ○ ○ ○ ○ ○ ○

Not intuitive at all Very intuitive

MODE: Dynamic Multi-Interface with Central Focus

You could choose which interface was shown in the center of the screen, while the others were shown smaller on the sides.

5) How much did the ability to switch between different views help you complete the task?

○ ○ ○ ○ ○ ○ ○

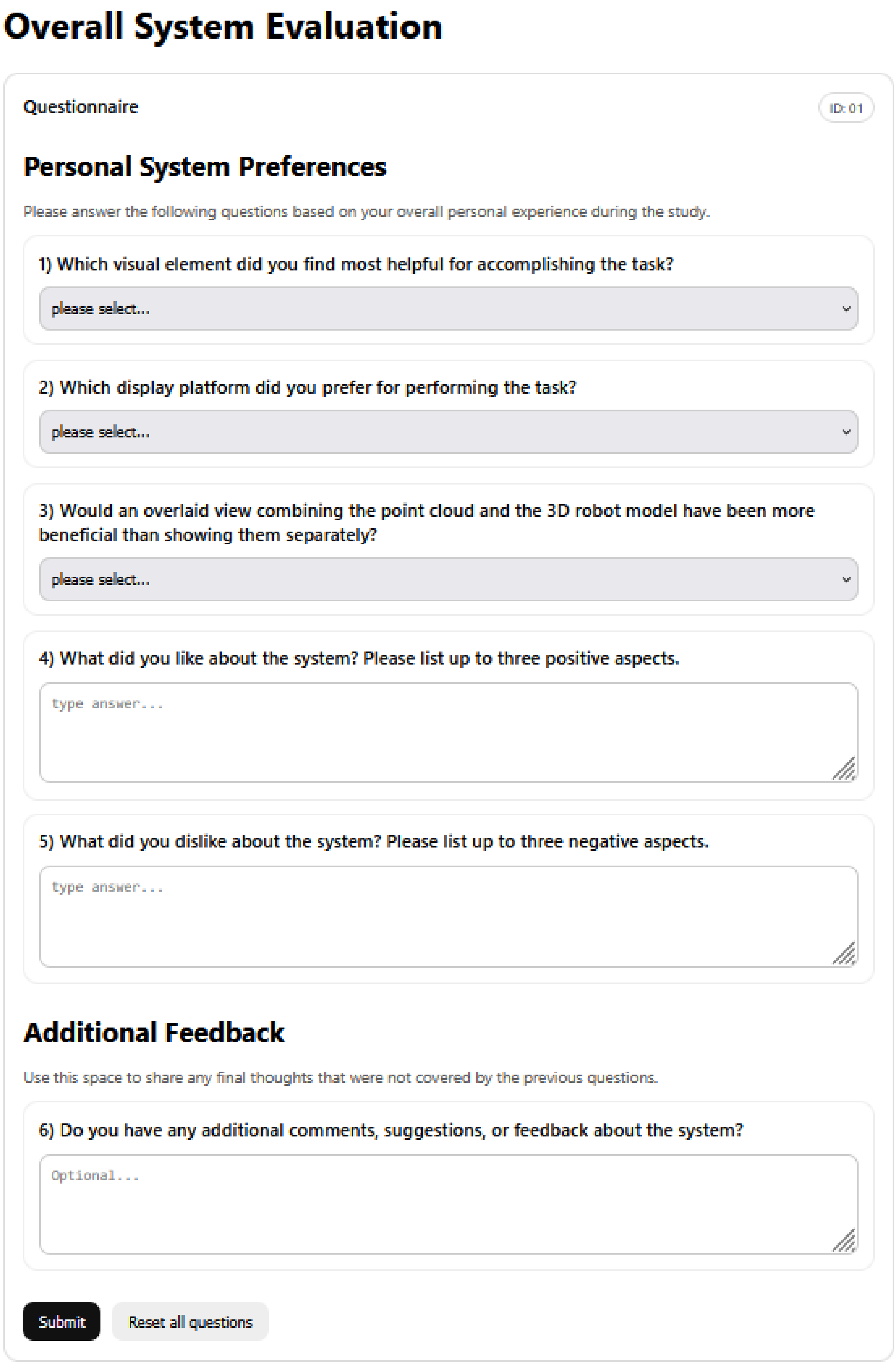
Not at all Very much

6) Compared to using a single interface in the first part, how much did the additional information shown in the smaller side windows help you perform the task?

○ ○ ○ ○ ○ ○ ○

Not at all Very much

Figure A.4.: Depth Perception Questionnaire (DPQ) used for the study



Overall System Evaluation

Questionnaire

ID: 01

Personal System Preferences

Please answer the following questions based on your overall personal experience during the study.

1) Which visual element did you find most helpful for accomplishing the task?

please select...

2) Which display platform did you prefer for performing the task?

please select...

3) Would an overlaid view combining the point cloud and the 3D robot model have been more beneficial than showing them separately?

please select...

4) What did you like about the system? Please list up to three positive aspects.

type answer...

5) What did you dislike about the system? Please list up to three negative aspects.

type answer...

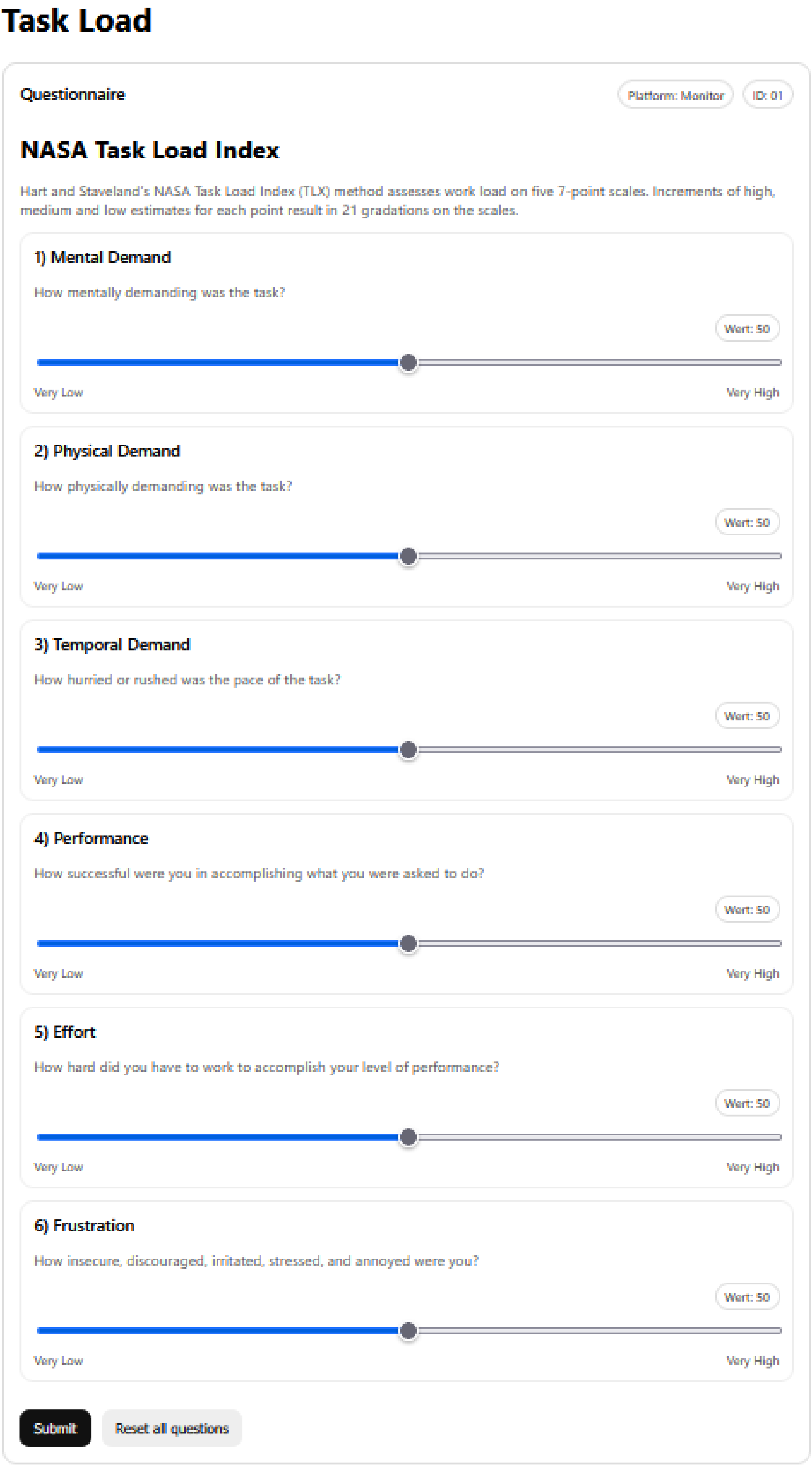
Additional Feedback

Use this space to share any final thoughts that were not covered by the previous questions.

6) Do you have any additional comments, suggestions, or feedback about the system?

Optional...

Figure A.5.: System Evaluation Questionnaire (SEQ) used for the study



Task Load

Questionnaire

NASA Task Load Index

Hart and Staveland's NASA Task Load Index (TLX) method assesses work load on five 7-point scales. Increments of high, medium and low estimates for each point result in 21 gradations on the scales.

1) Mental Demand

How mentally demanding was the task?

Wert: 50

2) Physical Demand

How physically demanding was the task?

Wert: 50

3) Temporal Demand

How hurried or rushed was the pace of the task?

Wert: 50

4) Performance

How successful were you in accomplishing what you were asked to do?

Wert: 50

5) Effort

How hard did you have to work to accomplish your level of performance?

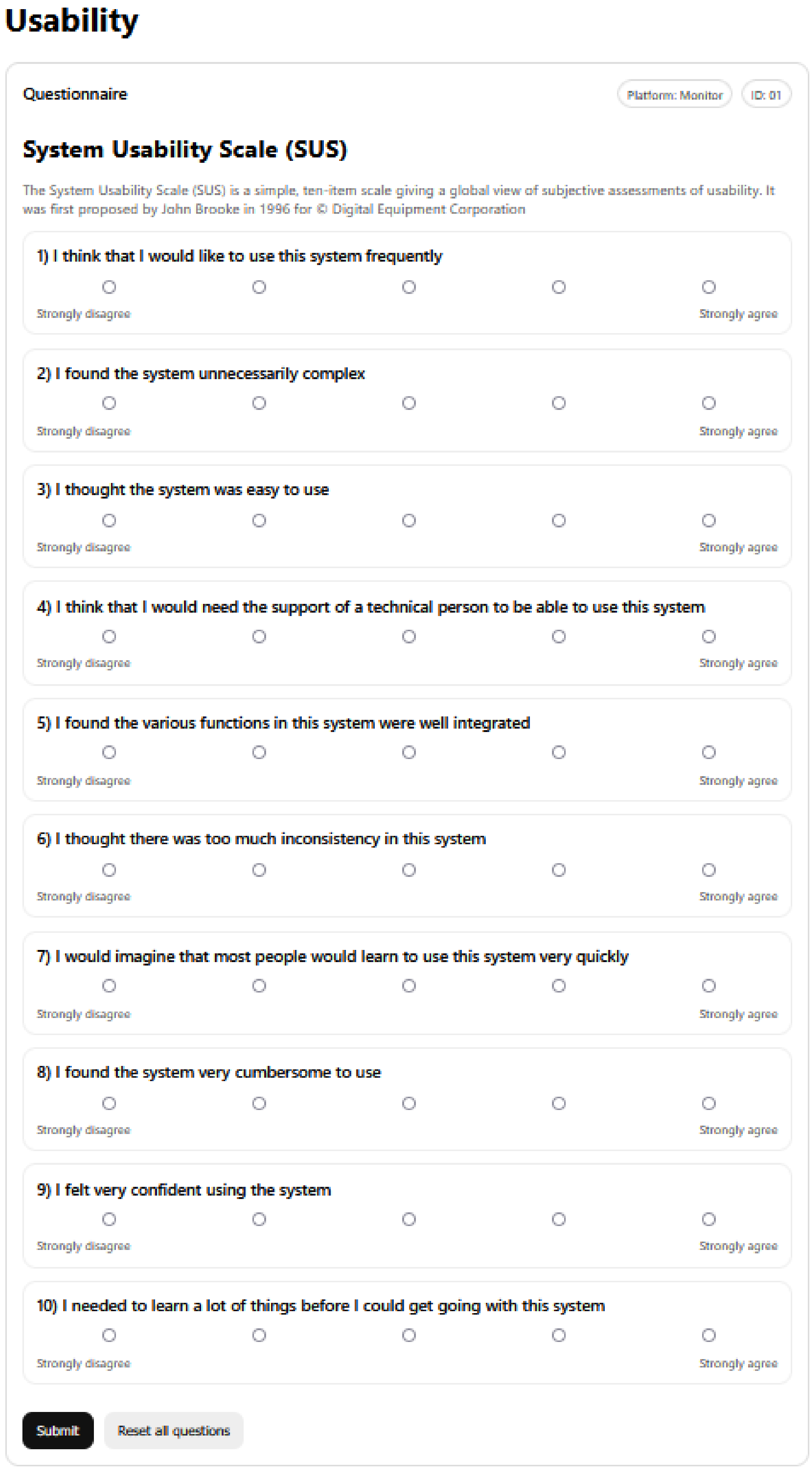
Wert: 50

6) Frustration

How insecure, discouraged, irritated, stressed, and annoyed were you?

Wert: 50

Figure A.6.: Task Load Index Questionnaire (NASA-TLX) as adapted from [HS88]



Usability

Questionnaire

Platform: Monitor

ID: 01

System Usability Scale (SUS)

The System Usability Scale (SUS) is a simple, ten-item scale giving a global view of subjective assessments of usability. It was first proposed by John Brooke in 1996 for © Digital Equipment Corporation

1) I think that I would like to use this system frequently

○ ○ ○ ○ ○

Strongly disagree Strongly agree

2) I found the system unnecessarily complex

○ ○ ○ ○ ○

Strongly disagree Strongly agree

3) I thought the system was easy to use

○ ○ ○ ○ ○

Strongly disagree Strongly agree

4) I think that I would need the support of a technical person to be able to use this system

○ ○ ○ ○ ○

Strongly disagree Strongly agree

5) I found the various functions in this system were well integrated

○ ○ ○ ○ ○

Strongly disagree Strongly agree

6) I thought there was too much inconsistency in this system

○ ○ ○ ○ ○

Strongly disagree Strongly agree

7) I would imagine that most people would learn to use this system very quickly

○ ○ ○ ○ ○

Strongly disagree Strongly agree

8) I found the system very cumbersome to use

○ ○ ○ ○ ○

Strongly disagree Strongly agree

9) I felt very confident using the system

○ ○ ○ ○ ○

Strongly disagree Strongly agree

10) I needed to learn a lot of things before I could get going with this system

○ ○ ○ ○ ○

Strongly disagree Strongly agree

Figure A.7.: System Usability Score Questionnaire (SUS) adapted from [Bro95]

B. Tabellen

C. User Study Procedure Sequence

The sequence of the user study procedure is layed out exemplary. For clarity every activity is allocated to the participant (P) or the experimenter (E).

1. **Preparations**
    - (E) system setup and testing
    - (E) specifiy participant ID and prepare consent form
2. **Introduction**
    - (E) welcome participant
    - (E) explain study concept and procedure
    - (P) sign informed consent form
    - (P) fill out personal data (age, gender, handedness)
3. **System Familiarization**
    - (E) system explanation and demonstration
    - (E) answer questions
    - (P) test GUI interaction
    - (P) test teleoperation input
    - (P) freely familiarize with the system
4. **Task Explanation**
    - (E) explain task in detail
    - (E) advise caution to avoid collisions
    - (E) explain phases and runs
    - (E) answer questions
    - (P) prepared for task execution
5. **Task Execution Runs**
    - **First Phase**: (e.g. Monitor)

  - **Run 1**: (e.g. Video Screen Mode)
    - (E) reset robot to initial condition
    - (E) start unity backend tracking
    - (P) execute teleoperation task
    - (E) watch for collision, emergency stop if necessary
    - (E) measure and note metrics
    - (E) record comments/issues/anomalies
    - (P) task completion or third failed grasp attempt
    - (E) stop teleoperation and backend tracking
  - **Run 2**: (e.g. Point Cloud Mode) ...
  - **Run 3**: (e.g. Free Focus Mode) ...
  - (P) fill out phase questionnaires (DPQ, TLX, SUS)
  - (E) change to other display platform (Mixed Reality setup)
- **Second Phase**: (e.g. Mixed Reality)
  - **Run 4**: (e.g. Video Screen Mode) ...
  - **Run 5**: (e.g. Point Cloud Mode) ...
  - **Run 6**: (e.g. Free Focus Mode) ...
  - (P) fill out phase questionnaires (DPQ, TLX, SUS)
- **System Evaluation**
  - (P) fill out final questionnaire (SEQ)
  - (P) final comments/suggestions
- **Conclusion**
  - (E) thanking and farewell