



**Politécnico
de Viseu**

Escola Superior
de Tecnologia
e Gestão de Viseu

Developing an MR-Based Robotic Control System for Object Grasping Applications

Carlos Daniel Ferreira Silva

Trabalho de Projeto
Mestrado em Sistemas de Informação

Trabalho efetuado sob a orientação de
Professor Ricardo João Luís Marques Correia

Março de 2026



**Politécnico
de Viseu**

Escola Superior
de Tecnologia
e Gestão de Viseu

Developing an MR-Based Robotic Control System for Object Grasping Applications

Carlos Daniel Ferreira Silva

Trabalho de Projeto
Mestrado em Sistemas de Informação

Trabalho efetuado sob a orientação de
Professor Ricardo João Luís Marques Correia

Março de 2026

Acknowledgements

First and foremost, I want to thank my mother. She never stopped believing that I could do this, even when I had doubts myself. Her encouragement and the little pushes at the right moments made a real difference, and they are a big part of why I kept going and finished this Master's degree.

I would also like to thank Professor Ricardo Correia, my supervisor, for his constant support and for genuinely staying involved throughout the entire process. He regularly checked in, asked how things were going, and made sure I did not lose momentum—both in the development work and in the writing. That combination of availability, attention, and accountability helped me stay focused and turn this project into a complete thesis.

I would also like to express my gratitude to the STAR Institute for proposing this project and for providing the conditions that made its development possible. Being welcomed into its facilities and having the opportunity to work directly with the robot were essential to the practical realization of this thesis and greatly enriched the overall experience.

A special thank you goes to Pedro Monteiro for helping me during the writing phase. He took the time to answer questions, clarify details, and point me in the right direction when I was stuck. Without his support, the writing process would have taken considerably longer.

Finally, I would like to acknowledge the School of Technology and Management of Viseu (ESTGV) and the Polytechnic Institute of Viseu (IPV) for the institutional support and conditions provided throughout the degree, which made the completion of this work possible.

Abstract

This dissertation presents the design, implementation, and evaluation of a mixed reality (MR) teleoperation system for controlling a Universal Robots UR5e using a Meta Quest 3S headset and a Unity-based application. The system integrates an XR interaction layer for capturing controller pose and inputs, a communication and control layer based on the UR dashboard interface and RTDE, and a synchronized digital twin rendered in MR to provide continuous visual feedback aligned with the physical workspace. Robot motion is generated through a clutch-based interaction mechanism: when the control button is held, relative controller motion is mapped to incremental robot motion commands; when released, motion commands stop and the user can reposition without affecting the robot. Runtime tuning parameters (e.g., translation and rotation multipliers and jitter thresholds) are exposed through in-headset UI panels to support sensitivity adjustment during operation.

A user study was conducted to assess perceived responsiveness, intuitiveness of translational and rotational control, and overall interface usability. Results indicate that translational control was generally easy to learn and rated as intuitive by most participants. Rotational control was rated positively overall, although roll was identified as the most challenging axis. Perceived delay was typically reported as having low impact, but qualitative feedback highlighted the importance of fine control tuning and interface readability, including accessibility concerns for users with strong vision correction needs. Overall, the findings support MR-based teleoperation with a digital twin as a practical and intuitive interaction paradigm, while highlighting concrete design trade-offs between responsiveness, stability, and interface accessibility.

Keywords: Mixed Reality, Teleoperation, Digital Twin, Universal Robots UR5e, RTDE, Meta Quest 3S, Human-Robot Interaction

Resumo

Esta dissertação apresenta o desenho, implementação e avaliação de um sistema de teleoperação em realidade mista (RM) para controlo de um Universal Robots UR5e utilizando um headset Meta Quest 3S e uma aplicação desenvolvida em Unity. O sistema integra uma camada de interação XR para aquisição da pose e inputs do controlador, uma camada de comunicação e controlo baseada na interface de *dashboard* e em RTDE, e um gémeo digital sincronizado, visualizado em RM, que fornece feedback visual contínuo alinhado com o espaço físico. O movimento do robô é gerado através de um mecanismo de *clutch*: enquanto o botão de controlo está premido, o movimento relativo do controlador é mapeado para comandos incrementais de movimento do robô; ao libertar o botão, o envio de comandos cessa e o utilizador pode reposicionar a mão/controlador sem afetar o robô. Parâmetros de afinação em tempo de execução (por exemplo, multiplicadores de translação e rotação e limiares de jitter) são disponibilizados em painéis de interface dentro do headset para permitir o ajuste de sensibilidade durante a operação.

Foi realizado um estudo com utilizadores para avaliar a responsividade percebida, a intuitividade do controlo translacional e rotacional e a usabilidade global da interface. Os resultados indicam que o controlo translacional foi, em geral, fácil de aprender e considerado intuitivo pela maioria dos participantes. O controlo rotacional obteve avaliações globalmente positivas, embora o eixo de *roll* tenha sido identificado como o mais desafiante. O atraso percebido foi tipicamente reportado como tendo baixo impacto, mas o feedback qualitativo salientou a importância de afinação para controlo fino e da legibilidade da interface, incluindo preocupações de acessibilidade para utilizadores com fortes necessidades de correção visual. No seu conjunto, os resultados suportam a teleoperação em RM com gémeo digital como uma abordagem prática e intuitiva, evidenciando simultaneamente compromissos de desenho entre responsividade, estabilidade e acessibilidade da interface.

Palavras-Chave: Realidade Mista, Teleoperação, Gémeo Digital, Universal Robots UR5e, RTDE, Meta Quest 3S, Interação Humano-Robô

Contents

List of Tables	ix
List of Figures	xi
Listings	xiii
Glossary	xv
List of Acronyms	xix
1 Introduction	1
1.1 Motivation	1
1.2 Contextualization	3
1.3 Problem Statement	4
1.4 Objectives	5
1.5 Expected Results	6
1.6 Work Plan	7
2 State of the Art	11
2.1 Definitions	11
2.2 Methodology and Requirement Analysis	12
2.2.1 Bibliographic Research	12
2.3 Related Work	13
3 System Architecture	19
3.1 System Implementation	19
3.1.1 Hardware Setup	19
3.2 Functional and Non-Functional Requirements	20
3.2.1 Functional Requirements	20
3.2.2 Non-Functional Requirements	20
3.2.3 Software Setup	20
3.2.3.1 Development Tools and Frameworks	20
3.2.3.2 Key Features	21
3.2.3.3 Testing	21
3.3 System Overview	21

3.4	Hardware Architecture	25
3.4.1	Meta Quest 3S HMD	25
3.4.2	UR5e Robotic Manipulator	25
3.4.3	UR5e Robot Specifications	27
3.4.4	Computing and Network Infrastructure	27
3.5	Software Architecture	28
3.5.1	Development Environment	28
3.5.2	Layered Software Structure	28
3.5.3	Digital Twin Representation	29
3.6	Communication Subsystem	29
3.6.1	Dashboard Interface Script	29
3.6.2	Simple Command Control Script	30
3.6.3	RTDE-Based Control	30
3.7	Data Flow and Control Loop	30
3.8	Interaction Logic	33
3.9	Summary	34
4	System Implementation	35
4.1	Overview	35
4.2	Unity Scene and Digital Twin	36
4.2.1	XR Rig and Hand Tracking	36
4.2.2	Development and Debugging Setup	36
4.2.3	UR5e Digital Twin	37
4.3	Communication Modules	39
4.3.1	Dashboard Interface	39
4.3.2	Simple Command Interface	39
4.3.3	RTDE Client and Real-Time Control	40
4.4	User Interface Implementation	40
4.4.1	Overall Layout	40
4.4.2	Connection Panel	40
4.4.3	Dashboard Control Panel	41
4.4.4	Arms Control Panel	41
4.5	Hand-Based Control Logic	41
4.5.1	Clutch Mechanism and Reference Pose	41
4.5.2	Mapping Hand Motion to Robot Commands	42
4.5.3	MR Teleoperation Examples	42
4.6	Summary	44
5	Results	47
5.1	Participants and Test Conditions	47
5.1.1	Evaluation Environment	48

5.2	Usability Results	50
5.2.1	Responsiveness and Perceived Delay (Q5)	50
5.2.2	Translational Control Intuitiveness (Q6–Q8)	51
5.2.3	Rotational Control Intuitiveness (Q9–Q11)	52
5.2.4	Overall Interface Intuitiveness (Q12)	54
5.3	Qualitative Feedback Summary	55
5.4	Summary	55
6	Conclusion and Future Work	57
6.1	Future Work	58
6.2	Final Thoughts	59
	Appendix A Participant Responses (Portuguese)	65
	Appendix B Participant Responses (English translation)	67
	Appendix C Gantt Chart	69

List of Tables

2.1	Comparison of Head-Mounted Displays (HMDs) (Based on Guo et al. (2022))	15
2.2	Systematic comparison of related work	18
3.1	Key technical specifications of the UR5e.	27

List of Figures

1.1	Gantt Chart	9
3.1	Controller input mapping used in the prototype. Translational control is driven by controller pose during the clutch period, while rotational control combines controller roll with joystick-driven pitch and yaw. .	22
3.2	Overall system architecture of the proposed Mixed Reality (MR)-based teleoperation system.	23
3.3	Physical UR5e robotic manipulator.	26
3.4	Physical UR5e robotic manipulator (back view).	26
3.5	Layered software architecture of the implemented system.	29
3.6	Runtime control and communication flow of the teleoperation system.	32
3.7	Hybrid input mapping used for robot teleoperation.	33
4.1	Development and debugging setup used during implementation, showing the UR5e, development workstation, and the HMD.	36
4.2	UR5e digital twin rendered in the Unity scene.	37
4.3	Unity hierarchy of the UR5e digital twin, showing the kinematic chain from <code>base_link</code> to <code>wrist_3_link</code>	38
4.4	MR visualization of the UR5e digital twin overlaid in the environment, shown alongside the physical robot.	39
4.5	MR user interface panels for robot control.	40
4.6	In-headset MR views during live teleoperation of the UR5e. The digital twin is overlaid onto the physical workspace, while axis indicators provide real-time feedback on tool pose and control direction.	43
5.1	Participant interacting with the MR teleoperation system during the evaluation session.	48
5.2	Example of a participant using the HMD and controller while testing the robot teleoperation interface.	49
5.3	Evaluation session showing a participant interacting with the system while observers monitored the test environment.	49
5.4	Q5: Perceived impact of delay on control ability (0–5).	50
5.5	Q6: Intuitiveness of up/down translation (0–5).	51
5.6	Q7: Intuitiveness of left/right translation (0–5).	51

5.7	Q8: Intuitiveness of forward/back translation (0–5).	52
5.8	Q9: Intuitiveness of pitch rotation (0–5).	53
5.9	Q10: Intuitiveness of roll rotation (0–5).	53
5.10	Q11: Intuitiveness of yaw rotation (0–5).	54
5.11	Q12: Overall interface intuitiveness (0–5).	54

Listings

4.1 Joystick-based pitch and yaw mapping corrected by the current wrist roll.	43
--	----

Glossary

Cartesian control

is a robot control approach in which motion commands are specified in terms of position and orientation in three-dimensional space rather than by individual joint values.

clutch-based interaction

is an interaction technique in which user input only affects the controlled system while a specific control button or trigger is held, allowing repositioning without issuing commands.

cobot

is a collaborative robot designed to work safely alongside humans in shared environments, typically supporting tasks that benefit from both automation and human oversight.

control loop

is the recurring sequence in which input is acquired, commands are generated, the system acts, and feedback is returned for continuous operation and adjustment.

controller tracking

is the process of continuously estimating the position and orientation of a handheld controller so that its motion can be mapped to actions in a virtual or robotic system.

coordinate frame

is a reference system used to describe positions and orientations in space, allowing transformations and motion to be expressed consistently between different entities.

dashboard interface

is a high-level robot communication interface used to manage operational functions such as connection, program control, and execution state.

digital twin

is a virtual representation of a physical system that mirrors its state and behaviour in real time, allowing monitoring, simulation, and interaction with the real counterpart.

end-effector

is the device attached to the end of a robotic arm that interacts directly with the environment, such as a gripper, tool, or sensor.

hand tracking

is the process of detecting and interpreting the position, orientation, and movement of the user's hands so they can be used as an input method for interaction.

headset

is a wearable device that combines displays and sensors to provide immersive visualisation and interaction in virtual, augmented, or mixed reality environments.

jitter

is unwanted small and rapid variation in tracked motion or control signals that can reduce stability and precision during interaction.

kinematic hierarchy

is the structured relationship between linked parts of a mechanism, where the motion of one component affects the position and orientation of connected components.

latency

is the delay between a user's input or a system event and the corresponding response produced by the system.

OpenXR

is an open standard for developing cross-platform virtual and mixed reality applications through a common application programming interface.

passthrough

is a feature in mixed reality headsets that allows the user to see the physical environment through camera-based video while digital elements are overlaid onto it.

pose

describes the position and orientation of an object in space, typically expressed relative to a reference coordinate frame.

real-time feedback

is the immediate return of system state information to the user during interaction, helping maintain responsiveness, awareness, and control accuracy.

responsiveness

is the degree to which a system reacts quickly and predictably to user input, contributing to the feeling of direct control.

state feedback

is information returned by the robot about its current condition, such as joint values, tool pose, or execution state, used to monitor and update the control system.

teleoperation

is the remote control of a robot or system by a human operator, typically through an interface that translates user inputs into robot actions.

tool center point

is the reference point on the robot tool used to define its effective working position and orientation during motion planning and control.

Unity

is a real-time development platform used to build interactive 3D applications, including mixed reality interfaces and simulations.

usability

is the degree to which a system can be used effectively, efficiently, and satisfactorily by users to achieve specific goals.

user interface

is the set of visual and interactive elements through which a user communicates with and controls a system.

List of Acronyms

AI	Artificial Intelligence
AR	Augmented Reality
DoF	Degrees of Freedom
EI	Emotional Intelligence
FOV	Field of View
HMD	Head-Mounted Display
HRC	Human-Robot Collaboration
HRI	Human-Robot Interaction
IP	Internet Protocol
LCD	Liquid Crystal Display
LoA	Level of Attention
MR	Mixed Reality
N/A	Not Available
OLED	Organic Light-Emitting Diode
OST	Optical See-Through
PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses
RTDE	Real-Time Data Exchange
TCP/IP	Transmission Control Protocol / Internet Protocol
ViT	Vision Transformer
VR	Virtual Reality
XR	Extended Reality

Chapter 1

Introduction

The convergence of Augmented Reality (AR) and robots possesses significant transformational value for creating more efficient, effective, and accessible robotic-man interfaces. By combining virtual information with real environments in a transparent manner, AR can have a profound impact in changing the way humans drive and work with robotic platforms. In this chapter, an introduction to the proposed work is delivered through a discussion of most significant motivation for integration, discussion of theoretical background, and a statement of a specific challenge addressed in this project. All these serve a basis for problem statement and objectives of this work.

1.1 Motivation

Integrating AR with robots transforms the way humans and machines collaborate, opening new capabilities for industries including manufacturing, medical, and research Fu et al. (2023). Controlling robots effectively has for a long time been a challenge, with traditional techniques having involved cumbersome hardware and sophisticated software that takes a lot of training and is prone to human errors. All these can make robotic technology complex, and in the process, hinder development in industries that could benefit from less complex collaboration between humans and robots Arents and Greitans (2022).

MR takes a new direction with its capacity for projecting virtual information onto reality, and with it, a more organic and instinctual use of robots is facilitated.

Products like the Meta Quest 3S, with its advanced MR capabilities, make for an ideal platform for developing such futuristic interfaces, and with them, one can access robots in a whole new manner. MR utilizes real-time observation of a robot's activity, direct feedback, and even controlling with a simple gesture, reducing the learning curve and enhancing efficiency in operations.

This research arises out of a desire to escape traditional robotic constraint and make use of our native mental and spatial capabilities, and MR can capitalize on them in offering a deeper, less complex experience. By following a robot in an MR environment, a user can have a deeper understanding of its location and capabilities, and act with a high level of accuracy and control, therefore.

MR-powered commands, for example, can enable intuitive teleoperation for positioning and orienting a robot end-effector, which is a practical foundation for a wide range of industrial and research tasks. With such a convenient form of control, flexible and high-precision work becomes less cumbersome than traditional interfaces (e.g., 2D keyboard/monitor or teach-pendant-style interaction), as prior studies report improved usability and/or reduced workload and task time with Virtual Reality (VR)/MR/AR-based control, Hetrick et al. (2020). In the long term, MR-robotics integration can make security and efficiency easier, and allow new channels that have been out of bounds with present technology.

In Human-Robot Collaboration (HRC), security is paramount, particularly in settings in which robots become increasingly present in shared spaces with humans, Martinelli et al. (2024). The integration of MR with robotics further expands the scope of security considerations, encompassing both physical and cognitive factors essential for safe and effective operation Hamad et al. (2024). Physical security involves processes such as collision avoidance, speed and force controls, and stop in case of an emergency, whose overall function lessens accidental peril. Cognitive security, on the other hand, is achieved through predictable motion of robots and transparent feedback systems whose function lessens hesitation in use and instills trust in robotic behavior.

Digital twins are revolutionizing modern industry by generating virtual replicas of physical objects Tao et al. (2022), enabling real-time monitoring and analysis. This technology effectively connects the physical and digital worlds, allowing industries to detect potential issues early, forecast outcomes with greater accuracy, and optimize operations. Its impact is especially significant when integrated with MR and robotics, as it enables improved visualization and control of robotic systems, resulting in greater precision and adaptability. By replicating the arm's movements and interactions in real time, it provides instant feedback, ensuring actions are accurate and controlled Zhao et al. (2024). This mirroring improves not only the precision of the robot's movements but also the safety and efficiency of Human-Robot Interaction (HRI) Zhao et al. (2024). With more intuitive and precise control

over robots, digital twins are enabling interactions with machines in ways that feel more natural and, ultimately, more effective.

1.2 Contextualization

Once considered a futuristic technology, AR is today a beneficial tool in numerous industries such as healthcare, education, manufacturing, and entertainment Villagran-Vizcarra et al. (2023). Robotics revolutionized industries with its unparalleled accuracy, efficiency, and dependability, at times even besting humans Song and Zhao (2024). Innovations such as the Meta Quest 3S have continued to build MR technology, with additional capabilities such as gesture tracking and spatial mapping. All these contribute to immersive experiences blending virtual and real environments even more.

In healthcare, AR is a valuable tool for surgeons projecting critical information during surgical operations and improving precision and care of patients Vávra et al. (2017). In education, AR makes learning interactive and engaging, helping students understand complex topics more effectively Bacca et al. (2014). In manufacturing, AR aids in maintenance, training, and complex assembly work, reducing errors and improving productivity Wang et al. (2016).

Robotics has greatly affected industries including manufacturing, logistics, and service. In factories, robots undertake repetitive and dangerous work, both enhancing productivity and security Ajewole et al. (2023). In healthcare, service industries, and even in home settings, service robots are improving quality of life and operational efficiency Holland et al. (2021). Conventional robots, however, require specialized skills to operate, and in most instances, such expertise can make them less accessible Goodrich and Schultz (2007).

Combining AR and robotics offers a promising solution to this problem. AR can simplify robotic system controls, creating interfaces that can be operated with ease even without technical expertise Bassyouni and Elhajj (2021). With real-time visualization of actions, AR promotes situational awareness and aids in decision-making Chang and Hayes (2024).

While research has proven that VR can enable robot teleoperation by allowing operators to remotely manipulate robots with interactive visual feedback Wonsick and Padir (2020), AR extends this by overlaying digital cues onto reality. Research further reveals that AR can enhance performance in a task, decrease mental burden, and improve learning in robot programming Jeffri and Awang Rambli (2021).

This research overcomes such barriers through a synergy between a robotic platform and a Meta Quest 3S by providing ease of controls, creating a MR environment, and facilitate manipulation of objects. Leveraging the advanced AR features of the Meta Quest 3S, the system allows the user to control the robot using gestures within

a fully immersive MR setting. MR environment mirrors in real-time the motion of a robot, seamlessly blending virtual and real worlds in a manner that permits real-time feedback for optimized accuracy in controls.

The study contributes to the field through exploration of real-life applications of MR in robotic control and demonstration of MR advantages over traditional AR approaches. With MR, the system creates a deeper and more immersive environment in which virtual and real elements coexist and dynamically respond to each other Parveau and Adda (2018). This type of integration creates a deeper level of user understanding and collaboration with the robot, and controlling it feels more organic and instinctual in comparison to AR systems that simply overlay information, with less integration.

1.3 Problem Statement

Although AR and robotics have advanced significantly, creating a seamless system that allows people to control and interact with robots in real time via devices like the Meta Quest 3S remains a challenge. Traditional robotic control systems often require complex training and lack real-time, meaningful feedback, making them hard to use and limiting their adoption in areas where robots could bring major benefits Goodrich and Schultz (2007).

The need for robots that can be easily controlled and adapted to different tasks is growing in sectors like manufacturing, healthcare, logistics, and service industries Holland et al. (2021). A user-friendly AR-based control system could make robots more accessible, increase productivity, and reduce errors. However, creating this kind of system is complex because it involves blending advanced technologies, including:

- **Intuitive Control Interfaces:** The challenge is to design AR interfaces that let people control robots naturally-using gestures;
- **Real-Time Virtual Environment Synchronization:** Developing a virtual model of the robot that precisely matches its physical movements in real-time involves complex spatial mapping and synchronization of data;
- **Task-Level End-Effector Control:** Enabling robust 6-Degrees of Freedom (DoF) control of the robot tool based on MR inputs requires careful mapping, stability constraints, and predictable feedback to support precise operation.

Some progress has been made toward using AR for robot visualization and remote operation, but despite recent progress, end-to-end systems that integrate direct control, virtual modelling, and object manipulation on devices such as the Meta Quest 3S are still relatively scarce, with many approaches remaining at a prototype stage.

- **Seamless Integration:** Achieving seamless integration of AR and robotic systems remains a challenge, especially for complex tasks like object manipulation Costa et al. (2022)
- **Precise Tracking and Calibration:** Ensuring precise tracking and calibration to avoid discrepancies between virtual and physical actions is crucial for safety and accuracy Liu et al. (2020)
- **User Acceptance:** Increasing user acceptance and comfort with AR interfaces is an ongoing area of research Quandt and Freitag (2021)
- **Advanced Technologies:** Developing more advanced technologies to provide realistic and immersive AR experiences is essential for improving HRI Suzuki et al. (2022)

This study aims to address these gaps by answering the following research questions:

- How can MR be integrated with robotics to create a control interface that's intuitive and efficient using the Meta Quest 3S?
- What methods can be used to build a real-time virtual environment that accurately mirrors a robot within an AR system?
- How can MR inputs be mapped to stable and precise 6-DoF control of the robot, while maintaining an intuitive user experience and reliable feedback?
- What technical challenges exist in synchronizing AR inputs with robotic actions, and how can these challenges be addressed?

1.4 Objectives

Create a MR system that makes controlling and manipulating a robot easy and natural, using the Meta Quest 3S headset.

- Develop a dependable communication system between the Meta Quest 3S and the robot, allowing smooth transmission of control commands.
- Build a real-time virtual environment that mirrors the physical robot precisely within the AR user interface.
- Design and implement a real-time MR teleoperation pipeline that maps user input to UR5e end-effector motion through Real-Time Data Exchange (RTDE), supported by a synchronized digital twin for feedback.

- Craft a user-friendly interface that leverages AR features like gestures to make controlling the robot straightforward.
- Sync the virtual environment with the physical movements of the robot, ensuring real-time feedback and a seamless user experience.
- Test the system’s performance, focusing on how easy, responsive, and accurate it is in controlling the robot and handling objects.

Although the system is designed as a foundation for object grasping applications, this work focuses on implementing and evaluating the MR teleoperation pipeline (6-DoF motion control and digital twin feedback), while grasp execution is left for future work.

1.5 Expected Results

By achieving the goals outlined in this project, we aim to create a MR system that significantly simplifies and enhances robot control and object manipulation through the Meta Quest 3S headset. Establishing a dependable communication link between the Meta Quest 3S and the robot should allow for seamless, real-time command transmission, ensuring low-latency responsiveness and a stable connection that keeps users engaged with a smooth experience.

Our creation of a real-time virtual environment, which will mirror the physical robot’s actions within the AR interface, is designed to help users intuitively see and understand what the robot is doing. This dynamic visualization will foster a more natural and immediate interaction, giving users clear feedback on the robot’s actions.

By creating algorithms that let the robot respond directly to the user’s hand movements, captured through the Meta Quest 3S’s hand tracking technology, it will be possible to expand what the robot can do, allowing it to handle more precise and intricate tasks. Designing an interface that uses AR features like hand tracking and gesture recognition should make controlling the robot feel more intuitive and natural. This approach will lower the learning curve, making it easier and more efficient for users of all experience levels to interact with and guide the robot.

Synchronizing the virtual environment with the physical robot’s movements will provide real-time feedback, giving users an immediate reflection of any actions performed by the robot in the AR space. This instant feedback is crucial to enhancing awareness and ensuring accuracy in control.

Finally, by testing the system’s usability, responsiveness, and accuracy, it is expected to highlight improvements over traditional robot control methods. The expectations from this work are that users will find the system easier to navigate, feel more satisfied with its precision, and see the potential for real-world applications.

Ultimately, this project seeks to showcase how AR and robotics integration can lead to a more intuitive, effective control system-potentially offering new insights and advancements in AR, robotics, and human-computer interaction.

1.6 Work Plan

The work carried out in this dissertation followed four continuous phases: *Research & Planning*, *Unity & XR Foundations*, *Development*, *Integration and Testing*, and *Finalization*. In addition to the implementation milestones, dissertation writing was carried out continuously throughout all phases, with a final consolidation and editing period in the *Finalization* phase. The main activities within each phase are outlined below.

- **Research & Planning**

- **Project scoping and requirements definition:** Define scope, functional/non-functional requirements, constraints (latency, stability, network reliability), and success criteria for the MR teleoperation prototype.
- **Literature review / state of the art:** Review MR/Extended Reality (XR) teleoperation, digital twin for robotics, UR communication interfaces, and usability evaluation methods to inform the system design and study design.
- **System architecture design:** Specify the end-to-end pipeline (XR input sampling, command generation, robot communication, feedback, and digital twin), including separation of concerns between UI, control logic, and robot interfaces.
- **Technology and tooling selection:** Select Unity + OpenXR/Meta runtime, decide on UR dashboard for lifecycle control and RTDE for real-time exchange, and establish the development workflow and tooling for debugging/logging.
- **Initial evaluation design planning:** Define candidate tasks, metrics (perceived delay, controllability, usability), and questionnaire structure, including practical considerations for safe lab testing.

- **Unity & XR Foundations**

- **Unity learning and prototyping:** Build familiarity with Unity transforms, coordinate frames, XR rig setup, and controller input acquisition needed for continuous teleoperation.
- **MR interface prototyping:** Prototype in-headset panels and interaction flows for connection management, robot control actions, and runtime tuning.

- **Digital twin integration in Unity:** Import the UR5e model, set up the kinematic hierarchy, and implement joint-driven updates so the twin can mirror the real robot state.
- **Control mapping analysis:** Explore clutch-based relative control, translation mapping, and orientation handling (yaw/pitch/roll), including strategies to keep inputs intuitive under roll.
- **Networking groundwork:** Prototype connectivity and messaging patterns, prepare the code structure for robot-side integration, and implement diagnostics to support later debugging with the physical UR5e.

- **Development, Integration and Testing**

- **Robot communication stack implementation:** Implement dashboard connectivity for robot/program lifecycle control and integrate RTDE for structured state feedback and command streaming, including watchdog/keepalive behaviour.
- **Real-time teleoperation control loop:** Implement the runtime loop from XR sampling to command generation and transmission, ensuring predictable updates and consistent control when the clutch is engaged.
- **Robust input mapping & tuning:** Refine mapping for translation and rotation (including hybrid rotation: roll via controller orientation and pitch/yaw via joystick), apply deadzones, smoothing, and adjustable gains, and calibrate frame alignment for intuitive behaviour.
- **MR visual feedback and usability aids:** Synchronize the digital twin from robot feedback, add on-screen status/diagnostics, and include visual cues to help users understand current tool pose and command direction.
- **System validation and iterative debugging:** Perform repeated end-to-end validation with the real robot, addressing edge cases such as jitter vs responsiveness, stability trade-offs, and connection robustness.
- **User study execution:** Conduct user testing sessions, collect quantitative ratings and qualitative feedback, and consolidate results for analysis and reporting.

- **Finalization**

Refine the final thesis chapters, finalize diagrams/figures and appendices, and incorporate feedback from review iterations.

An overview of the project timeline is shown in Figure 1.1. For readability, a full-size version of the Gantt chart is provided in Appendix C.

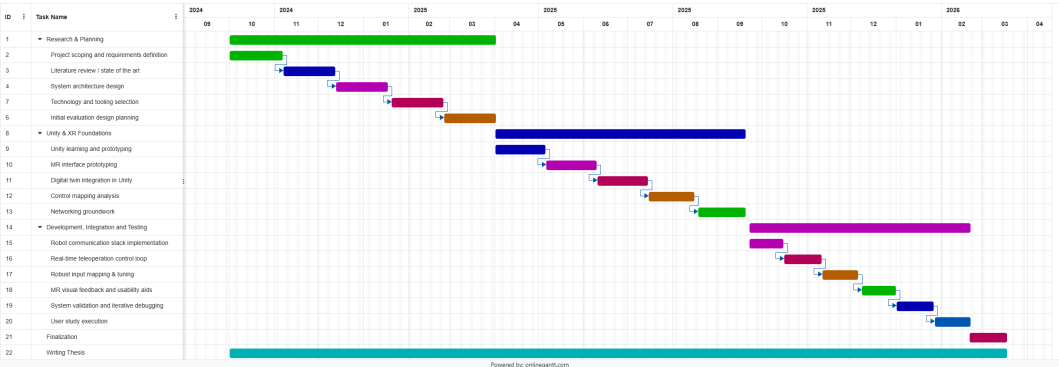


Figure 1.1: Gantt Chart

Chapter 2

State of the Art

This chapter presents the methodological and theoretical foundation of the work. It first introduces the main definitions relevant to AR/MR-based HRI and robot teleoperation, then describes the methodological approach adopted for the development of the proposed MR teleoperation system, including the bibliographic research process. Finally, it reviews related studies and the technological context that support the design and development of the proposed system.

2.1 Definitions

XR is a collective term used to describe immersive technologies that merge or replace perceptions of the physical world, including VR, AR, and MR. VR places the user in a fully simulated digital environment, effectively replacing the surrounding physical world. In contrast, AR enhances the real environment by superimposing digital elements onto it while maintaining the user's perception of the physical space. MR lies between these concepts, enabling virtual and real objects to coexist and interact in ways that are spatially integrated. Therefore, XR can be understood as the overarching category under which VR, AR, and MR are positioned according to the level of immersion and integration with reality.

AR is a technology that overlays digital information on real environments in a form that enriches a view of reality through a combination of real and virtual items in one, unobtrusive and interactive environment Manring et al. (2020). Hand tracking, at the heart of AR's techniques for interaction, allows for virtual information

manipulation with natural motions in place of real controllers. Hand motion and interpretation are captured with this technology, and thus AR interactions become fluent and easier Soares et al. (2021).

Such advancements in interaction become increasingly significant in the case of HRI, in which efficient and effective communications between humans and machines and between machines and robots become a necessity. By tracking hand-movements in AR, HRI can make even more transparent and organic collaborations, in which humans can understand and maintain robotic systems with ease Soares et al. (2021). In collaborative robots (cobots), for safe collaboration with humans in factories, for example, such an improvement in interaction is particularly important. Cobots combine robotic efficiency and accuracy with the adaptability and intelligence of humans, taking care of complex and repetitive tasks and offering security and adaptability Robotec (2024).

To further enable collaboration between humans and robots, Emotional Intelligence (EI) is becoming a new direction in robotics. With an analysis of face and hand movements of an operator, robots with EI can assess human intention and Level of Attention (LoA) and enable flexible and adaptable behavior Antonelli et al. (2024). All these capabilities depend in most cases on complex machine architectures such as Vision Transformer (ViT), processing information through partitioning an image into a sequence of patches and processing them sequentially. The perceptual accuracy of Artificial Intelligence (AI) systems is enhanced with this mechanism, and robots can understand human actions and make fact-based decisions in real time Antonelli et al. (2024).

2.2 Methodology and Requirement Analysis

2.2.1 Bibliographic Research

This systematic review followed the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) 2020 guidelines to ensure transparency, replicability, and robustness in the research process.

The eligibility criteria focused on peer-reviewed journal articles, conference papers, and systematic reviews published in English that addressed AR, MR, HRI, Digital Twin Technology, Gesture Recognition, and Robot Teleoperation. Excluded were non-peer-reviewed publications (e.g., white papers, preprints), studies unrelated to robotics or AR/MR applications, and articles without substantial methodological insights or practical applications.

The bibliographic research process began by identifying seminal works and recent advancements in AR, robotics, and MR applications. Foundational articles such as Azuma (1997)'s survey on AR applications provided critical insights into the potential of AR in various fields such as medicine, manufacturing, and visualization.

Additional sources like Song and Zhao (2024) highlight advancements in robotics applications, which emphasize their impact across industry and healthcare. Villagran-Vizcarra et al. (2023) provided a comprehensive analysis of AR implementations, challenges, and opportunities in education, marketing, and industrial applications. These resources, among others, were selected from reputable databases, ensuring a robust theoretical foundation for the study.

The sources for the bibliographic research were used for their reliability and relevance to the objectives. Platforms like MDPI and ScienceDirect provide extensive peer-reviewed articles on AR and robotics and ensuring access to high-quality, cutting-edge research. Additionally, Google Scholar and IEEE Xplore were utilized to identify seminal works and diverse publications, while offering robust repositories of technical papers related to augmented and MR technologies. These sources ensure a comprehensive and credible foundation for the study.

For each bibliographic element, relevant keywords were identified to streamline the search process and ensure a focused collection of literature. Keywords like *Augmented Reality*, *AR*, *Mixed Reality*, *MR*, *Human-Robot*, *Human-Robot Interaction*, *HRI*, *Digital Twin*, *Robot Control*, *Object Manipulation*, *Collaborative Robotics*, *Cobots*, *Gesture Recognition* and *Hand Tracking*. These keywords were selected based on their relevance to the objectives of the study and their frequent appearance in state-of-the-art literature in AR and robotics. They ensured that the collected materials addressed the core themes and provided actionable insights for the development of the MR system.

The collected bibliography was reviewed and summarized to extract the most relevant information. The documents were analyzed to identify their contributions to the study's themes, such as AR/MR integration, HRI, and robotics applications. The focus centered on summarizing key findings, identifying gaps in the existing research, and contextualizing insights to align with the project's objectives. This approach provided a clearer understanding of the state of the art while ensuring that only the most impactful studies contributed to the development of the MR system.

2.3 Related Work

In the study Soares et al. (2021) the authors developed an AR system that enables operators without programming knowledge to program industrial robots through demonstration. Using the Microsoft HoloLens 2, the system records an operator's hand movements to capture desired trajectories, translating them into executable robot programs. This approach allows users to draw geometric figures that the robot replicates, simplifying robot programming and fostering intuitive HRI.

Similarly, Manring et al. (2020) address the challenges of human supervision and remote control of robots in high-risk dynamic scenarios, which traditionally rely on

joystick interfaces requiring significant training. By employing an AR-based control system, they enhance usability and reduce training demands, enabling intuitive gesture-based interactions. This strategy provides a creative platform for developing more practical and user-friendly robot control methods.

Building on these efforts, De la Cruz et al. (2022) developed a mobile application utilizing AR to facilitate learning in operating industrial robotic arms. The application leverages advanced control algorithms that help users simulate trajectories, monitor errors, and visualize animations for better comprehension. This intuitive AR environment supports interactive training experiences, demonstrating its potential as a practical educational tool in robotics.

Focusing on accessibility, Zhang et al. (2023) present a hands-free control system for the Boston Dynamics Spot robot using the HoloLens 2. By integrating eye gaze and head motion tracking for navigation and robotic arm control, as well as voice commands for mode switching, they enable users to guide the robot in ways not possible with traditional interfaces. Although tasks took longer to complete than with joystick controls, the experience was comparable, demonstrating the viability of MR technologies for inclusive robot control solutions.

Moving beyond programming and accessibility, Antonelli et al. (2024) introduce an EI system for collaborative robots (cobots) to enhance safety in HRI. By assessing an operator's LoA through a ViT trained on facial expressions and hand gestures, the EI system adapts the cobot's behavior to maintain safe and efficient collaboration. Integrating this system into a digital twin of an Omron TM5-700 cobot shows how EI can refine trajectory planning and attention management, advancing the overall effectiveness of human-robot partnerships.

To understand the various HMDs, the authors of Guo et al. (2022) present a comprehensive survey on the measurement and evaluation of AR and MR technologies under various environmental conditions. Their work highlights the role of HMDs in the advancement of AR/MR applications, emphasizing how different environmental factors affect user perception and system performance. By reviewing 87 studies over 32 years, they propose a classification system that categorizes HMDs based on four environmental factors and their impact on user interactions. To provide a comparative analysis of HMD performance, Guo et al. present key specifications for various devices, including display type, resolution, Field of View (FOV), tracking method, weight, and intended usage Guo et al. (2022). Table 2.1 summarizes this comparison and extends it by including the Meta Quest 3S, which is the HMD selected for this work, and the Magic Leap 2, which was also considered during the decision-making process.

The authors of Wang and Dunston (2009) examine the comparative effectiveness of virtual environments based on MR in collaborative design. In the experiment, two MR systems for real-time collaboration and communication, face-to-face and

HMD Model	Display	Resolution	FOV	Tracking	Weight	Usage
NVIS nVisor	LCD	1280x1024 per eye	60°	Optical	N/A	Military/ Industrial
Google Glass	OST	640x360	N/A	Camera-Based	50g	Wearable/ Computing
HTC Vive	OLED	2160x1200	110°	External Sensors	555g	Gaming
HoloLens 2	OST	2048x1080 per eye	52°	Inside-Out	566g	Augmented Reality
Oculus Rift	OLED	2160x1200	110°	External Sensors	470g	Gaming
Oculus Quest 2	LCD	1832x1920 per eye	97°	Inside-Out	503g	Standalone VR
Meta Quest 3S	LCD	1832x1920 per eye	96°	Inside-Out	514g	Standalone VR
Magic Leap 2	OST	1440x1760 per eye	70°	Inside-Out	260g	Enterprise/ Industrial

Table 2.1: Comparison of HMDs (Based on Guo et al. (2022))

remotely, are designed and evaluated in the experiments. It is revealed that these systems speed up the process of detecting errors in design tasks and also reduce the cognitive effort required compared to traditional methods like paper-based drawings and commercial tools such as NavisWorks Roamer. It also reveals that MR technology can improve spatial awareness and facilitate smoother information exchange, making it a promising tool for fields like engineering, architecture, and construction. Furthermore, the knowledge gained from this research is valuable for HRI and other MR applications, particularly for streamlining remote collaboration and minimizing the mental load involved in complex tasks.

To complement the discussion above, Table 2.2 provides a systematic comparison of the reviewed studies in terms of device, control method, target robot or platform, and evaluation focus.

System	Device	Control Method	Robot / Platform	Evaluation
Soares et al. (2021)	Microsoft HoloLens 2	Programming by demonstration using an AR application that records the operator's hand movements/- path coordinates and translates them into robot programs	Universal Robots UR5; ABB IRB 2600	Concept validation through experiments where the user draws geometric figures (triangle, rectangle, 3D solid) and the robot replicates the recorded path
Manring et al. (2020)	Microsoft HoloLens	AR-based manual and automatic robot control; the user can either directly manipulate the holographic end effector or preview and execute a planned motion	Yaskawa Motoman SIA5D	Compared against joystick control; secondary sources report reduced mission completion time and assessment of usability and operator workload
De la Cruz et al. (2022)	Mobile device / smartphone	AR mobile application for trajectory selection, motion simulation, animation, and control-error visualization for training in robotic-arm operation	Industrial robotic manipulators	Application tests reported intuitive use and easier understanding of the operation of programmable robotic manipulators

Continued on next page

System	Device	Control Method	Robot / Platform	Evaluation
Zhang et al. (2023)	Microsoft HoloLens 2	Hands-free MR robot control using eye-gaze tracking for navigation and head-motion tracking for robot-arm control	Boston Dynamics Spot	Compared against joystick control; reported comparable performance in time efficiency and user experience
Antonelli et al. (2024)	ViT-based vision system with digital twin	LoA-aware adaptive trajectory and speed selection based on a ViT system estimating the operator's attention/emotional state	Omron TM5-700 cobot	Tested on a pick-and-place task in simulation and then experimentally validated with the physical cobot
Guo et al. (2022)	Multiple HMDs (survey)	Not applicable; survey of AR/MR user-measurement studies with HMDs under different environmental factors	Not robot-specific	Survey of 87 papers over 32 years; proposes a classification using four environmental factors, user-feedback presence/absence, and ten study-characteristic categories

Continued on next page

System	Device	Control Method	Robot / Platform	Evaluation
Wang and Dunston (2009)	Two MR environments	Collaborative design review in a shared MR workspace	Non-robot collaborative design task	Comparative study with two experiments assessing the effectiveness of MR environments for collaborative design and error-detection tasks

Table 2.2: Systematic comparison of related work

The reviewed studies demonstrate the increasing role of AR and MR in improving robotic interaction, control methods, and accessibility. Research on gesture-based control Soares et al. (2021); Manring et al. (2020); De la Cruz et al. (2022) highlights how AR can simplify robotic programming and improve user experience, reinforcing the importance of intuitive control mechanisms in robotic manipulation. Furthermore, advances in hands-free and adaptive interfaces Zhang et al. (2023) provide insight into accessibility improvements, relevant to designing more inclusive and flexible control systems. The integration of EI in robotics Antonelli et al. (2024) further emphasizes the importance of real-time human intent recognition, a concept that aligns with efforts to improve natural HRC. Similarly, Wang and Dunston (2009) shows improvements in collaboration using MR environments, which reduce cognitive load and optimize real-time interactions, which are also critical considerations for a better HRI in MR-based systems. Lastly, the comparative analysis of HMDs Guo et al. (2022) supports the selection of the Meta Quest 3S for this work, ensuring that the hardware chosen is in line with performance and usability considerations. By incorporating these findings, this study builds upon existing work to develop an MR-based robotic control system that leverages intuitive interaction, real-time visualization, and adaptive control mechanisms.

Chapter 3

System Architecture

This chapter presents the system architecture of the MR application designed to enhance HRI through intuitive teleoperation and aligned digital twin feedback. The architecture encompasses both hardware and software components, detailing their integration to create a seamless MR environment. The chapter outlines the key modules, data flow, and interaction mechanisms that underpin the system's functionality, with particular emphasis on continuous hand tracking and button-activated robot control.

3.1 System Implementation

The implementation of the MR system involves the integration of hardware and software components to create an interactive MR environment. This process can be divided into the following stages:

3.1.1 Hardware Setup

The Meta Quest 3S will be configured as the primary interface for the MR system. Key features utilized included:

- Hand Tracking and Gesture Recognition: Enabled intuitive control of the robotic system.
- Spatial Mapping: Facilitated accurate overlay of virtual objects onto the physical environment.
- High-Resolution Display: Provided users with a clear and immersive MR experience.

- **Affordability:** The Meta Quest 3S offers advanced MR features at a comparatively lower cost, making it an accessible solution for research and application development.

The Microsoft HoloLens 2 was initially considered due to its advanced features, such as eye tracking. However, it was ultimately discarded because the device has been discontinued. Similarly, the Magic Leap 2 was evaluated for its industrial-grade capabilities but was set aside as its higher cost did not justify the additional benefits for this application.

3.2 Functional and Non-Functional Requirements

The system will be designed to meet the following requirements:

3.2.1 Functional Requirements

- Enable real-time control of the robotic platform using MR inputs.
- Provide an accurate visualization of the robot's movements within the MR environment.
- Expose real-time system status and diagnostic information within the MR interface (e.g., connectivity, control mode, and active parameter values) to enable transparent monitoring during teleoperation.

3.2.2 Non-Functional Requirements

- **Safety:** Integrate safety mechanisms within the application to minimise risk to the user and bystanders, and to reduce the likelihood of damage to the robot.
- **Ease of Use:** The intuitiveness of the control interface and the learning curve for new users.
- **System Responsiveness:** The latency between user inputs and the robot's actions.
- **Task Performance:** The accuracy and efficiency of the robot in completing teleoperation tasks.

3.2.3 Software Setup

The software development environment for the MR system was designed to leverage the advanced capabilities of the Meta Quest 3S and to ensure seamless integration with the robotic platform. This section outlines the tools, frameworks, and configurations to be utilized to achieve a robust and responsive MR application.

3.2.3.1 Development Tools and Frameworks

Unity will be used as the primary development platform for creating the MR application due to its flexibility, extensive library of assets, and compatibility with the Meta Quest 3S. It enables the implementation of complex 3D environments and real-time interaction. Meta OpenXR, an extension of OpenXR, will be used to simplify the integration of MR-specific

functionalities, such as hand tracking, gesture recognition, and spatial mapping. This toolkit provided a framework for rapid prototyping and ensured compatibility with the Meta Quest 3S hardware.

3.2.3.2 Key Features

The application incorporated the Meta Quest 3S's built-in hand-tracking capabilities to enable intuitive user interactions with the robotic system.

- Algorithms will be developed to ensure that the physical robot's movements are mirrored precisely within the MR interface. This involves optimizing communication between the device and the robotic platform to minimize latency.
- The interface will be designed with usability in mind, using gesture-based controls to lower the learning curve and improve interaction efficiency.

3.2.3.3 Testing

The software will undergo iterative testing cycles to ensure stability, performance, and user satisfaction, such as evaluating response times between user inputs and robot actions to ensure real-time synchronization and conducting usability tests with participants to refine interface design and interaction mechanics.

3.3 System Overview

The proposed system enables a user wearing a Meta Quest 3S HMD to visualize and control a UR5e robotic arm within a shared physical and virtual environment. A digital twin of the UR5e is rendered in the MR scene and kept synchronized with the physical robot. The system continuously tracks the user's hand or controller pose, but the robot only moves when a specified button is actively held down.

When the grip button is pressed, the system records the current hand pose as a reference, and subsequent hand movements are interpreted as relative displacements with respect to this pose. Subsequent controller translations are mapped to robot tool center point (TCP) translations (forward/backward, left/right, and up/down), as illustrated in Figure 3.1(a). Rotational control is split across inputs: roll is derived from controller orientation (roll left/right), while pitch and yaw are controlled using the controller joystick to enable orientation adjustment without requiring large wrist rotations. The direction conventions used for these rotations are shown in Figure 3.1(b). From the robot's frame of reference, movements to the right, left, up, or down are mapped so that "right" from the user's point of view corresponds to "right" from the robot's perspective, ensuring a consistent and intuitive mapping between user input and robot response.



(a) Translational mapping and clutch (grip) input. (b) Rotational mapping: roll from controller orientation, pitch/yaw via joystick.

Figure 3.1: Controller input mapping used in the prototype. Translational control is driven by controller pose during the clutch period, while rotational control combines controller roll with joystick-driven pitch and yaw.

At a high level, the architecture is composed of three main subsystems:

- The **MR interaction subsystem**, running on the Meta Quest 3S, responsible for rendering the MR scene, tracking the user's head and hands/controllers, and handling button-based activation of robot motion.
- The **robotic subsystem**, represented by the UR5e cobot, which executes motion commands received over the network and provides state feedback such as joint positions or tool center point pose.
- The **communication and control subsystem**, which maps relative hand motion to robot commands, maintains the digital twin, and manages network communication with the robot through dashboard and RTDE interfaces.

These subsystems form a closed loop in which the user's input is translated into robot motion in a controlled and predictable way, while the digital twin provides real-time feedback in the MR environment.

Figure 3.2 summarizes the overall structure of the proposed system. It highlights the separation between the MR interaction subsystem running on the HMD, the communication and control subsystem responsible for mapping user input into robot commands, and the physical robotic subsystem. The diagram also emphasizes the role of the digital twin as an intermediate visual layer that reflects the state of the physical UR5e and provides feedback to the user.

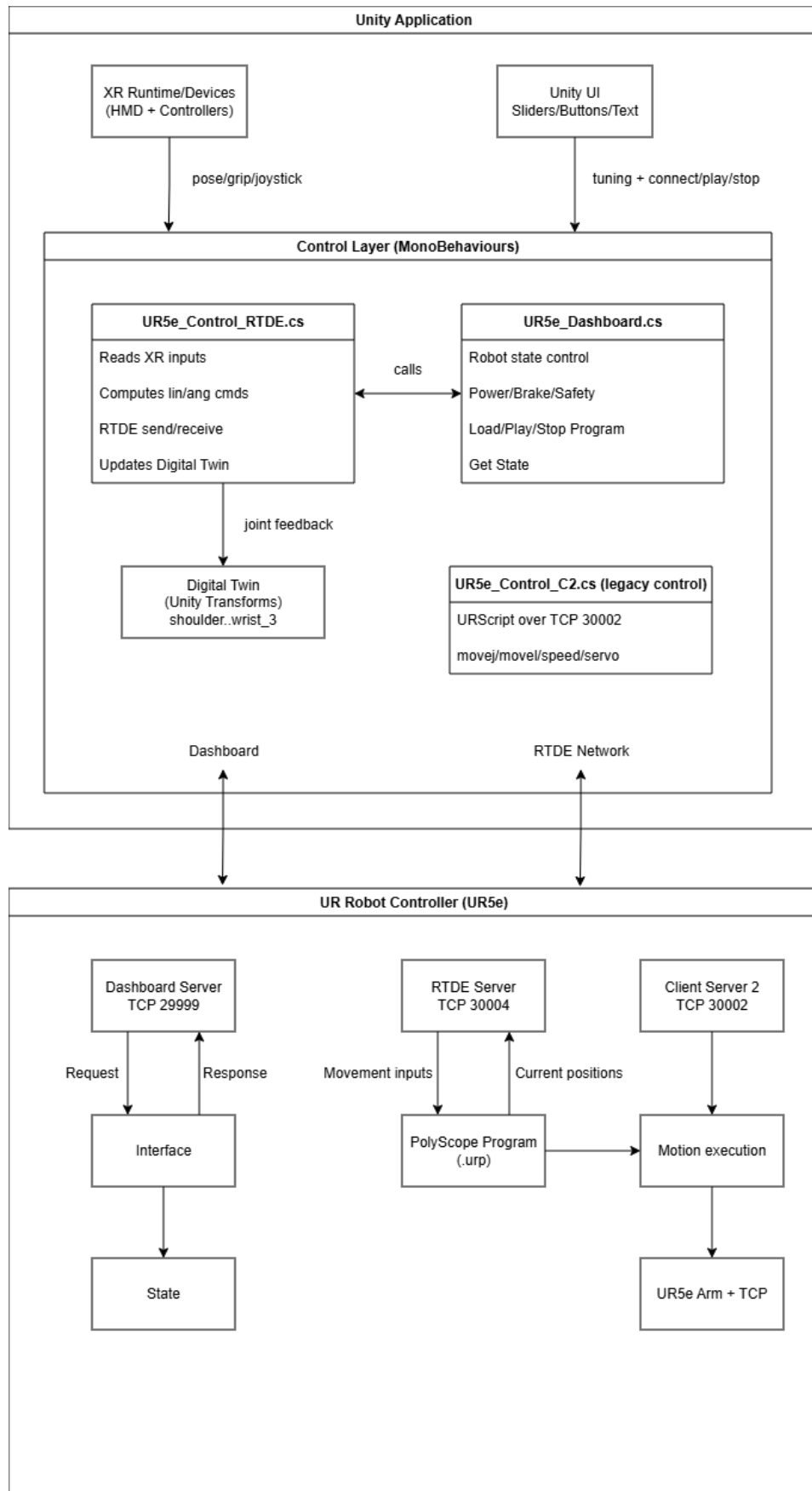


Figure 3.2: Overall system architecture of the proposed MR-based teleoperation system.

A key implementation detail is that the Unity application communicates with the UR controller through multiple UR-provided network interfaces, each serving a different purpose. The Dashboard server is used for high-level lifecycle control (loading programs, play/pause/stop, power/brake release, and status queries). It runs on TCP port 29999 and accepts newline-terminated commands, Universal Robots (2025).

For motion control, several approaches were considered:

- Primary/Secondary client interfaces (30001 and 30002 respectively) allow sending URScript commands directly while also receiving a continuous robot-state stream. However, the state stream must be manually parsed and its structure may change across software updates; the guide explicitly recommends preferring RTDE for structured state access, Universal Robots (2025).
More importantly for teleoperation, these script-sending interfaces are non-blocking, meaning the external application must manage timing and completion of robot actions. This increases the risk of issuing a new motion command before the previous one has completed, which can lead to hard stops and unintended motion, Universal Robots (2025).
- Interpreter mode (port 30020) supports injecting URScript statements into a running program at runtime. While it adds useful acknowledgements and enables more structured remote program execution than the non-blocking primary/secondary approach, it comes with practical constraints (single-line statements, buffer management to avoid “runtime too far behind”, and scope limitations) that make it less suitable as the main real-time teleoperation channel for continuous, high-rate incremental motion updates, Universal Robots (2025).
- RTDE (port 30004) was ultimately selected as the main real-time control and feedback mechanism. RTDE is designed to synchronize an external application with the UR controller without breaking real-time properties, using a structured input/output “recipe” defined in XML and a continuous synchronization loop, Universal Robots (2025).

In this architecture, Unity sends incremental motion parameters through RTDE input registers, while the robot runs a dedicated PolyScope program that reads those registers and executes the corresponding motion (e.g., pose setpoints or servo/speed commands). The guide describes this pattern explicitly: robot motion is commanded by writing target values into RTDE registers, with the robot program consuming those values and optionally enforcing a watchdog mechanism to pause/stop on lost communication, Universal Robots (2025).

This design choice reflects the requirements of MR teleoperation: low-latency state feedback, predictable command application, and safer behaviour under packet loss. The dashboard channel remains responsible for program management (ensuring the correct robot-side RTDE consumer program is running), while RTDE provides the high-frequency closed-loop path between user input, robot motion execution, and digital-twin updates.

3.4 Hardware Architecture

3.4.1 Meta Quest 3S HMD

The Meta Quest 3S serves as the primary interaction device. It provides:

- Inside-out tracking of head and hand/controller poses.
- Passthrough capabilities to support MR visualization of the physical environment.
- Standalone execution of the Unity-based MR application.

The Quest 3S continuously tracks the pose of the active hand or controller. This tracking data is used to compute relative movements when the control button is pressed, forming the basis for generating robot motion commands.

3.4.2 UR5e Robotic Manipulator

The UR5e cobot provides six DoF and an Ethernet interface for communication. It is operated using:

- The **dashboard interface**, which allows higher-level operations such as powering the robot on and off, releasing brakes, and retrieving basic status information.
- The **RTDE interface**, which enables real-time exchange of robot state and control commands.

The robot is mounted in a fixed position within the workspace. Its base frame is used as the reference coordinate frame for both the digital twin in Unity and the mapping between user motion and robot motion. To contextualize the hardware platform used in this work, Figures 3.3 and 3.4 show the physical UR5e robotic manipulator used during development and evaluation. These reference images support the discussion of the robot interfaces and specifications presented in the following subsection.



Figure 3.3: Physical UR5e robotic manipulator.

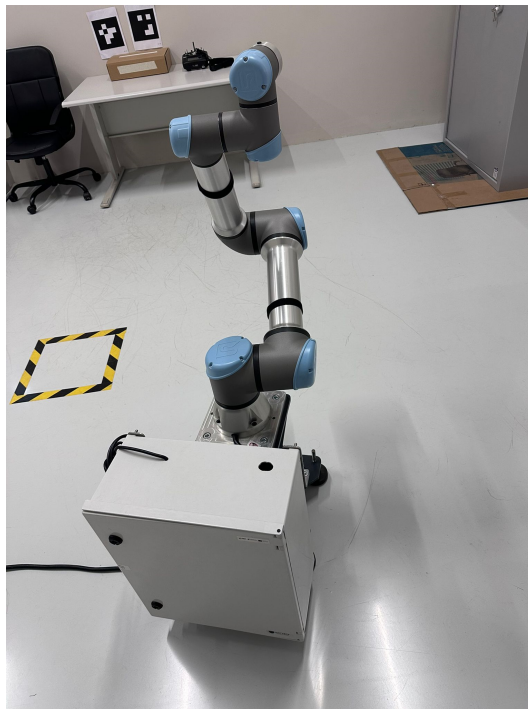


Figure 3.4: Physical UR5e robotic manipulator (back view).

3.4.3 UR5e Robot Specifications

The physical robot used in this work is the Universal Robots UR5e collaborative manipulator. The UR5e provides six rotational DoF with a maximum payload of 5 kg and a reach of 850 mm. The robot is specified with a pose repeatability of ± 0.03 mm (ISO 9283), and each joint supports a working range of $\pm 360^\circ$.

From a control and integration perspective, the UR5e controller exposes a high-frequency system update rate (500 Hz) and provides multiple industrial communication and integration options, including Ethernet-based interfaces. The platform also includes built-in safety functionality (reported as 17 configurable safety functions with performance level PLd Category 3 according to EN ISO 13849-1).

Table 3.1 summarizes the main UR5e specifications relevant to this work Universal Robots (2021).

Specification	UR5e Value
DoF	6 rotational joints
Maximum payload	5 kg
Reach	850 mm
Pose repeatability	± 0.03 mm (ISO 9283)
Joint working range	$\pm 360^\circ$ (all joints)
Max joint speed	$180^\circ/\text{s}$
Max TCP speed	≈ 1 m/s
Robot weight	$\approx 20.6\text{--}20.7$ kg
Internet Protocol (IP) classification	IP54
Safety functions	17 configurable safety functions (PLd Cat. 3)
System update frequency	500 Hz

Table 3.1: Key technical specifications of the UR5e.

3.4.4 Computing and Network Infrastructure

The overall system requires that the Meta Quest 3S and the UR5e share a common network. During development and testing, a PC is used as an intermediary to configure the network and assist with debugging. The network setup includes:

- A wired Ethernet connection between the PC and the UR5e.
- A wireless connection between the Meta Quest 3S and the same network as the PC, for example through a router or via connection sharing.

Router and network configuration steps are carried out so that the HMD can communicate directly with the UR5e using its IP address. This configuration allows the Unity application running on the Quest 3S to establish Transmission Control Protocol / Internet Protocol (TCP/IP) connections to the robot’s dashboard and RTDE ports.

3.5 Software Architecture

3.5.1 Development Environment

The MR application is developed using the Unity engine with XR support for the Meta Quest 3S. The development environment comprises:

- Unity as the core platform for 3D rendering, physics, and scene management.
- XR frameworks to handle head and controller tracking and interaction.
- Custom C# scripts implementing network communication, RTDE handling, and robot control.

The project is configured to support rapid iteration on scripts and assets, simplifying testing of communication logic, interaction mechanisms, and digital twin behavior both in the editor and on the target device.

3.5.2 Layered Software Structure

Conceptually, the software is organized into several layers:

- **Presentation Layer:** Responsible for rendering the MR scene, including the digital twin of the UR5e, visual indicators of robot state, and user interface elements such as connection buttons, host input fields, and validation messages.
- **Interaction Layer:** Handles input from the HMD, including controller tracking and button events. This layer detects when the control button is pressed or released and manages the recording of the initial hand pose as a reference for relative motion.
- **Control and Mapping Layer:** Translates relative hand/controller movements into robot motion commands. It maintains the logic that ensures that motion to the right from the user's point of view is interpreted as motion to the right from the robot's perspective. This layer also updates the digital twin's articulation bodies to mirror the intended robot motion.
- **Communication Layer:** Encapsulates all TCP/IP communication with the UR5e. It includes scripts for interacting with the robot dashboard and RTDE interfaces, constructing and sending commands, receiving responses, and handling connection management.

The layered organization described above is represented in Figure 3.5. This representation makes explicit the separation between presentation, interaction, control, and communication concerns. Such a separation is important for maintainability, since modifications to one layer, such as the user interface or the communication backend, can be introduced without requiring major restructuring of the remaining components.

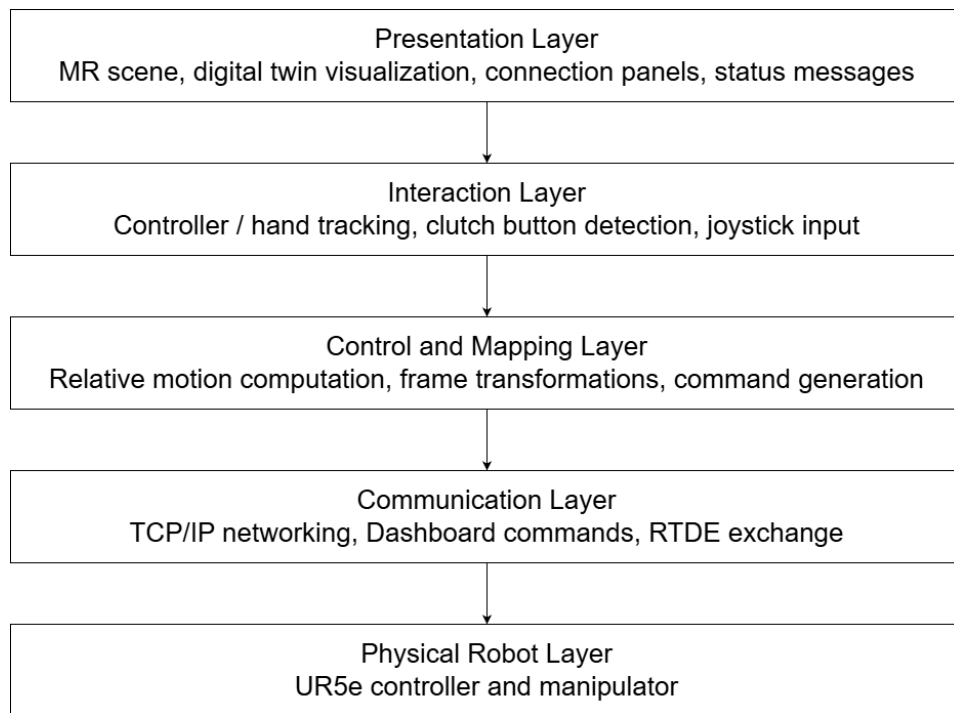


Figure 3.5: Layered software architecture of the implemented system.

This layered architecture separates MR interaction concerns from robot-specific communication and control, enabling focused development and easier maintenance.

3.5.3 Digital Twin Representation

The UR5e is represented in Unity as a digital twin, built from a kinematic hierarchy of `GameObjects` corresponding to the robot's physical links. Each joint is modeled using an `ArticulationBody` component, which allows joint targets to be set programmatically. The digital twin is used to:

- Visualize the robot pose in the MR environment for improved spatial awareness.
- Preview the effect of user input on the robot before and during teleoperation.
- Support testing and validation of control logic without always requiring physical motion of the robot.

The alignment between the digital twin and the physical robot is maintained through calibration and the use of consistent reference coordinate frames.

3.6 Communication Subsystem

3.6.1 Dashboard Interface Script

A dedicated script is created to interact with the robot dashboard interface over TCP/IP. This script provides:

- Functions to connect and disconnect from the robot.

- Retrieval of basic robot information and status.
- Execution of the power-on protocol for the robot arm, including powering on and releasing brakes.
- Execution of power-off procedures for safely disabling the robot arm.

The Unity project is configured so that changes to this script can be easily integrated and tested, allowing iterative refinement of connection handling and error reporting.

3.6.2 Simple Command Control Script

A second script is implemented for sending simple control instructions to the robot over TCP/IP. This script includes:

- Connection handling for the relevant control port.
- A command constructor for building correctly formatted robot instructions.
- Support for sending basic motion or state commands, useful during early development and testing.

As with the dashboard script, the Unity project is set up to facilitate rapid testing of these control functions in both editor and standalone builds.

3.6.3 RTDE-Based Control

To support real-time control, an RTDE configuration is introduced. This includes:

- A C# RTDE implementation, based on the structure and behavior of an existing Python RTDE library, adapted to integrate with Unity.
- A URP script deployed on the robot controller to handle incoming RTDE data and apply the received commands to the robot.
- A control loop that exchanges state information and control targets between the MR application and the robot over RTDE.

The RTDE layer enables higher-frequency updates and more responsive control than simple text-based command interfaces, which is particularly important for mapping continuous hand motion to robot movement.

3.7 Data Flow and Control Loop

The system's operation can be described through a sequence of stages forming a closed control loop:

1. **Tracking and Input Acquisition:** The Meta Quest 3S continuously tracks the pose of the user's head and the active hand/controller, and also captures joystick inputs used for rotational control (pitch and yaw) in the teleoperation phase.

2. **Button Activation and Reference Pose:** When the user holds down the designated control button, the system records the current hand pose as a reference. As long as the button remains pressed, subsequent hand movements are interpreted as relative displacements from this reference pose.
3. **Relative Motion Mapping:** The control and mapping layer computes the relative motion of the hand and maps it to the robot's coordinate frame. The mapping is designed so that directional movements (for example, right, left, forward, backward) are consistent between the user's perspective and the robot's perspective.
4. **Command Generation:** Based on the relative motion, the control logic constructs appropriate robot commands. Depending on the active communication mode, these may be:
 - High-level commands transmitted over a simple TCP/IP control interface.
 - Real-time joint or Cartesian control targets transmitted via RTDE.
5. **Digital Twin Update:** The digital twin's joint targets are updated to reflect the intended motion. This provides immediate real-time feedback in the MR environment, helping the user understand how the robot is responding to the hand movement.
6. **Robot Actuation and Feedback:** The physical UR5e receives the commands and moves accordingly. When RTDE is used, the robot's state (for example, joint angles or tool center point pose) can be read back and used to further refine the digital twin's pose and ensure synchronization between the virtual and physical robots.

Figure 3.6 provides a compact representation of the runtime control loop. User input (HMD tracking and controller actions) is sampled in Unity and mapped to motion commands by the control script, which are transmitted to the UR5e via RTDE. Robot state feedback is then received back over RTDE and mirrored in the MR environment through the digital twin. This closed loop is essential to the system's responsiveness and perceived usability.

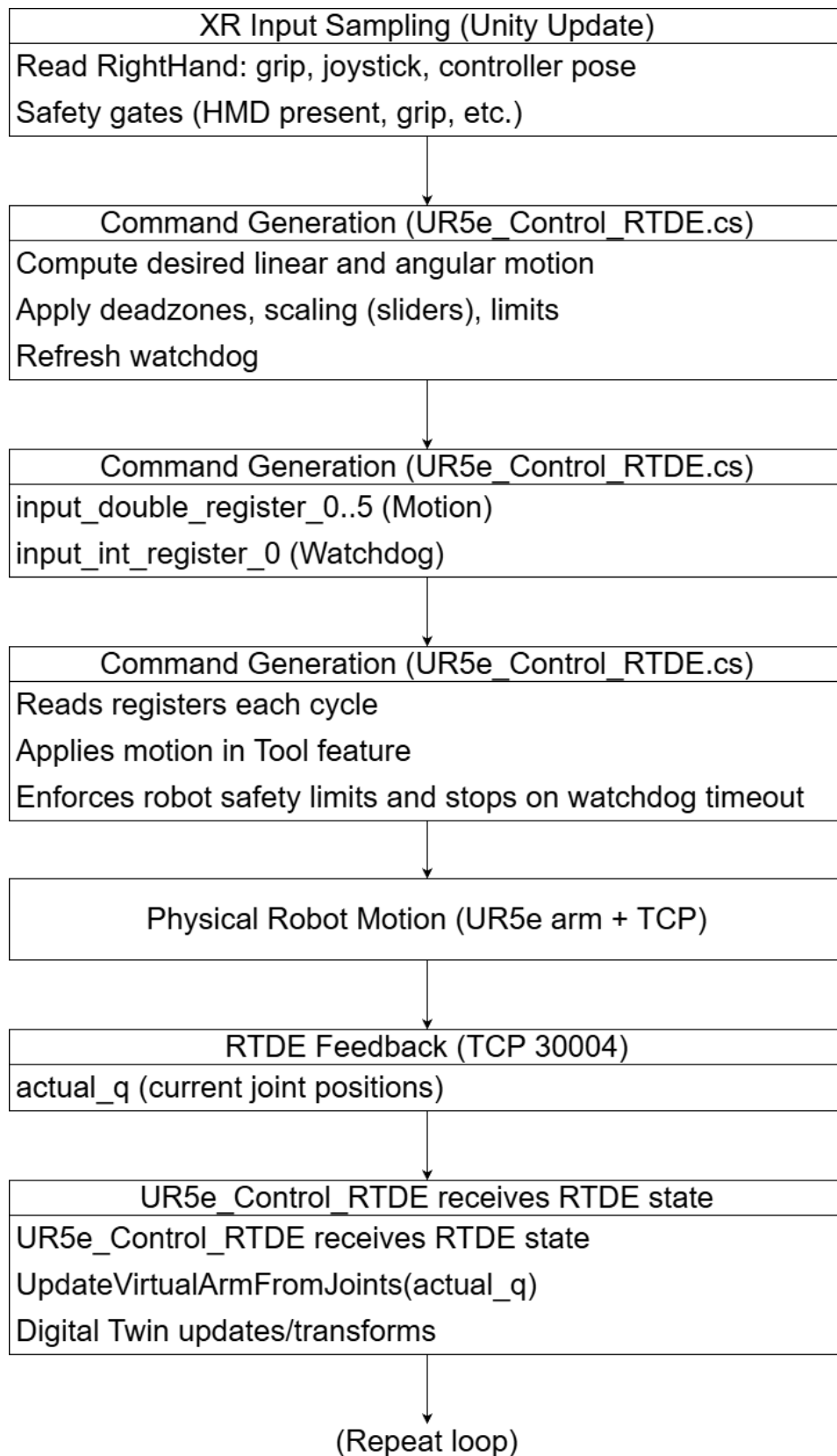


Figure 3.6: Runtime control and communication flow of the teleoperation system.

When the button is released, the system stops generating new commands based on hand movement. The robot maintains its current pose or follows any default behavior defined in the control logic, while the digital twin continues to reflect the robot's latest known state.

3.8 Interaction Logic

Unlike architectures that define multiple discrete interaction modes, this system follows a single, continuous interaction paradigm:

- The digital twin is always visible in the MR scene, providing a persistent visualization of the robot.
- Hand or controller tracking runs continuously in the background.
- Robot motion is only triggered while the designated control button is held down.

The mapping between user input and robot motion is not uniform across all axes. As shown in Figure 3.7, translational movement is derived from the controller pose, roll is derived from controller rotation, and pitch/yaw are commanded using the joystick. This mapping is also summarised visually in Figure 3.1(a–b), which illustrates the clutch (grip) mechanism and the direction conventions for translation and rotation. This hybrid mapping was adopted to improve usability and reduce ambiguity during teleoperation.

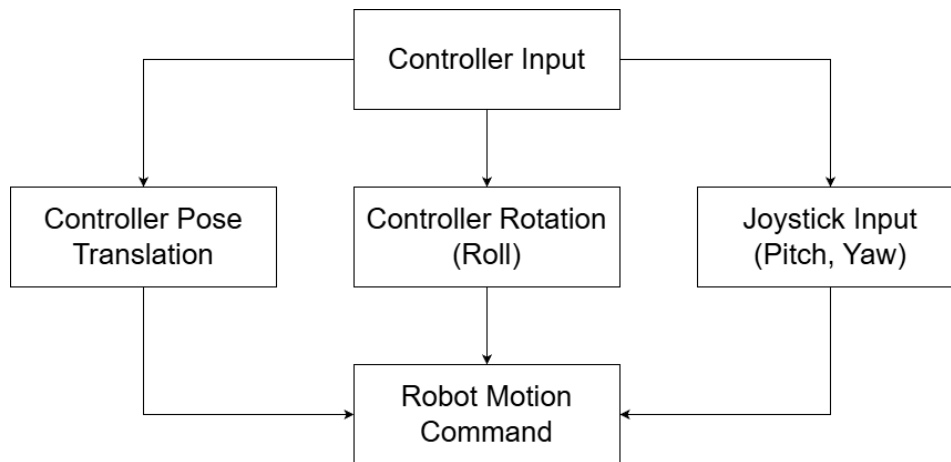


Figure 3.7: Hybrid input mapping used for robot teleoperation.

This design has several advantages. It reduces the need for explicit mode switching, simplifies the user experience, and provides a clear and immediate understanding of when the robot is under direct control. The button effectively acts as a clutch-based interaction mechanism: pressing it explicitly enables motion commands relative to a newly defined reference pose, while releasing it immediately disables command generation. This provides an important safety property by preventing unintended robot motion when the user adjusts posture, regrips the controller, or tracking momentarily degrades.

3.9 Summary

This chapter presented the system architecture for a MR application that enables intuitive control of a UR5e robotic arm using a Meta Quest 3S HMD. The architecture integrates hardware components (HMD, robot, and network infrastructure) with a layered software stack in Unity, comprising presentation, interaction, control, and communication layers. A digital twin of the UR5e provides real-time visualization and supports safe testing of control strategies.

Continuous tracking of the user's hand or controller is combined with button-activated control, ensuring that robot motion is only triggered under explicit user intent. Networked communication with the robot is handled through dedicated scripts for the dashboard interface, simple command control, and RTDE-based real-time exchange. Together, these elements form a coherent platform for experimenting with MR-based HRI and for implementing the methods and evaluations described in subsequent chapters.

Chapter 4

System Implementation

This chapter describes the implementation of the MR system that enables hand-based teleoperation of a UR5e robot using a Meta Quest 3S HMD. While the previous chapter focused on the conceptual architecture, here the emphasis is on how the Unity application, communication modules, and user interface were implemented and integrated into a working prototype.

4.1 Overview

The implementation follows the architecture outlined in Chapter 3. A Unity project targeting the Meta Quest 3S was created, integrating hand tracking, a digital twin of the UR5e, and networking components for communication with the physical robot. The system continuously tracks the user's hand or controller pose, while robot motion is only generated when a designated button is pressed. During this clutch period, relative hand motion is mapped to robot motion, and the same commands are sent to the physical robot.

The implementation can be divided into three main parts:

- the Unity scene and digital twin setup;
- the communication modules for the dashboard and RTDE interfaces;
- the user interface and interaction logic for hand-based control.

4.2 Unity Scene and Digital Twin

4.2.1 XR Rig and Hand Tracking

The Unity project is configured with XR support for the Meta Quest 3S. An XR origin is placed in the scene, providing tracking for the HMD and hands/controllers through the Meta OpenXR runtime. The right hand (or controller) pose is exposed to the rest of the system via a dedicated anchor Transform, which is sampled at each update cycle. This anchor provides the translational component of teleoperation and the roll component of tool rotation. For pitch and yaw rotation, the system reads the controller joystick input, allowing rotational adjustments without requiring the user to physically rotate the controller for those axes.

4.2.2 Development and Debugging Setup

During development, the system was tested in a MR workspace that included the physical UR5e robot, the Meta Quest 3S HMD, and a development PC. Figure 4.1 shows an example of this setup during debugging and iterative testing.



Figure 4.1: Development and debugging setup used during implementation, showing the UR5e, development workstation, and the HMD.

Although the Meta Quest 3S was the intended target device for the system, the Meta Quest 3 was used in this specific development session because the Quest 3S was not available at that time.

At this stage, the Meta Quest 3 was connected to the development PC through a cable, allowing APK builds to be sideloaded directly onto the headset for rapid testing. This setup simplified the deployment cycle, since modifications in the Unity project could be compiled and transferred to the device immediately after each iteration. The wired connection was also useful for debugging, log inspection, and validation of interaction changes while the application was being refined.

This development arrangement made it possible to test the full integration between the Unity application, the XR interaction layer, and the physical robot under realistic conditions. It also allowed the teleoperation interface, controller mappings, and communication modules to be adjusted incrementally before moving to more stable standalone tests.

4.2.3 UR5e Digital Twin

The digital twin of the UR5e is imported into Unity as a hierarchy of `GameObjects` that mirrors the physical robot's kinematic chain. Each joint in the hierarchy (from `shoulder_link` to `wrist_3_link`) is associated with an `ArticulationBody` component, enabling joint-level updates in Unity that correspond to the real robot joints. Figure 4.2 shows the UR5e digital twin rendered in the Unity scene, while Figure 4.3 illustrates the hierarchical structure used to represent the robot links and joints. The digital twin was validated during iterative tests in the development setup shown in Figure 4.1.

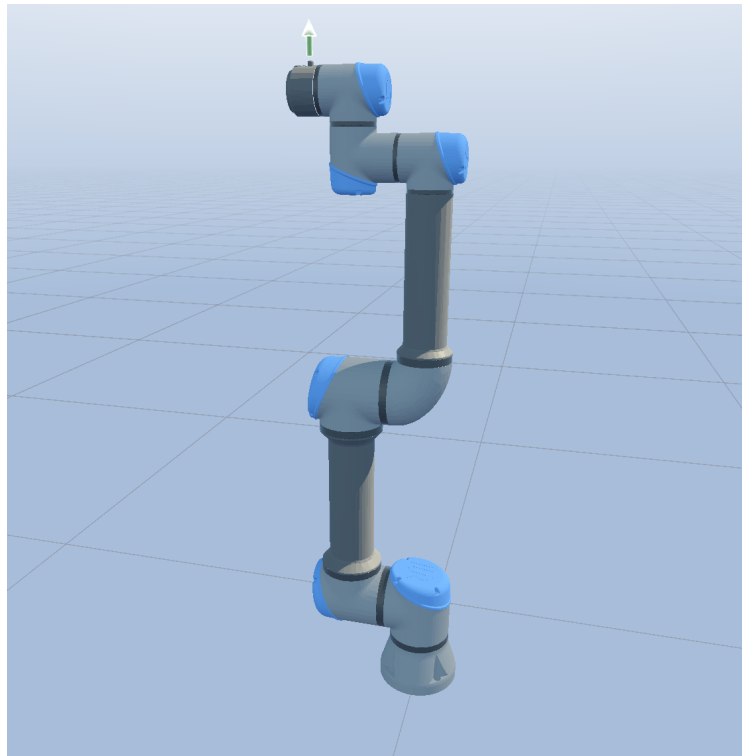


Figure 4.2: UR5e digital twin rendered in the Unity scene.



Figure 4.3: Unity hierarchy of the UR5e digital twin, showing the kinematic chain from `base_link` to `wrist_3_link`.

Joint targets are updated via the RTDE connection: the robot streams current joint positions to the Unity application at 60 Hz, allowing real-time feedback-level synchronization without requiring additional prediction mechanisms. This approach reduces the likelihood of visual desynchronization between the physical robot and the digital twin, since the Unity representation is continuously corrected using the robot’s measured joint state.

The base of the digital twin is aligned with the robot’s base frame. This alignment is used consistently by the control logic so that movements to the right, left, up, and down, match between the user’s perspective and the robot’s reference frame. The flange at the end of `wrist_3_link` is treated as the virtual tool center point (TCP) and is used when visualizing reach and potential contact with objects. The MR overlay presented later in this section is aligned to the physical UR5e platform shown in Figures 3.3 and 3.4, ensuring that the virtual and physical coordinate frames remain consistent during teleoperation. Figure 4.4 shows the digital twin overlaid in the MR view, positioned relative to the physical UR5e in the environment.

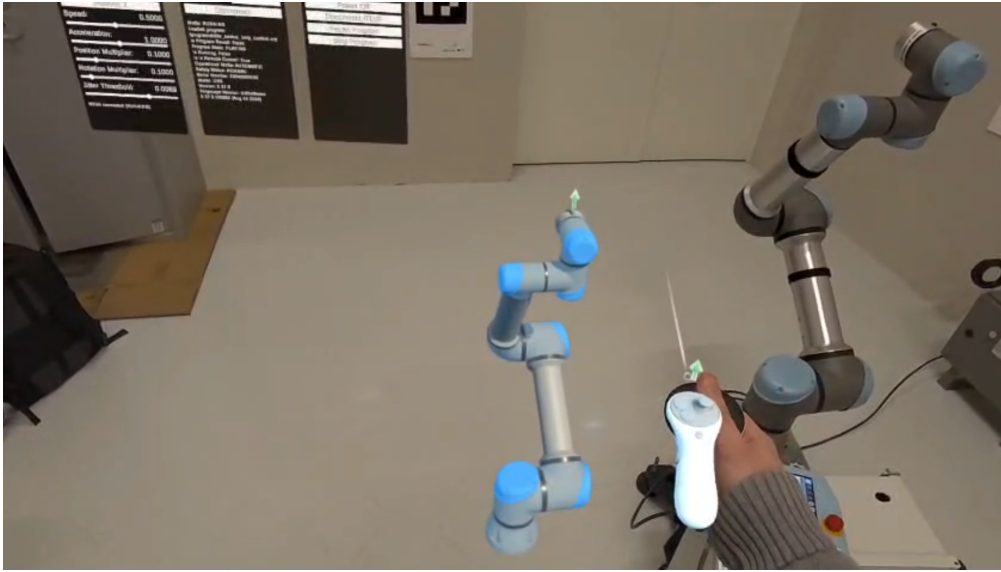


Figure 4.4: MR visualization of the UR5e digital twin overlaid in the environment, shown alongside the physical robot.

4.3 Communication Modules

4.3.1 Dashboard Interface

Connection to the UR5e dashboard server is handled by the `UR5e_Dashboard.cs` script. This script establishes a TCP/IP connection to the dashboard port and exposes high-level operations such as:

- connect and disconnect from the robot;
- power on the robot and release the brakes as part of the initialization sequence;
- power off the robot arms;
- start, pause, and stop the execution of UR program on the controller.

Dashboard commands follow the textual protocol defined in the official Universal Robots documentation. The script parses basic responses to confirm success or failure and reports the current connection status to the Unity application. These functions are later bound to buttons in the MR user interface.

4.3.2 Simple Command Interface

For initial testing and validation, a simpler control path is provided by `UR5e_Control_C2.cs`. This script sends pre-defined URScript commands over TCP/IP, without relying on RTDE. It is mainly used to verify network connectivity, test basic motion primitives, and prototype motion sequences before integrating them into the full control loop.

4.3.3 RTDE Client and Real-Time Control

Real-time exchange of state and control commands is implemented through `RTDEClient.cs`. This script is an adaptation of the official `RTDEClient.py` example provided by Universal Robots, translated into C# and integrated into Unity.

The RTDE client:

- establishes a connection to the RTDE port of the UR5e;
- negotiates input and output recipes based on an XML configuration;
- periodically sends control packets (for example, joint targets or TCP pose commands);
- receives robot state packets, such as the current joint positions or TCP pose.

The script `UR5e_Control_RTDE.cs` builds on top of `RTDEClient.cs`. It maintains the control loop in which relative hand motions are converted into robot commands. It also ensures that the same targets are applied both to the robot and the digital twin, keeping them synchronized. A small UR program running on the robot is responsible for consuming RTDE targets and executing them on the robot.

4.4 User Interface Implementation

4.4.1 Overall Layout

The MR user interface consists of three floating panels, as shown in Figure 4.5. These panels are positioned in front of the user in the MR environment and can be interacted with using hand or controller input.



Figure 4.5: MR user interface panels for robot control.

4.4.2 Connection Panel

The *Connect* panel provides a text input field labeled *Host* and a *Connect* button. The text field allows specifying the robot's IP address, which is then passed to the dashboard and RTDE scripts.

A simple validation mechanism is implemented: if the host field is empty or invalid when the user presses the *Connect* button, a red validation message is displayed under the field, indicating that the IP address must be specified before proceeding. Only after a successful connection does the system enable the RTDE controls and the robot initialization buttons.

4.4.3 Dashboard Control Panel

The *Dashboard Control* panel exposes the higher-level operations offered by the dashboard interface. It includes buttons for:

- *Initialize*: powers on the robot and releases the brakes in sequence;
- *Power Off*: safely powers down the robot arms;
- *Disengage Safety*: issues the command to disengage protective stops when permitted;
- *Connect RTDE* and *Disconnect RTDE*: open and close the RTDE connection;
- *Play Program*, *Pause Program*, and *Stop Program*: control the execution of the UR program that processes RTDE commands.

Each button is linked to a corresponding public method in `UR5e_Dashboard.cs` or `UR5e_Control_RTDE.cs`. This design keeps the UI logic separate from the communication logic, improving maintainability.

4.4.4 Arms Control Panel

The *Arms Control* panel provides controls for teleoperation parameters. It includes:

- two buttons (*Position 1* and *Position 2*) that store or recall preset joint configurations for the UR5e;
- sliders for *Speed* and *Acceleration*, which directly influence the limits used when constructing motion commands;
- a *Position Multiplier* that scales the translational component of hand motions;
- a *Rotation Multiplier* that scales the rotational component;
- a *Jitter Threshold* that filters out small hand movements considered as noise.

These parameters are read at runtime by the control logic and allow the user to adapt the robot's responsiveness and smoothness without modifying the underlying code.

4.5 Hand-Based Control Logic

4.5.1 Clutch Mechanism and Reference Pose

The interaction follows the single continuous mode described in Chapter 3. The system continuously tracks the hand or controller pose but only generates robot motion when a designated button is held. The button is treated as a clutch:

- when the button is pressed, the current hand pose is stored as a reference;

- while the button remains pressed, each new pose is compared to the reference to compute a relative displacement;
- when the button is released, the system stops sending motion commands, and the robot maintains its current pose.

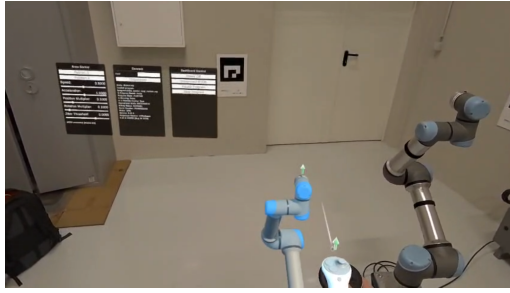
4.5.2 Mapping Hand Motion to Robot Commands

At each update during an active clutch-based interaction period, the system:

1. reads the current hand pose and computes the relative translation (and roll, when applicable) with respect to the stored reference, while pitch and yaw commands are obtained from joystick input;
2. applies the jitter threshold, ignoring translations and rotations whose magnitude is below a user-defined minimum;
3. scales the translational component and the rotation components by the corresponding multipliers to control sensitivity, including joystick-driven pitch and yaw;
4. transforms the resulting movement from the XR origin frame into the robot's base frame, preserving the intuitive mapping in which "right" from the user's point of view corresponds to "right" from the robot's perspective;
5. constructs a motion command (for example, an updated TCP pose or incremental joint targets) and sends it via `UR5e_Control_RTDE.cs`;
6. updates the digital twins' articulation bodies with the same targets, providing immediate real-time feedback.

4.5.3 MR Teleoperation Examples

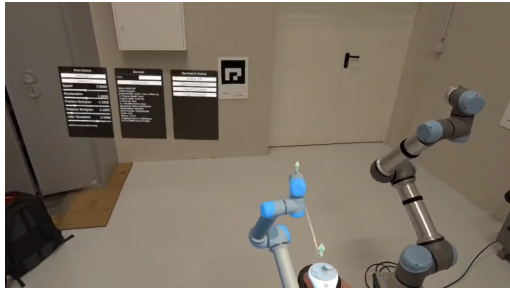
Figure 4.6 presents example in-headset views captured during development. The sequence illustrates the clutch-based interaction: a reference pose is established at the start, after which controller translation and rotation inputs produce corresponding motion of the physical UR5e. The on-screen axis indicators (arrows) are rendered to support usability by exposing the tool pose and the current control direction to the operator.



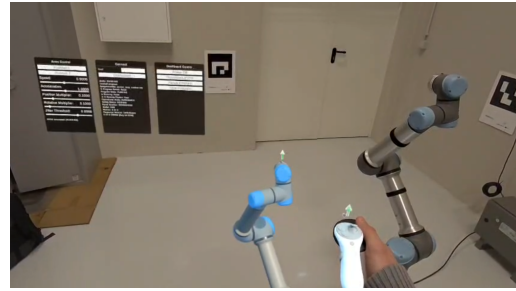
(a) Starting pose (reference established).



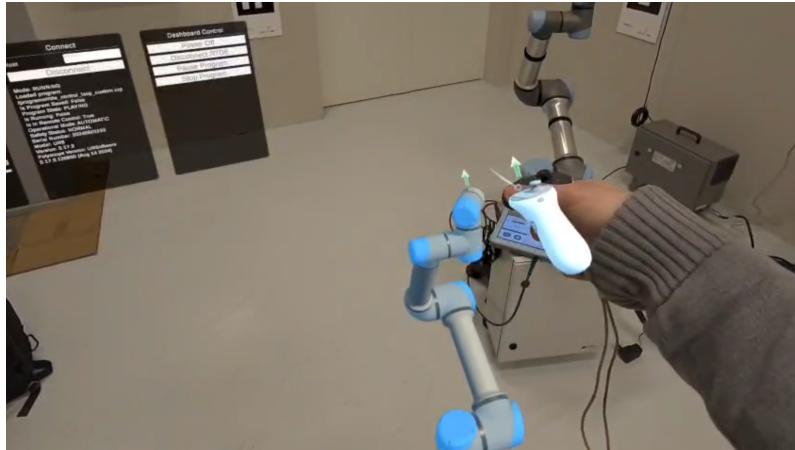
(b) Controller translation upward (TCP follows).



(c) Controller translation to the left (TCP follows).



(d) Joystick pitch input (TCP orientation adjusted).



(e) Controller rotation (roll) and visual pose cues via axis indicators.

Figure 4.6: In-headset MR views during live teleoperation of the UR5e. The digital twin is overlaid onto the physical workspace, while axis indicators provide real-time feedback on tool pose and control direction.

To illustrate how joystick input was integrated into the control loop, Listing 4.1 shows the implementation used to map controller joystick commands to robot pitch and yaw velocities. Since the tool may already be rotated around the roll axis, the joystick inputs are first expressed in a user-oriented frame and then transformed into the tool frame using the current wrist roll angle. This correction ensures that the commanded pitch and yaw remain intuitive from the user's point of view, even when the robot tool is already rotated.

```

1 float stickDeadzone = 0.15f;
2 float joystickLimiter = 0.1f;
3
4 float sx = stickX;
5 float sy = stickY;
6
7 // Apply joystick deadzone
8 if (Mathf.Abs(sx) < stickDeadzone) sx = 0f;
9 if (Mathf.Abs(sy) < stickDeadzone) sy = 0f;
10
11 // User-oriented commands
12 float rx_world = -sy * joystickLimiter; // forward -> pitch
    forward
13 float ry_world =  sx * joystickLimiter; // right    -> yaw right
14
15 // Current wrist roll angle
16 float r = wristRollDeg * Mathf.Deg2Rad;
17 float cosR = Mathf.Cos(r);
18 float sinR = Mathf.Sin(r);
19
20 // Transform pitch/yaw into tool frame
21 float rx_tool =  cosR * rx_world + sinR * ry_world;
22 float ry_tool = -sinR * rx_world + cosR * ry_world;
23
24 // Final angular command
25 float rx = rx_tool;
26 float ry = ry_tool;

```

Listing 4.1: Joystick-based pitch and yaw mapping corrected by the current wrist roll.

This implementation detail is particularly relevant because it highlights that the rotational control was not based solely on raw controller motion. Instead, it required a transformation step to preserve intuitive behavior under different tool orientations, which was an important aspect of the final teleoperation experience.

Speed and acceleration slider values are incorporated into the command generation step, acting as bounds for how quickly the robot is allowed to move and how aggressively it can change direction. This helps maintain safe and predictable behavior, especially when the hand moves abruptly.

4.6 Summary

This chapter detailed the implementation of the MR-based robotic control system. The Unity project integrates hand tracking, a digital twin of the UR5e, and dedicated communication modules for both the dashboard and RTDE interfaces. A set of MR panels provides connection management, dashboard control, and tuning of teleoperation parameters.

Hand-based control is realized through a clutch mechanism that converts relative hand motions into robot commands while maintaining a consistent mapping between the user's and robot's frames of reference. The combination of adjustable control parameters, real-time feedback digital twin visualization, and robust communication with the UR5e establishes a practical platform for experimenting with MR-based HRI and for future evaluation of object grasping tasks.

Chapter 5

Results

This chapter presents the results obtained from a user-based evaluation of the MR teleoperation prototype. Each participant received an approximately one-minute explanation of the interaction concept and safety considerations, followed by around 5–10 minutes of hands-on testing time. After the trial, participants completed a questionnaire containing demographic questions and usability items (Q5–Q12) rated on a 0–5 scale. The full participant responses are provided in Appendix A (Portuguese) and Appendix B (English translation). The following sections summarize the participant profile and discuss the usability outcomes, supported by the charts associated with each question.

5.1 Participants and Test Conditions

A total of 14 participants took part in the evaluation. The sample included 11 male and 3 female participants. Age distribution was concentrated between 30–40 (7 participants) and 40–60 (5 participants), with 2 participants in the 20–30 range. Prior experience with VR/MR varied, with an average self-reported level of 1.36 on a 0–5 scale, indicating that most participants had limited familiarity with immersive interfaces. Vision correction was evenly distributed: 7 participants reported wearing glasses and 7 reported not wearing glasses.

The test protocol was designed to be short and exploratory, focusing on initial usability impressions rather than long-term learning effects. Participants were encouraged to attempt basic translational and rotational control actions and to form an opinion on responsiveness, intuitiveness, and interface clarity.

5.1.1 Evaluation Environment

Figures 5.1–5.3 illustrate the evaluation environment and show participants interacting with the system during the testing sessions. The images provide visual context for the conditions under which the usability questionnaire was completed, including the physical UR5e setup, the Meta Quest HMD, and the workstation used for monitoring the application.



Figure 5.1: Participant interacting with the MR teleoperation system during the evaluation session.

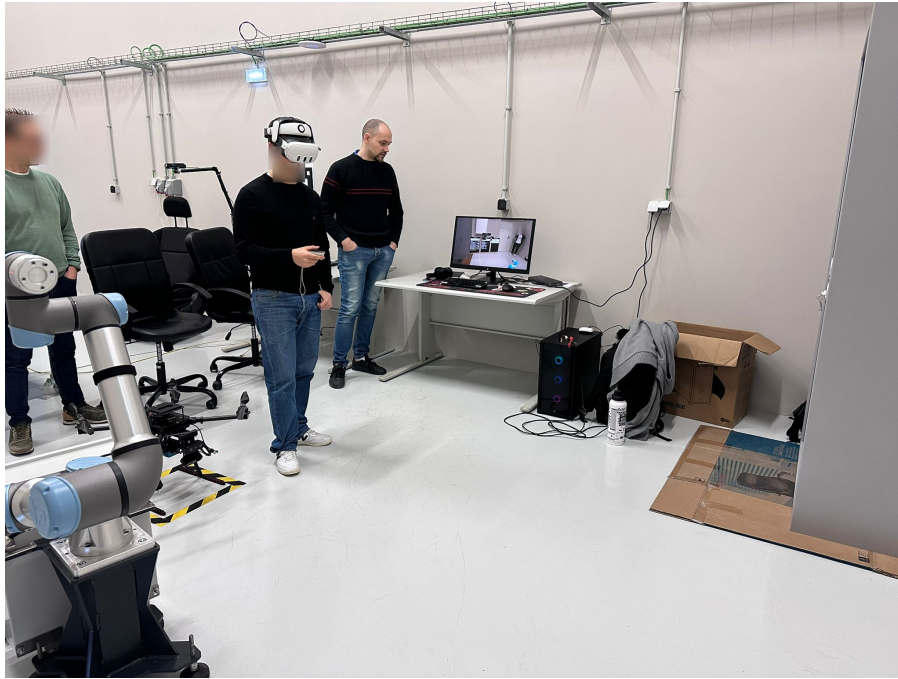


Figure 5.2: Example of a participant using the HMD and controller while testing the robot teleoperation interface.



Figure 5.3: Evaluation session showing a participant interacting with the system while observers monitored the test environment.

These images also highlight the practical nature of the evaluation: participants were able to test the interaction directly in the same environment where the robot and the MR application were deployed. This supports the interpretation of the questionnaire results,

since the reported usability impressions were formed under realistic operating conditions rather than through simulation alone.

5.2 Usability Results

The usability questionnaire items (Q5–Q12) are grouped below according to the aspect of interaction they evaluate: responsiveness/latency, translational control, rotational control, and overall interface intuitiveness. Each subsection references the corresponding chart for that item.

5.2.1 Responsiveness and Perceived Delay (Q5)

Q5 evaluated how much the delay between user movement and robot movement affected the ability to control the system (0–5). Overall, the reported impact of delay was low-to-moderate, with a mean score of 1.57 (median 1). Most ratings clustered between 0 and 2, though a small number of participants reported higher impact (including one rating of 5). Figure 5.4 summarizes the distribution.

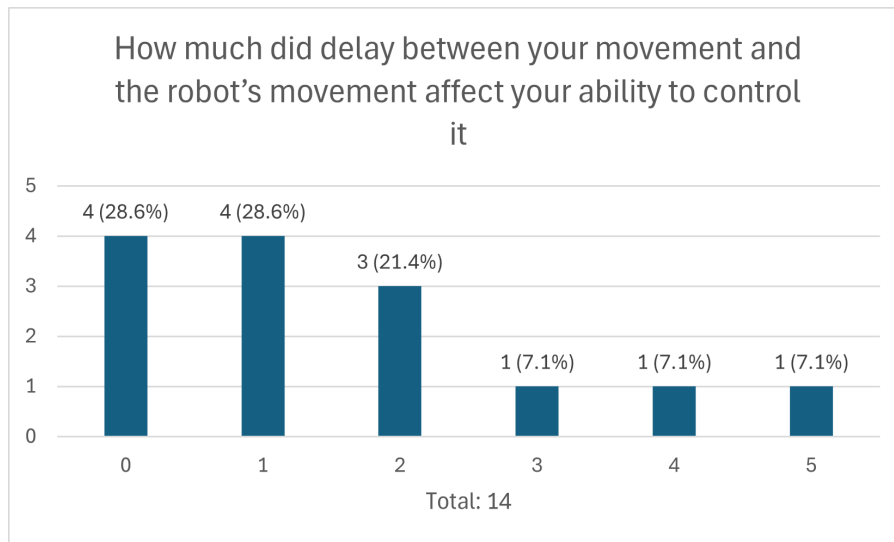


Figure 5.4: Q5: Perceived impact of delay on control ability (0–5).

Qualitative feedback supported this result, with some participants indicating that improved tuning (e.g., speed responsiveness) could further enhance controllability. Increasing responsiveness is feasible, but it also increases physical jitter in the UR5e, which in turn induces small vibrations in the supporting structure. A heavier and more rigid mounting could mitigate the vibration, but given the constraints of the available setup, the control parameters were tuned to a balanced level of responsiveness without inducing noticeable motion of the support structure.

5.2.2 Translational Control Intuitiveness (Q6–Q8)

Questions Q6–Q8 assessed whether the translational movements were intuitive: up/down (Q6), left/right (Q7), and forward/back (Q8). Participants rated translational control positively across all three directions:

- **Q6 (Up/Down):** mean 4.50 (median 5), Figure 5.5.

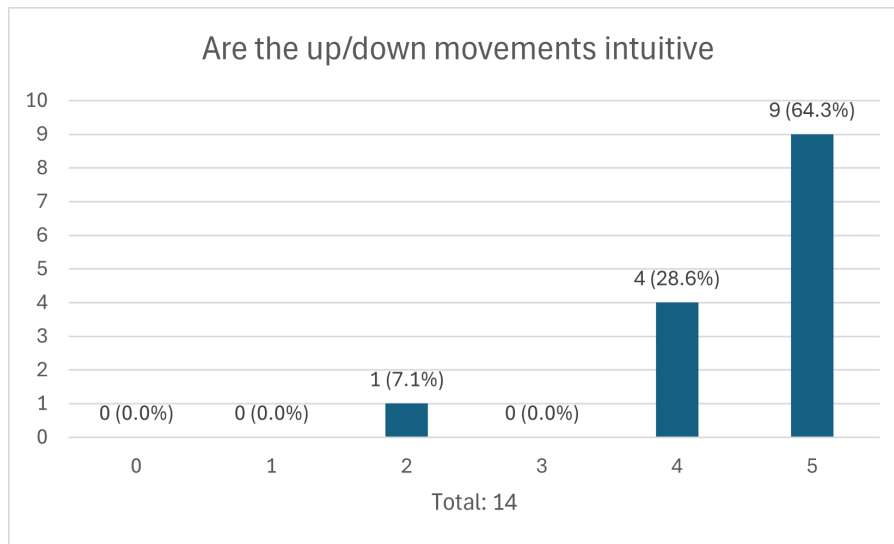


Figure 5.5: Q6: Intuitiveness of up/down translation (0–5).

- **Q7 (Left/Right):** mean 4.36 (median 5), Figure 5.6.

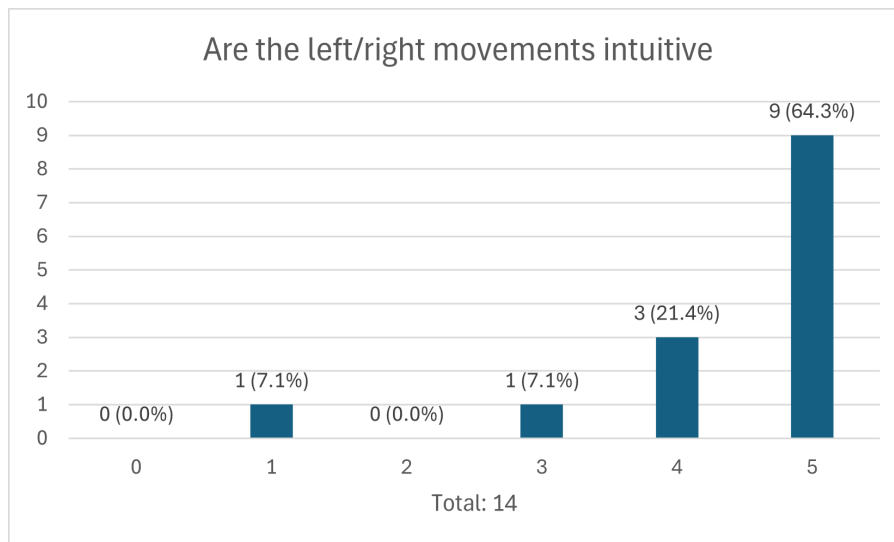


Figure 5.6: Q7: Intuitiveness of left/right translation (0–5).

- **Q8 (Forward/Back):** mean 4.21 (median 4), Figure 5.7.

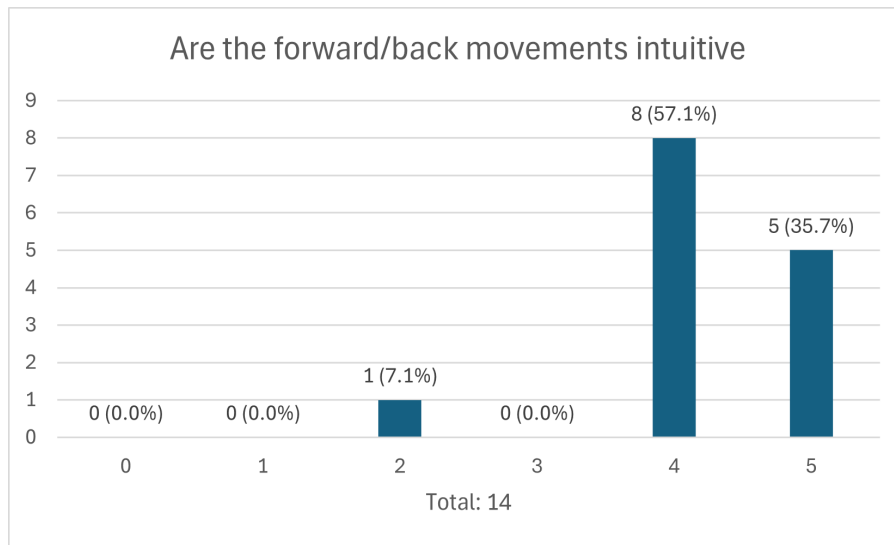


Figure 5.7: Q8: Intuitiveness of forward/back translation (0–5).

Comments indicate that translational control was generally easy to learn, although precision in some scenarios was perceived as challenging. One participant explicitly noted that precision could be difficult at times, suggesting that fine control (e.g., sensitivity adjustments) may be important for demanding tasks. To address this, the interface provides runtime tuning sliders such as position multiplier and speed/acceleration limits, allowing users to reduce responsiveness for finer, more precise adjustments when needed.

5.2.3 Rotational Control Intuitiveness (Q9–Q11)

Questions Q9–Q11 assessed rotational control: pitch (Q9), roll (Q10), and yaw (Q11). In this prototype, pitch (Q9) and yaw (Q11) were controlled using the controller joystick, whereas roll (Q10) was controlled through controller rotation. Results were positive overall, though slightly lower than translation for some axes:

- **Q9 (Pitch):** mean 4.21 (median 4), Figure 5.8.

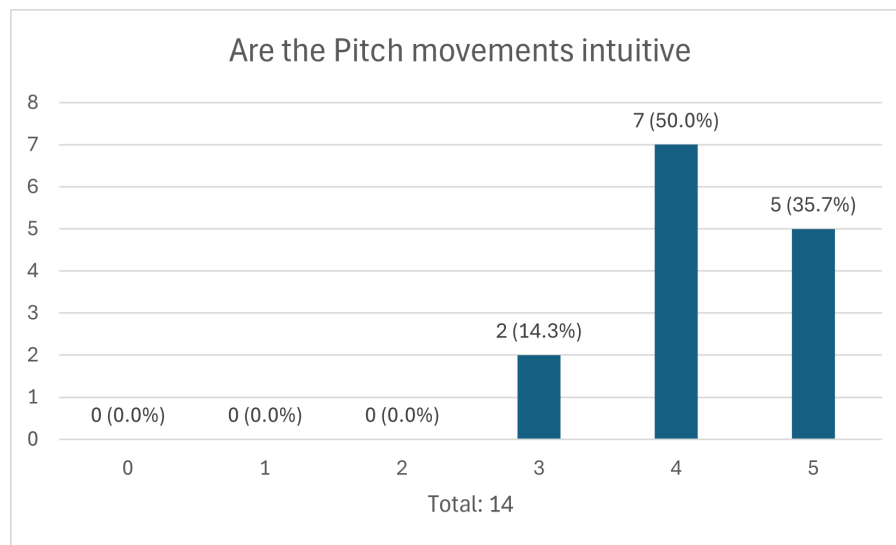


Figure 5.8: Q9: Intuitiveness of pitch rotation (0–5).

- **Q10 (Roll):** mean 3.93 (median 4), Figure 5.9.

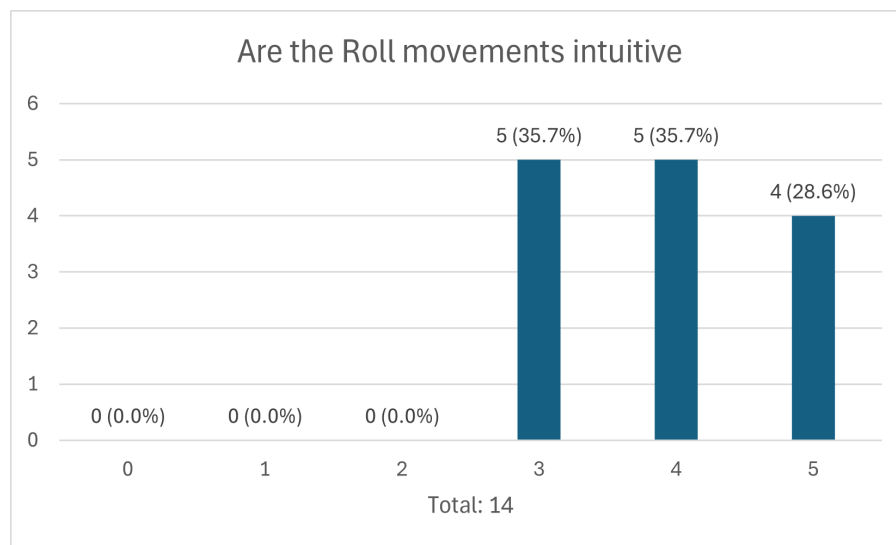


Figure 5.9: Q10: Intuitiveness of roll rotation (0–5).

- **Q11 (Yaw):** mean 4.14 (median 4), Figure 5.10.

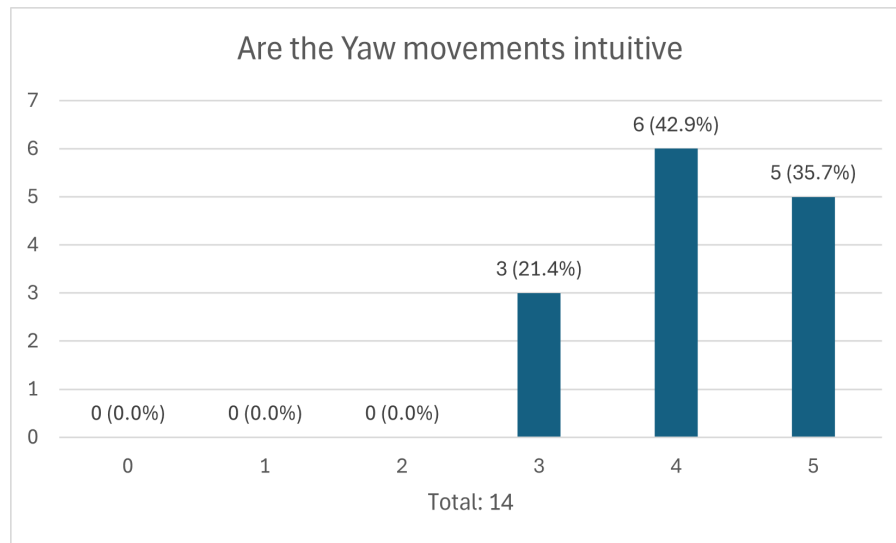


Figure 5.10: Q11: Intuitiveness of yaw rotation (0–5).

Roll received the lowest mean among the rotational axes, and participant comments align with this outcome. For example, one participant reported that roll precision was sometimes difficult, even though the overall interaction remained usable. Another participant highlighted challenges related to left-right movement and left-right rotation, suggesting that the mapping and/or sensitivity may require careful tuning for consistent rotational precision.

5.2.4 Overall Interface Intuitiveness (Q12)

Q12 evaluated whether the interface was intuitive (0–5). The overall rating was positive, with a mean of 4.00 and a median of 4. The distribution included one very low rating (0), as shown in Figure 5.11.

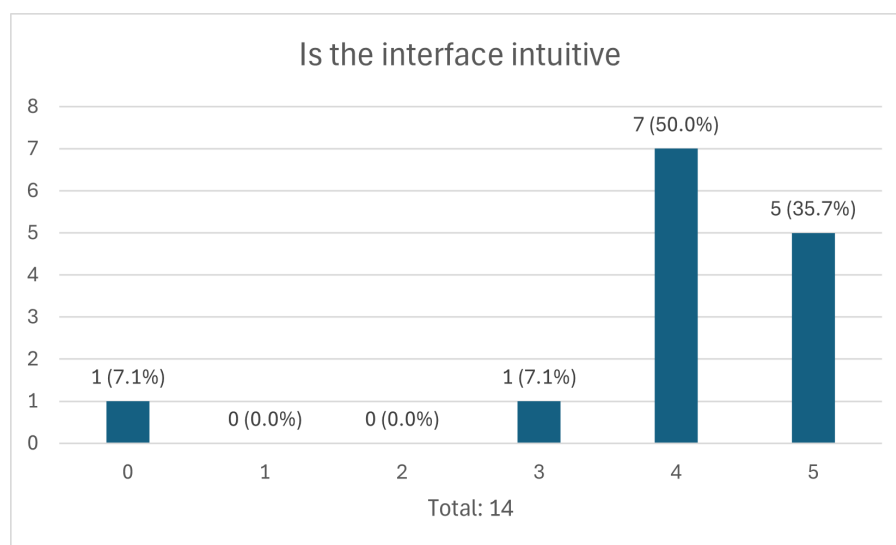


Figure 5.11: Q12: Overall interface intuitiveness (0–5).

The lowest interface rating was explicitly explained by a participant who could not read the interface due to vision constraints and the inability to use glasses comfortably with the HMD. This feedback indicates that interface readability and accessibility are critical factors for broader usability, independent of the control mapping itself. Additionally, another participant reported image blur, reinforcing the importance of display clarity and user-specific comfort in MR systems.

5.3 Qualitative Feedback Summary

In addition to numerical ratings, participants provided open feedback on potential use cases and general comments. Suggested application areas included operating in hazardous environments (e.g., toxic, radioactive, extreme temperature, or ATEX-like contexts), remote technical operations, training and demonstration (including programming-by-demonstration), and other scenarios where human risk reduction is valuable.

Several comments also suggested directions for future improvement, such as refining motion smoothness/linearity, adding support for two-handed control, and introducing a simple gripper-based task to better evaluate manipulation capability under realistic conditions.

5.4 Summary

Overall, participants reported high intuitiveness for translational control and generally positive ratings for rotational control, with roll identified as the most challenging axis. Perceived delay was typically reported as having low impact, though individual variation was observed. Interface usability was rated positively overall, but accessibility concerns (notably readability for users with strong vision correction needs) emerged as a relevant limitation. These results suggest concrete refinements in three areas. **Responsiveness:** although delay was generally reported as having low impact, participants indicated that improved tuning could further enhance controllability. Increasing responsiveness can be achieved by raising the commanded speed/acceleration limits and reducing filtering, but this also amplifies mechanical jitter in the UR5e and induced slight vibration of the supporting structure during testing. In the current setup, parameters were therefore tuned to a balanced level of responsiveness to avoid noticeable motion of the support, while still allowing users to adjust sensitivity at runtime via the implemented multipliers and limits. **Rotational precision:** roll was consistently the most challenging axis and comments suggest that further refinement of rotational sensitivity and smoothness is needed. Practical improvements include introducing a dedicated “fine rotation” mode (lower gain/step size near the target), gain scheduling and/or hysteresis to reduce oscillation around the desired orientation, and optional two-handed interaction to decouple translation from rotation during demanding adjustments; however, these measures may reduce perceived responsiveness and increase task time. **Interface accessibility:** the lowest interface rating was attributed to the inability to read the panels due to vision constraints, and additional comments reported blur. Future iterations should therefore provide adjustable UI scaling and distance, larger fonts and higher contrast, and a simplified layout for critical controls; the main trade-off is increased visual clutter and potential occlusion of the workspace, which must be balanced against readability.

Chapter 6

Conclusion and Future Work

This work presented and validated a MR teleoperation system for controlling a Universal Robots UR5e using a Meta Quest 3S headset and a Unity-based application. The proposed system integrates (i) an XR interaction layer for capturing controller pose and inputs, (ii) a real-time communication and control layer based on the UR dashboard interface and RTDE, and (iii) a digital twin rendered in MR to provide continuous visual feedback aligned with the physical workspace. Together, these components form a closed-loop teleoperation pipeline in which user actions are mapped to robot motion commands and the robot state is reflected back to the user through the MR overlay.

From an implementation perspective, the main contributions are the end-to-end integration of the Quest-based MR interface with the UR5e control stack, the clutch-based interaction mechanism that maps relative controller motion to robot motion only while the control button is held, and the MR user interface panels that expose connection management, dashboard operations, and runtime tuning parameters (e.g., translation/rotation scaling and jitter filtering). The digital twin is continuously synchronized from robot joint feedback, enabling the user to monitor robot motion directly in the MR scene and supporting iterative development and debugging.

A user study was conducted to evaluate the usability of the prototype, focusing on perceived responsiveness, translational control intuitiveness, rotational control intuitiveness, and overall interface intuitiveness. Results indicate that translational control was generally easy to learn and rated as intuitive by most participants, while rotational control was rated positively overall but with roll identified as the most challenging axis. Perceived delay was typically reported as having a low impact on controllability, although individual variation was observed. Importantly, qualitative feedback highlighted that fine control benefits from tuning (e.g., adjusting responsiveness), and that interface readability is a critical usability

factor: at least one participant reported difficulty reading the MR panels due to vision constraints, and another reported image blur.

These findings directly motivate the next technical refinements. Increasing responsiveness is feasible via higher gains and/or less aggressive filtering, but in the available setup this tended to amplify mechanical jitter, inducing slight vibration in the robot’s supporting structure; therefore, tuning must balance agility with physical stability (or alternatively be supported by a stiffer/heavier mounting). For rotational precision—particularly roll—practical improvements include a dedicated fine-control mode (lower gains near the target, gain scheduling and hysteresis to reduce oscillation), and optionally a two-handed interaction mode to decouple translation from rotation during precise adjustments. Finally, interface accessibility can be improved by offering adjustable panel scale and distance, larger typography, higher contrast, and a simplified layout for critical controls; the trade-off is increased visual clutter and potential occlusion of the workspace, which must be managed carefully in MR.

Overall, this project demonstrates that MR-based teleoperation with a synchronized digital twin can provide an intuitive and effective interaction paradigm for real robot control, while also revealing concrete engineering constraints that influence usability in practice. The implemented prototype provides a robust foundation for future work on higher-precision teleoperation and task execution, broader accessibility, and more advanced control strategies for improving stability without sacrificing responsiveness.

6.1 Future Work

Several extensions can improve robustness, usability, and applicability of the proposed MR teleoperation system.

Persistence and operator workflow. A practical next step is to persist user settings across sessions, including the robot host/IP, passthrough configuration, selected UR program, and interface parameters (e.g., multipliers and thresholds). This would reduce setup time and improve repeatability between test sessions.

Remote operation support. The current prototype assumes local network connectivity. A future iteration could enable secure remote control outside the local network (e.g., via VPN or a relay service), supported by a dedicated camera system mounted in the robot workspace to provide situational awareness during remote operation.

Scene understanding and virtual obstacle modelling. The MR environment could be augmented with the ability to place virtual objects that represent real-world obstacles in the workspace. This can be implemented either manually (operator-driven placement) or automatically through camera-based scanning and reconstruction. Once available, these virtual objects can be used for digital twin collision detection to provide warnings or constrain motion commands to reduce collision risk.

Expanded controller-based rotation control. Although the current implementation supports yaw, pitch, and roll, yaw and pitch are adjusted via joystick input for ergonomics and stability. A natural extension is full controller-based orientation control, where controller yaw and pitch are mapped directly to the robot tool orientation, enabling a unified 6-DoF controller interaction without relying on the joystick.

Hands-free interaction. Additionally, speech recognition could be integrated to trigger non-critical functions without manual interaction (e.g., connect/disconnect, toggling passthrough, switching modes, or setting presets), which may improve accessibility and reduce cognitive load during complex tasks.

6.2 Final Thoughts

This thesis demonstrated that MR can provide an effective interface for intuitive teleoperation when real-time feedback and consistent input mapping are maintained. The evaluation highlights that perceived controllability is shaped as much by practical engineering constraints (e.g., stability-responsiveness trade-offs and interface readability) as by the core control logic. The resulting prototype offers a solid foundation for extending MR teleoperation toward higher-precision control and more robust deployment scenarios, and for supporting future task-level extensions when needed.

References

- Ajewole, F., Kelkar, A., Moore, D., Shao, E., and Thirtha, M. (2023). Unlocking the industrial potential of robotics and automation. [Cited on page 3]
- Antonelli, M. G., Beomonte Zobel, P., Manes, C., Mattei, E., and Stampone, N. (2024). Emotional intelligence for the decision-making process of trajectories in collaborative robotics. *Machines*, 12(2). [Cited on pages 12, 14, and 18]
- Arents, J. and Greitans, M. (2022). Smart industrial robot control trends, challenges and opportunities within manufacturing. *Applied Sciences*, 12(2). [Cited on page 1]
- Azuma, R. T. (1997). A Survey of Augmented Reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385. [Cited on page 12]
- Bacca, J., Baldiris, S., Fabregat, R., Graf, S., and Kinshuk (2014). Augmented reality trends in education: A systematic review of research and applications. *Journal of Educational Technology I& Society*, 17(4):133–149. [Cited on page 3]
- Bassyouni, Z. and Elhajj, I. H. (2021). Augmented reality meets artificial intelligence in robotics: A systematic review. *Frontiers in Robotics and AI*, 8. [Cited on page 3]
- Chang, C. T. and Hayes, B. (2024). A survey of augmented reality for human-robot collaboration. *Machines*, 12(8). [Cited on page 3]
- Costa, G. d. M., Petry, M. R., and Moreira, A. P. (2022). Augmented reality for human-robot collaboration and cooperation in industrial applications: A systematic literature review. *Sensors*, 22(7). [Cited on page 5]
- De la Cruz, E. I., Salazar, E. R., Romero, J. A., Jiménez, L. M., and Rodríguez, J. J. (2022). Control to manipulate robotic arms using augmented reality. In Yang, X.-S., Sherratt, S., Dey, N., and Joshi, A., editors, *Proceedings of Sixth International Congress on Information and Communication Technology*, pages 101–112, Singapore. Springer Singapore. [Cited on pages 14 and 18]
- Fu, J., Rota, A., Li, S., Zhao, J., Liu, Q., Iovene, E., Ferrigno, G., and De Momi, E. (2023). Recent advancements in augmented reality for robotic applications: A survey. *Actuators*, 12(8). [Cited on page 1]
- Goodrich, M. and Schultz, A. (2007). Human-robot interaction: A survey. *Foundations and Trends in Human-Computer Interaction*, 1:203–275. [Cited on pages 3 and 4]

- Guo, H.-J., Bakdash, J. Z., Marusich, L. R., and Prabhakaran, B. (2022). Augmented reality and mixed reality measurement under different environments: A survey on head-mounted devices. *IEEE Transactions on Instrumentation and Measurement*, 71:1–15. [Cited on pages ix, 14, 15, and 18]
- Hamad, M., Nertinger, S., Kirschner, R. J., Figueredo, L., Naceri, A., and Haddadin, S. (2024). A concise overview of safety aspects in human-robot interaction. In Piazza, C., Capsi-Morales, P., Figueredo, L., Keppler, M., and Schütze, H., editors, *Human-Friendly Robotics 2023*, pages 1–18, Cham. Springer Nature Switzerland. [Cited on page 2]
- Hetrick, R., Amerson, N., Kim, B., Rosen, E., Visser, E. J. d., and Phillips, E. (2020). Comparing virtual reality interfaces for the teleoperation of robots. In *2020 Systems and Information Engineering Design Symposium (SIEDS)*, pages 1–7. [Cited on page 2]
- Holland, J., Kingston, L., McCarthy, C., Armstrong, E., O'Dwyer, P., Merz, F., and McConnell, M. (2021). Service robots in the healthcare sector. *Robotics*, 10(1). [Cited on pages 3 and 4]
- Jeffri, N. F. S. and Awang Rambli, D. R. (2021). A review of augmented reality systems and their effects on mental workload and task performance. *Heliyon*, 7(3):e06277. [Cited on page 3]
- Liu, Y., Li, Y., Zhuang, Z., and Song, T. (2020). Improvement of robot accuracy with an optical tracking system. *Sensors*, 20(21). [Cited on page 5]
- Manring, L., Pederson, J., Potts, D., Boardman, B., Mascarenas, D., Harden, T., and Cattaneo, A. (2020). Augmented reality for interactive robot control. In Dervilis, N., editor, *Special Topics in Structural Dynamics & Experimental Techniques, Volume 5*, pages 11–18, Cham. Springer International Publishing. [Cited on pages 11, 13, and 18]
- Martinelli, M., Lopes, S. I., and Migliardi, M. (2024). Predictive technologies and methodologies for human operator assessment in industry 5.0: A conceptual framework. In *2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8. [Cited on page 2]
- Parveau, M. and Adda, M. (2018). 3ivclass: a new classification method for virtual, augmented and mixed realities. *Procedia Computer Science*, 141:263–270. The 9th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2018) / The 8th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2018) / Affiliated Workshops. [Cited on page 4]
- Quandt, M. and Freitag, M. (2021). A systematic review of user acceptance in industrial augmented reality. *Frontiers in Education*, 6. [Cited on page 5]
- Robotec (2024). What is a collaborative robot (cobot)? Accessed: 2024-12-20. [Cited on page 12]

- Soares, I., Petry, M., and Moreira, A. P. (2021). Programming robots by demonstration using augmented reality. *Sensors*, 21(17). [Cited on pages 12, 13, and 18]
- Song, Q. and Zhao, Q. (2024). Recent advances in robotics and intelligent robots applications. *Applied Sciences*, 14(10). [Cited on pages 3 and 13]
- Suzuki, R., Karim, A., Xia, T., Hedayati, H., and Marquardt, N. (2022). Augmented reality and robotics: A survey and taxonomy for ar-enhanced human-robot interaction and robotic interfaces. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA. Association for Computing Machinery. [Cited on page 5]
- Tao, F., Xiao, B., Qi, Q., Cheng, J., and Ji, P. (2022). Digital twin modeling. *Journal of Manufacturing Systems*, 64:372–389. [Cited on page 2]
- Universal Robots (2021). *UR5e User Manual (SW 5.21) — Compliance / Introduction*. Universal Robots A/S. [Cited on page 27]
- Universal Robots (2025). *Remote operation of robots*. Universal Robots A/S. [Cited on page 24]
- Vávra, P., Roman, J., Zonča, P., Ihnát, P., Němec, M., Kumar, J., Habib, N., and El-Gendi, A. (2017). Recent development of augmented reality in surgery: A review. *Journal of Healthcare Engineering*, 2017(1):4574172. [Cited on page 3]
- Villagran-Vizcarra, D. C., Luviano-Cruz, D., Pérez-Domínguez, L. A., Méndez-González, L. C., and Garcia-Luna, F. (2023). Applications analyses, challenges and development of augmented reality in education, industry, marketing, medicine, and entertainment. *Applied Sciences*, 13(5). [Cited on pages 3 and 13]
- Wang, X. and Dunston, P. S. (2009). Comparative effectiveness of mixed reality based virtual environments in collaborative design. In *2009 IEEE International Conference on Systems, Man and Cybernetics*, pages 3569–3574. [Cited on pages 14 and 18]
- Wang, X., Ong, S. K., and Nee, A. Y. C. (2016). A comprehensive survey of augmented reality assembly research. *Advances in Manufacturing*, 4(1):1–22. [Cited on page 3]
- Wonsick, M. and Padir, T. (2020). A systematic review of virtual reality interfaces for controlling and interacting with robots. *Applied Sciences*, 10(24). [Cited on page 3]
- Zhang, G., Zhang, D., Duan, L., and Han, G. (2023). Accessible robot control in mixed reality. *arXiv preprint arXiv:2306.02393*. [Cited on pages 14 and 18]
- Zhao, F., Deng, W., and Pham, D. T. (2024). A robotic teleoperation system with integrated augmented reality and digital twin technologies for disassembling end-of-life batteries. *Batteries*, 10(11). [Cited on page 2]

Appendix A

Participant Responses (Portuguese)

Question	Answer	User 1	User 2	User 3	User 4	User 5	User 6	User 7
1 Sex	M/F/Prefer not say	M	M	F	M	M	M	M
2 Age Range	0-10/20-30/40-60/60+	30-40	30-40	40-60	20-30	30-40	40-60	30-40
3 VR experience	0-5	3	3	0	0	2	4	0
4 Do you wear glasses	Yes/No	No	No	Yes	Yes	No	No	Yes
5 How much did delay between your movement and the robot's movement affect your ability to control it?	0-5	0	1	1	2	3	0	1
6 Are the up/down movements intuitive	0-5	5	4	5	4	5	5	4
7 Are the left/right movements intuitive	0-5	5	4	5	4	5	5	4
8 Are the forward/back movements intuitive	0-5	5	4	5	4	4	5	4
9 Are the Pitch movements intuitive	0-5	5	4	4	3	4	4	4
10 Are the Roll movements intuitive	0-5	5	3	5	3	3	4	3
11 Are the Yaw movements intuitive	0-5	5	3	5	3	4	4	4
12 Is the interface intuitive	0-5	5	5	4	0	5	4	3
13 What do you think this could be useful for?	Open Response	Podrá ser útil para situaciones de resgate en locais peligrosos como locais ATEX, entre muitos outras ocasioes em que a vida humana possa estar em risco.	Para trabalhos de precisao e trabalhos em que seja util preservar o risco humano.	Manipulação de substâncias tóxicas, radioativas ou com temperaturas extremas, além de atividades de pesquisa envolvendo virus e bactérias.	Poderia ser util para aplicações onde a presença manual de um operador poderia ser ansuvida, como tomada de decisão por colaboradores em certas funções, para procedimentos que impliquem o uso de ferramentas muito especificas mas que têm de ser controlados manualmente.	manipulação de peças pesadas, manipulação em ambientes perigosos para pessoas, manipulação remota	tarefas manuais que afetem humanos do perigo, capacidade de interação manual remota	Realizar alguma atividade que pudessem representar algum perigo para o operador, por exemplo manusear algum produto nocivo que seria desajavel o operador estar a distancia
14 Comments	Open Response	Para mim, creio que faltará apenas ajustar a velocidade das juntas do robot em relação ao movimento do comando.	Os movimentos do robot serem mais lineares, e o uso da mão esquerda para poder controlar o robot.	Deu 0 na interface, porque como tenho uma elevada graduação e não da para estar com óculos, não consegui ler nada da interface: Não sei se é difícil de fazer porque faz parte das configurações do "capacete", mas se houvesse uma forma de "tocar", quase como fazes com um pioleto, podia ser uma vertente interessante de explorar, uma vez que cada vez mais temos uma população com algum problema de saúde relacionado com a visão.	podia ser interessante para a experiencia integrar uma tool simples como um gripper e dar uma tarefa ao utilizador de manipular ou transportar um corpo. Depois torna o utilizador teria mais noção de capacidades e limitações do sistema.			Por vezes é um pouco complicado ser preciso no movimento de roll, mas no geral parece ser algo aplicavel

Question	Answer	User 8	User 9	User 10	User 11	User 12	User 13	User 14
1 Sex	M/F/Prefer not say	F	M	F	M	M	M	M
2 Age Range	0-10/20-30/40-60/60+	30-40	40-60	30-40	40-60	30-40	20-30	40-60
3 VR experience	0-5	5	0	1	0	0	0	1
4 Do you wear glasses	Yes/No	No	No	Yes	Yes	No	Yes	Yes
5 How much did delay between your movement and the robot's movement affect your ability to control it?	0-5	4	0	1	2	5	0	2
6 Are the up/down movements intuitive	0-5	5	5	5	4	5	2	5
7 Are the left/right movements intuitive	0-5	5	5	5	3	1	5	5
8 Are the forward/back movements intuitive	0-5	5	5	4	4	2	4	4
9 Are the Pitch movements intuitive	0-5	5	5	4	4	5	3	5
10 Are the Roll movements intuitive	0-5	5	5	3	4	4	4	4
11 Are the Yaw movements intuitive	0-5	5	5	5	3	4	4	4
12 Is the interface intuitive	0-5	5	4	5	4	4	4	4
13 What do you think this could be useful for?	Open Response	Tenho e "programação por demonstração" com trajetórias suaves em linhas de produção com mudanças frequentes da trajetória do robo	Não consigo discernir a utilidade destes movimentos virtuais.	Vejo esse tipo de tecnologia implementadas em fabricas, por exemplo, permitindo aos humanos manusear os robos em situações específicas, não estando em contacto direto com o ambiente onde o robo está. Por exemplo, imaginando que o robo opera num local muito rto, o humano consegue controlar e manuseá-lo a distancia, sem ter de se expor a essas condições climáticas.	Penso que pode ser útil na demonstração de conceitos, e na utilização em modo remoto	operações técnicas a distancia	Imitation Learning	Trinlar robots, operar robots remotamente
14 Comments	Open Response	Bastante interativo		O facto de usar o osculos com os osculos de VR dificultou um pouco a leitura do menu - ainda que desse para me chegar mais perto para ver melhor. Além disso, achei que o movimento do braço real não era muito fluido. Em relação ao facto de haver um braço virtual muito perto do real, fez com que me confundisse um pouco, tendo optado por me focar nos movimentos reais e não nos virtuais	Os movimentos esquerda direita e rotação esquerda direita não eram muito precisos, a sensibilidade podia ser melhorada, porque com um movimento que não era necessariamente brusco, o robot respondia "aos arranques".	Existo algum destoque de imagem		O facto de alguns movimentos terem de ser feitos com o joystick são mais difíceis de serem usados e menos intuitivos

Appendix B

Participant Responses (English translation)

Question	Answer	User 1	User 2	User 3	User 4	User 5	User 6	User 7
1 Sex	M/F/Prefer not say	M	M	F	M	M	M	M
2 Age Range	0-10/20-30/40-60/60+	30-40	30-40	40-60	20-30	30-40	40-60	30-40
3 VR experience	0-5	3	3	0	0	2	4	0
4 Do you wear glasses	Yes/No	No	No	Yes	Yes	No	No	Yes
5 How much did delay between your movement and the robot's movement affect your ability to control it?	0-5	0	1	1	2	3	0	1
6 Are the up/down movements intuitive	0-5	5	4	5	4	5	5	4
7 Are the left/right movements intuitive	0-5	5	4	5	4	5	5	4
8 Are the forward/back movements intuitive	0-5	5	4	5	4	4	5	4
9 Are the pitch movements intuitive	0-5	5	4	4	3	4	4	4
10 Are the Roll movements intuitive	0-5	5	3	5	3	3	4	3
11 Are the Yaw movements intuitive	0-5	5	3	5	3	4	4	4
12 Is the interface intuitive	0-5	5	5	4	0	5	4	3
13 What do you think this could be useful for?	Open Response	It could be useful for rescue situations in dangerous places such as ATEX areas, among many other occasions where human life may be at risk.	For precision work and tasks where reducing human risk is useful.	Handling toxic or radioactive substances, or substances at extreme temperatures, as well as research activities involving viruses and bacteria.	It could be useful for applications where an operator's physical/manual presence could be risky, as a way to train employees in certain functions, for procedures that require very specific tools but still need to be manually controlled.	Handling heavy parts, handling in environments that are dangerous for people, remote handling.	Manual tasks that keep humans away from danger, ability for remote manual interaction.	Carrying out an activity that could represent danger for the worker, for example handling a harmful product where it would be preferable for the worker to stay at a distance.
14 Comments	Open Response	For me, I think it will only be necessary to adjust the speed of the robot joints relative to the controller movement.	The robot's movements should be more linear, and there should be support for using the left hand to control the robot.		I gave 0 to the interface because, as I have a strong prescription and cannot wear glasses, I could not read anything on the interface. I do not know if it is difficult to do because it is part of the headset settings, but if there were a way to "focus", almost like you do with a projector, it could be an interesting direction to explore, since more and more we have a population with some health issue related to vision.	It could be interesting for the experiment to integrate a simple tool like a gripper and give the user a task to manipulate or transport an object. This way the user would have a better sense of the system's capabilities and limitations.		Sometimes it is a bit complicated to be precise in the roll movement, but overall it seems applicable.

Question	Answer	User 8	User 9	User 10	User 11	User 12	User 13	User 14
1 Sex	M/F/Prefer not say	F	M	F	M	M	M	M
2 Age Range	0-10/20-30/40-60/60+	30-40	40-60	30-40	40-60	30-40	20-30	40-60
3 VR experience	0-5	5	0	1	0	No	0	1
4 Do you wear glasses	Yes/No	No	No	Yes	Yes	No	Yes	Yes
5 How much did delay between your movement and the robot's movement affect your ability to control it?	0-5	4	0	1	2	5	0	2
6 Are the up/down movements intuitive	0-5	5	5	5	4	5	2	5
7 Are the left/right movements intuitive	0-5	5	5	5	3	1	5	5
8 Are the forward/back movements intuitive	0-5	5	5	4	4	2	4	4
9 Are the pitch movements intuitive	0-5	5	5	4	4	5	3	5
10 Are the Roll movements intuitive	0-5	5	5	3	4	4	4	4
11 Are the Yaw movements intuitive	0-5	5	5	5	3	4	4	4
12 Is the interface intuitive	0-5	5	4	5	4	4	4	4
13 What do you think this could be useful for?	Open Response	Training and "programming by demonstration" with smooth trajectories on production lines with frequent changes to the robot trajectory.	I cannot see the usefulness of these virtual movements.	I see this type of technology being implemented in factories, for example allowing humans to handle robots in specific situations without being in direct contact with the environment where the robot is. For instance, if the robot operates in a very cold place, a human can control and handle it remotely without having to be exposed to those climatic conditions.	I think it can be useful for demonstrating concepts, and for use in remote mode.	Remote technical operations.	Imitation Learning	Train robots, operate robots remotely
14 Comments	Open Response		Quite interactive.	Using glasses together with the VR glasses made it a bit harder to read the menu - although I could move closer to see better. Also, I felt the real arm movement was not very smooth. Regarding the virtual arm being very close to the real one, it confused me a little, so I chose to focus on the real movements and not the virtual ones.	The left-right movements and left-right rotation were not very precise, the sensitivity could be improved because, with a movement that was not necessarily abrupt, the robot would respond "in jerks".	There is some image blur.		The fact that some movements have to be done with the joystick makes them harder to use and less intuitive.

Appendix C

Gantt Chart

