Universidade de Aveiro
2024

João Luís Fernandes Clemente

Uso de um braço robótico para auxiliar cenários de Colaboração Remota apoiada por Realidade Mista

Using a Robotic Arm to Assist during Scenarios of Remote Collaboration supported by Mixed Reality

Universidade de Aveiro
2024

João Luís Fernandes Clemente

Uso de um braço robótico para auxiliar cenários de Colaboração Remota apoiada por Realidade Mista

Using a Robotic Arm to Assist during Scenarios of Remote Collaboration supported by Mixed Reality

Tese apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Robótica e Sistemas Inteligentes , realizada sob a orientação científica do Doutor (nome do orientador), Professor associado do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, do Doutor (co-orientador), Professor auxiliar convidado do Departamento de Matemática da Universidade de Aveiro, da Doutora (co-orientadora), Professora associada c/ agregação do Departamento de Biologia da Universidade de Aveiro, e do Doutor (co-orientador), Professor auxiliar convidado do Departamento de Física da Universidade de Aveiro.

ponho aqui algo?

o júri / the jury

presidente / president	Prof. Doutor João Antunes da Silva professor associado da Universidade de Aveiro
vogais / examiners committee	Prof. Doutor João Antunes da Silva professor associado da Universidade de Aveiro
	Prof. Doutor João Antunes da Silva professor associado da Universidade de Aveiro
	Prof. Doutor João Antunes da Silva professor associado da Universidade de Aveiro
	Prof. Doutor João Antunes da Silva professor associado da Universidade de Aveiro
	Prof. Doutor João Antunes da Silva professor associado da Universidade de Aveiro

agradecimentos / acknowledgements

Esta dissertação consiste no culminar de um percurso académico que se estendeu ao longo de 17 anos.
Deste modo, gostaria de começar por agradecer à minha família, especialmente à minha mãe e ao meu pai por me terem proporcionado esta aventura, bem como, por terem celebrado todas as minhas conquistas.
Em segundo lugar, gostaria de agradecer aos meus orientadores, Professor Doutor Bernardo Marques e Professor Doutor Eurico Pedrosa, pelo apoio e orientação na conceção deste trabalho. Pelas diversas horas nas quais se dispuseram a ajudar-me, de modo a poder levar este projeto a bom porto. Gostaria também de agradecer ao Professor Doutor João Alves, por me ter transmitido o seu conhecimento e experiência, fundamentais para a realização deste trabalho.
Por fim, mas não menos importante, gostaria de agradecer a todos os amigos que fiz ao longo deste percurso, tendo um especial carinho pela "familia" que criei em Aveiro, por terem partilhado comigo belos momentos e pela constante presença e apoio nos últimos 5 anos.
A todos aqueles que se cruzaram comigo e de algum modo contribuíram para o meu crescimento pessoal e académico, o meu muito obrigado.

palavras-chave

texto livro, arquitetura, história, construção, materiais de construção, saber tradicional.

resumo

Um resumo é um pequeno apanhado de um trabalho mais longo (como uma tese, dissertação ou trabalho de pesquisa). O resumo relata de forma concisa os objetivos e resultados da sua pesquisa, para que os leitores saibam exatamente o que se aborda no seu documento.
Embora a estrutura possa variar um pouco dependendo da sua área de estudo, o seu resumo deve descrever o propósito do seu trabalho, os métodos que você usou e as conclusões a que chegou.
Uma maneira comum de estruturar um resumo é usar a estrutura IMRaD. Isso significa:

- Introdução
- Métodos
- Resultados
- Discussão

Veja mais pormenores aqui:
`https://www.scribbr.com/dissertation/abstract/`

keywords textbook, architecture, history, construction, construction materials, traditional knowledge.

abstract An abstract is a short summary of a longer work (such as a thesis, dissertation or research paper).
The abstract concisely reports the aims and outcomes of your research, so that readers know exactly what your paper is about.
Although the structure may vary slightly depending on your discipline, your abstract should describe the purpose of your work, the methods you've used, and the conclusions you've drawn.
One common way to structure your abstract is to use the IMRaD structure. This stands for:

- Introduction
- Methods
- Results
- Discussion

Check for more details here:
`https://www.scribbr.com/dissertation/abstract/`

reconhecimento do uso de ferramentas de AI

Reconhecimento do uso de tecnologias e ferramentas de Inteligência Artificial (IA) generativa, softwares e outras ferramentas de apoio.
Reconheço a utilização do ChatGPT 4 (Open AI, https://chat.openai.com) para melhorar a escrita académica e para fornecer sugestões de código e assistência no desenvolvimento do software.

I acknowledge the use of ChatGPT 4 (Open AI, https://chat.openai.com) for refining academic language and offering code suggestions, aiding in the development of the software components.

Contents

List of Figures

List of Tables

Acronyms

AI	Artificial Intelligence
AR	Augmented Reality
AR	Augmented Reality
AV	Augmented Virtuality
CPS	Cyber-Physical System
CPSS	Cyber-Physical Systems
DRs	Digital Realities
DT	Digital Twin
DTs	Digital Twins
FOV	Field of View
HHD	Handheld Device
HHDs	Handheld Devices
HMDs	Head-Mounted Displays
HMI	Human-Machine Interaction
HRC	Human-Robot Collaboration
HRCp	Human-Robot Cooperation
HRCx	Human-Robot Coexistence
HRI	Human-Robot Interaction
IDE	Integrated Development Environment
IOT	Internet of Things
ML	Machine Learning
MR	Mixed Reality
RL	Reinforcement Learning
ROS	Robot Operating System
SMEs	Small and Medium Enterprises
TCP/IP	Transmission Control Protocol/Internet Protocol
UI	User Interface
URDF	Unified Robot Description Format
VR	Virtual Reality
XR	Extended Reality

CHAPTER 1

Introduction

1.1 Motivation

The First Industrial Revolution, powered by steam engines, paved the way for subsequent revolutions driven by electricity, automation, machinery, and the internet. Each revolution introduced groundbreaking technologies that reshaped industries, emphasizing the companies' need to prioritize reskilling and upskilling their workforce.

However, experts argue that complete removal of humans from the manufacturing processes is not feasible [1]. Instead, there is a growing emphasis on fostering collaborative partnerships between humans and intelligent machinery.

As collaborative environments evolve, robots have become indispensable in various domains, leading to increased complexity in these scenarios. Therefore, advanced solutions are needed to enhance HRC. One promising approach consists on integrating Mixed Reality (MR) technologies, which encompass both Virtual Reality (VR) and Augmented Reality (AR) by blending the physical and digital world, thus providing immersive experiences that transcend traditional reality and overcome geographical constraints, enabling real-time collaboration among individuals from different locations.

However, the potential of MR to enhance remote collaboration is currently hindered by several critical limitations. These include, not only limited perspective and context capture, which impede remote collaborators' understanding and decision-making capabilities, as well as a lack of multisensory data collection, restricting comprehensive environmental comprehension. Additionally, MR interaction with physical objects often lacks the precision required for detailed tasks, particularly in dynamic scenarios.

These challenges diminish the effectiveness of MR in facilitating thorough context sharing and impact the overall efficiency and safety of collaborative tasks.

1.2 Goals

The primary goal of this dissertation consists on leveraging Mixed Reality (MR) alongside a static robotic arm (UR10e) to support remote collaboration scenarios. This involves transforming Human-Robot Collaboration (HRC) by integrating Mixed-Realities (MR) technologies and robotic capabilities to enhance both on-site and remote collaboration experiences.

In order to properly achieve this, the main goal can be broken down into the following objectives:

- **On-Site Interaction:**
  - Enable dynamic and real-time collaboration with the robot within the designated environment.
  - Utilize Handheld Devices (HHDs), such as tablets or smartphones, to share live views of the surroundings, allowing remote collaborators to gain a comprehensive understanding of the collaborative space.
- **Remote Visualization and Interaction:**
  - Provide remote participants with a foundational 2D interface, such as a laptop screen, to visualize the collaboration scenario and task context.
  - Establish a bidirectional communication, enabling remote operation of the robot via the MR application, enhancing the user's ability to interact and manipulate the collaborative environment.
- **Automation and Immersion:**
  - Integrate a camera into the robot to automate the process of environment sharing with remote participants, assisting on-site participants by delegating visual sharing to the robot.
  - Incorporate visual and audio cues to enhance the user's perception and interaction within the collaborative space.

By addressing these objectives, this dissertation aims to establish a framework that enhances remote collaboration through the innovative use of Mixed-Reality (MR) and robotic technologies.

1.3 Thesis Structure

This dissertation is structured across six chapters, each representing a logical progression in the development of this work within the context of Human-Robot Collaboration (HRC) and Industry 5.0.

The first chapter introduces the project by framing the motivation behind this work as well as defining the baseline for this dissertation's development. The following chapter presents a thorough state-of-the-art review, with a focus on key concepts such as Industry 5.0, Human-Robot Collaboration (HRC), Collaborative Robots, Digital Realities (DRs), and Digital Twins (DTs). It also includes an analysis of two case studies involving Augmented Reality (AR)-Digital Twin (DT) solutions in industrial settings. The third chapter discusses the various implementation tools utilized in the project development, such as Unity, Vuforia, and ROS. Chapter four outlines the framework established for the project and details the specific features implemented, distinguishing those meant for remote members from those designed for on-site collaborators. The fifth chapter is devoted to evaluating the developed features, identifying their limitations, and presenting an application example where the solution could be effectively implemented. Finally, the sixth chapter concludes the thesis by summarizing the project's achievements and suggesting possible directions for future work.

CHAPTER 2

State of Art

2.1 Key Drivers of Industry 5.0

With the advent of Industry 4.0 and the emerging concept of Industry 5.0, the industry environment has witnessed significant transformations. Understanding this progress is crucial to contextualize the technological advancements and the shift towards more human-centric manufacturing processes.

2.1.1 Industry 4.0: The Fourth Industrial Revolution

Industry 4.0 represents the integration of cutting-edge digital technologies into manufacturing processes, leading to the emergence of smart factories. It leverages advanced systems such as Cyber-Physical Systems (CPSS), the Internet of Things (IOT), robotics, Augmented Reality (AR), simulation, cloud computing, and big data analytics, as illustrated in Figure 2.1. This paradigm signifies a fundamental shift towards interconnected, intelligent, and digitally-driven manufacturing ecosystems, revolutionizing the way products are conceived, manufactured, and delivered, while enhancing production efficiency, flexibility, and innovation [2, 3].

With advancements in Artificial Intelligence (AI), industrial processes can achieve unprecedented performance levels, often exceeding human capabilities. These AI-driven systems enable robots to perform tasks that may be too hazardous, complex, or delicate for humans, such as handling dangerous materials or managing microscopic elements. Despite this extraordinary potential, it is important to recognize that current industrial robots are not as "smart" as humans in many contexts and, even though these robots are capable of performing highly skilled tasks, they frequently operate under strict, pre-programmed limits [3].

Although Industry 4.0 has undoubtedly increased productivity, flexibility, and automation in industrial environments, it has also led to concerns regarding the diminishing role of human operators. This relentless push towards full automation has, in some cases, reduced human involvement in critical decision-making processes, leading to a more machine-centric production landscape [4].

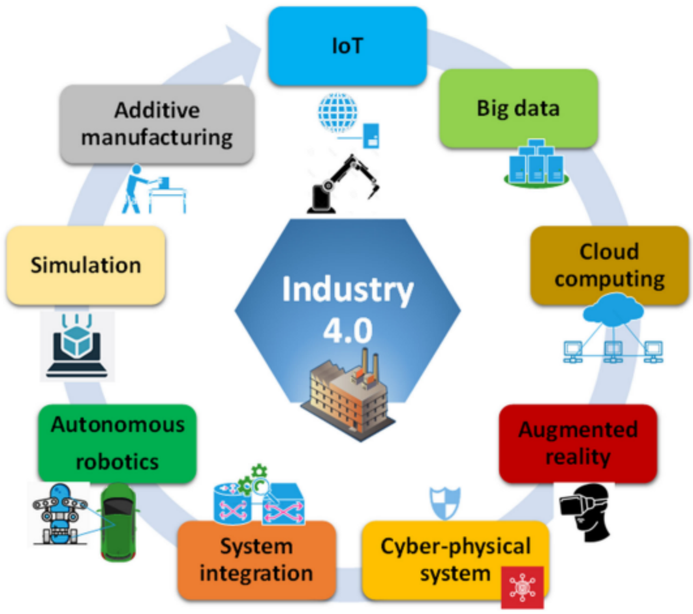


Figure 2.1: Key enabling technologies in Industry 4.0 [3]

2.1.2 Industry 5.0: Reintegrating the Human Element

Approximately a decade after the launch of Industry 4.0, the European Commission introduced the Industry 5.0 concept in response to new societal challenges [5]. The growing concerns about the exclusion of human operators in Industry 4.0 systems, coupled with the limitations of full automation, paved the way for this new industrial paradigm. Industry 5.0 seeks to reintroduce the human element into industrial ecosystems, emphasizing greater human involvement in manufacturing processes [6].

The main goal consists on combining the strengths of humans and machines to achieve more sustainable, efficient, and human-centered production systems. This shift reflects the realization that, while machines excel at repetitive, dangerous, or complex tasks, humans provide irreplaceable creativity, adaptability, and problem-solving abilities [7]. Industry 5.0 aims to strike a balance between technological advancement and human-centric values, fostering environments where humans work alongside advanced technologies to achieve greater societal and environmental outcomes [4].

Recognizing that humans and machines each possess distinct strengths that can complement one another, the following key technological drivers of Industry 5.0 build upon the advancements of Industry 4.0:

- **Artificial Intelligence and Cognitive Computing:** These technologies continue to evolve, enabling robots and automation systems to work alongside humans in ways that enhance productivity without fully replacing them. AI allows for more intuitive Human-Machine Interaction (HMI), where machines can understand and respond to human needs more effectively.
- **Collaborative Robots (Cobots):** are engineered to ensure safe, collaborative operation alongside human workers, facilitating not only intuitive interactions but also

fostering efforts that leverage the unique strengths of both humans and robots. Their integration is driven by the need to create systems that enable seamless, user-friendly HRC, in full alignment with the guiding principles of Industry 5.0. This paradigm shift redefines traditional employment roles by emphasizing HRI, with a focus on communication and coordination with robotic systems and advanced AI [7].

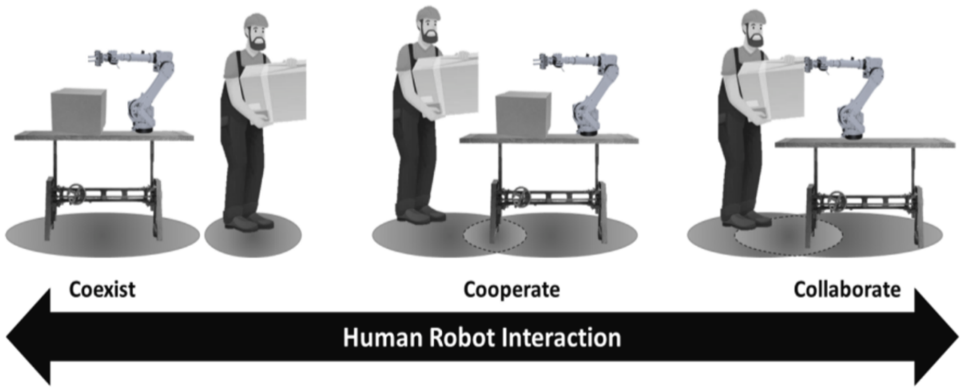
- **Digital Twins:** represent a pivotal technological advancement in Industry 5.0. They provide visual models that enhance comprehension and facilitate the evaluation of goods, processes, and production systems. By allowing real-time monitoring and simulation, DT help optimize manufacturing processes, bridging the gap between the virtual and physical worlds [7].
- **Human-Centric Automation:** Emphasis is placed on using technology to augment human capabilities rather than replace them, fostering a more inclusive, creative, and flexible manufacturing environment. This approach ensures that technology empowers human workers, enabling them to focus on tasks requiring intuition and creativity.
- **Advanced Human-Machine Interfaces:** The development of intuitive interfaces, by integrating technologies such as AR and VR, facilitates better communication between humans and machines. These interfaces allow for more natural interactions, improving understanding and efficiency in collaborative tasks.
- **Sustainable and Resilient Manufacturing:** Industry 5.0 also focuses on sustainability and resilience, integrating environmental considerations into manufacturing processes. This includes optimizing resource usage and reducing waste, aligning technological advancement with ecological responsibility.

By integrating these key drivers, Industry 5.0 addresses the challenges identified in Industry 4.0, promoting harmonious collaboration between humans and machines. This synergy aims to enhance productivity while preserving the unique contributions of human workers, ultimately leading to more innovative, sustainable, and human-centered industrial practices.

2.2 Human-Robot Collaboration

Human-Robot Interaction (HRI) is an extensive field dedicated to examining the interactions and coexistence of humans and robots in shared spaces, whose objective consists on enhancing these interactions by designing robots that are safe, effective and compatible for assisting and cooperating with humans in diverse roles, rather than replacing them [8]. This involves developing robots that are, not only autonomous, but also capable of understanding and communicating with humans, as well as predicting human-behavior and learning from human feedback.

However, HRI can be broken down into different forms of interaction, whose categorization is based on various factors that define how humans and robots share the workspace. This disctinction is represented in the Figure 2.2 and can be broken down into **Coexistence**, **Cooperation** and **Collaboration** [9]. Each one can be distinguished by the degree of interaction and task sharing between humans and robots:



Space	Separated	Shared	Shared
Tasks	Unrelated	Linked	Shared
Contact	Impossible	Possible	Desired
Context Awareness	Not necessary	Minimal	Necessary

Figure 2.2: Different levels of HRI - [9]

- **Human-Robot Coexistence (HRCx)**: In this form of interaction, humans and robots operate in the same environment but perform entirely independent tasks without interaction. The workspace is separated, there is no contact, and context awareness is unnecessary since the tasks are unrelated. Usually this interaction does not involve synchronized work or communication between the two parties.
- **Human-Robot Cooperation (HRCp)**: Here, humans and robots share the workspace and work on linked tasks, towards a common goal. Advanced technologies such as sensors or machine vision may be used to detect and prevent collisions. Contact is possible, though not essential and actions are largely independent with occasional coordinated efforts.
- **Human-Robot Collaboration (HRC)**: represents the most advanced and integrated form of HRI. In HRC, humans and robots not only share a common workspace but also actively collaborate on shared objectives. This collaboration can involve direct physical contact, such as the joint manipulation of objects, or non-physical interaction, including verbal communication, gestures, or pattern recognition. Within such environments, humans often handle tasks that require fine motor skills, decision-making, or creative problem-solving, while robots take on repetitive, strenuous, or hazardous activities, ensuring efficiency and safety. This synergy enhances productivity by leveraging the unique strengths of both humans and robots, creating a dynamic partnership where each complements the other's capabilities.

Underneath the image, the table further breaks down these distinctions, highlighting key factors like space, task relationship, possibility of contact, and the need for context awareness. This gradient from coexistence to collaboration shows the increasing complexity and interdependence in HRI, as technology evolves to make robots more capable partners in industrial and service environments.

Below, the Figure 2.3 illustrates a HRC workspace, showcasing the interaction between a human worker and a robot within a shared environment. The workspace is divided into distinct yet overlapping areas: the robot's work envelope and the human's work envelope. These areas reflect the respective tasks of each party, with the robot likely performing repetitive or automated tasks, such as material handling within the feeding system, while the human focuses on more intricate tasks requiring dexterity and decision-making.

The overlapping shared workspace demonstrates the core principle of HRC, where humans and robots work together toward common goals, necessitating real-time coordination and communication. In this setup, advanced sensing technologies or machine vision are essential to ensure safety and prevent collisions, allowing both the human and robot to operate efficiently within close proximity.

This image underscores how robots, rather than replacing humans, complement human skills by taking on routine, physically demanding tasks, while humans contribute with their cognitive abilities. This partnership reflects the broader vision of Industry 5.0, where human creativity and robotic precision are combined to create adaptable, human-centered industrial processes, enhancing both efficiency and safety in collaborative environments.

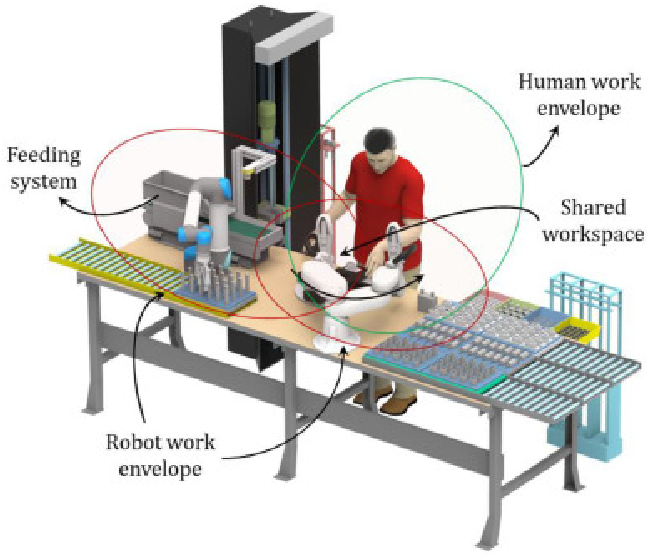


Figure 2.3: Human-robot collaborative workstation - [10]

These new robots featuring intelligent sensing and vision systems, envisioned to integrate the production line, are called "cobots". They represent the alternative to full automation, since industry specialists have stated it is not possible to completely remove the human within the manufacturing environment [1].

2.3 Collaborative Robots (Cobots)

In the past decade, the market has introduced a new category of robots known as collaborative robots, or "cobots." These are designed to interact physically with humans within the same workspace, devoid of the traditional barriers or protective cages typical in conventional robotic systems. Cobots are celebrated for their ability to quickly and inexpensively adapt layouts, although properly harnessing their potential requires a thorough understanding of their proper uses and characteristics, which can otherwise pose barriers to industry adoption [11].

Unlike traditional industrial robotic systems, which require extensive safety guarding and consequently reduce flexibility while increasing both costs and spatial demands, cobots present a solution tailored to the current market's demand for shorter lead times and mass customization, particularly for Small and Medium Enterprises (SMEs) [12].

This cobots emergence represents a paradigm shift in industrial automation, emphasizing HRC over the traditional model of robotic isolation. Cobots facilitate direct physical interaction between humans and machines while being designed for intuitive use, enabling even non-experts to reprogram them effortlessly [13]. By leveraging the complementary strengths of human cognitive capabilities and robotic precision, cobots offer substantial productivity gains and reduced operational costs.

The concept of cobots was first introduced by J. Edward Colgate and Michael Pashkin in 1996 [14], laying the groundwork for practical applications in HRC. Commercializing cobots began with the release of the UR5 model by Universal Robots in 2008 [11], marking a significant milestone in the advancement of HRC and facilitating the integration of collaborative robotics into industrial workflows.

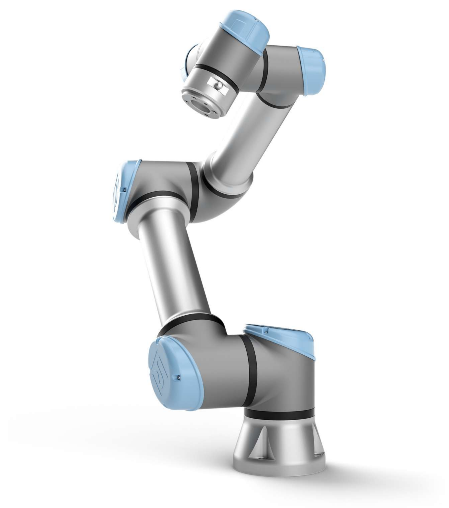


Figure 2.4: Universal Robots UR5 cobot - [15]

Cobots differentiate themselves from traditional industrial robots by prioritizing safety, ergonomics, and user accessibility. Unlike conventional robots that require extensive safety enclosures, cobots are equipped with advanced features such as force and torque sensors, vision systems, and anti-collision mechanisms. These capabilities enable them to operate safely in close proximity to humans without the need for restrictive barriers [16]. The inherent design of cobots supports flexibility and ease of deployment, avoiding the high costs and complexity associated with retrofitting traditional robotic systems for similar functionality.

The adoption of cobots in industrial settings is driven by a combination of economic, operational, and health-related factors:

- **Cost Efficiency:** Cobots can significantly reduce labor costs by performing repetitive tasks, thereby lowering direct unit production costs compared to traditional automation solutions [17].
- **Enhanced Workplace Safety:** Their design minimizes occupational hazards, which leads to improved worker safety and health, addressing ergonomic challenges in manual labor.
- **Spatial Efficiency:** The compact and flexible nature of cobots allows them to be easily relocated and reconfigured within different production areas, optimizing factory space utilization [18].

These attributes are particularly beneficial in high-risk applications and industries that demand frequent changes in production layouts, such as electronics, automotive, and aerospace manufacturing.

When assessing the applicability of cobots versus traditional robots, several distinctions emerge. Cobots excel in tasks that require adaptability and human-like dexterity, such as assembly, placement, handling, and quality inspection. Their versatility and ease of integration make them suitable for low-volume, high-mix production environments, where agility is crucial. However, traditional robots retain an advantage in scenarios that demand high payload capacity, speed, and precision, particularly in heavy-duty manufacturing applications [11].

Despite their advantages, integrating cobots into existing workflows poses challenges. Issues such as interoperability with legacy systems, programming complexity for non-standard tasks, and optimizing cobot performance in dynamic environments require further research and development. Additionally, advancements in AI and Machine Learning (ML) could unlock new capabilities for cobots, enabling them to autonomously adapt to changing tasks and work conditions, thereby extending their utility beyond predefined, structured environments.

2.4 Digital Realities

DRs encompass a wide spectrum of technologies that merge virtual elements with real-world environments to varying extents. In 1994, Milgram and Kishino introduced the Reality-Virtuality Continuum, a theoretical framework that characterizes the progression from a purely physical environment to a fully virtual one, as illustrated in Figure 2.5 [19]. This continuum is divided into four principal stages: Reality, Augmented Reality AR, Augmented Virtuality (AV), and Virtual Reality VR.

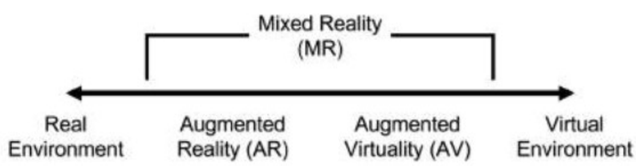


Figure 2.5: Reality-Virtuality Continuum, from [19]

In this continuum, Reality represents the perception of an unaltered physical environment, devoid of any virtual modifications. As we progress along the continuum towards the virtual side, different digital realities offer increasingly immersive experiences by blending or replacing real-world content with virtual elements.

Augmented Reality

AR enhances a user's interaction with their physical environment by overlaying dynamic digital content, such as 3D objects, information layers, or media, onto the real world [20]. The main goal of AR is to seamlessly integrate virtual objects with the user's surrounding physical context, facilitating real-time interaction between the virtual and physical realms [21]. The ultimate goal is for users to experience both virtual and real entities as coexisting within the same space, generating a cohesive and interactive environment.

Achieving this level of integration requires accurate spatial registration, a process that ensures that virtual elements are properly aligned with real-world objects in both location and scale. The spatial coherence between the two realities is critical for creating an effective AR experience, where virtual objects respond to changes in the environment and user interaction in real-time.

Various AR devices are employed to deliver these experiences, including AR-Head-Mounted Displays (HMDs), tablets, HHDs, projectors, and see-through VR headsets with built-in cameras. Each device offers different degrees of environmental awareness and interaction capabilities, such as hand tracking and holographic projection.

Virtual Reality

Within the Reality-Virtuality Continuum proposed by Milgram and Kishino, VR occupies the extreme end of the spectrum, representing a complete substitution of a user's perception of the physical world with a fully immersive synthetic environment. In this stage, the user is entirely isolated from their real surroundings, perceiving only the artificially constructed virtual environment, which is typically presented through a range of immersive devices such as HMDs [19].

Modern VR systems achieve this full immersion by leveraging advanced HMDs, such as the Oculus Quest 2, which present stereoscopic images directly to the user's eyes through built-in displays or projection systems. These devices often incorporate additional features like head tracking, which enables the user's head movements to influence their viewpoint in the virtual environment, further enhancing the sense of immersion and presence [22]. Some systems also include positional tracking through external sensors or inside-out tracking via integrated cameras, allowing users to physically navigate virtual spaces, augmenting both interaction fidelity and spatial awareness.

VR is particularly effective in applications that require the user to be completely enveloped in an artificial environment, thus enabling the simulation of real-world scenarios, historical reconstructions, or entirely imaginative worlds. This sense of "presence," wherein users perceive the virtual environment as real, is fundamental to VR's efficacy across various domains, including gaming, education, training, and simulation. Additionally, VR potential to create deeply immersive and isolated experiences makes it especially valuable in fields like remote collaboration, where users can interact with simulated environments or models that are otherwise inaccessible [23].

Mixed Reality

MR continues to elude a universally accepted definition, with interpretations diverging significantly across academic and industrial domains. According to the Milgram and Kishino Reality-Virtuality Continuum, depicted in Figure 2.5, MR occupies a transitional space between AR and AV, bridging the two concepts [19]. Expanding on this, Microsoft's MR spectrum defines MR as spanning a range of technologies, from AR (where physical reality predominates, augmented with digital overlays) to AV (where the virtual environment dominates, supplemented by real-world data) [24].

In MR environments, digital and physical elements coexist and interact in real time, creating a dynamic interface where virtual and physical worlds seamlessly blend. This enables immersive, bi-directional interaction, where users engage with both digital objects and real-world elements, facilitating fluid communication between virtual entities and the physical environment. This integration enhances the user experience by enabling virtual objects to influence, and be influenced by, real-world contexts in a highly interactive manner, enabling new forms of collaboration, visualization, and interaction.

The inherent complexity of MR arises from the challenge of ensuring a natural and intuitive integration of digital and physical elements. This requires advanced environmental sensing, real-time data fusion, and contextual understanding to deliver interactions that appear natural to the user.

Despite considerable advancements, the definition of MR remains contested. Speicher, Hall, and Nebeling (2019) identified six competing notions of MR across both scholarly literature and industry practice, underscoring the fragmentation in its interpretation. Some experts argue that MR represents an enhanced form of AR, where users are not merely passive observers but active participants interacting with a responsive augmented space. In this interpretation, MR is seen as a "stronger" form of AR, exemplified by technologies such as Microsoft's HoloLens, where users can manipulate virtual elements within their physical environment [22].

Other perspectives view MR as a convergence of AR and VR, where the boundary between the real and virtual worlds is fluid and adaptable, creating immersive, hybrid experiences. For example, the widely known game Pokémon Go is sometimes cited as an MR application, where a VR-based digital environment enables users to interact with augmented digital elements such as Pokémons, overlaid onto the physical world, as depicted in Figure 2.6 [22].

However, the definition most relevant to this project's development emphasizes MR as a medium for collaboration, enabling users to interact across different realities—whether physical, augmented, or virtual. In this context, MR supports shared experiences between users situated in distinct environments. For example, a physical space visualized by an on-site AR user can be simultaneously recreated and experienced by a remote VR user, allowing for real-time, collaborative interactions between participants in different realities. This collaborative dimension of MR is central to the system proposed in this research, which adopts this definition as the foundational framework for enabling seamless interaction and cooperation between users in distinct digital and physical environments [22].

Figure 2.6: Pokémon GO - an example of MR application

2.5 Digital Twins

DTs are sophisticated digital replicas of physical entities, allowing for the simulation, analysis, and control of systems within a digital framework. These digital counterparts have emerged as pivotal technologies in a variety of domains, particularly in enhancing HRC, as they offer real-time, interactive environments that mirror physical systems. The ability to replicate physical entities with high fidelity enables improved decision-making, operational efficiency, and flexibility across a wide range of industrial applications.

This DT concept was first introduced by NASA in the 1980s as part of its spacecraft monitoring systems, where virtual models were employed to replicate the conditions and behavior of spacecraft during missions. Over the past decades, advances in IOT, sensor technology, and computational power have significantly improved the capabilities of DTs, moving beyond their original use case. Modern systems leverage real-time sensor data and advanced simulation techniques to enhance the accuracy and reliability of digital models, enabling more sophisticated predictions, real-time analytics, and simulations of complex systems [20].

In nowadays manufacturing and industry scenarios, DTs play a transformative role, particularly in smart manufacturing systems, where they enable detailed examination and prediction of the behavior of physical systems. This capability allows companies to optimize operations, reduce downtime, and improve overall system efficiency. Furthermore, in HRC, DTs facilitate safer and more productive work environments by dynamically adjusting robotic movements and operations to better align with human needs, thereby enhancing ergonomic interactions and mitigating safety risks [25].

A prominent real-world implementation of this technology is seen in Singapore's Smart Nation Initiative, where the Land Transport Authority employs a DT to simulate and evaluate potential policy decisions before their implementation. This application exemplifies the wide-ranging potential of DTs to support decision-making processes in urban planning, infrastructure management, and beyond [26]. As these technologies evolve, their applications in both academic research and industrial practice are rapidly expanding.

The academic landscape surrounding DTs has seen extensive research exploring their versatility and potential, having been applied to a broad spectrum of areas, illustrating the profound impact of DTs on enhancing system efficiency, predictive maintenance, and overall operational performance [26–30].

However, despite the growing prominence of DTs, there is still no universally accepted formal definition of the concept. The majority of scholars and industry experts agree that a DT constitutes a Cyber-Physical System (CPS) that consists of at least three essential components:

- A physical system,
- A virtual model,
- Bidirectional communication between the physical and virtual models [25, 31, 32].

This interaction is fundamental to the operation of true DTs, allowing for a continuous feedback loop where changes in the physical world can inform the virtual model, and, in turn, decisions or optimizations made in the digital realm can directly influence the physical system.

In contrast to true DTs, some critics argue that many commercially available implementations, such as those provided by companies like Siemens, represent "digital shadows" rather than full DTs. The distinction lies in the capability for bidirectional communication. In many digital shadow systems, changes in the physical system are reflected in the virtual model, but there is no capacity for the virtual model to directly control or alter the physical system. This one-way communication limits the interactive and predictive capabilities that define a true DT [33].

To illustrate the distinction, Figure 2.7 presents a reference model of a DT, showcasing the bidirectional flow of information between physical and digital entities. This structure is essential for enabling real-time interaction and feedback, core characteristics that differentiate DTs from digital shadows. True DTs must facilitate a continuous, reciprocal exchange of data between physical and virtual domains, allowing the virtual model to reflect and affect the physical system [34].

Bidirectional communication's importance in DTs is further emphasized by Liu et al. [35], who argue that "a true DT must include bidirectional communication instead of having a virtual model that only updates according to a physical system." This distinction between the types of communication and interaction is critical, as it defines the extent to which a DT can be leveraged for control, simulation, and predictive purposes.

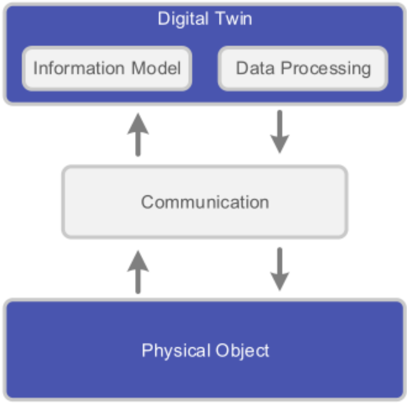


Figure 2.7: A Digital Twin reference model, emphasizing the importance of bidirectional communication between the physical object and its digital counterpart [34]

Table 2.1 outlines different interaction levels within DT systems, ranging from no interaction, where the physical and virtual systems are disconnected, to unidirectional data flow, where data is transferred from the physical to the virtual model, and culminating in bidirectional communication. The latter represents a fully functional DT, where continuous exchange of data allows the virtual model to influence and control the physical system, achieving real-time synchronization and interaction.

Table 2.1: Levels of Interaction in a Digital Twin system between the physical model and its digital counterpart, adapted from [35]

Level of Interaction	Description
No interaction	Virtual model and physical system are not connected through a network. The virtual model only simulates and models a physical system without any real-time updates.
Unidirectional	The physical system feeds sensor data to the virtual model through a network. The virtual model utilizes data to update the current state and predict future states.
Bidirectional	Both the physical system and the virtual model can send data to each other. The virtual model updates using physical data while the physical system can be controlled through data sent by the virtual model.

2.6 Human-Robot Collaboration in Industrial Applications

Following the detailed exploration of DT and MR, this section discusses how these technologies integrate into HRC, demonstrating significant improvements in interaction, safety and efficacy in practical examples from industry solutions with specific emphasis on remote collaboration.

2.6.1 Enhancing Human-Robot Collaboration through Augmented Reality and Digital Twin Implementation

Augmented Reality for Enhanced Interaction and Safety

Chu et al. [36] reviewed various studies on integrating AR with DT to improve HRC using visual and haptic feedback interfaces. Their work emphasizes the value of AR in enhancing human-robot communication by providing both egocentric (shared remote views) and exocentric (spatial visualization of the robot relative to the workspace) perspectives, thereby improving spatial awareness and interaction quality.

Green et al. [37] further highlight that multimodal AR interfaces—incorporating visual, haptic, and acoustic cues—can significantly enhance HRC. Multimodal approaches help overcome challenges like limited Field of View (FOV) in HMDs, making the interaction more intuitive. For instance, audio-tactile feedback is particularly beneficial for individuals with visual impairments, providing alternative sensory channels without compromising performance. Visual AR cues, implemented through HMDs, help users navigate complex environments by overlaying relevant information, thus enhancing navigation without impeding robotic movement.

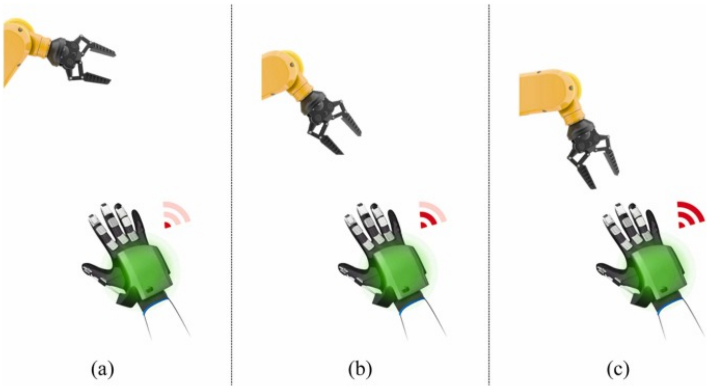
Human-Aware Motion Planning and Safety Improvements

Lasota et al. [38] conducted experiments to evaluate the impact of human-aware motion planning on HRC, demonstrating substantial improvements in task performance and team fluency. Compared to standard robotic systems, participants working with human-aware robots completed tasks more efficiently, exhibited greater concurrent motion, and experienced less idle time for both human and robot. Moreover, they maintained greater separation distances, which reduced collision risks and increased perceived safety. These results illustrate the dual advantage of human-aware planning: it not only enhances task efficiency but also elevates worker comfort and safety, which are critical for minimizing stress-related risks in industrial environments.

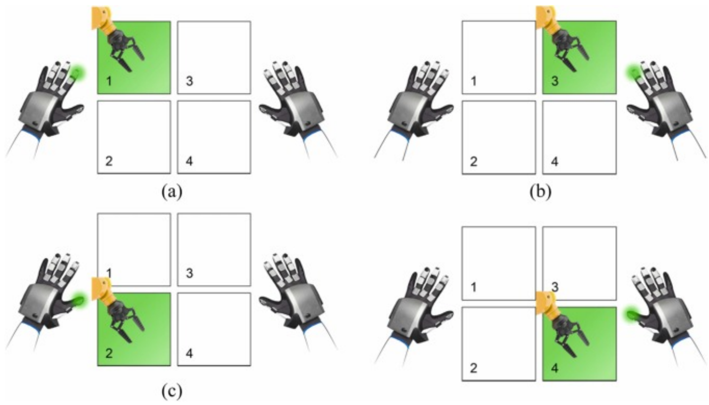
Implementation Techniques

The study utilized the Robot Operating System (ROS) to control a WidowX 250 Robot Arm, using the MoveIt framework for motion planning. This approach ensures modularity, adaptability, and ease of management in shared HRC environments. Both AR and DT models were developed using the Microsoft HoloLens 2 for visual feedback and the SenseGlove Nova™ for haptic feedback, offering a comprehensive multimodal experience.

Through the HoloLens, users could visualize the robot's planned trajectory and swept volume, thus anticipating its actions. Concurrently, the haptic interface provided vibration feedback to signal the robot's proximity and target destinations, as illustrated in Figure 2.8. These multimodal cues offered varying levels of detail, aiding coordination in tasks that required awareness of robot movements.



(a) Cues indicating the gripper's destination using vibration on different human fingers.



(b) Cues indicating the proximity of the gripper via vibration frequency changes.

Figure 2.8: Visual and haptic interfaces used in the experiment [36].

Benefits of Multimodal Feedback

Findings show that combining visual, haptic, and acoustic cues significantly improve task performance. Visual interfaces, especially those indicating proximity, excelled in usability, while haptic feedback proved invaluable in scenarios where visual input was insufficient or overloaded. Acoustic signals also served as alerts for sudden changes in the robot's motion, helping reduce operator anxiety in unpredictable environments.

In a task where the operator and robot worked independently but in close proximity, the system enabled efficient coordination. The robot delivered materials while the operator performed assembly tasks, highlighting the potential for AR-based interfaces to optimize HRC in industrial settings.

2.6.2 Augmented Reality-Assisted Multi-Robot Systems for Enhanced Control and Coordination

Integrating AR into multi-robot manufacturing systems offers significant improvements in interaction, operational safety, and efficiency, especially when applied to real-time and planned control modes.

Ong et al. [39] further explored AR-assisted robot programming for welding applications, demonstrating that user-friendly interfaces can significantly reduce the complexity and duration of the programming process. These interfaces enable operators to define welding points and orientations using handheld pointers, thus enhancing task accuracy and efficiency by allowing validation within the actual robot workspace.

Malí et al. [40] developed an AR application that permits users to adjust robot axis values, visualize specific robot points through 3D arrows, and navigate hidden points using leading lines. Evaluated in an industrial setting, this application showed improvements in usability and interaction capabilities.

Puljiz et al. [41, 42] explored various AR-based methods for robotic arm programming using devices like the Microsoft HoloLens, implementing techniques such as hand-guided task programming, augmented trajectory visualization, and the creation of spatial maps for virtual waypoint placement. These methods facilitate intuitive and accurate robotic arm programming, enabling seamless integration between virtual commands and real-world operations.

Modern manufacturing trends are characterized by a shift toward mass customization and increased flexibility, driven by the demand for individualized products. This necessitates more adaptable manufacturing systems where human operators collaborate with industrial robots to handle complex tasks [43–45]. However, existing robotic systems primarily execute pre-programmed tasks with limited intelligence.

To address this, two promising approaches have emerged: leveraging advanced AI techniques for robot learning [46] and integrating a human-in-the-loop strategy for robot teleoperation. The latter, more aligned with Industry 5.0 principles, extends the capabilities of both humans and robots by incorporating human expertise into collaborative multi-robot processes [47].

Unlike traditional HRC, multi-robot manufacturing with a human in the loop allows operators to interact with robots from remote locations, not limited to physical workspaces. This paradigm facilitates safer and more flexible manufacturing, by bridging the gap between fully automated and manual operations [47]. However, significant challenges remain, including the need for more user-friendly teleoperation interfaces and systems that can be easily utilized by manufacturing operators without extensive robotics training [48].

Research in multi-agent collaborative manufacturing has focused on enhancing safety, productivity, and cost reduction. Wearable AR-assisted systems and DT technologies enable accurate and intuitive robot teleoperation. By combining AR with robot teleoperation, workers can access physical and virtual information simultaneously in a hybrid environment, interacting with virtual objects [49, 50]. An example is an AR-based teleoperation system utilizing RGB-D imaging, allowing operators to perceive the remote robot's environment and perform teleoperation [51]. Another system transforms robot workspaces into AR environments for rapid and intuitive path planning and task programming [52]. These systems improve task performance by providing additional visual cues to enhance the operator's situational awareness.

Recent advancements in smart manufacturing have led to the development of DT models for robot control. For example, [53] used the Unity engine to create a DT of a robot arm that could learn manufacturing tasks virtually and replicate them in the physical world. The integration of DT and VR interfaces has also been proposed to design immersive human-in-the-loop robotic systems, where the DT acts as an intermediary layer for task execution and quality monitoring [54, 55].

Li et al. [56] demonstrated how AR-assisted DTs enable operators to manage and coordinate multiple robots more effectively. Figure 2.9 depicts an immersive dual view where users can interact with the physical setup of two collaborating robots while simultaneously observing their virtual counterparts through Microsoft HoloLens AR glasses. This setup allows for real-time monitoring and simulation of manufacturing processes, improving robot operation control.

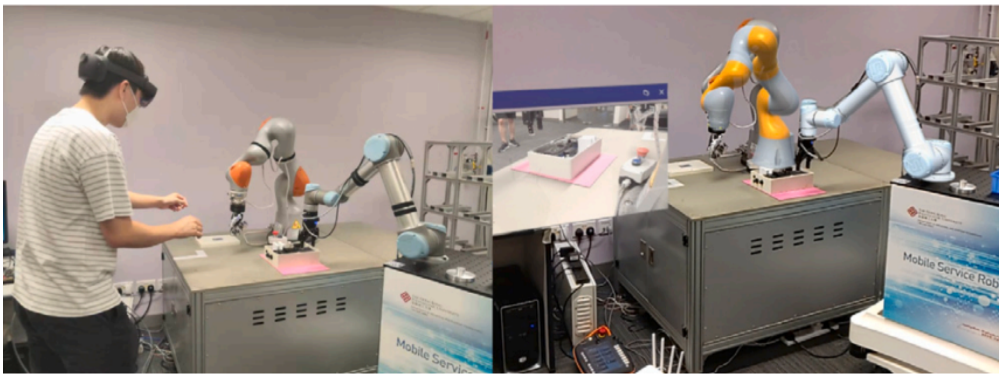


Figure 2.9: AR-assisted DT-enabled multi-robot collaborative manufacturing system [56].

Their proposed comprehensive framework for an AR-assisted, DT-enabled robot collaborative manufacturing system features human-in-the-loop control. The system architecture, shown in Figure 2.10, introduces a multi-node communication mechanism to facilitate interactions among multiple robots and clients. It includes the design of an AR-based teleoperation system for pose registration and motion planning, coupled with three DT-enabled interaction approaches to achieve closed-loop interaction between virtual and physical robots.

The DT of the physical robot, modeled using the Unity engine, is displayed as a hologram in the remote workspace via AR glasses, enabling teleoperation and remote monitoring. Pose registration involves aligning the virtual and physical robot models using the Vuforia Engine, while joint alignment translates DT joint values into real-world coordinates.

Therefore, a robot control approach aided by AR technology offers several benefits:

- Enhanced predictability of robot posture and motion trajectories.
- Trajectory visualization to prevent safety issues.
- An intuitive interface that overcomes spatial and physical limitations.

However, observing workspace and robot state information during task execution presents challenges, such as networking latency and positioning accuracy. Proposed solutions include time-sensitive networks and advanced communication technologies such as 5G [56].

The proposed system also utilizes IP cameras for workspace monitoring, projecting video feeds onto AR glasses for enhanced remote monitoring, as shown in Figure 2.11.

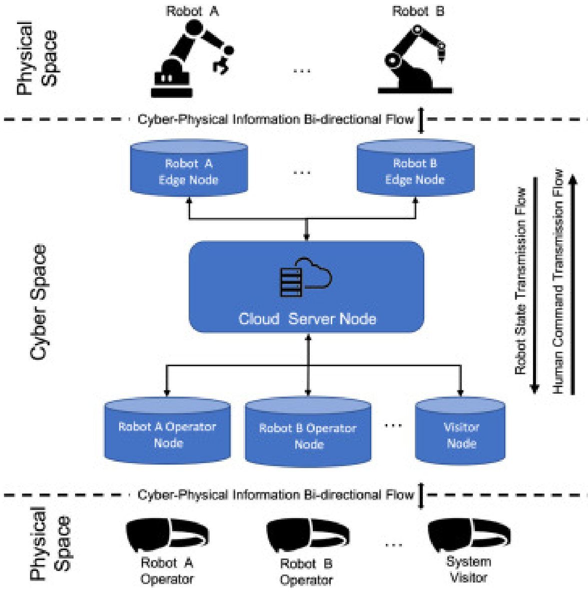


Figure 2.10: The architecture of a multi-robot, multi-client communication mechanism [56].

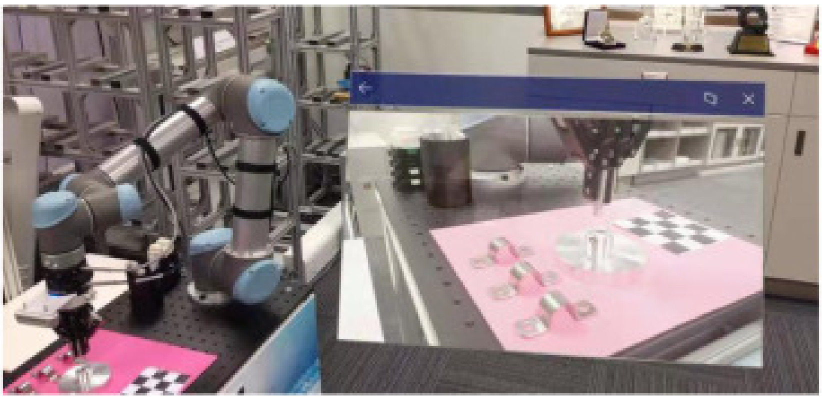


Figure 2.11: Demonstration of the workspace observation approach [56].

2.7 Future Trends in Human-Robot Collaboration

According to the 2019 article *Human–Robot Collaboration in Manufacturing Applications: A Review* [11], future directions in HRC are evolving due to advancements in cobot technologies, sensing methodologies, and algorithmic developments. The key trends identified include:

- **Enhanced Scene Understanding:** Next-generation HRC systems will prioritize deeper contextual awareness of the workspace and tasks at hand. This involves not only detecting the physical environment but also interpreting operator intentions, recognizing task progression, and continuously monitoring environmental dynamics. Such enhanced scene understanding will enable robots to anticipate human actions, predict potential safety risks, and adjust their behavior accordingly, thus fostering a higher level of operational safety and efficiency.
- **Advanced Sensing and Data Fusion:** To facilitate this enhanced scene understanding, advanced sensing methodologies and sophisticated data fusion techniques will be

critical. By integrating multi-modal sensor data—such as visual, tactile, and auditory inputs—robots will be able to construct more comprehensive models of their surroundings and human collaborators. Real-time fusion of such data will allow systems to process information more effectively, ensuring safer interactions by predicting hazardous movements and improving overall system transparency. This, in turn, will enhance user trust and accelerate the adoption of HRC solutions across industries.

- **Integration of Learning Techniques: (does it make sense to leave this, since it is not relevant for the project?)** The incorporation of ML and adaptive learning algorithms into HRC systems represent a transformative leap forward. Techniques such as learning-by-demonstration, and Reinforcement Learning (RL) will enable robots to more accurately mimic human dexterity and decision-making processes, allowing them to learn from human input and adapt to non-repetitive, complex tasks. These adaptive systems will continuously improve based on interaction data, leading to more intuitive and efficient HRC in dynamic industrial environments.
- **Improved Task Planning and Adaptive Learning:** Future HRC systems will be distinguished by advanced task planning capabilities, driven by more sophisticated task modeling and real-time adaptation mechanisms. As robots become more capable of autonomously learning from both structured and unstructured environments, their ability to handle a wider array of tasks will expand, reducing human involvement in routine planning stages. The deployment of these capabilities in manufacturing and service sectors will enable robots to shift between tasks seamlessly, dynamically adjusting their behavior to respond to real-time changes in production or workflow.
- **User-Friendly Interfaces and Interaction Methods:** As the complexity of HRC systems increases, the need for intuitive and accessible human-robot interfaces will become paramount. Developing user interfaces that enable seamless human control without requiring advanced technical expertise is a key area of research. Implementing AR and VR technologies is expected to play a pivotal role in this domain, offering operators immersive and intuitive control mechanisms. These interfaces will reduce cognitive load and enable operators to interact with robots more effectively, thus improving operational efficiency and overall system usability.

Historically, the focus has been on increasing the relevance of HRI by addressing higher safety requirements and enabling robots to perform more complex tasks. Recently, the scope has expanded to include more sophisticated methods aimed at enhancing system performance, applying these methods across different application fields and tackling more intricate tasks. This expansion is driven by the emergence of new cobots, advancements in sensing technologies, matured algorithms, and accumulated experience in designing collaborative workcells [11].

2.8 Summary

Even though significant advancements in AR-DT implementations over the past, the state-of-the-art literature predominantly focuses on developing applications for on-site personnel. Remote collaboration, particularly in HRC scenarios, has received comparatively less attention, highlighting the need for systems that effectively facilitate both on-site and remote collaboration.

Therefore, the proposed project aims to facilitate and integrate better remote collaboration by proposing a generalized conceptual system applicable across various application scenarios.

Despite having developed on-site features, such as digital twin pose registration alongside audio and visual cues, focus will be mainly on the remote collaboration part implementation and the billateral communication between users.

Unity 3D engine will be further explored for robot model development, ROS will be used for robot control, and Vuforia for pose registration. It will also incorporate visual and audio cues to enhance user safety and awareness, MR elements also implemented with Unity 3D. A camera will enable workspace monitoring.

In conclusion, the proposed system will enable remote users to manipulate the robot using Handheld Device (HHD), with the robot's real-time position displayed in the Unity DT. By addressing these challenges, the project aims to enhance remote collaboration in HRC scenarios, contributing to the broader field of MR-DT applications.

introduction to implementation tools, unity game engine, vuforia for pose registration, urdf importer for unity, robot operating system, aruco marker for robot alignment, networking and protocols, conclusion

The chapter will explore each tool in detail, providing a thorough explanation of its selection, its integration within the project, and its contribution to the broader MR-based HRC framework.

CHAPTER 3

Implementation Tools

Achieving the successful integration of Human-Robot Collaboration (HRC) and Mixed Reality (MR) technologies requires a careful selection of advanced tools, spanning both software and hardware domains. The goal of this dissertation is to construct a robust MR-based DT framework that allows real-time, remote collaboration and bi-directional robot control. The implementation of such a system introduces several key challenges, including the alignment of physical and digital entities, real-time data communication, and the creation of immersive, user-friendly interfaces. This chapter systematically introduces the key implementation tools used throughout the development of the project, explaining their roles in addressing these challenges. The tools discussed in this chapter were chosen for their specific contributions to the project's overall goal.

3.1 UR10e Robot

In order to start adressing the aforementioned challenges, a first effort has been made. A robotic arm from Universal Robots, UR10e, shown in the figure 3.1, available at IRIS LAB.

The UR10e model is one of Universal Robots' most advanced cobots, featuring a payload capacity of 12.5 kg, and a reach of 1300 mm, being designed to automate a wide range of tasks that typically require human input, such as assembly, packaging, and pick-and-place operations [1].

UR10e's integrated force sensors and collision detection technologies allow to collaborate safely with humans in shared workspaces, making it ideal for HRC scenarios. Besides, it offers significant flexibility in terms of programming and adaptability. Its console is intuitive and allows imported pre-programmed scripts, therefore being easily deployed across various industrial tasks with minimal programming experience required by the operator.

This robot, as the core physical entity in this human-robot collaborative system, serves as the dynamic agent for performing collaborative tasks, where its physical attributes were mirrored in an immersive digital environment, regarding the fundamental DT concept.

[1] UR10e `https://www.universal-robots.com/products/ur10-robot/` Accessed: 2024-10-15

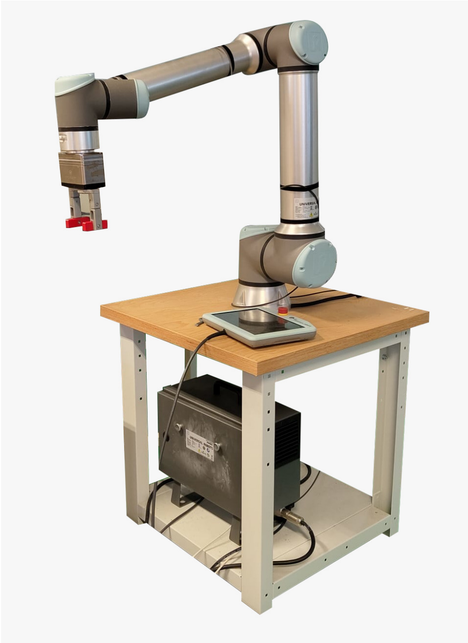


Figure 3.1: UR10e Robot used in the IRIS-LAB, University of Aveiro

3.2 Simulation Environment

3.2.1 Unity

Unity, developed by Unity Technologies, was selected as the primary platform for developing the MR environment in this project. Originally designed for game development, Unity has evolved into a powerful tool for creating interactive 3D applications, including AR, VR, and MR. Its robust architecture and versatility in rendering complex virtual environments make it an ideal choice for building a dynamic DT of the UR10e robotic arm. This ability to seamlessly integrate external data sources, such as sensor inputs from real-world hardware, enables a high degree of interactivity and realism in the simulation.

Unity's (Integrated Development Environment (IDE)) allows for rapid prototyping and iterative design of both the virtual space and the MR user interface. Moreover, the engine's cross-platform compatibility supports a range of devices, including desktop systems and mobile platforms, and can handle real-time rendering of high-fidelity 3D models, essential for MR applications. Through the Unity-based simulation, operators can not only control the physical UR10e robot remotely but also visualize real-world tasks in a digital, augmented environment, ensuring accurate synchronization between physical and virtual realms.

In addition, Unity's robust asset management and scripting support, primarily through C#, provide developers with the tools to easily simulate complex environments, manage interactive objects, and implement advanced functionality such as collision detection and user input handling. These features enable a realistic and immersive experience for both on-site and remote users, improving the overall effectiveness of the HRC system.

3.3 Digital Model Implementation of the Robot

3.3.1 Digital Robot Model - URDF Importer Package

In order to correctly mirror the UR10e physical robot model and establish a functional DT bidirectional synchronization, the (Unified Robot Description Format (URDF)) model of the robot was imported to Unity simulation environment. Unity Robotics Hub's URDF Importer package was used for this very purpose [2]. This package provided several key advantages during the implementation of the DT for the UR10e robot.

The *Unity URDF Importer* package significantly facilitated the integration of the URDF file, enabling the precise recreation of the UR10e robot's physical structure, including its joints and linkages. This seamless integration supported real-time simulation of the robot's movements and configurations within the **Unity!** (**Unity!**) environment. Additionally, it allowed the visualization and fine-tuning of physics properties to accurately reflect the robot's real-world dynamics. Essential for bilateral communication between virtual and physical robots, the package also streamlined sensor data import from ROS, enhancing the realism of simulations and optimizing development and testing processes.

3.4 Pose Registration

Pose registration is a crucial step in aligning the digital model with the physical robot. In this project, Vuforia, a cutting-edge AR software platform, was integrated with Unity to accomplish this alignment.

3.4.1 Vuforia

Vuforia, a powerful AR platform, offers robust object recognition and tracking solutions, crucial for MR applications. In this project, Vuforia was integrated to facilitate precise alignment of the digital robot model with its physical counterpart by detecting and tracking an ArUco marker placed on the physical robot. This alignment is essential for ensuring accurate representation within the MR environment.

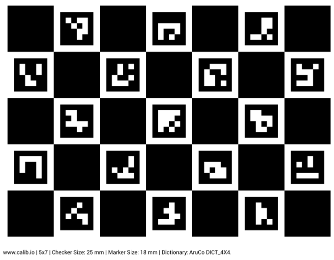
3.4.2 Marker Detection

An ArUco marker, illustrated in the Figure 3.2a, served as a medium to perform accurate pose estimation of the robot within the Unity environment. Besides, the Logitech c922 camera shown in Figure 3.2b scanned this marker, enabling the system to overlay the digital model accurately over the physical robot. This ensured precise positioning and manipulation of the digital twin in the MR space.

In Figure 3.3 the digital robot model can be seen positioned realitve to the above described marker.

* TODO: Add a figure showing the digital UR10 model aligned with the real ArUco marker and the real robot in the background. This will visually represent the alignment process.

[2] Unity Robotics Hub `https://github.com/Unity-Technologies/Unity-Robotics-Hub` Accessed: 2024-02-02



(a) ArUco marker for alignment of the digital twin

(b) Logitech c922 camera used for marker detection

Figure 3.2: Marker and camera setup for digital and physical robot alignment

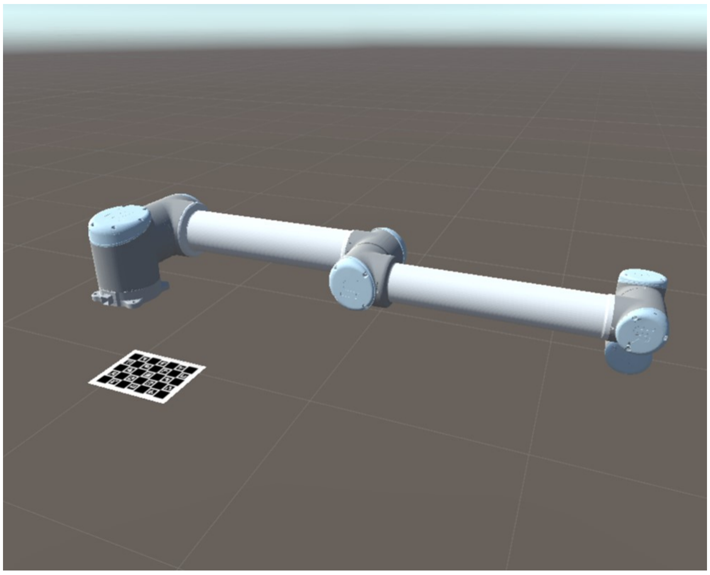


Figure 3.3: Digital UR10 model aligned with ArUco marker in Unity

3.5 Bidirectional Communication

After having the digital model correctly aligned with the physical robot, the next step was to establish bidirectional communication between the Unity and the UR10e. This communication was essential for enabling remote control of the physical robot through the DT, as well as for synchronizing the robot's state between the real and virtual environments.

3.5.1 ROS

The Robot Operating System (ROS) was chosen as the middleware for facilitating real-time communication between the physical robot and the Unity environment, since the UR10e robot from IRIS-LAB already had prior developed ROS packages, such as `iris_ur10e` [3] and `iris_sami` [4].

[3] IRIS-LAB github Repository `https://github.com/iris-ua/iris_ur10e`, Acessed: 2024-02-02

[4] IRIS-LAB github Repository for UR10e Robot Manipulation `https://github.com/iris-ua/iris_ur10e`, Acessed: 2024-02-02

These packages provided a comprehensive ROS setup for controlling the UR10e robot, including trajectory planning, RViz visualization, and real-time operation. ROS offers a robust and flexible framework for developing complex robotic systems. In this project, it enables seamless integration between hardware components and software modules. Specifically, ROS facilitates the exchange of sensor data, control commands, and state information between the physical UR10e robot and its digital twin in Unity, ensuring synchronization for remote control and collaborative operations.

CHAPTER 4

Mixed Reality for Human-Robot Collaboration

4.1 Framework

Figure 4.1 illustrates the proposed framework of the MR-based HRC system. This framework seamlessly integrates two distinct environments—on-site and remote—through a robust communication pipeline designed to enable real-time, collaborative robot manipulation.

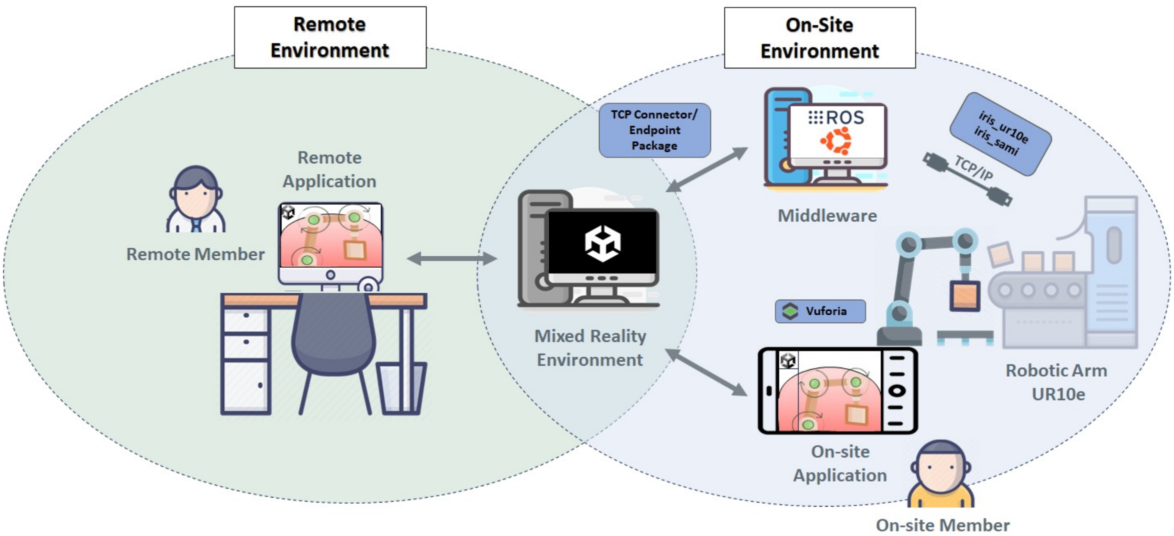


Figure 4.1: Overview of the proposed MR-based HRC system framework integrating remote and on-site environments.

In the **On-Site Environment**, the UR10e robotic arm operates as the central physical entity to be controlled and manipulated. The on-site member interacts with this robotic arm through a custom MR application developed in Unity, enabling intuitive real-time robot manipulation. Three distinct control methods were developed within the application to facilitate manipulation, which will be explained in subsequent sections.

Pose registration between the physical robot and its DT is executed using Vuforia's capabilities, integrated within Unity. The system employs ArUco markers for precise pose estimation, ensuring that the digital representation of the UR10e is accurately aligned with its physical counterpart. The DT of the robot, rendered in Unity, provides a visually synchronized, real-time mirror of the robot's movements and configurations, thus facilitating enhanced interaction.

The robot is connected via Ethernet to a laptop running Ubuntu 20.04 with ROS Noetic, which serves as the middleware layer. This middleware enables seamless data exchange, including robot's data and control commands, between the Unity DT and the physical robot. The `iris_ur10e` and `iris_sami` ROS packages, developed by the IRIS Lab and available on GitHub[1], provide a pre-established ROS environment that supports key functionalities such as trajectory planning and robotic manipulation and visualization through RViz. These packages were further enhanced, enabling bidirectional communication between the MR environment and the UR10e.

Communication between the ROS middleware and Unity's MR environment is established using Unity's ROS-TCP-Connector and ROS-TCP-Endpoint packages. These packages enable bidirectional communication over a Transmission Control Protocol/Internet Protocol (TCP/IP) protocol, ensuring real-time synchronization between the virtual and physical environments. This communication architecture is fundamental for maintaining the DT's fidelity, reflecting real-world changes in the Unity model and vice versa.

Regarding the **Remote Environment**, a remote participant accesses the same Unity-based application. This allows the remote member to visualize and manipulate the robot in real time, from a separate location. The remote UI provides real-time visualization of the robot's state and its workspace, enabling remote collaboration. The synchronization between the remote and on-site environments is facilitated through Unity's mixed-reality capabilities, which, in conjunction with the ROS-based control, enable the remote user to execute commands and receive real-time feedback.

The Middleware layer, acting as the system's backbone, ensures the continuous synchronization of data between the physical robot and its DT. It manages the real-time feedback loop, maintaining bidirectional data flow between the virtual robot in Unity and the physical robot in the on-site environment. This configuration guarantees that any actions performed by either the on-site or remote user are consistently reflected in both the physical and digital realms, preserving operational coherence and maximizing collaborative efficiency.

This framework provides an immersive and responsive mixed-reality environment, bridging the gap between physical and digital spaces. The system enables real-time robot manipulation and monitoring from both on-site and remote locations, making it a versatile platform for collaborative tasks in advanced industrial applications. The seamless integration of MR, DT, and HRC technologies significantly enhances user interaction, safety, and productivity, while offering an intuitive interface for remote and on-site collaboration.

[1] https://github.com/iris-ua

4.2

After having properly implemented the bidirectional communication between both environments, the on-site and the remote, allowing both members to interact with the robot, the next step was to complement the already built application as well as its interface.

Afterwards, the framework required to integrate the MR technologies with the robotic arm was thoroughly discussed with the supervisors, leading to the following components:

- **On-Site Interaction:**
  - Implement an UR10e digital model into the Unity 3D simulation environment
  - Utilize marker detection, utilizing Vuforia, to align the digital model with the physical robot
  - Perform pose registration to ensure accurate spatial alignment between the virtual and physical models
  - Develop a user-friendly interface for robot manipulation using HHDs
  - Implement visual and audio cues for user awareness and accident prevention
- **Remote Visualization and Interaction:**
  - Enable bilateral communication between the robot and the Unity digital twin
  - Provide remote participants with a foundational 2D interface, such as a laptop screen, to visualize the collaboration scenario and task context.
  - Implement real-time updates of the robot's position and workspace visualization
  - Develop the capability for remote operation of the robot via the MR application, enhancing the remote participant's ability to interact and manipulate the collaborative environment.
- **Automation and Immersion:**
  - Integrate a camera into the robot and develop a camera feed transmission to provide real-time updates of the robot's position and workspace visualization
  - Share this information with remote participants, assisting on-site participants by delegating visual sharing to the robot.

4.3 On-site Application Features

Regarding the on-site member's collaboration experience, as explained in the state of art review, by implementing different sensorial cues, such as visual and audio, it will enhance the user experience into a more intuitive and immersive way.

4.3.1 Virtual Safety Zones

The implementation of virtual safety zones is a critical feature designed to enhance on-site member's safety when interacting with the robot.

Two safety zones were developed, as shown in figure 4.2, to address specific safety and user experience concerns:

- **Outer Safety Zone**: Initially, only the outer safety zone was developed. The purpose of creating this zone was to provide an early warning to users as they approach the hazardous area near the robot. This approach consisted on changing its color as a visual alert. However, this method proved ineffective because, once inside it, users could not perceive the color change, rendering the warning system inadequate.
- **Inner Safety Zone**: To overcome the limitations of the outer zone, an additional, inner safety zone was introduced. This design ensures a two-step safety mechanism:

  Visual Alert: Upon entering the outer safety zone, the color of the inner sphere changes to red. This alteration serves as a visual cue, indicating that the user is getting closer to a high-risk area.

  Auditory Warning: Entering the inner safety zone triggers an auditory alarm. This sound alert signifies that the user has breached into the robot working area, enhancing the effectiveness of the safety mechanism.

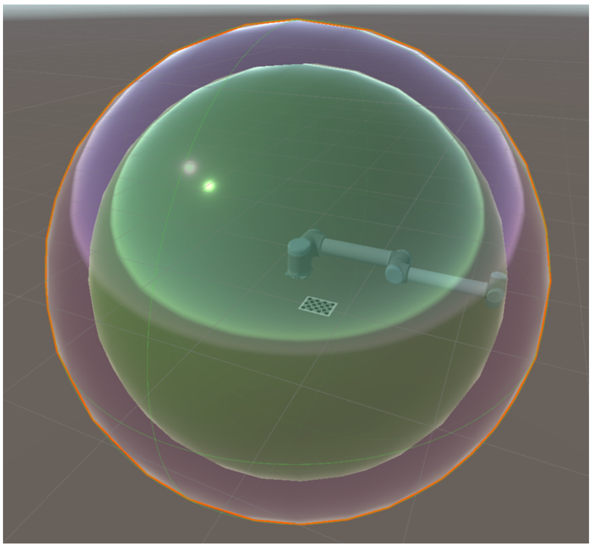


Figure 4.2: Simulated environment showing the display of both the inner and outer safety zones, as well as the robot and the marker inside them

Safety Zone Breach Protocol

A crucial safety feature is that if the on-site member enters the safety zone area while the robot is in motion, the robot automatically stops. This immediate halt ensures that potential accidents or injuries are avoided by preventing any interaction with the robot when a user is within a designated dangerous area.

4.3.2 Interface

The interface, which incorporates the safety-zone features detailed earlier, is illustrated in figure **??** (take another picture with the most recent interface) Within this interface, one can observe the panel for controlling the robot joints. Additionally, there are two green buttons positioned at the top and bottom right corners of the interface, designated for activating the safety-zone and joint movement functions, respectively. This subsequent features will be explained below.

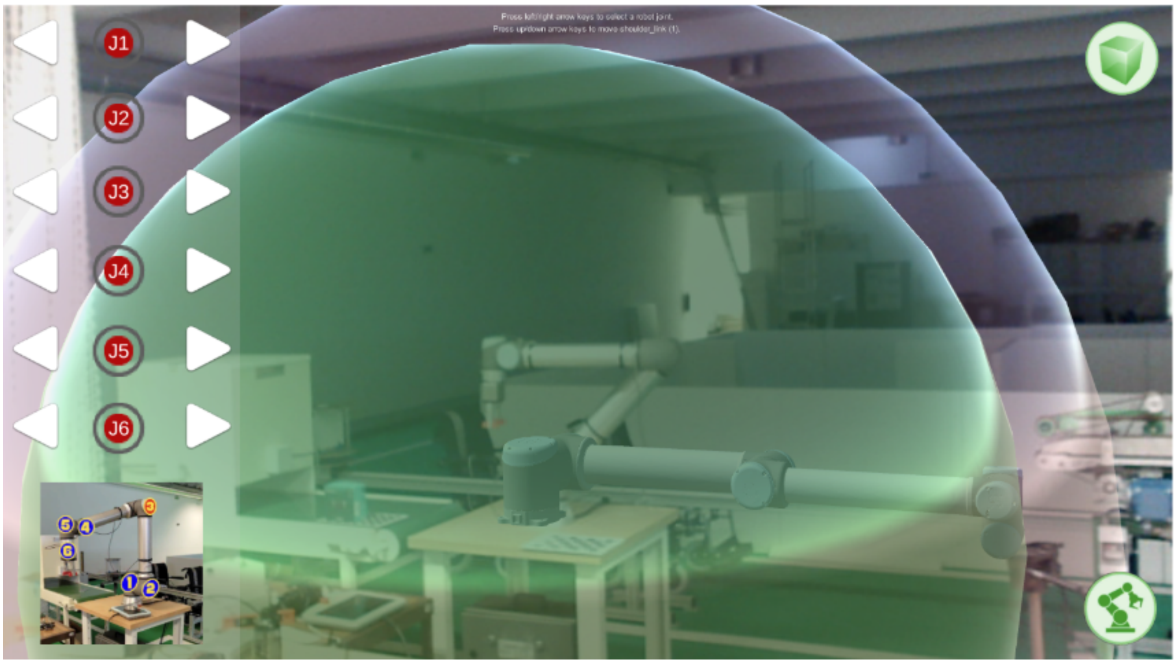


Figure 4.3: Interface featuring developed interactions

Joint Movement

- this is explained also in the previous chapter (3) - includes a pseudo-code snippet

This feature allows users to manipulate the robot's joints via a user-friendly interface, as depicted in the figure 4.4a by the first joint. Its functionality works as follows:

- **Joint Selection**: Users can activate a joint by clicking on it. Upon activation, the selected joint's central red circle turns green.
- **Movement Control**: By clicking on the selected joint's directional arrows within the interface, it moves in either a positive or negative direction. This functionality mimics the real-time movement control similar to using keyboard arrow keys, ensuring intuitive operation.
- **Continuous Movement**: The selected joint continues to move until we deactivate its button.
- **Single Joint Activation**: To ensure precise control, joint movement is only possible when one joint is selected at a time. This prevents unintended actions and enhances the accuracy of adjustments.

Buttons

Two buttons, shown in figure 4.4, were introduced to toggle the activation and deactivation of the joint menu functionalities 4.4b and the safety zones 4.4c. Upon deactivation, the button will turn gray. This design allows users to easily switch these features on or off, providing flexibility in controlling the safety mechanisms and movement operations within the Extended Reality (XR) environment.

(a) Joint Control for first Joint **(b)** Controller Menu Toggle **(c)** Safety-zone Toggle

Figure 4.4: Example from first joint control menu and toggle buttons for controller menu and safety-zones

4.4 Remote Collaboration

After having successfully implemented the robot's digital model into the Unity environment, as well as performed the pose registration,

Building on the foundation of the application that facilitated on-site member interactions, the next critical step was to establish a robust connection to the UR10e robot. This connection was essential for accurately developing a digital twin of the robotic arm, allowing for real-time manipulation and visualization within a Unity environment.

To achieve this, it was necessary to extend the initial project by developing a complementary application focused on the remote manipulation of the robotic arm through Unity.

This bridge facilitated communication between the ROS environment on Ubuntu and the Unity application on Windows, enabling real-time visualization and control of the robot within Unity.

However, this integration posed significant challenges. Despite the potential of the ROS-TCP-Connector, the documentation provided limited guidance on adapting the connection to different robots beyond the examples in the online tutorial. As a result, the development process relied heavily on trial and error, requiring iterative testing and debugging to achieve a functional ROS-Unity connection.

4.5 Robot Manipulation

When it it came to manipulate the digital version of the robot in the Unity environment, it was necessary to understand the Unity Robotics Hub package's way of doing so. A C# script named `Controller.cs`, contained the necessary functions to control the robot's joints.

After further analysis, three control methods were implemented to control the robot's joints:

- **UI Control:** This method allowed the user to control the DT version of the robot by moving each joint individually through an Unity UI. Its purpose consisted on being user-friendly and intuitive manner of controlling the robot, where a panel with a button for each joint was displayed, as shown in figure 4.5.

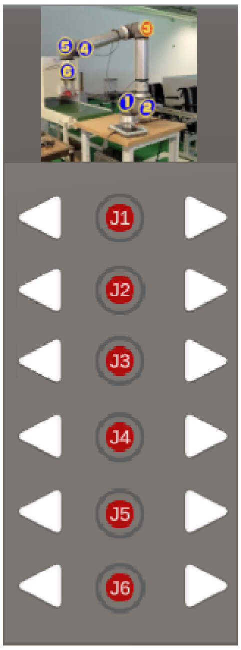


Figure 4.5: UI panel to control the robot's joints individually when using the UI Control method

Apart from the joints' buttons, there is also a figure of the robot displayed in the top part of the panel, showing each joint's relative number, and thus allowing an easier identification of the joint to be controlled.

Upon activating a joint by pressing the desired button, its color turns green instead of the default red button and the user is able to choose between rotating this joint in either the positive or negative direction, also represented by a change in the default color of the corresponding directional arrow. These features are represented in figure — add figures joint being selected as well as the direction of UI control.

- **Unity-ROS Control:** This method enabled the robot to be controled during the runtime simulation in the Unity environment via the keyboard arrows of the laptop, then sending its position into the ROS environment via Wi-fi. In order to change the Unity DT robot state, the user has to press the right/left arrow keyboard keys to select the following/previous joint as well as the up/down keys to rotate the selected joint in the positive/negative direction, respectively.

  After updating the DT robot's state, the user must press the "Publish" button within the user interface, shown in figure A.5. This action publishes the current joint states over Wi-Fi to the ROS environment in a different ROS topic than the real robot joint states are defined.

  Below, a pseudo-code showcasing how to manipulate the robot in Unity-ROS control mode is described.

  In order to handle this communication process, two ROS nodes were created. The `unity_joint_subscriber.py` script was developed within the `iris_ur10e` package. By creating a new node, called `joint_state_listener`, it then subscribes to the `unity_joint_states` topic, expecting to receive a JointState message type. Afterwards,

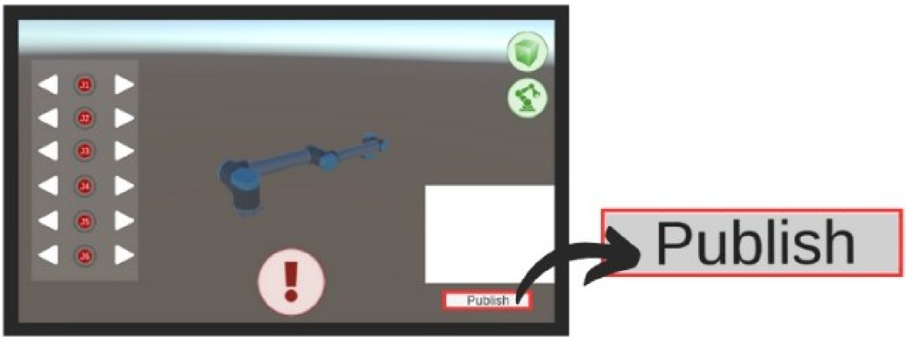


Figure 4.6: Publish button that sends Unity's DT robot joint states into ROS the environment

Algorithm 1 Unity Input for Joint Selection and Movement

```
Step 1: User Input for Joint Selection and Movement in Unity
while Unity Simulation is running AND Unity-ROS Control is selected do
    if RightArrowKeyPressed then
        Select next joint
    else if LeftArrowKeyPressed then
        Select previous joint
    end if
    if UpArrowKeyPressed then
        Rotate selected joint in positive direction
    else if DownArrowKeyPressed then
        Rotate selected joint in negative direction
    end if
    if PublishButtonPressed then
        joint_states = GetCurrentJointStates()
        PublishToROSTopic('unity_joint_states', joint_states)
    end if
end while
```

it initializes a publisher for the `move_joint_unity` topic, that converts this data into a Float64MultiArray format that will be further received by the second node.

Regarding the second node, the `move_unity.py` script was created in the `iris_sami` package. It initializes the `test_arm_movement` node that listens for joint position commands on the `/move_joint_unity` topic. Upon receiving this data, it moves the robotic arm to the desired position.

This robot position update can be performed either on the simulation environment, where it can be visualized through Rviz, or by utilizing the real UR10e robot.

A pseudo-code explanation for the ROS nodes is presented in algorithm 2. (add a picture of the framework exchange between Unity and ROS and a pseudo code of the ROS nodes created)

Algorithm 2 Combined ROS Node for Receiving Unity Joint States and Moving the Robot

```
Step 2 and 3: Combined ROS Node for Receiving Unity Joint States and Moving the Robot
Initialize ROS Node: joint_state_listener
Subscribe to Topic: 'unity_joint_states'
while Receiving JointState message from Unity do
    float_array_data = ConvertToFloat64MultiArray(joint_states)
    PublishToROSTopic('move_joint_unity', float_array_data)
end while
Precondition: The joint_state_listener node must be running and publishing to the 'move_joint_unity' topic.
Initialize ROS Node: test_arm_movement
Subscribe to Topic: 'move_joint_unity'
while Receiving Float64MultiArray message from move_joint_unity do
    MoveRobotArmTo(joint_positions)
    if ConnectedToRealRobot then
        MoveRealRobot()
    else
        VisualizeInRviz()
    end if
end while
```

- **ROS-Unity Control:** This method allowed the robot to be controlled either in simulation in ROS environment with the Rviz interface, or manually in the robot itself and this would be reflected on the DT robot model displayed in the Unity environment.

  Upon selecting this method, the DT robot model would be updated in real-time according to the data that is being published into the `joint_states` ROS topic.

  A new script was created in Unity, called `JointStateSubscriber.cs`, that subscribes to the `joint_states` topic, and stores the information regarding the joint positions in a dictionary structure that is saved at each frame into a specific `.json` file. This file is then read by the `Controller.cs` script, updating the DT robot model in Unity.

  By maintaining this synchronization between the real robot and the virtual environment, the Unity scene accurately reflects the robot's live state, ensuring a consistent digital twin representation as well as enabling the bidirectional communication and robot manipulation.

  Below, there is another pseudo-code that explains how the ROS-Unity control method works.

Algorithm 3 ROS-Unity Control via Joint States Subscription

Step 1: Subscribe to ROS `joint_states` topic
Attach the Unity Script to the Digital Robot Model Asset: `JointStateSubscriber.cs`
Upon Initialization, it subscribes to topic: `/joint_states`
while Receiving JointState message from ROS **do**
    Extract joint names and positions from the message
    Store joint positions in a dictionary structure
    Save the joint positions to a `jointStateSubscriber.json` file
end while
Step 2: Update Unity DT Robot Model
while Simulation is Running **do**
    Read the `jointStateSubscriber.json` file
    Update the Unity DT robot model using the joint positions from the file
end while
Step 3: Synchronize Real Robot with DT Robot
The Unity DT robot model moves according to the real robot's joint positions, ensuring a consistent Digital Twin representation.

Integration Highlights

These two nodes addressed key aspects of system performance:

- **Synchronization:** Ensures that changes in Unity's control environment are accurately and timely reflected in the robot's physical movements.
- **Modularity:** Separates data handling and robot control into different nodes to improve system reliability and ease of maintenance.

do below here - add photos or video that mimic the process, same images

4.6 Camera Feed Transmission

In order to enhance the remote participant's understanding of the on-site environment and task performance, I integrated a live camera feed from a camera that was attached to the robot. This feed was then displayed in the Unity application, allowing the remote user to observe the robot's environment in real-time, providing critical visual feedback necessary for effective remote collaboration.

4.6.1 Hardware and Software Setup

An Orbbec Astra camera, displayed in the figure 4.7, was provided by the project supervisors was used to capture the live video feed. This 3D camera was chosen for its high-quality video output and compatibility with the ROS environment, enabling seamless integration into the Unity application.

Figure 4.7: Astra 3D Orbbec Camera used to transmit real-time video feed from robot environment to remote user

To integrate the camera into the ROS environment, an existing GitHub repository [2] tailored for the Astra camera integration was used. This repository contained the necessary drivers and ROS nodes to enable the camera's functionality within the ROS framework.

4.6.2 ROS Camera Node

To initiate the camera feed, the `astra_camera_node` from the `astra_camera` package is initialized. Afterwards, an RVIZ image viewer was used to visualize the live feed, ensuring the camera was functioning correctly and capturing the desired video data.

4.6.3 Unity Camera Feed Integration

From the Unity side, the `CameraFeedReceiver.cs` (verify the name of the script) script was developed to receive and display the live camera feed. This script was then atached to an UI interface (confirm the name of the element) that displayed the video feed in real-time. add a figure of the UI interface with the view of the camera - The figure (add figure) illustrates the camera feed in the Unity application, showcasing the live video stream from the robot's environment.

4.6.4 Data Transmission to Unity

However, the raw image data generated by the camera was too heavy to be transmitted efficiently over Wi-Fi, so this data needed to be republished using the `image_transport` package. By utilizing the following ROS command

```
rosrun image_transport republish raw
in:=/camera/color/image_raw out:=/camera/image_repub
```

[2] Github Repository used to integrate Astra Orbbec Camera in the ROS environment `https://github.com/orbbec/ros_astra_camera` Accessed: 2024-10-04

the image data was republished in a more efficient format, allowing for smoother and real-time transmission to the Unity environment.

CHAPTER 5

Discussion and Evaluation

Issues found during the development:

The following developed features, for enhancing the on-site environment, were tested while using a laptop with the previously described camera.

Despite many attempts, trying to build the MR application into android handheld devices was not effective. Since the digital version of the robot showed physics limitations when entering on the simulation environment.

However, due to the complexity of importing the UR10e model directly, a closely related UR10 model was used from an open-source repository [1].

[1]PositronicsLab `https://github.com/PositronicsLab/reveal_packages/tree/master/industrial_arm/scenario/models/urdf/ur10` Accessed: 2024-02-05

CHAPTER 6

Conclusion and Future Work

6.1 Conclusion

This dissertation has explored the development of an Mixed-Reality-assisted, Digital Twin-enabled robot collaborative system with human-in-the-loop control. The primary objective consisted on enhancing Human-Robot Collaboration by integrating advanced technologies that bridge the gap between physical and virtual environments supporting remote collaboration and, thereby, fostering more efficient and intuitive interactions in manufacturing settings.

On-Site interaction was a crucial aspect of the project. By utilizing Handheld Devices such as tablets and smartphones, on-site participants were able to share live views of their surroundings. Furthermore, augmented reality elements were layed upon the mixed-reality application to provide visual and audio cues to the operator and enhance the robot's movements awareness.

In terms of **remote visualization and interaction**, a foundational 2D interface was accessible via standard devices like laptops. This interface enabled remote participants to visualize the collaboration scenario and understand the task context effectively. Furthermore, the system's capabilities allow remote operation of the robot through the MR application, given that bi-directional communication was implemented. This enhancement empowered remote users to interact with and manipulate the collaborative environment, bringing them closer to the on-site experience and improving overall collaboration efficacy.

Regarding **automation and immersion**, a camera was mounted on the robot to automate the process of environment sharing with remote participants. This feature relieved on-site participants from the responsibility of manually sharing visual information, as the robot could now autonomously provide live feeds of the workspace. This automation not only improved efficiency but also enhanced the immersive experience for remote users by offering real-time visual insights into the operational environment.

Throughout the development process, we leveraged the Unity game engine for robot model development and employed the Robot Operating System for seamless communication between the physical robot and its digital twin. The use of Vuforia facilitated precise pose registration, ensuring accurate alignment between virtual and physical models. By integrating both visual and audio cues as well as intuitive controls within the shared environment, we enhanced user awareness of the robot's movements and provided a user-friendly interface for robot manipulation.

In conclusion, the project successfully met its defined objectives by creating a system that enhances Human-Robot Collaboration through advanced Mixed-Reality and Digital Twin technologies, focusing on the remote member capabilities. Integrating these technologies resulted in a more intuitive, efficient, and safe collaborative environment, aligning with the core values of Industry 5.0. This proposed system demonstrates significant potential for improving manufacturing processes by combining human expertise with robotic precision, ultimately contributing to more flexible and human-centric industrial practices.

6.2 Future Work

Despite having achieved its primary goals, there are several areas for future exploration to further enhance the system's capabilities and impact.

Conducting User Experience Research and Evaluation is a vital next step to refine the system based on user feedback and performance metrics. By performing usability studies, we can identify pain points and areas for improvement, ensuring that the system meets the needs of its users effectively. Analyzing task performance data will help optimize workflows and enhance overall efficiency. Future work related to this includes:

- **Enhancing Immersive Technologies**: Exploring the potential of advanced devices like the Microsoft HoloLens 2 can further enhance the immersive experience of remote collaboration. Integrating mixed reality headsets can provide users with more natural and intuitive interactions within the collaborative environment.
- **Improving Communication Tools**: Integrating information-sharing tools such as voice communication and real-time annotations will facilitate more effective collaboration between on-site and remote participants. This enhancement can lead to better understanding, quicker decision-making, and a more cohesive teamwork experience.
- **System Performance Optimization**: Addressing challenges related to network latency and positioning accuracy will improve the system's responsiveness and reliability. Implementing advanced communication protocols and optimizing data processing can enhance real-time interactions.
- **Longitudinal Studies**: Assessing the long-term impact of the technology on collaboration efficiency will provide insights into how the system influences productivity over time. This can reveal trends and patterns that inform further enhancements.
- **Ergonomic Assessments**: Ensuring that prolonged use of AR devices does not cause discomfort or health issues is essential for user well-being. Conducting ergonomic studies will help optimize device usage and interface design to promote comfort and reduce fatigue.

In addition to user experience research, future work could focus on:

- **Advanced Interaction Modalities**: Incorporating gesture recognition and voice commands can make the system more accessible and reduce reliance on manual input devices. These modalities can provide a more intuitive control mechanism, especially in environments where traditional input devices are impractical.
- **Security and Privacy Measures**: Strengthening encryption and authentication protocols will protect user data and ensure secure communication channels. This is crucial for maintaining trust and compliance with data protection regulations.

By pursuing these future developments, the system can significantly improve its effectiveness and user satisfaction. Continuous refinement based on user feedback and technological advancements will contribute to its adoption in various industrial contexts, ultimately enhancing human-robot collaboration and advancing the principles of Industry 5.0.

6.2.1 Final Remarks

This dissertation has laid the groundwork for an innovative approach to human-robot collaboration in manufacturing environments. By integrating AR and DT technologies, we have demonstrated the potential for creating systems that are not only efficient but also human-centric. The fusion of human intuition with robotic capabilities opens new avenues for productivity and innovation.

The journey does not end here. The insights gained and the foundation established through this work pave the way for future explorations that can further bridge the gap between humans and machines. Embracing continuous improvement and adaptation will ensure that such systems remain relevant and impactful in the ever-evolving landscape of industrial automation.

References

[1] A. Weiss, A. K. Wortmeier, and B. Kubicek, «Cobots in Industry 4.0: A Roadmap for Future Practice Studies on Human-Robot Collaboration», *IEEE Transactions on Human-Machine Systems*, vol. 51, pp. 335–345, 4 Aug. 2021, ISSN: 21682305. DOI: 10.1109/THMS.2021.3092684.

[2] D. P. Moller, H. Vakilzadian, and R. E. Haas, «From Industry 4.0 towards Industry 5.0», vol. 2022-May, IEEE Computer Society, 2022, pp. 61–68, ISBN: 9781665480093. DOI: 10.1109/eIT53891.2022.9813831.

[3] I. Ahmed, G. Jeon, and F. Piccialli, «From Artificial Intelligence to Explainable Artificial Intelligence in Industry 4.0: A Survey on What, How, and Where», *IEEE Transactions on Industrial Informatics*, vol. 18, pp. 5031–5042, 8 Aug. 2022, ISSN: 19410050. DOI: 10.1109/TII.2022.3146552.

[4] M. Golovianko, V. Terziyan, V. Branytskyi, and D. Malyk, «Industry 4.0 vs. Industry 5.0: Co-existence, Transition, or a Hybrid», *Procedia Computer Science*, vol. 217, pp. 102–113, 2023, 4th International Conference on Industry 4.0 and Smart Manufacturing, ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2022.12.206. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050922022840.

[5] E. Commission, D.-G. for Research, Innovation, M. Breque, L. De Nul, and A. Petridis, *Industry 5.0 – Towards a sustainable, human-centric and resilient European industry.* Publications Office of the European Union, 2021. DOI: doi/10.2777/308407.

[6] S. Nahavandi, «Industry 5.0—A Human-Centric Solution», *Sustainability*, vol. 11, no. 16, 2019, ISSN: 2071-1050. DOI: 10.3390/su11164371. [Online]. Available: https://www.mdpi.com/2071-1050/11/16/4371.

[7] N. Y. Mulongo, «Industry 5.0 a Novel Technological Concept», in *2024 International Conference on Smart Applications, Communications and Networking (SmartNets)*, 2024, pp. 1–6. DOI: 10.1109/SmartNets61466.2024.10577684.

[8] U. E. Ogenyi, J. Liu, C. Yang, Z. Ju, and H. Liu, «Physical Human-Robot Collaboration: Robotic Systems, Learning Methods, Collaborative Strategies, Sensors, and Actuators», *IEEE Transactions on Cybernetics*, vol. 51, pp. 1888–1901, 4 Apr. 2021, ISSN: 21682275. DOI: 10.1109/TCYB.2019.2947532.

[9] R. Jahanmahin, S. Masoud, J. Rickli, and A. Djuric, *Human-robot interactions in manufacturing: A survey of human behavior modeling*, Dec. 2022. DOI: 10.1016/j.rcim.2022.102404.

[10] A. A. Malik and A. Brem, «Digital twins for collaborative robots: A case study in human-robot interaction», *Robotics and Computer-Integrated Manufacturing*, vol. 68, p. 102 092, 2021, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2020.102092. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584520303021.

[11] E. Matheson, R. Minto, E. G. G. Zampieri, M. Faccio, and G. Rosati, «Human–Robot Collaboration in Manufacturing Applications: A Review», *Robotics*, vol. 8, no. 4, 2019, ISSN: 2218-6581. DOI: 10.3390/robotics8040100. [Online]. Available: https://www.mdpi.com/2218-6581/8/4/100.

[12] L. Barbazza, M. Faccio, F. Oscari, and G. Rosati, «Agility in Assembly Systems: A Comparison Model», *Assembly Automation*, vol. 37, no. 4, pp. 411–421, 2017. DOI: 10.1108/AA-10-2016-128. [Online]. Available: https://doi.org/10.1108/AA-10-2016-128.

[13] K. R. Guerin, C. Lea, C. Paxton, and G. D. Hager, «A framework for end-user instruction of a robot assistant for manufacturing», in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 6167–6174. DOI: 10.1109/ICRA.2015.7140065.

[14] J. Colgate, W. Wannasuphoprasit, and M. Peshkin, «Cobots: robots for collaboration with human operators», English (US), Proceedings of the 1996 ASME International Mechanical Engineering Congress and Exposition ; Conference date: 17-11-1996 Through 22-11-1996, Dec. 1996, pp. 433–439.

[15] JUGARD+KÜNSTNER GmbH, *UR5e - Collaborative Robot from Universal Robots*, `https://www.jugard-kuenstner.de/universal-robots/cobots/ur5e`, Accessed: 2024-09-23, n.d.

[16] G. Michalos, S. Makris, P. Tsarouchi, T. Guasch, D. Kontovrakis, and G. Chryssolouris, «Design Considerations for Safe Human-robot Collaborative Workplaces», *Procedia CIRP*, vol. 37, pp. 248–253, 2015, CIRPe 2015 - Understanding the life cycle implications of manufacturing, ISSN: 2212-8271. DOI: `https://doi.org/10.1016/j.procir.2015.08.014`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S2212827115008550`.

[17] M. Faccio, M. Bottin, and G. Rosati, «Collaborative and Traditional Robotic Assembly: A Comparison Model», *International Journal of Advanced Manufacturing Technology*, vol. 102, pp. 1355–1372, 2019. DOI: `10.1007/s00170-018-03247-z`. [Online]. Available: `https://doi.org/10.1007/s00170-018-03247-z`.

[18] M. Fechter, P. Foith-Förster, M. S. Pfeiffer, and T. Bauernhansl, «Axiomatic Design Approach for Human-robot Collaboration in Flexibly Linked Assembly Layouts», *Procedia CIRP*, vol. 50, pp. 629–634, 2016, 26th CIRP Design Conference, ISSN: 2212-8271. DOI: `https://doi.org/10.1016/j.procir.2016.04.186`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S2212827116305686`.

[19] P. Ladwig and C. Geiger, «A Literature Review on Collaboration in Mixed Reality», in *Smart Industry & Smart Education*, M. E. Auer and R. Langmann, Eds., Cham: Springer International Publishing, 2019, pp. 591–600, ISBN: 978-3-319-95678-7.

[20] Y. Liu, S. Ong, and A. Nee, «State-of-the-art survey on digital twin implementations», *Advances in Manufacturing*, vol. 10, pp. 1–23, 2022. DOI: `10.1007/s40436-021-00375-w`.

[21] R. T. Azuma, *A Survey of Augmented Reality*, 1997. [Online]. Available: `http://www.cs.unc.edu/~azumaW:`.

[22] M. Speicher, B. D. Hall, and M. Nebeling, «What is Mixed Reality?», in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, ser. CHI '19, Glasgow, Scotland Uk: Association for Computing Machinery, 2019, pp. 1–15, ISBN: 9781450359702. DOI: `10.1145/3290605.3300767`. [Online]. Available: `https://doi.org/10.1145/3290605.3300767`.

[23] D. Chusetthagarn, V. Visoottiviseth, and J. Haga, «A Prototype of Collaborative Augment Reality Environment for HoloLens», in *2018 22nd International Computer Science and Engineering Conference (ICSEC)*, 2018, pp. 1–5. DOI: `10.1109/ICSEC.2018.8712803`.

[24] Microsoft, *Mixed Reality*, `https://learn.microsoft.com/en-us/windows/mixed-reality/discover/mixed-reality`, Accessed: September 16, 2024, 2024.

[25] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, «Digital Twin in Industry: State-of-the-Art», *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2405–2415, 2019. DOI: `10.1109/TII.2018.2873186`.

[26] K. H. Soon and V. H. S. Khoo, «CITYGML MODELLING FOR SINGAPORE 3D NATIONAL MAPPING», *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-4/W7, pp. 37–42, 2017. DOI: `10.5194/isprs-archives-XLII-4-W7-37-2017`. [Online]. Available: `https://isprs-archives.copernicus.org/articles/XLII-4-W7/37/2017/`.

[27] W. Luo, T. Hu, W. Zhu, and F. Tao, «Digital twin modeling method for CNC machine tool», in *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, 2018, pp. 1–4. DOI: `10.1109/ICNSC.2018.8361285`.

[28] F. Tao, M. Zhang, Y. Liu, and A. Nee, «Digital twin driven prognostics and health management for complex equipment», *CIRP Annals*, vol. 67, no. 1, pp. 169–172, 2018, ISSN: 0007-8506. DOI: `https://doi.org/10.1016/j.cirp.2018.04.055`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0007850618300799`.

[29] M. N. Mamatha, «Design of Single Patient Care Monitoring System and Robot», in *Cyber-physical Systems and Digital Twins*, M. E. Auer and K. Ram B., Eds., Cham: Springer International Publishing, 2020, pp. 203–216, ISBN: 978-3-030-23162-0.

[30] C. Doukas and I. Maglogiannis, «Bringing IoT and Cloud Computing towards Pervasive Healthcare», in *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, 2012, pp. 922–926. DOI: 10.1109/IMIS.2012.26.

[31] F. Tao, F. Sui, A. Liu, Q. Qi, M. Zhang, B. Song, Z. Guo, S. Lu, and A. Nee, «Digital twin-driven product design framework», *International Journal of Production Research*, vol. 57, pp. 1–19, Feb. 2018. DOI: 10.1080/00207543.2018.1443229.

[32] R. Rosen, G. von Wichert, G. Lo, and K. D. Bettenhausen, «About The Importance of Autonomy and Digital Twins for the Future of Manufacturing», *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 567–572, 2015, 15th IFAC Symposium onInformation Control Problems inManufacturing, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2015.06.141. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315003808.

[33] C. Cimino, E. Negri, and L. Fumagalli, «Review of digital twin applications in manufacturing», *Computers in Industry*, vol. 113, p. 103 130, 2019, ISSN: 0166-3615. DOI: https://doi.org/10.1016/j.compind.2019.103130. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166361519304385.

[34] Y. Lu, C. Liu, K. I.-K. Wang, H. Huang, and X. Xu, «Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues», *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101 837, 2020, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2019.101837. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584519302480.

[35] Y. K. Liu, S. K. Ong, and A. Y. C. Nee, «State-of-the-art survey on digital twin implementations», *Advanced Manufacturing*, vol. 10, no. 1, pp. 1–23, 2022. DOI: 10.1007/s40436-021-00375-w. [Online]. Available: https://doi.org/10.1007/s40436-021-00375-w.

[36] C.-H. Chu and Y.-L. Liu, «Augmented reality user interface design and experimental evaluation for human-robot collaborative assembly», *Journal of Manufacturing Systems*, vol. 68, pp. 313–324, 2023, ISSN: 0278-6125. DOI: https://doi.org/10.1016/j.jmsy.2023.04.007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0278612523000717.

[37] S. A. Green, M. Billinghurst, X. Chen, and J. G. Chase, «Human-Robot Collaboration: A Literature Review and Augmented Reality Approach in Design», *International Journal of Advanced Robotic Systems*, vol. 5, no. 1, p. 1, 2008. DOI: 10.5772/5664. eprint: ://doi.org/10.5772/5664. [Online]. Available: ://doi.org/10.5772/5664.

[38] P. A. Lasota and J. A. Shah, «Analyzing the Effects of Human-Aware Motion Planning on Close-Proximity Human–Robot Collaboration», *Human Factors*, vol. 57, no. 1, pp. 21–33, 2015, PMID: 25790568. DOI: 10.1177/0018720814565188. eprint: https://doi.org/10.1177/0018720814565188. [Online]. Available: https://doi.org/10.1177/0018720814565188.

[39] S. Ong, A. Nee, A. Yew, and N. Thanigaivel, «AR-Assisted Robot Welding Programming», *Advances in Manufacturing*, vol. 8, no. 1, pp. 40–48, 2020, ISSN: 2195-3597. DOI: 10.1007/s40436-019-00283-0. [Online]. Available: https://doi.org/10.1007/s40436-019-00283-0.

[40] I. Malý, D. Sedláček, and P. Leitão, «Augmented reality experiments with industrial robot in industry 4.0 environment», in *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, 2016, pp. 176–181. DOI: 10.1109/INDIN.2016.7819154.

[41] D. Puljiz and B. Hein, *Concepts for End-to-end Augmented Reality based Human-Robot Interaction Systems*, 2019. arXiv: 1910.04494 [cs.RO]. [Online]. Available: https://arxiv.org/abs/1910.04494.

[42] D. Puljiz, E. Stöhr, K. S. Riesterer, B. Hein, and T. Kröger, «General Hand Guidance Framework using Microsoft HoloLens», in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 5185–5190. DOI: 10.1109/IROS40897.2019.8967649.

[43] B. Tipary and G. Erdős, «Generic development methodology for flexible robotic pick-and-place workcells based on Digital Twin», *Robotics and Computer-Integrated Manufacturing*, vol. 71, p. 102 140, 2021, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2021.102140. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584521000259.

[44] M. Parigi Polverini, A. M. Zanchettin, and P. Rocco, «A constraint-based programming approach for robotic assembly skills implementation», *Robotics and Computer-Integrated Manufacturing*, vol. 59,

pp. 69–81, 2019, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2019.03.005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584518305726.

[45] J. Jiang, Z. Huang, Z. Bi, X. Ma, and G. Yu, «State-of-the-Art control strategies for robotic PiH assembly», *Robotics and Computer-Integrated Manufacturing*, vol. 65, p. 101 894, 2020, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2019.101894. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584519302418.

[46] G.-Z. Yang, «Robot learning—Beyond imitation», *Science Robotics*, vol. 4, no. 26, eaaw3520, 2019. DOI: 10.1126/scirobotics.aaw3520. eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.aaw3520. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.aaw3520.

[47] G. Boschetti, M. Bottin, M. Faccio, and R. Minto, «Multi-robot multi-operator collaborative assembly systems: a performance evaluation model», *Journal of Intelligent Manufacturing*, vol. 32, no. 5, pp. 1455–1470, 2021, ISSN: 1572-8145. DOI: 10.1007/s10845-020-01714-7. [Online]. Available: https://doi.org/10.1007/s10845-020-01714-7.

[48] F. Azimifar, M. Abrishamkar, B. Farzaneh, A. Sarhan, and H. Amini, «Improving teleoperation system performance in the presence of estimated external force», *Robotics and Computer-Integrated Manufacturing*, vol. 46, pp. 86–93, 2017, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2016.12.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584516301879.

[49] T. Masood and J. Egger, «Augmented reality in support of Industry 4.0—Implementation challenges and success factors», *Robotics and Computer-Integrated Manufacturing*, vol. 58, pp. 181–195, 2019, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2019.02.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584518304101.

[50] A. Hietanen, R. Pieters, M. Lanz, J. Latokartano, and J.-K. Kämäräinen, «AR-based interaction for human-robot collaborative manufacturing», *Robotics and Computer-Integrated Manufacturing*, vol. 63, p. 101 891, 2020, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2019.101891. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584519307355.

[51] Y. Pan, C. Chen, D. Li, Z. Zhao, and J. Hong, «Augmented reality-based robot teleoperation system using RGB-D imaging and attitude teaching device», *Robotics and Computer-Integrated Manufacturing*, vol. 71, p. 102 167, 2021, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2021.102167. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S073658452100051X.

[52] S. Ong, A. Yew, N. Thanigaivel, and A. Nee, «Augmented reality-assisted robot programming system for industrial applications», *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101 820, 2020, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2019.101820. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584519300250.

[53] M. Matulis and C. Harvey, «A robot arm digital twin utilising reinforcement learning», *Computers & Graphics*, vol. 95, pp. 106–114, 2021, ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2021.01.011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S009784932100011X.

[54] E. Yigitbas, K. Karakaya, I. Jovanovikj, and G. Engels, *Enhancing Human-in-the-Loop Adaptive Systems through Digital Twins and VR Interfaces*, 2021. arXiv: 2103.10804 [cs.HC]. [Online]. Available: https://arxiv.org/abs/2103.10804.

[55] I. A. Tsokalo, D. Kuss, I. Kharabet, F. H. P. Fitzek, and M. Reisslein, «Remote Robot Control with Human-in-the-Loop over Long Distances Using Digital Twins», in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9013428.

[56] C. Li, P. Zheng, S. Li, Y. Pang, and C. K. Lee, «AR-assisted digital twin-enabled robot collaborative manufacturing system with human-in-the-loop», *Robotics and Computer-Integrated Manufacturing*, vol. 76, p. 102 321, 2022, ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2022.102321. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584522000102.

APPENDIX A

Additional content

A.1 Unity Robotics Hub Overview

Before diving into the specifics of establishing the ROS-Unity connection and further develop the project, both tutorials and resources available in the Unity Robotics Hub were studied. This GitHub repository serves as a central hub for tools, tutorials, and documentation tailored for robotic simulation in Unity.

A.1.1 Available Documentation

It offers a range of tutorials that are invaluable for setting up and extending ROS-Unity integration, as well as to understand how ROS concepts work inside Unity's environment:

- **ROS–Unity Integration: Initial Setup** - Guides you through the initial steps of setting up communication between ROS and Unity, including package installation and network configuration.
- **ROS–Unity Integration: Network Description** - Provides a detailed overview of network settings and offers troubleshooting tips for connectivity issues.
- **ROS–Unity Integration: Publisher** - Teaches you how to publish messages from a Unity scene to ROS, with practical examples involving GameObject data.
- **ROS–Unity Integration: Subscriber** - Demonstrates how to subscribe to ROS topics in Unity and use the received messages to alter objects in a Unity scene.
- **ROS–Unity Integration: Unity Service** - Covers the implementation of ROS services within Unity, allowing Unity to respond to ROS service requests.
- **ROS–Unity Integration: Service Call** - Explains how to call external ROS services from Unity, enabling Unity to request data or actions from ROS nodes.

The repository also includes example scripts that correspond to each tutorial.

A.2 Establishing the Network Connection

After reviewing the Unity Robotics Hub tutorial on network integration, it became clear that establishing a network connection between the Unity and ROS environments was the first crucial step in remote application development. The process involves:

- **Setting up the network:** Connect the Unity laptop to a Wi-Fi network, then connect the Ubuntu laptop running ROS to the hotspot created by the Unity laptop.
- **Configuring the IP address:** Use the IP address from the Unity laptop within the Unity inspector as shown in Figure A.1, and in the `ROSConnection.cs` script to ensure proper communication.
- **Specify the IP Address in ROS Workspace:** A new *.launch* file was created to initialize new nodes, including the `server_endpoint` node from the `ros_tcp_endpoint` package, crucial for establishing a proper connection between the ROS and Unity environments. An example of the IP definition in the *.launch* file can be seen below.

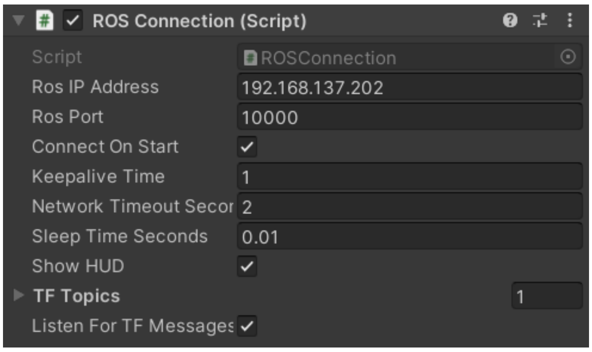


Figure A.1: Unity Connection Inspector

```
<arg name="tcp_ip" default="192.168.137.202"/>
<arg name="tcp_port" default="10000"/>

<node name="server_endpoint" pkg="ros_tcp_endpoint"
    type="default_server_endpoint.py" args="--wait" output="screen"
    respawn="true">
    <param name="tcp_ip" type="string" value="$(arg tcp_ip)"/>
    <param name="tcp_port" type="int" value="$(arg tcp_port)"/>
</node>
```

This setup is fundamental for the Unity environment to interact effectively with ROS, allowing for real-time data exchange and control commands to be sent between the two systems. Further details are available in the Unity Robotics Hub tutorial on ROS-Unity integration.

A.3 ROS Message Generation

After establishing the communication between ROS and Unity, as well as the other basic communication channels—publishers, subscribers, and services, the next step was to customize these components to handle the specific types of messages required for this project.

A.3.1 Adapting to Specific Message Types

To begin, I revisited the previously mentioned GitHub repositories for the UR10e robot, specifically focusing on identifying which ROS nodes and topics were critical for my Unity-ROS integration. This involved:

- **Identifying Key Nodes and Topics:** The primary focus was on finding which topic is responsible for publishing the current state of the robot's joints.
- **Message Subscription and Retrieval:** With guidance from my supervisors, I determined that subscribing to the `/joint_states` topic would be essential for retrieving real-time data about the robot's joint positions. This topic uses the `sensor_msgs/JointState` message type, a standard message in ROS that provides the positions, velocities, and efforts of the robot's joints. It is well-documented and includes the necessary fields for capturing joint states. This message type is part of the broader `common_msgs/sensor_msgs` package, which is available on the ROS wiki and in the sensor_msgs GitHub repository.

A.3.2 ROS-TCP Connector and C# Message Generation

To correctly handle the `sensor_msgs/JointState` messages within Unity, it was necessary to generate corresponding C# classes. This process, which is detailed in the ROS-TCP Connector documentation, involves:

1. **Generating C# Message Classes:** By following the steps provided in the documentation, I was able to generate the necessary C# JointState class that represent the corresponding ROS message, as depicted in the figure A.2. This step was crucial for storing and manipulating the joint state data within the Unity environment.

Figure A.2: JointState message generation in Unity, corresponding to the desired ROS message

2. **Compiling and Verifying the Message Classes:** After generating the message classes, I compiled them in the Windows environment, ensuring they matched the expected structure as described in the UnityRoboticsTutorial repository. This process took some time to fully understand, but was critical for the successful integration of ROS data into Unity.

A.4 Position Control (1 control type) - Unity to ROS

- confirm the purpose of this section - remove it? This control type enables operators to interact directly with the robot's joints in real-time, providing a responsive and highly interactive control environment. The implementation of Position Control involves several key steps to ensure seamless operation and integration with the ROS environment.

A.4.1 Saving and Sending Joint States

The first step in implementing Position Control was to capture and save the current states of the robot's joints in Unity. These joint states are then packaged and sent to the ROS environment upon user command. This process involves:

1. **Capturing Joint States:** The Unity application continuously monitors and records the positions of each joint of the digital twin robot, visible in the debug console represented in the figure A.3

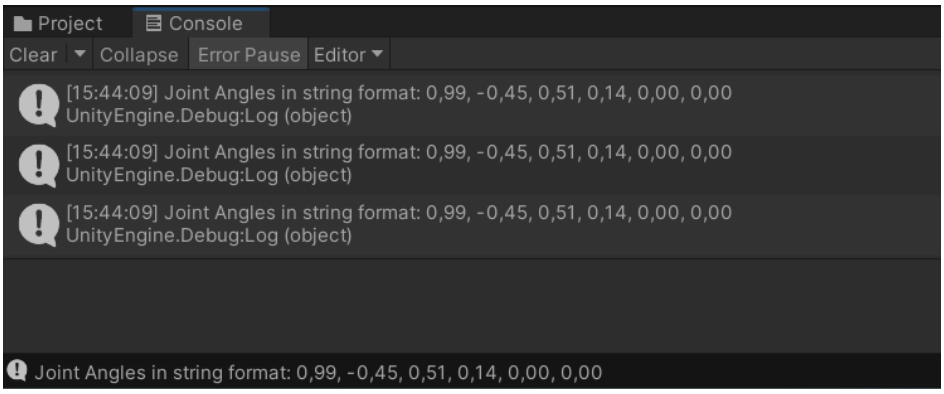


Figure A.3: Unity debug console showing the current digital twin's joint positions

2. **Data Serialization:** The joint states are serialized into a format suitable for transmission over the network. This was done by updating the joint positions into a list that is constantly saved into a JSON format, as depicted in the picture A.4.
3. **Sending Data:** Upon pressing a UI button trigger, shown in figure A.5, the serialized joint states are sent from Unity to the ROS environment using the previously established TCP/IP connection.

jointStatePublisher.json

Assets > Resources > jointStatePublisher.json > ...

```
{
    "jointPositions": [
        {
            "jointName": "shoulder_pan_joint",
            "position": 0.9948359727859497
        },
        {
            "jointName": "shoulder_lift_joint",
            "position": -0.4398227035999298
        },
        {
            "jointName": "elbow_joint",
            "position": 0.5078903436660767
        },
        {
            "jointName": "wrist_1_joint",
            "position": 0.1413716971874237
        },
        {
            "jointName": "wrist_2_joint",
            "position": 0.0
        },
        {
            "jointName": "wrist_3_joint",
            "position": 0.0
        }
    ]
}
```

Figure A.4: JSON format used to store Unity's digital twin joint states

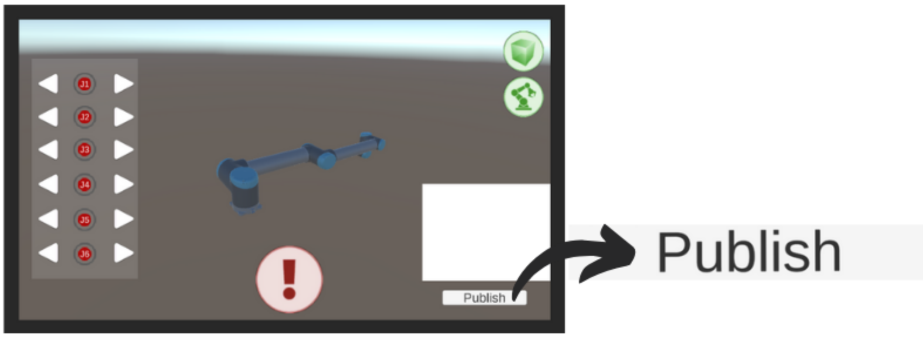


Figure A.5: UI Interface with Publishing button to send Unity's robot joint states into ROS the environment

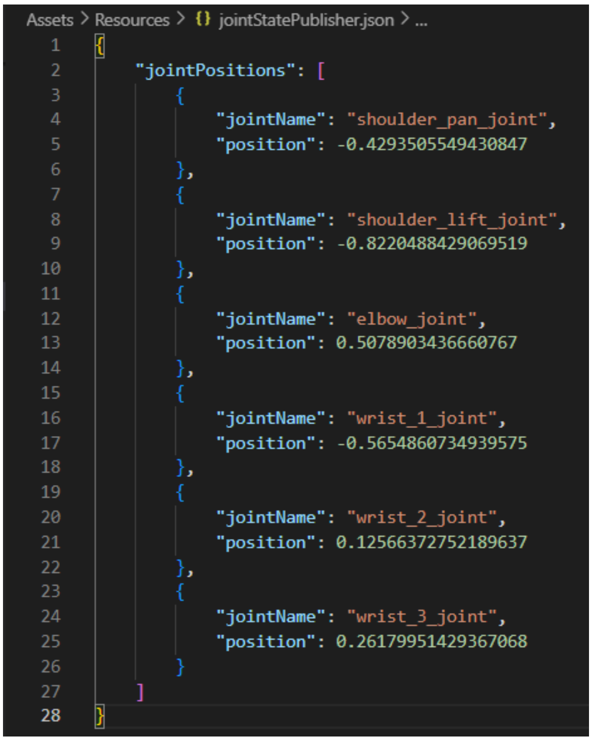
A.4.2 ROS Integration for Position Control

In order to receive the data, the ROS environment needed some adjustments to process these joint states coming from Unity environment. Two new nodes were added to the ROS setup, enabling the following key functionalities:

1. **Joint State Listener Node:**
   - **Purpose:** Receives joint states from Unity, ensuring that the digital inputs are translated into actionable data within the ROS environment.
   - **Functionality:** This node listens to a dedicated topic from Unity, processes this data, and republishes it to a different control topic. Figure A.6 illustrates an example of data transmitted from Unity to ROS. The accompanying debug log below demonstrates its accuracy.

```
rosrun iris_ur10e unity_joint_subscriber.py
[INFO] [1725629583.306162, 720.978000]:
/joint_state_listener_7441_1725629572125I
heard (-0.4293505549430847, -0.8220488429069519,
0.5078903436660767, -0.5654860734939575,
0.12566372752189636, 0.26179951429367065)
```

2. **Robot Control Node:**
   - **Purpose:** Directly controls the robot's movements based on processed joint states from Unity.



Assets > Resources > {} jointStatePublisher.json > ...


```json
{
    "jointPositions": [
        {
            "jointName": "shoulder_pan_joint",
            "position": -0.4293505549430847
        },
        {
            "jointName": "shoulder_lift_joint",
            "position": -0.8220488429069519
        },
        {
            "jointName": "elbow_joint",
            "position": 0.5078903436660767
        },
        {
            "jointName": "wrist_1_joint",
            "position": -0.5654860734939575
        },
        {
            "jointName": "wrist_2_joint",
            "position": 0.12566372752189637
        },
        {
            "jointName": "wrist_3_joint",
            "position": 0.26179951429367068
        }
    ]
}
```


Figure A.6: JSON file containing the unity's joint states that were sent to ROS environment

- **Functionality:** This node requires that the previous one is also running, and subscribes to the control topic */move_joint_unity* to fetch and apply joint state data to the physical robot, mirroring the Unity operator's interactions. The following debug log outputs the correct utilization of this node

```
rosrun iris_sami move_unity.py
[ INFO] [1725629576.546376282]: Loading robot model 'ur10e'.
[ INFO] [1725629577.773871662, 715.485000000]: Ready to take
commands for planning group manipulator.
[INFO] [1725629583.310087, 720.982000]: Moving arm to joint
positions: (-0.4293505549430847, -0.8220488429069519,
0.5078903436660767, -0.5654860734939575,
0.12566372752189636, 0.26179951429367065)
```

and the figure A.7 displays two cases where the robot and its digital twin are properly synchronized using this control type.

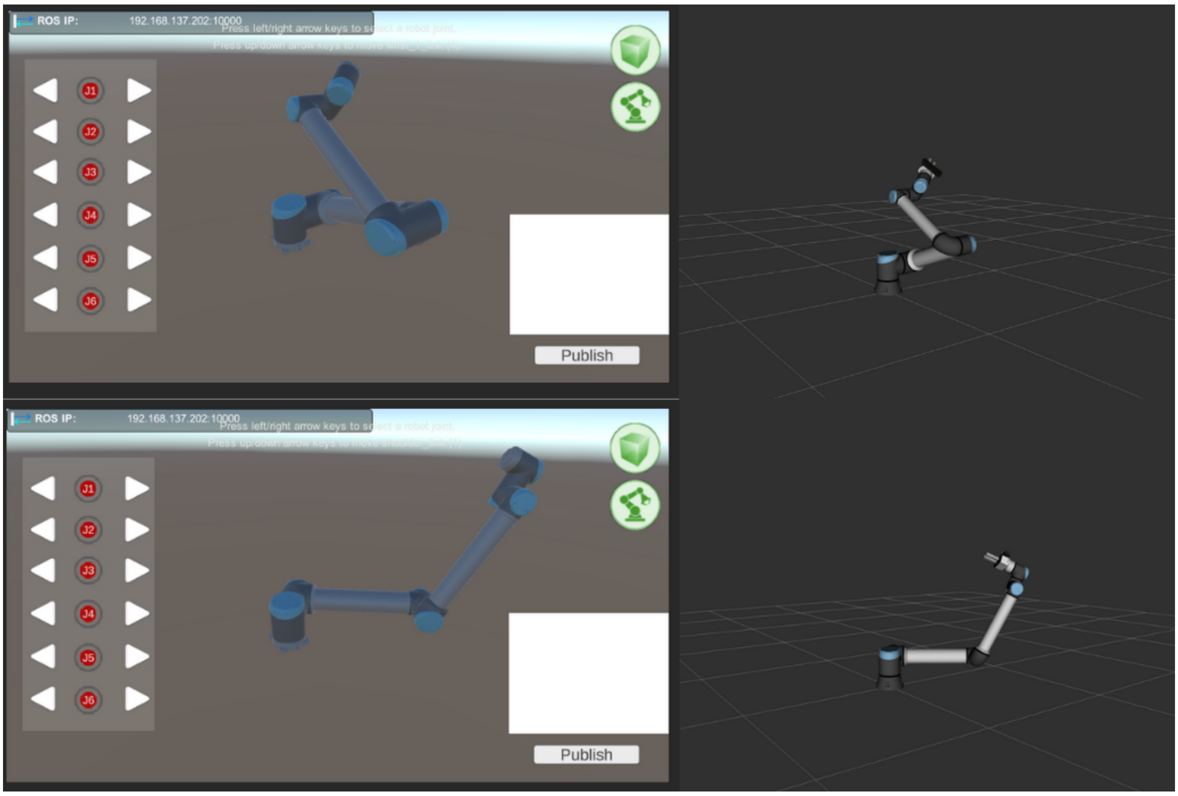


Figure A.7: Two scenarios showcasing the synchronization between Unity and ROS environments using Joint Control

A.5 Joint State Subscription (2 control type) - ROS to Unity

While the Position Control type allows operators to interactively manipulate the robot's joints and send these changes to the ROS environment, the Joint State Subscription control type operates in an opposite manner. It is designed to continually update the Unity digital twin's joint positions in synchronization with movements from the physical robot or its simulation in RViz.

A.5.1 Saving ROS Data

In order to properly synchronize Unity's digital twin with real-time robot movements from the ROS environment, a new script `JointStateSubscriber.cs` was created. It subscribes to the */joint_states* topic to continuously capture and store the robot's joint positions. This process is critical for maintaining a live reflection of the robot's state within the Unity simulation.

Subscribing to ROS Topics and Data Serialization

- **Subscribing to Joint States:** The script actively listens to the `/joint_states` topic, ensuring that any movement in the robot is promptly reflected in Unity.
- **Storing Joint Data:** Captured joint positions are stored in a C# dictionary, facilitating efficient data access and manipulation.
- **Serializing Data to JSON:** The joint data is serialized into a JSON file, which provides a persistent and accessible format for storing the robot's state, overcoming potential restrictions from Unity packages that limit direct folder access.

Explanation and Integration

The implementation of this system ensures that Unity's digital twin is consistently updated with the latest joint states from ROS, offering an accurate virtual representation for monitoring or interaction. Using the `SaveJointPositionsToFile()` method, joint data is structured and saved to enhance development workflow and project scalability.

Practical Application and Visualization

The real-time state of the robot's joints, whether it is operating in a simulated environment or in real-time with the physical robot, is represented in figure A.8. Initially, the script was designed to save these robot joints' values upon pressing a UI button, but it was later adapted to continuously update the `float64[] position` array from the ROS side, visible in the below terminal log snippet, into the Unity environment as shown in the figure A.9, ensuring that the most recent joint positions were always available.

```
name:
    - elbow_joint
    - left_finger_joint
    - right_finger_joint
```

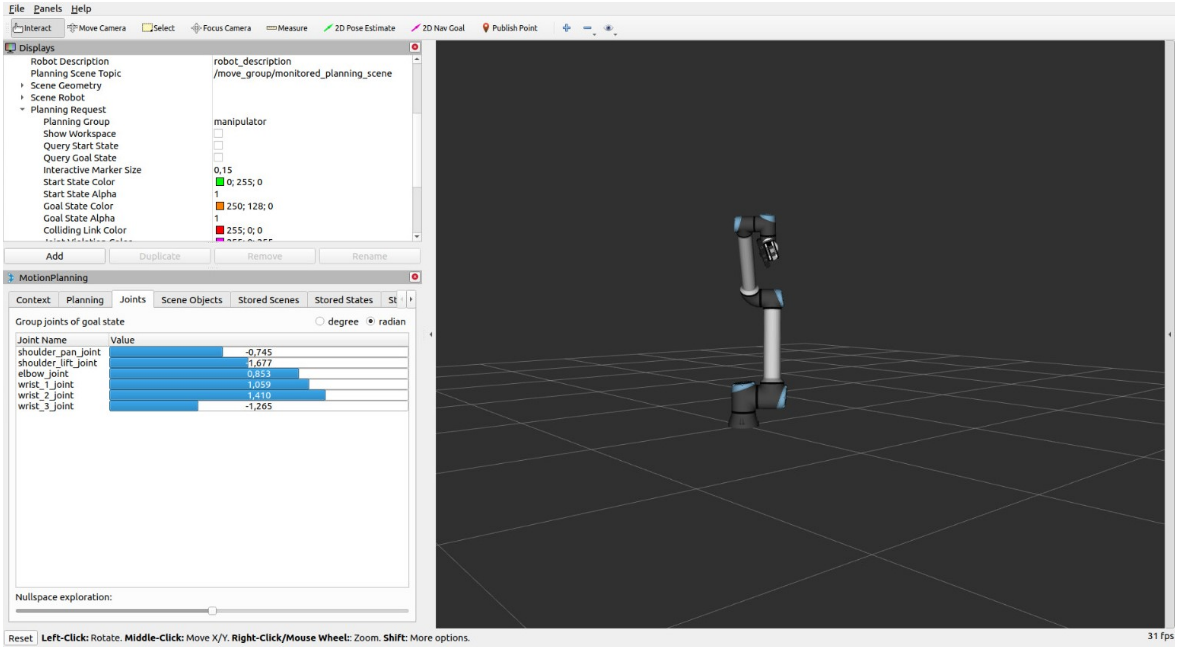


Figure A.8: Rviz environment with simulated Robot and correspondent joint values, in radians

```
  - shoulder_lift_joint
  - shoulder_pan_joint
  - wrist_1_joint
  - wrist_2_joint
  - wrist_3_joint
position: [0.8535921338670764, -5.695760001087214e-08, 5.587315793023694e-08,
-1.6766575764811593, -0.7445104810535929, 1.059219605892937, 1.410329834615414,
-1.265097896821196]
velocity: [-4.84869176906394e-05, -0.00020121543757810655, 0.00019870884754583778,
0.006705746353461187, 0.00061522726960112, -0.00033804213228767756,
0.001088906936911304, 0.00029848778568498711]
effort: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
```

Assets > Resources > {} jointStateSubscriber.json

```
Joint Name: elbow_joint, Position: 0.8535922
Joint Name: shoulder_lift_joint, Position: -1.676658
Joint Name: shoulder_pan_joint, Position: -0.7445105
Joint Name: wrist_1_joint, Position: 1.059219
Joint Name: wrist_2_joint, Position: 1.41033
Joint Name: wrist_3_joint, Position: -1.265098
Joint Name: left_finger_joint, Position: -5.699415E-08
Joint Name: right_finger_joint, Position: 5.590857E-08
```

Figure A.9: JointStateSubscriber Json file with Robot's joint values, in radians