



**João Luís  
Fernandes Clemente**

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

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





**João Luís  
Fernandes Clemente**

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

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

Tese apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, 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.

Texto Apoio financeiro do POCTI  
no âmbito do III Quadro Comu-  
nitário de Apoio.

Texto Apoio financeiro da FCT e do  
FSE no âmbito do III Quadro Comu-  
nitário de Apoio.

Dedico este trabalho à minha esposa e filho pelo incansável apoio.

**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**

Agradeço toda a ajuda a todos os meus colegas e companheiros.

**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

<b>Contents</b>	<b>i</b>
<b>List of Figures</b>	<b>iii</b>
<b>List of Tables</b>	<b>v</b>
<b>Acronyms</b>	<b>vi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Goals . . . . .	2
1.3 Thesis' Structure . . . . .	2
<b>2 State of Art</b>	<b>3</b>
2.1 Industry 4.0 . . . . .	3
2.2 Industry 5.0 . . . . .	5
2.2.1 The Key Drivers of Industry 5.0 . . . . .	5
2.2.2 Human-Robot Collaboration . . . . .	6
2.2.3 Collaborative Robots (Cobots) . . . . .	7
2.2.4 Collaboration Scenarios . . . . .	8
2.3 Digital Realities . . . . .	9
2.4 Digital Twins . . . . .	10
2.4.1 Introduction to Digital Twins . . . . .	10
2.4.2 Historical Origins of Digital Twins . . . . .	11
2.4.3 Characteristics and Applications of Digital Twins . . . . .	11
2.4.4 Digital Twins in Academia and Industry . . . . .	11
2.4.5 Integration of Augmented Reality with Digital Twins . . . . .	12
2.5 Human-Robot Collaboration and Industrial Applications . . . . .	13
2.5.1 Enhancing Human-Robot Collaboration through AR and DT Implementations	13
2.6 Summary . . . . .	16

<b>3</b>	<b>Methodology</b>	<b>18</b>
<b>4</b>	<b>Augmented Reality on On-site Collaboration</b>	<b>19</b>
4.1	Marker Detection . . . . .	20
4.2	Digital Model Implementation of the Robot . . . . .	21
4.3	On-site Application Features . . . . .	21
4.3.1	Virtual Safety Zones . . . . .	21
4.3.2	Interface . . . . .	22
4.4	Video Demo . . . . .	24
<b>5</b>	<b>Remote Member's Application</b>	<b>25</b>
5.1	Overview and Contextualization . . . . .	25
5.2	Unity Robotics Hub Overview . . . . .	26
5.2.1	Available Documentation . . . . .	26
5.3	Establishing the Network Connection . . . . .	26
5.4	ROS Message Generation . . . . .	27
5.4.1	Adapting to Specific Message Types . . . . .	28
5.4.2	ROS-TCP Connector and C# Message Generation . . . . .	28
5.5	New Control Types for Remote Operation . . . . .	29
5.6	Position Control - Unity ROS . . . . .	29
5.6.1	Saving and Sending Joint States . . . . .	29
5.6.2	ROS Integration for Position Control . . . . .	31
5.7	Joint State Subscription - ROS Unity . . . . .	33
5.7.1	Saving ROS Data . . . . .	33
5.8	Other features - (Camera Feed Transmission) - add more . . . . .	35
<b>6</b>	<b>Discussion and Evaluation</b>	<b>36</b>
<b>7</b>	<b>Conclusion and Future Work</b>	<b>37</b>
	<b>References</b>	<b>38</b>
<b>A</b>	<b>Additional content</b>	<b>40</b>

# List of Figures

2.1	Key enabling technologies in Industry 4.0 - [3] . . . . .	4
2.2	Different levels of Human-Robot Interaction (HRI) - [9] . . . . .	7
2.3	The mixed reality spectrum, from [13] . . . . .	10
2.4	A Digital Twin reference model [24] . . . . .	12
2.5	AR displayed work envelope of robotic arm <b>Chu2023</b> . . . . .	14
2.6	Schematic diagram of the building and programming of a robotic station alongside VR technologies [33] . . . . .	15
2.7	Augmented Reality (AR)-assisted Digital Twin (DT)-enabled multi-robot collaborative manufacturing system [34] . . . . .	16
4.1	Robot UR10e used for the development of the dissertation work, available at IRIS-LAB, University of Aveiro . . . . .	20
4.2	ArUco marker used with the Logitech c922 camera for segmentation and manipulation of virtual environment . . . . .	20
4.3	Digital UR10 model related to the aruco marker, on Unity environment . . . . .	21
4.4	Simulated environment showing the display of both the inner and outer safety zones, as well as the robot and the marker inside them . . . . .	22
4.5	Interface featuring developed interactions . . . . .	23
4.6	Example from first joint control menu and toggle buttons for controller menu and safety-zones . . . . .	24
5.1	Unity Connection Inspector . . . . .	27
5.2	JointState message generation in Unity, corresponding to the desired ROS message . . . . .	28
5.3	Unity debug console showing the current digital twin's joint positions . . . . .	30
5.4	JSON format used to store Unity's digital twin joint states . . . . .	30
5.5	UI Interface with Publishing button to send Unity's robot joint states into ROS the environment . . . . .	30
5.6	JSON file containing the unity's joint states that were sent to ROS environment . . . . .	31
5.7	Two scenarios showcasing the synchronization between Unity and ROS environments using Joint Control . . . . .	32
5.8	Rviz environment with simulated Robot and correspondent joint values, in radians . . . . .	34

5.9	JointStateSubscriber Json file with Robot's joint values, in radians . . . . .	35
-----	--	----

# List of Tables

2.1	Levels of Control, adapted from [25]	12
-----	--------------------------------------	----

# Acronyms

<b>HRI</b>	Human-Robot Interaction	<b>XR</b>	Extended Reality
<b>AI</b>	Artificial Intelligence	<b>AR</b>	Augmented Reality
<b>CPSS</b>	Cyber-Physical systems	<b>VR</b>	Virtual Reality
<b>IOT</b>	Internet of Things	<b>DR's</b>	Digital Realities
<b>ML</b>	Machine Learning	<b>ROS</b>	Robot Operating System
<b>VR</b>	Virtual Reality	<b>MR</b>	Mixed Reality
<b>AR</b>	Augmented Reality	<b>HHD</b>	Handheld Device
<b>HRC</b>	Human-Robot Collaboration	<b>HMD</b>	Head-Mounted Display
<b>DT</b>	Digital Twin	<b>CPS</b>	Cyber-Physical System

# Introduction

*This chapter aims to describe the context of this dissertation, outlining the primary goals and contributions of the developed work. Additionally, it will present the structure of this document.*

## 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, necessitating companies to prioritize reskilling and upskilling their workforce.

However, experts argue that complete automation, eliminating humans from 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. The complexity of these scenarios necessitates advanced solutions to enhance the HRI. One promising approach is the integration of Extended Reality (XR) technologies, which encompass Virtual Reality (VR), AR, and Mixed Reality (MR). XR technologies blend the physical and digital worlds, providing immersive experiences that transcend traditional reality and overcome geographical constraints, enabling real-time collaboration among individuals from different locations.

However, the potential of XR to enhance remote collaboration is currently hindered by several critical limitations. These include limited perspective and context capture, which impede remote collaborators' understanding and decision-making capabilities, and a lack of multisensory data collection, restricting comprehensive environmental comprehension. Additionally, XR's interaction with physical objects often lacks the precision required for detailed tasks, particularly in dynamic scenarios. These challenges diminish the effectiveness of XR in facilitating thorough context sharing and impact the overall efficiency and safety of collaborative tasks.



## 1.2 GOALS

The primary goal of this dissertation is to leverage MR alongside a static robotic arm (UR10e) to support remote collaboration scenarios. This involves transforming human-robotic collaboration by integrating XR technologies and robotic capabilities to enhance both on-site and remote collaboration experiences.

To achieve this, the dissertation proposes 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.
- Develop the capability for remote operation of the robot via the XR application, enhancing the remote participant'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.

By addressing these objectives, this dissertation aims to create a robust framework that enhances remote collaboration through the innovative use of MR and robotic technologies.

## 1.3 THESIS' STRUCTURE

fazer!!!! no fim

## State of Art

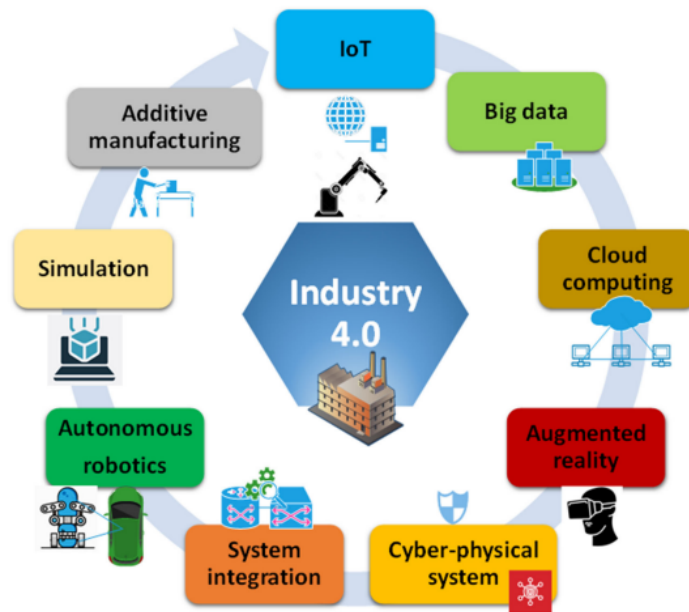
*In this chapter, the evolution leading to Industry 5.0 contextualizes the importance of technological advancements. The focus is on Industry 5.0, particularly its transition from autonomous robots to the emerging dynamics of Human-Robot Collaboration (HRC). The critical need for collaborative scenarios between humans and robots is highlighted, emphasizing the transformative potential of such partnerships. Additionally, the role of digital realities alongside DT within Industry 5.0, assessing their effectiveness in facilitating HRI, marking a new era of efficiency and collaboration in the industry.*

### 2.1 INDUSTRY 4.0

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

With 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. While these robots can perform highly skilled tasks, they often operate within 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. The relentless push towards full automation has, in some cases, reduced the



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

involvement of humans in critical decision-making processes, leading to a more machine-centric production landscape [4]

## 2.2 INDUSTRY 5.0

Approximately a decade after the launch of Industry 4.0, the European Commission introduced Industry 5.0 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 a new industrial paradigm: Industry 5.0. As a response to the increasing automation and the need for greater human involvement in manufacturing processes, Industry 5.0 seeks to reintroduce the human element into industrial ecosystems [6].

The goal is to combine 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 seeks 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].

### 2.2.1 The Key Drivers of Industry 5.0

While Industry 4.0 focused on pushing the limits of automation and data-driven production, Industry 5.0 reintroduces the importance of human collaboration, recognizing that humans and machines each have unique advantages that can complement one another. Key technological drivers of Industry 5.0 include:

- **Artificial Intelligence and Cognitive Computing:** These technologies continue to evolve, enabling robots and automation systems to work alongside humans in ways that enhance human productivity without fully replacing them.
- **Collaborative Robots (Cobots):** Cobots are designed to work safely alongside humans, facilitating intuitive interaction and allowing both to collaborate on tasks that leverage the strengths of each. The development of collaborative robots focuses on creating systems that enable intuitive, seamless interaction between human users and robotic systems, a key aspect of Industry 5.0's vision. This redefines the employment roles of human workers, emphasizing their communication with robotic systems and artificial intelligence [7].
- **Digital Twins:** The incorporation of digital twins represents a pivotal technological advancement in Industry 5.0. Digital twins provide visual models that enhance comprehension and facilitate the evaluation of goods, processes, and production. By allowing real-time monitoring and simulation, digital twins help optimize manufacturing processes, bridging the gap between the virtual and physical worlds [7].
- **Human-Centric Automation:** The emphasis is placed on using technology to augment human capabilities rather than replace them, fostering a more inclusive, creative, and flexible manufacturing environment.

### 2.2.2 Human-Robot Collaboration

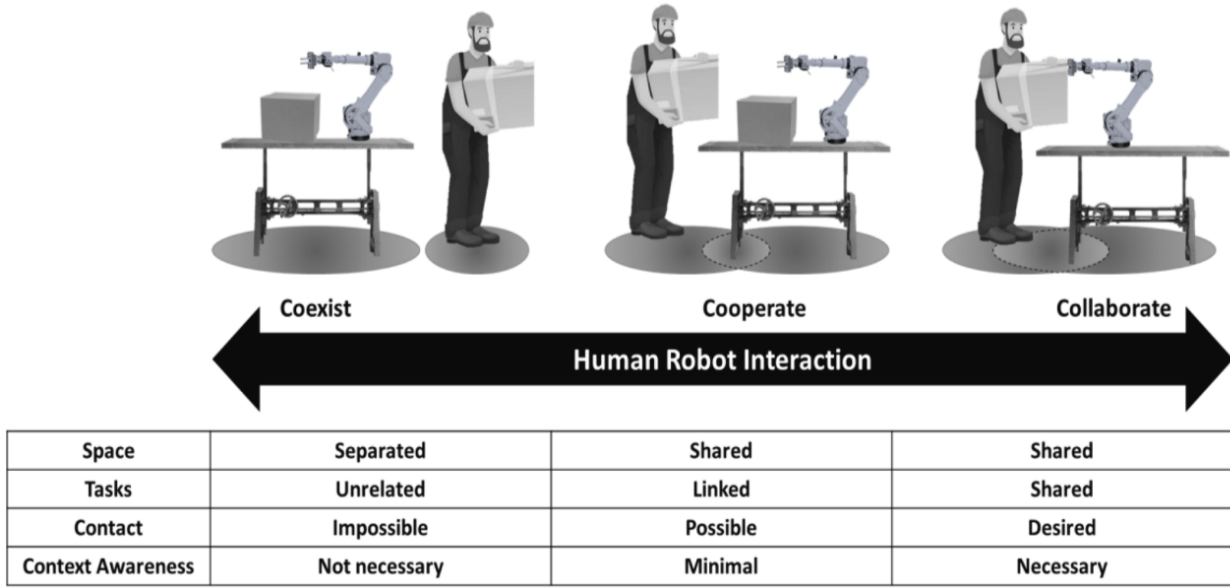
HRI focuses on creating robots that are safe, effective, and compatible for collaboration with humans rather than replacing them. This involves developing robots that are autonomous and also capable of understanding and predicting human actions, communicating with humans, and learning from human feedback. HRI is an extensive field dedicated to examining the interactions and coexistence of humans and robots in shared spaces. Its objective consists on enhancing these interactions by designing robots that can assist and cooperate with humans in diverse roles [8].

One key area within HRI is HRC, which specifically focuses on scenarios where humans and robots cooperate directly to achieve common goals. 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:

- **Human-Robot Coexistence (HRCx):** In this form of interaction, humans and robots share the same workspace but do not directly interact. Both carry out their tasks independently, with the primary focus on avoiding collisions. Usually this interaction does not involve synchronized work or communication between the two parties.
- **Human-Robot Cooperation (HRCp):** Here, humans and robots share both the workspace and the task objective. Cooperation occurs when humans and robots work toward a common goal, and advanced technologies such as sensors or machine vision may be used to detect and prevent collisions. However, there is limited physical interaction between them.
- **Human-Robot Collaboration (HRC):** This is the most integrated form of HRI, where humans and robots not only share the workspace and objectives but also interact directly. In HRC, physical contact may occur (e.g., through shared manipulation of an object), or there can be non-physical collaboration, such as verbal communication, gestures, or recognizing intentions through eye movement. In collaborative environments, humans usually perform tasks requiring fine motor skills or decision-making, while robots handle repetitive, strenuous, or hazardous tasks.

Despite both coexistence, cooperation and collaboration terms being used in HRI, as well as both allowing for a close interaction and a shared goal, this categorization differs not only based on the dependence and sequence of operations performed by the agents, but also based on the shared environment area. Therefore, collaboration can be defined as gathering human operators and robots sharing the sequence of a task towards achieving the same goal, allowing direct contact between both agents, as depicted in the figure 2.2 [9].

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].



**Figure 2.2:** Different levels of HRI - [9]

### 2.2.3 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.

#### *The Role and Benefits of Cobots*

Unlike traditional industrial robotic systems that necessitate extensive safety guarding, thus reducing flexibility and increasing both cost and spatial demands, cobots offer a solution geared towards the current market's need for reduced lead times and mass customization[1]. This demand is particularly prevalent among small- and medium-sized enterprises (SMEs). Cobots facilitate direct physical interaction between humans and machines, and their design allows for straightforward reprogramming by non-experts [3]. Collaboratively, humans and cobots can increase productivity while cutting production costs by combining human cognitive abilities with robotic precision and strength.

#### *Background and Evolution of Cobots*

The cobot concept was initially proposed by J. Edward Colgate and Michael Pashkin in 1996 [2,4 - confirmar], with the first commercial cobot, the UR5 model by Universal Robots, being sold in 2008 [6]. This marks the beginning of practical human-robot collaboration developments.

### *Understanding Collaboration in Cobots*

Clarifying the modes of collaboration is crucial as the term often leads to misunderstandings. Müller et al [7]. proposed a classification system for different collaboration methods between humans and cobots, indicating that these categorizations are not fixed and may vary across different literatures. - add figure 1 - summarizes that humans and cobots can work together.

### *Distinguishing Cobots from Industrial Robots*

Cobots are distinct from traditional industrial robots in that they are specifically designed with safety and ergonomics in mind [13]. They often include advanced features such as force and torque sensors, vision systems, and anti-collision systems, which are not typically part of traditional robotic setups. These features allow cobots to be deployed without the extensive additional costs and setup required for retrofitting traditional robots with similar capabilities.

### *Economic and Practical Advantages of Cobots*

The shift towards cobots in industrial settings is largely driven by economic factors, improvements in occupational health, and the efficient utilization of factory space. Cobots offer significant flexibility—they can be easily relocated within different parts of a plant and quickly adapted to changes in production layouts [16]. This is particularly advantageous in high-risk applications, although it does necessitate certain constraints similar to those in traditional systems. Cobots can also lead to lower direct unit production costs compared to traditional robotic systems [17].

### *Comparative Analysis with Traditional Robots*

- ver tabela 1 - comparacao between collaborative and traditional When comparing cobots and traditional robots across various tasks such as assembly, placement, handling, and picking, cobots often show superior daptability and cost-effectiveness. However, traditional robots still offer advantages in scenarios requiring high payload [23 serve?- confirmar. questao do payload?] and precision.

### *Future Trends in HRC*

figura 7 - ver Recent studies suggest future enhancements in HRC will focus on better understanding and integrating environmental and operator intentions into robotic operations. This involves employing advanced sensing and algorithmic strategies to make interaction more intuitive and safer, thereby increasing reliability and system adoption.

## **2.2.4 Collaboration Scenarios**

Cobots are different from traditional automation robots, being designed for direct interaction with humans within industrial settings. These cobots, which are now being developed for a range of industrial applications, eliminate the need for protective barriers, allowing them to work alongside humans and expand beyond the spatial constraints, typical of conventional industrial robots **Kadir2018** CITE THIS IN THE MENDELEY.

Key features that set cobots apart include their enhanced safety protocols for close human interaction and their simplified programming, which supports flexible application within manufacturing environments. This allows for straightforward deployment and reassignment throughout various factory processes, significantly aiding tasks such as assembly, packaging, and organizing **Faccio2019** CITE THIS IN THE MENDELEY.

The integration of HRC systems in smart manufacturing is opening doors to increased productivity and efficiency. Industries like automotive have begun implementing cobots equipped with advanced sensing and vision systems, demonstrating the potential for collaborative robots to revolutionize these sectors. Since the automotive industry requires a significant portion of production involving assembly tasks, combining human skills and cobot capabilities is proving essential for tasks requiring greater flexibility and robustness than what robots can provide on their own **Tsarouchi2016** CITE THIS IN THE MENDELEY.

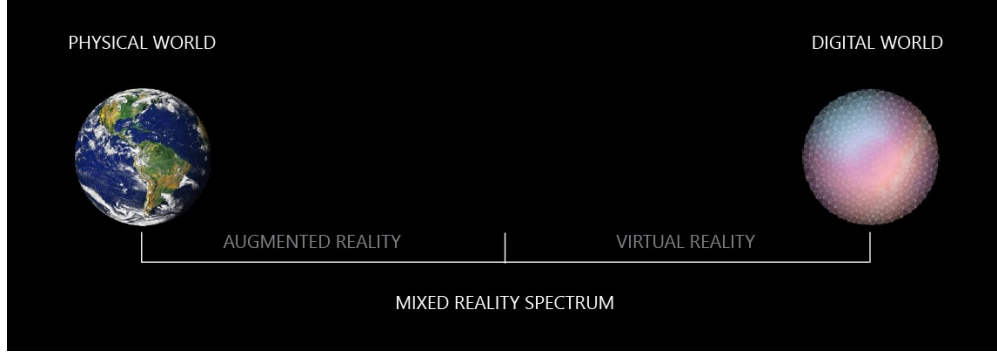
This shift towards a more collaborative manufacturing environment, pairing human intelligence with robotic efficiency, represents a significant stride towards the future of industrial automation.

### 2.3 DIGITAL REALITIES

Digital Realities (DR's) encompass a spectrum of technologies that blend virtual elements with the real world to varying degrees. In exploring the capabilities and applications of DR's, these can be broadly categorized into three main types:

- **AR:** Enhances the user's perception of the real world by overlaying digital content such as 3D graphics, information, or media directly onto their physical surroundings [10]. This technology enables the virtual content to interact dynamically with the environment in real-time. The ideal outcome is for users to perceive both virtual and real entities as inhabiting the same shared space, creating a unified and interactive experience [11]. AR requires spatial registration to ensure that virtual additions are contextually and spatially aligned with the physical world. A prime example of AR is Pokémon GO (add a picture of pokemon go view), where virtual creatures appear as if they exist within the real world, allowing users to interact with them through their mobile devices [12].
- **VR:** Immerses users in a completely synthetic environment, isolating them from the physical world. This immersion is achieved through Head-Mounted Display (HMD) (add picture of an example of HMD) that provide a fully virtual view, with technologies such as head tracking enhancing the experience by allowing natural movements to influence the virtual environment. VR typically features individual experiences and can transport users to remote or imaginary locations, offering a profound sense of presence in a digitally constructed reality [12].
- **MR:** Since MR lacks a universally agreed-upon definition, various sources describe it in different ways. In [13], MR encompasses a spectrum, as depicted in the picture 2.3 that includes everything from AR where the physical reality is dominant, to augmented virtuality where virtual elements are more prominent, facilitating a seamless transition between real and virtual worlds.





**Figure 2.3:** The mixed reality spectrum, from [13]

Given that MR is not an universally concept and is understood differently depending on the context, ten experts in AR/VR from academia and industry were interviewed [12].

Therefore, MR can encompass a wide range of notions, such as:

- **MR as a continuum:** Involving a range of experiences from AR to augmented virtuality, depending on the degree of virtual content integrated into the real world.
- **MR as Collaboration:** : Describes the interactive processes between users in different realities, such as one in AR and another in VR, emphasizing the collaborative potential of MR in shared or interconnected spaces.
- **MR as Strong AR:** Viewing MR as an advanced form of AR where interactions with and within the environment are more complex and integrated.
- **MR as a Synonym for AR:** In this usage, the terms MR and AR are used interchangeably. This approach typically occurs when a system or experience aligns closely with AR, and the definition of AR is utilized to clarify what is meant by MR.
- **MR as a Combination of AR and VR:** This definition describes MR as an integration of both AR and VR technologies. It refers to systems or applications that feature distinct AR and VR components, which may interact but are not necessarily seamlessly integrated. This can also apply to devices or apps capable of switching between AR and VR modes as needed.
- **MR as an Alignment of Environments:** This concept of MR involves the synchronization or alignment of virtual and physical environments. It describes systems where the virtual and real components are distinct yet aligned, similar to how collaboration setups work but without requiring actual collaborative interactions or physically separate environments.

## 2.4 DIGITAL TWINS

### 2.4.1 Introduction to Digital Twins

DT are sophisticated virtual replicas of physical entities that enable simulation, analysis, and control of systems within a digital framework. These digital counterparts are pivotal in

enhancing HRC, as they provide real-time, interactive environments that mirror the physical world. This mirroring capability facilitates improved decision-making, efficiency, and flexibility across various industrial applications.

#### **2.4.2 Historical Origins of Digital Twins**

The concept of DT was originally formulated by NASA in the 1980s for monitoring spacecrafts from Earth. With advancements in IOT and computational technologies, modern DT systems now utilize real-time sensor data to enhance the accuracy and reliability of simulations, significantly evolving from their initial conception [14].

#### **2.4.3 Characteristics and Applications of Digital Twins**

DT transform smart manufacturing by enabling detailed examination and prediction of the behavior of physical systems, thereby optimizing operations and minimizing downtime. In the realm of HRC, DT significantly enhance ergonomic interactions by adapting robotic movements to meet human needs, thus ensuring safer and more productive workplace conditions [15].

A notable implementation of DT technology can be observed in Singapore’s Smart Nation Initiative, where the Land Transport Authority utilizes a DT to simulate and evaluate potential policies before implementation, showcasing the vast potential of DT for modern and future technologies [16].

#### **2.4.4 Digital Twins in Academia and Industry**

Digital Twins (DT) have gained significant traction in both academic research and industrial applications in recent years. Despite their growing prominence, there is no universally accepted formal definition of a DT.

Nevertheless, the majority of reputable researchers concur that a DT is a Cyber-Physical System (CPS) comprising at least three essential components:

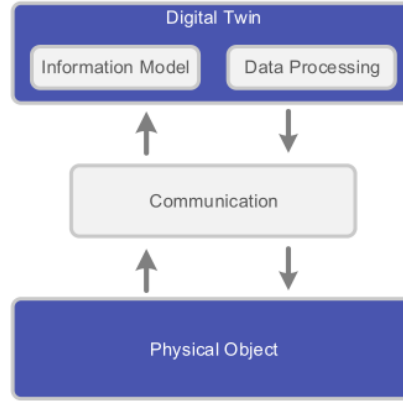
- A physical system,
- A virtual model,
- Connections facilitating bidirectional communication between the physical and virtual models [15], [17], [18].

In academic settings, DTs have been explored across a wide array of applications, including machine tool life management, product health management, smart cities, and patient health monitoring, among others [16], [19]–[22]. These applications demonstrate the versatility and potential of DTs in enhancing efficiency, predictive maintenance, and system optimization.

In the industrial sector, numerous companies, such as Siemens, offer software and platform solutions aimed at creating DTs for their industrial partners. However, some critics argue that these solutions represent digital shadows rather than true DTs. The primary contention is the lack of bidirectional communication capabilities [23]. Specifically, while changes in the physical system can influence the virtual model, the reverse—where modifications in the virtual model impact the physical system—is often absent.

In Figure 2.4, [24] outlines the DT reference model, illustrating the bidirectional communication between the physical and digital entities. This configuration underscores the distinction

between true DTs and digital shadows, the latter often criticized for lacking interactive communication capabilities [23].



**Figure 2.4:** A Digital Twin reference model [24]

As emphasized by Liu et al. [25], "A true DT must include bidirectional communication instead of having a virtual model that updates according to a physical system." Given that the concept of DT lacks a standardized definition, leading to a variability in the scope and execution of DT implementations and defining a misconception, the critical distinction between three types of interaction is further illustrated in Table 2.1.

**Table 2.1:** Levels of Control, adapted from [25]

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.

#### 2.4.5 Integration of Augmented Reality with Digital Twins

The integration of AR with DT represents a significant advancement in enhancing user interactions with complex systems. AR overlays virtual data directly onto physical objects, thereby providing a more intuitive and engaging interface for users. This capability is particularly beneficial in industrial settings where AR can facilitate more efficient and effective operation and maintenance processes [26], [27].

Despite the numerous advantages of DTs, a significant challenge lies in the effective visualization of the complex data they generate. As interest in DTs grows, it becomes

increasingly important to provide users with intuitive and accessible methods to interpret and interact with this data. According to [26], the benefits of real-time data and information provided by DTs cannot be fully realized without proper visualization techniques that present complex information in a clear and uncluttered manner, especially for users with limited technical expertise.

AR offers a promising solution to this challenge by enhancing data visualization and user interaction. Traditionally, users interact with digital information through 2D interfaces such as screens and monitors. However, AR expands this interaction into the real world by overlaying virtual data onto physical objects, creating immersive 3D experiences that align with users' familiar environments [27]. This 3D visualization enables users to better understand and absorb information through visual cues, which are often more effective than textual or auditory information [28]. For instance, projecting maintenance instructions directly onto a machine allows users to perform maintenance tasks more efficiently compared to consulting physical manuals [29]. Moreover, AR has been successfully utilized in various manufacturing processes to provide guidance for maintenance procedures and to train novice workers, demonstrating its versatility and effectiveness in industrial settings [30].

By integrating DT with AR, it is possible to monitor the state of a system in real-time using DTs while simultaneously displaying relevant data and information through AR interfaces. This integration not only enhances the user experience but also improves operational efficiency and safety by providing actionable insights in an intuitive and accessible manner.

## 2.5 HUMAN-ROBOT COLLABORATION AND INDUSTRIAL APPLICATIONS

Following the detailed exploration of DT and AR, this section discusses how these technologies integrate into HRC, with specific emphasis on remote collaboration and practical examples from the industry that utilize DT and AR solutions.

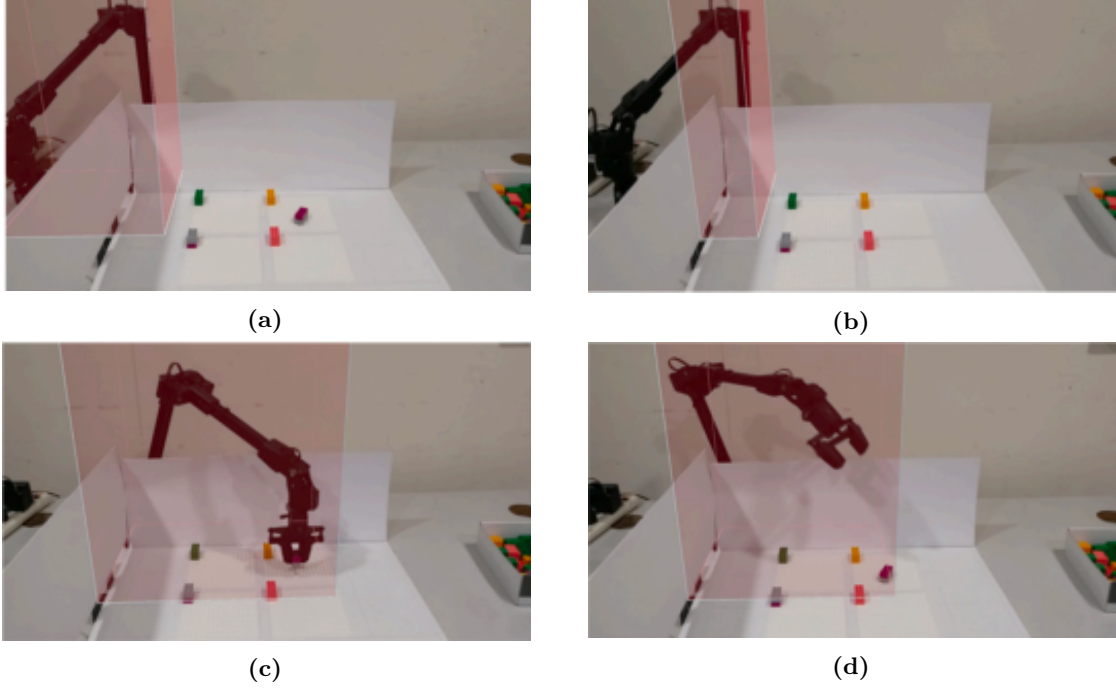
### 2.5.1 Enhancing Human-Robot Collaboration through AR and DT Implementations

Integrating Augmented Reality (AR) and Digital Twin (DT) technologies into Human-Robot Collaboration (HRC) has demonstrated significant improvements in interaction, safety, and efficiency. This section explores various solutions and their implementation advantages, drawing insights from recent studies.

#### 1. Augmented Reality for Enhanced Interaction and Safety

Leveraging AR can significantly improve the interaction between humans and robots by providing intuitive visualizations and reducing the need for continuous visual contact. As demonstrated by Chu et al. **Chu2023**, effectively conveying robot intentions through AR visuals enhances worker awareness of their surroundings, thereby improving safety and efficiency in collaborative tasks. The AR interfaces clearly visualize the robot's work envelope, as depicted in Figure 2.5, enabling human operators to navigate shared workspaces safely and efficiently.

Additionally, Lotsaris et al. [31] presented an AR application that facilitates operator work in human-robot environments by enabling coexistence and improving communication. Their system displays safety fields around the robot using different colors to represent safe and dangerous regions, enhancing situational awareness and reducing the risk of accidents. However, a significant challenge remains in the complexity of recognizing detailed information through haptic feedback, indicating a need for more sophisticated algorithms and sensors to capture and translate complex robot actions into intuitive AR visualizations.



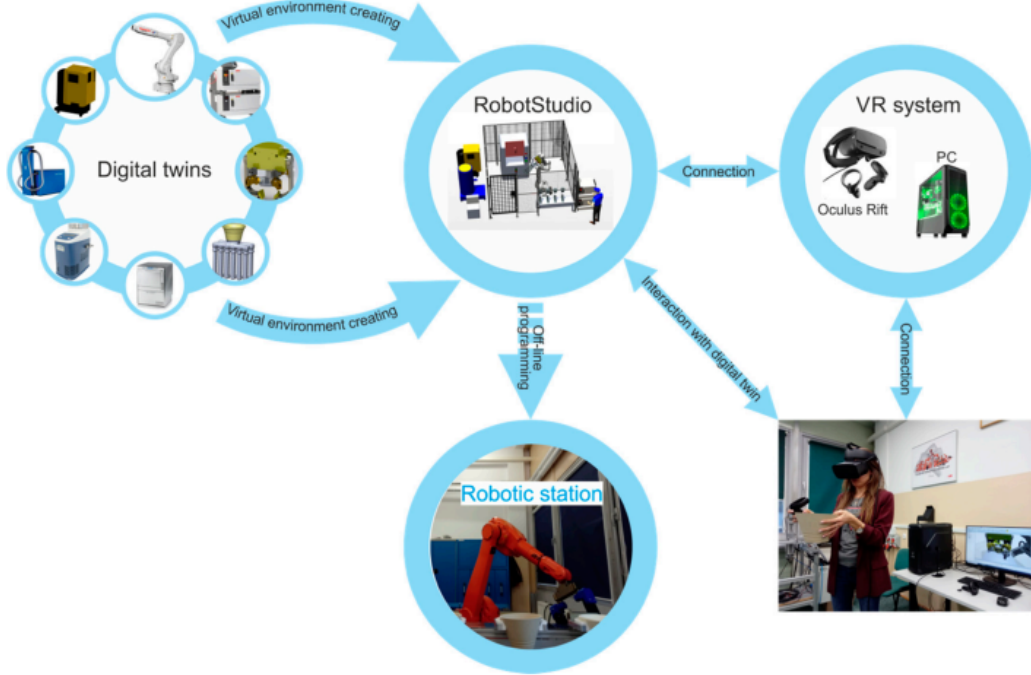
**Figure 2.5:** AR displayed work envelope of robotic arm **Chu2023**

## 2. Virtual and Augmented Reality for Intuitive Robot Programming

Programming robots using Virtual Reality (VR) and DT introduces innovative methods that capture human movements, reducing the learning curve for non-experts and enabling intuitive programming of complex tasks. Bolano et al. [32] demonstrated that bilateral communication between the VR environment and the robot's hardware allows changes to be made in both the virtual and physical systems. This full immersive experience enables users to interact directly with the robot's end effector through holographic interfaces, facilitating the selection and application of desired operations seamlessly.

Similarly, Burghardt et al. [33] explored programming DTs using VR, where the robot reproduces human movements within a virtual environment. This approach ensures that the robot can accurately mimic complex human actions, which is particularly beneficial in tasks that are challenging to automate traditionally. The integration of Unity and Robot Operating System (ROS) platforms further enhances the functionality by providing robust communication and real-time data processing capabilities.

In the context of mobile platforms, Lotsaris et al. [31] highlighted the challenges of achieving friendly interaction in shared working environments due to the absence of effective communication between operators and robots. Their AR application addresses this by displaying safety fields and facilitating better communication, thereby improving the overall interaction experience.



**Figure 2.6:** Schematic diagram of the building and programming of a robotic station alongside VR technologies [33]

Despite these advancements, challenges such as the need for high precision in replicating human movements and the complexity of VR tracking technology persist. Improvements in algorithms, particularly those utilizing Machine Learning (Machine Learning (ML)) techniques, are essential to ensure that robotic movements closely mimic human operators' actions **Burghardt2020**.

### 3. AR-Assisted Multi-Robot Systems for Enhanced Control and Coordination

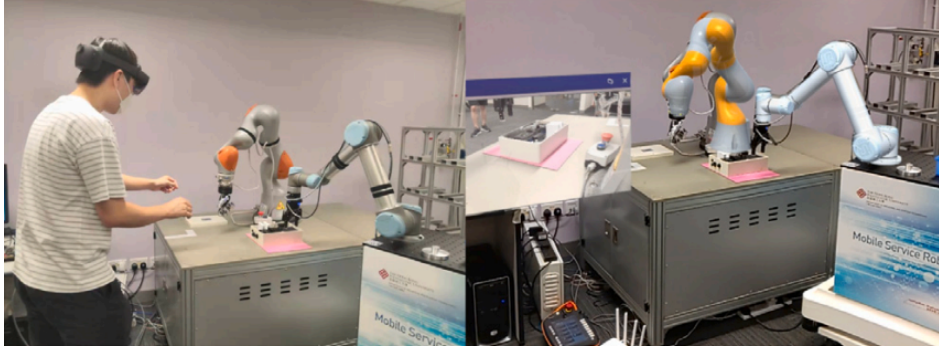
The use of AR in real-time and planned control modes for multi-robot manufacturing systems significantly improves interaction, operational safety, and efficiency. Li et al. [34] demonstrated how operators can manage and coordinate multiple robots more effectively using AR-assisted DTs. Figure 2.7 illustrates the dual view where users interact with the physical setup of two collaborating robots and view their virtual counterparts via Microsoft HoloLens AR glasses. This immersive and intuitive control experience allows for real-time simulation and monitoring of manufacturing processes and robot operations.

Additionally, Ong et al. [35] presented an AR-assisted robot programming system for welding applications. Their user-friendly interface simplifies and accelerates the programming process by allowing users to define welding points and orientations using a handheld pointer. The AR interface enables users to validate programmed tasks within

the real robot workspace, enhancing both accuracy and efficiency.

Further advancements include the use of smart glasses and smartphones for robot manipulation. Malí et al. [36] developed an AR application that allows users to adjust robot axis values, highlight specific robot points with 3D arrows, navigate to invisible robot points using leading lines, and provide instructions via text and touch interactions. This system was evaluated in a production cell with an industrial robot, demonstrating improved usability and interaction capabilities.

Puljiz et al. [37], [38] explored various AR-based methods for robotic arm programming using devices like Microsoft HoloLens. Their approaches include hand-guided task programming, augmented trajectories, and the creation of spatial maps for waypoint placement and virtual trajectory execution. These methods enable intuitive and precise programming of robotic arms, facilitating seamless integration of virtual instructions with real-world robot operations.



**Figure 2.7:** AR-assisted DT-enabled multi-robot collaborative manufacturing system [34]

Despite the promising advancements, issues such as DT model accuracy and network latency continue to affect system performance. Future work must focus on enhancing the fidelity and responsiveness of AR-assisted DT systems to fully realize their potential in industrial applications [34].

## 2.6 SUMMARY

Therefore, the proposed approach aligns with recent research emphasizing the potential of XR to enhance interaction, safety, and efficiency in HRC:

- **Interconnection via Robotic Arm:** Leveraging the UR10e, it facilitates a dynamic bridge between distributed members, offering a tangible link for both on-site and remote collaboration.
- **Enhanced On-site and Remote Interaction:** Utilizing Handheld Device (HHD) and potentially more immersive tools like the HoloLens 2 for on-site participants, and a 2D interface evolving towards VR technologies like Oculus Quest for remote participants, it proposes significant advancements in how collaborators perceive and interact with the shared environment.
- **Technological Advancements for Collaboration:**

- **Camera Integration on the Robot:** This automation improves environmental sharing, allowing on-site participants to focus more on their tasks.
- **Efficient Communication:** Enhancing information sharing through voice and annotations to streamline collaboration.



# CHAPTER 3

## Methodology

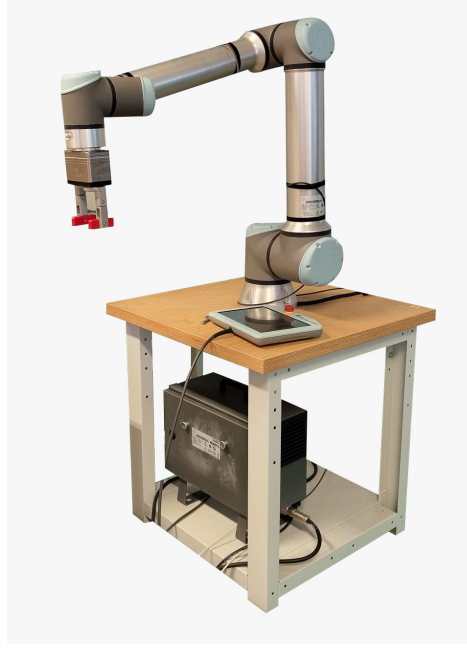
*verificar depois qual nome dar e o que por?*

# Augmented Reality on On-site Collaboration

*In this chapter, the first part implementation will be explained thoroughly.*

Initially, Unity software was used in order to develop an application that enabled the on-site member to interact with the robot by utilizing virtual and augmented reality to its favor.

In order to start addressing the mentioned challenges, a first effort has been made. A robotic arm, UR10e, from Universal Robots, available at IRIS LAB 4.1, was used as a dynamic agent to assist in shared activities.

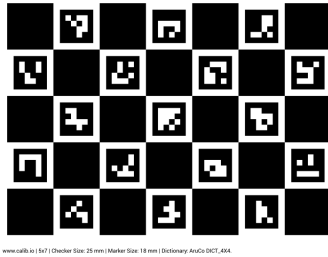


**Figure 4.1:** Robot UR10e used for the development of the dissertation work, available at IRIS-LAB, University of Aveiro

#### 4.1 MARKER DETECTION

Vuforia is a software platform that enables the creation of AR experiences. of AR into mobile and digital apps. It uses computer vision technologies to recognize and track images and objects in the real world, allowing developers to overlay digital content precisely.

The marker illustrated in Figure 4.2a was selected following initial attempts that yielded inconsistent results when using the laptop's camera, shown in the figure 4.2b, to scan the environment. This particular marker demonstrated greater stability, enabling the precise positioning of the digital UR10 model in alignment with the physical surroundings. Consequently, this facilitated the accurate overlay of the digital UR10 model onto the actual UR10e robot, enhancing the integration of virtual and real-world elements.



(a) ArUco marker used to allow the segmentation for aligning the digital twin accordingly to the real environment



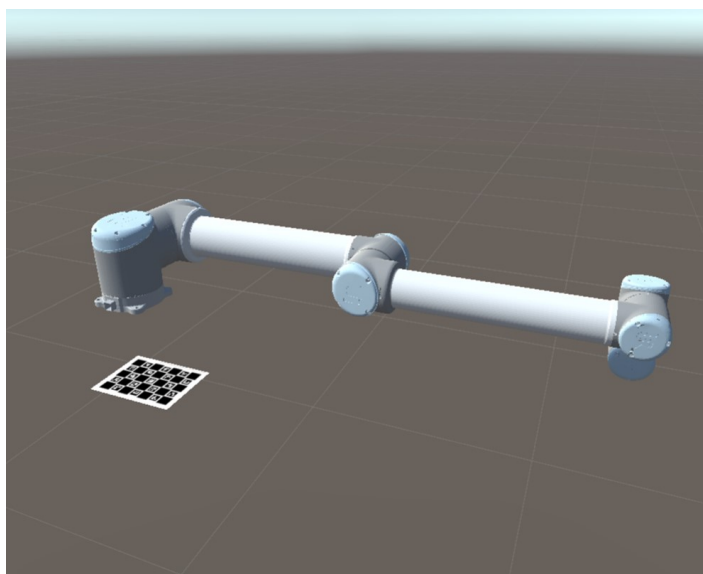
(b) Logitech c922 camera, used for testing on-site application developments

**Figure 4.2:** ArUco marker used with the Logitech c922 camera for segmentation and manipulation of virtual environment

## 4.2 DIGITAL MODEL IMPLEMENTATION OF THE ROBOT

To successfully develop the XR application for controlling the UR10e robot, it was essential to identify a model that closely mirrors the actual robot. The Unity Robotics Hub <sup>1</sup> facilitates the integration of robotics into Unity projects via the URDF-Importer package <sup>2</sup>. However, challenges arose when attempting to import the UR10e *.urdf* model into the Unity environment. To overcome this, a UR10 model, sourced from a GitHub repository <sup>3</sup> was used, due to its resemblance to the UR10e robot. This was discussed with the educators overseeing the dissertation development, and it was agreed that this approach would not pose any issues.

In the figure 4.3 it is possible to see the digital UR10 model positioned related to the aruco marker (4.2a), in a simulated Unity environment.



**Figure 4.3:** Digital UR10 model related to the aruco marker, on Unity environment

## 4.3 ON-SITE APPLICATION FEATURES

The following developed features, for enhancing the on-site environment, work while using a laptop with the camera displayed in the figure 4.2b. Despite many attempts, the usage of android handheld devices was not effective, since the digital version of the robot showed physics limitations when entering on the simulation environment.

### 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.

---

<sup>1</sup>Unity Technologies <https://github.com/Unity-Technologies/Unity-Robotics-Hub> Accessed: 2024-02-02

<sup>2</sup>Unity Technologies <https://github.com/Unity-Technologies/URDF-Importer> Accessed: 2024-02-02

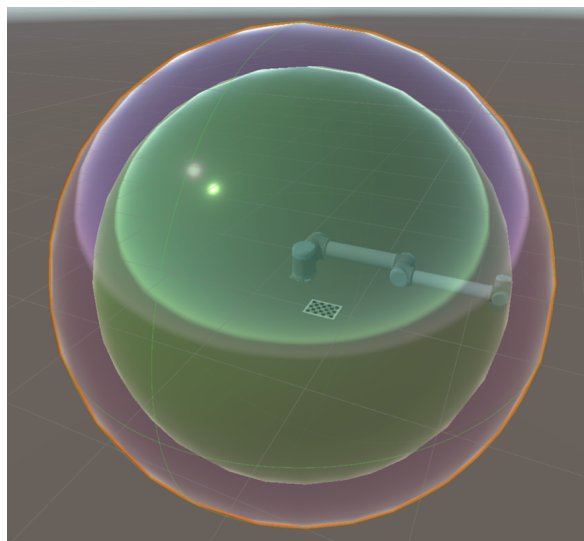
<sup>3</sup>PositronicsLab [https://github.com/PositronicsLab/reveal\\_packages/tree/master/industrial\\_arm/scenario/models/urdf/ur10](https://github.com/PositronicsLab/reveal_packages/tree/master/industrial_arm/scenario/models/urdf/ur10) Accessed: 2024-02-05

Two safety zones were developed, as shown in figure 4.4, 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 4.4:** Simulated environment showing the display of both the inner and outer safety zones, as well as the robot and the marker inside them

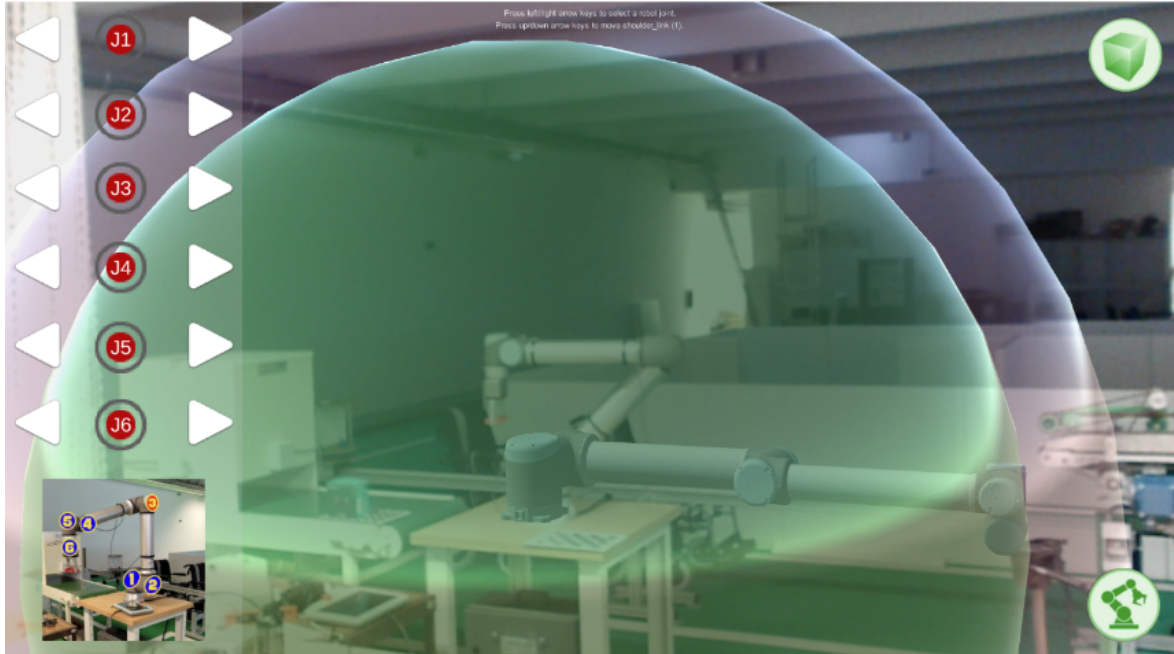
#### *Safety Zone Breach Protocol*

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 4.5. 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 4.5:** Interface featuring developed interactions

### *Joint Movement*

This feature allows users to manipulate the robot's joints via a user-friendly interface, as depicted in the figure 4.6a 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 4.6, were introduced to toggle the activation and deactivation of the joint menu functionalities 4.6b and the safety zones 4.6c. 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 XR environment.



(a) Joint Control for first Joint



(b) Controller Menu Toggle



(c) Safety-zone Toggle

**Figure 4.6:** Example from first joint control menu and toggle buttons for controller menu and safety-zones

#### 4.4 VIDEO DEMO

The video demonstration (<https://www.youtube.com/watch?v=SMQ0yXhdnUo>) showcases the successful implementation of the previously described features. Testing of the application was conducted using a laptop paired with the camera depicted in the figure 4.2b. This approach was chosen due to encountered challenges in properly building the application for android HHD.

# Remote Member's Application

*The development of the remote member's application will be thoroughly explained in this chapter.*

## 5.1 OVERVIEW AND CONTEXTUALIZATION

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. Unity was chosen for its interactive capabilities, but this choice introduced additional complexity due to the need to operate across different operating systems—Windows for Unity and Ubuntu 20.04 for ROS Noetic, the version supporting the required ROS packages.

A key advantage was the existence of pre-existing resources from the IRIS Lab, where the robot is housed. Specifically, there were two GitHub repositories—`iris_sami` and `iris_ur10e`, developed by previous students and guided by Professor Eurico. These repositories provided a well-established ROS environment, enabling control of the UR10e robot through RViz, trajectory planning, and real-time execution. The `iris_ur10e` package is integral to operating the physical robot in the lab, while `iris_sami` allows for the robotic arm's manipulation.

Given this existing ROS setup on Ubuntu 20.04, the challenge was to integrate it with Unity running on Windows, requiring a connection between two laptops.

The solution was to implement a ROS-Unity bridge, which was made possible using the ROS-TCP-Connector and ROS-TCP-Endpoint packages from the Unity Robotics Hub (Unity-Technologies/ROS-TCP-Connector). Selecting this package over other ROS-Unity bridges was recommended by João Alves, one of the instructors who provided valuable insights during the development of this project.



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.

## 5.2 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.

### 5.2.1 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.

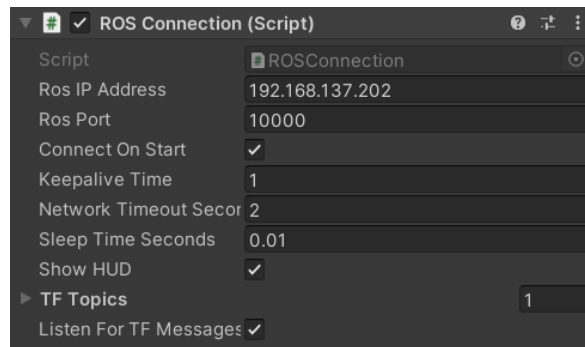
The repository also includes example scripts that correspond to each tutorial.

## 5.3 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 5.1, 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 5.1:** 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.

#### 5.4 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.

### 5.4.1 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.

### 5.4.2 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 5.2. This step was crucial for storing and manipulating the joint state data within the Unity environment.
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.



Figure 5.2: JointState message generation in Unity, corresponding to the desired ROS message

## 5.5 NEW CONTROL TYPES FOR REMOTE OPERATION

To maintain previously developed code and introduce necessary functionalities for remote operation, three distinct control types were implemented in the `Controller.cs` script, responsible for controlling the digital twin version of the robot:

1. **UIButtonControl:** Originally developed for the on-site member application, this control type features a UI interface with safety-zone functionalities. It allows operators to interact safely and efficiently with the robot. Its implementation was detailed in the previous chapter.
2. **Position Control:** This control type enables the manual control of the robot's joints through direct user input. It captures and sends the Unity digital twin's joint positions to the ROS environment upon user command, facilitating precise adjustments and real-time interaction.
3. **Joint State Subscription:** This control type continuously updates the Unity digital twin based on joint state data received from ROS. It ensures the digital twin accurately reflects the physical robot's status, automating synchronization between the Unity model and the ROS data.

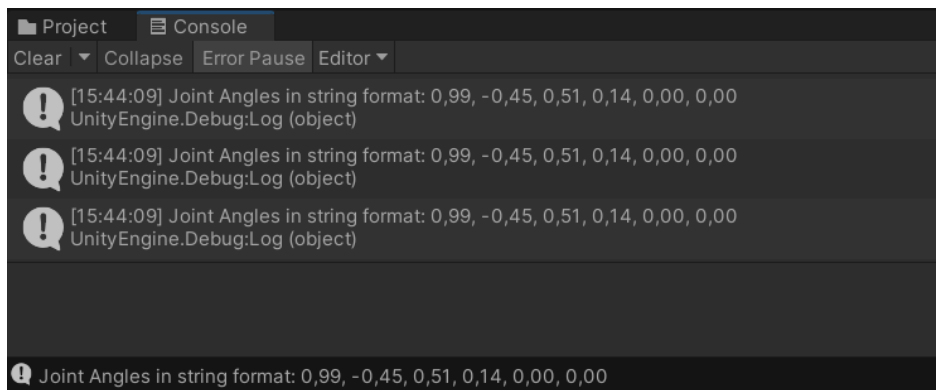
## 5.6 POSITION CONTROL - UNITY ROS

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.

### 5.6.1 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:

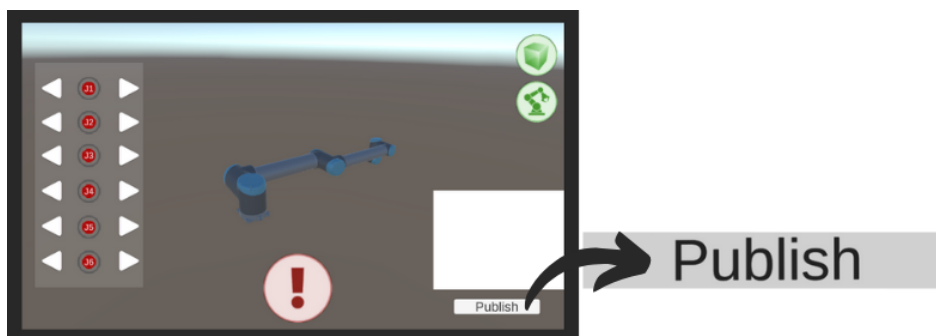
1. **Capturing Joint States:** The Unity application continuously monitors and records the positions of each joint of the digital twin robot, visible in the debug console represented in the figure 5.3
2. **Data Serialization:** The joint states are serialized into a format suitable for transmission over 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 5.4.
3. **Sending Data:** Upon pressing a UI button trigger, shown in figure 5.5, the serialized joint states are sent from Unity to the ROS environment using the previously established TCP/IP connection.



**Figure 5.3:** Unity debug console showing the current digital twin's joint positions



**Figure 5.4:** JSON format used to store Unity's digital twin joint states



**Figure 5.5:** UI Interface with Publishing button to send Unity's robot joint states into ROS the environment

### 5.6.2 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:

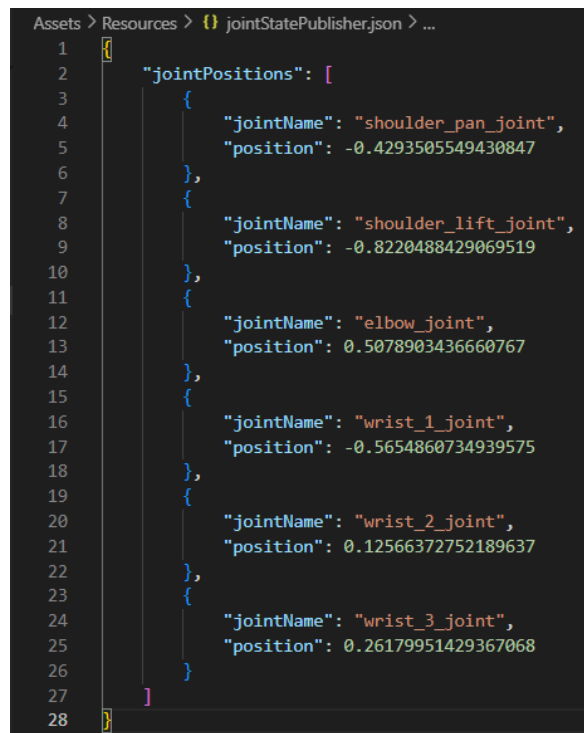
#### 1. Joint State Listener Node:

- **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 5.6 illustrates an example of data transmitted from Unity to ROS. The accompanying debug log below demonstrates its accuracy.

```
roslaunch 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.



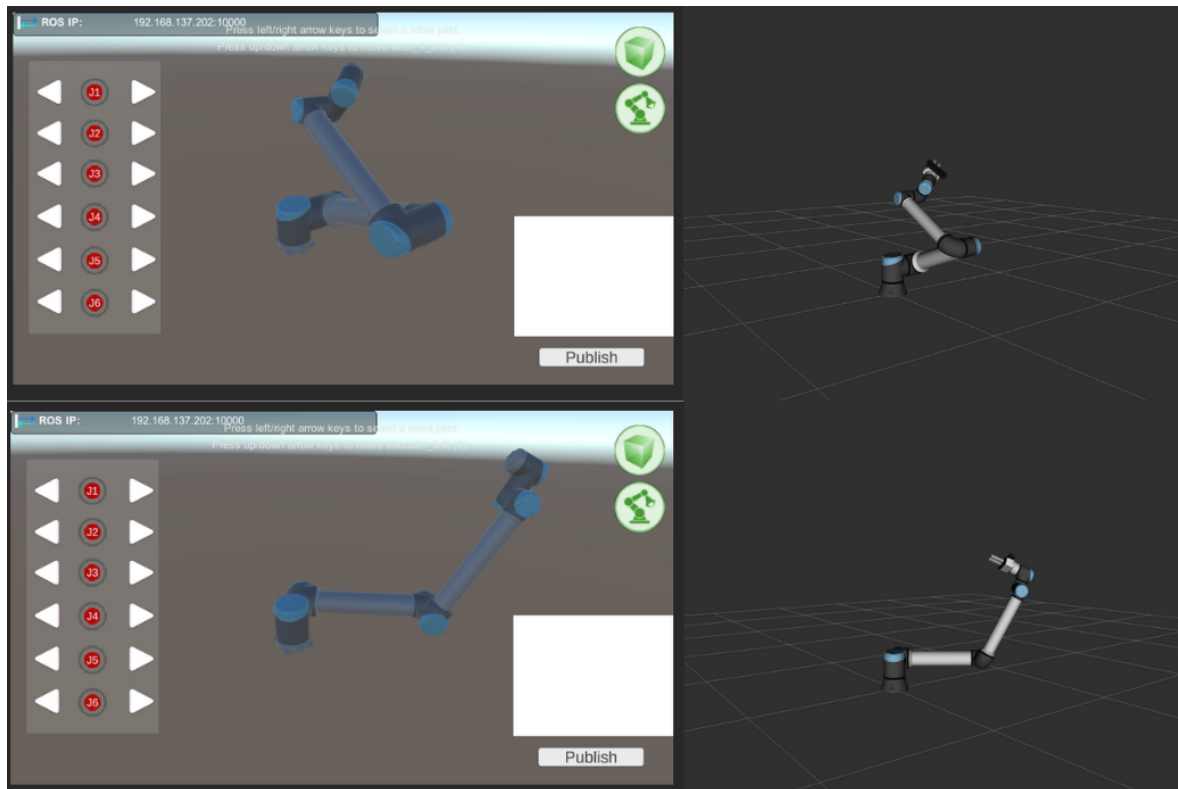
```
Assets > Resources > jointStatePublisher.json > ...
1  {
2    "jointPositions": [
3      {
4        "jointName": "shoulder_pan_joint",
5        "position": -0.4293505549430847
6      },
7      {
8        "jointName": "shoulder_lift_joint",
9        "position": -0.8220488429069519
10     },
11     {
12       "jointName": "elbow_joint",
13       "position": 0.5078903436660767
14     },
15     {
16       "jointName": "wrist_1_joint",
17       "position": -0.5654860734939575
18     },
19     {
20       "jointName": "wrist_2_joint",
21       "position": 0.12566372752189637
22     },
23     {
24       "jointName": "wrist_3_joint",
25       "position": 0.26179951429367068
26     }
27   ]
28 }
```

Figure 5.6: 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

```
roslaunch 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 5.7 displays two cases where the robot and its digital twin are properly synchronized using this control type.



**Figure 5.7:** Two scenarios showcasing the synchronization between Unity and ROS environments using Joint Control

### *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.

## 5.7 JOINT STATE SUBSCRIPTION - ROS 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.

### 5.7.1 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*

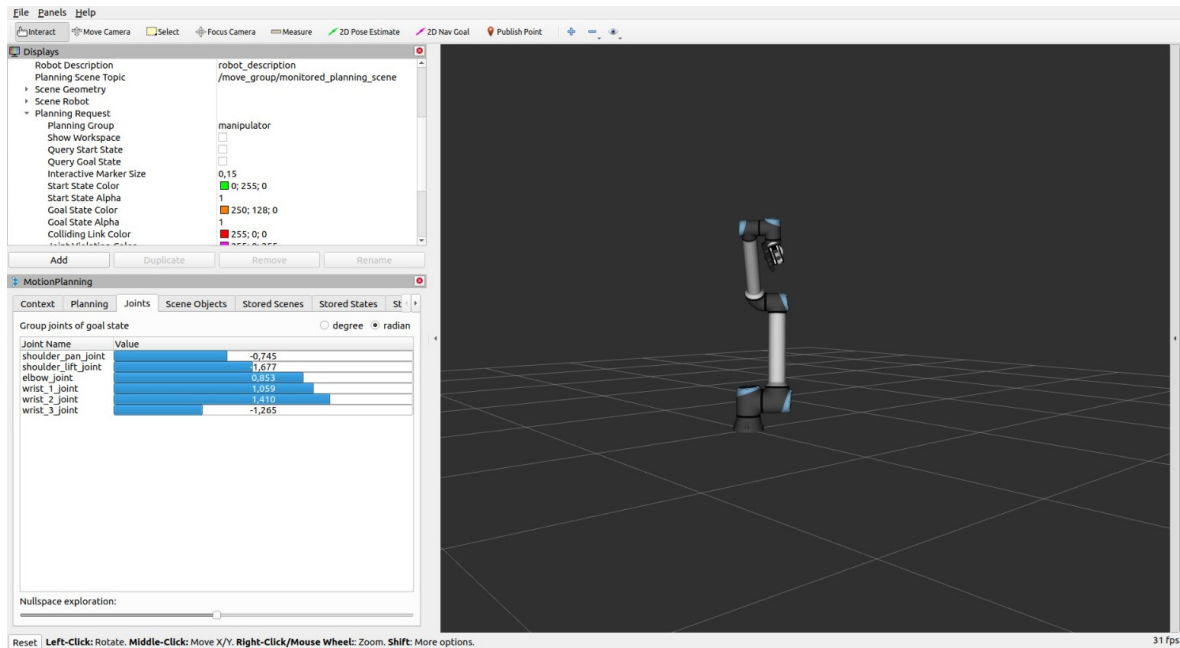
The implementation of this system ensures that Unity’s digital twin is consistently updated with the latest joint states from ROS, offering an accurate virtual representation for monitoring or interaction. Using the `SaveJointPositionsToFile()` method, joint data is structured and saved to enhance development 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 5.8. 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 5.9, ensuring that the most recent joint positions were always available.



**Figure 5.8:** Rviz environment with simulated Robot and correspondent joint values, in radians

name:

- elbow\_joint
- left\_finger\_joint
- right\_finger\_joint
- 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, 0.0001987088475458378, 0.006705746353461187, 0.00061522726960112, -0.0003380421322876756, 0.001088906936911304, 0.0002984877856849871]

effort: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]

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

```
Assets > Resources > {} jointStateSubscriber.json
1  Joint Name: elbow_joint, Position: 0.8535922
2  Joint Name: shoulder_lift_joint, Position: -1.676658
3  Joint Name: shoulder_pan_joint, Position: -0.7445105
4  Joint Name: wrist_1_joint, Position: 1.059219
5  Joint Name: wrist_2_joint, Position: 1.41033
6  Joint Name: wrist_3_joint, Position: -1.265098
7  Joint Name: left_finger_joint, Position: -5.699415E-08
8  Joint Name: right_finger_joint, Position: 5.590857E-08
9
```

**Figure 5.9:** JointStateSubscriber Json file with Robot's joint values, in radians

## 5.8 OTHER FEATURES - (CAMERA FEED TRANSMISSION) - ADD MORE

# CHAPTER 6

## Discussion and Evaluation

*something*

## Conclusion and Future Work

*something*

# 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](https://doi.org/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] 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.
- [11] R. T. Azuma, *A Survey of Augmented Reality*, 1997. [Online]. Available: <http://www.cs.unc.edu/~azumaW:>.
- [12] 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>.
- [13] Microsoft, *Mixed Reality*, <https://learn.microsoft.com/en-us/windows/mixed-reality/discover/mixed-reality>, Accessed: September 16, 2024, 2024.
- [14] M. Shafto, M. Conroy, R. Doyle, E. Glaessgen, C. Kemp, J. LeMoigne, and L. Wang, *Modeling, Simulation, Information Technology and Processing Roadmap*, May 2010.

- [15] 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.
- [16] 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/>.
- [17] 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.
- [18] 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 on Information Control Problems in Manufacturing, ISSN: 2405-8963. DOI: <https://doi.org/10.1016/j.ifacol.2015.06.141>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2405896315003808>.
- [19] 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.
- [20] 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>.
- [21] 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.
- [22] 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.
- [23] 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>.
- [24] 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>.
- [25] 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>.
- [26] K. Al-Kodmany, «Public Participation: Technology and Democracy», *Journal of Architectural Education*, vol. 53, pp. 220–228, Mar. 2006. DOI: 10.1162/104648800564635.
- [27] J. Peddie, «Augmented Reality: Where We All Live», in *Augmented Reality: Where We All Live*, New York: Springer International Publishing, 2017, pp. 1–28. DOI: 10.1007/978-3-319-54502-8.
- [28] S. Stokes, «Visual Literacy in Teaching and Learning: A Literature Perspective», *Electronic Journal for the integration of Technology in Education*, vol. 1, Nov. 2001.
- [29] D. Mourtzis, K. Vlachou, V. Zogopoulos, and F. Xanthi, «Integrated Production and Maintenance Scheduling Through Machine Monitoring and Augmented Reality: An Industry 4.0 Approach», Aug. 2017, pp. 354–362, ISBN: 978-3-319-66922-9. DOI: 10.1007/978-3-319-66923-6\_42.
- [30] S. Ong and A. Nee, *Virtual and Augmented Reality Applications in Manufacturing*. London: Springer-Verlag, 2004, pp. 1–11. DOI: 10.1007/978-1-4471-3873-0.

- [31] K. Lotsaris, N. Fousekis, S. Koukas, S. Aivaliotis, N. Kousi, G. Michalos, and S. Makris, «Augmented Reality (AR) based framework for supporting human workers in flexible manufacturing», *Procedia CIRP*, vol. 96, pp. 301–306, 2021, 8th CIRP Global Web Conference – Flexible Mass Customisation (CIRPe 2020), ISSN: 2212-8271. DOI: <https://doi.org/10.1016/j.procir.2021.01.091>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2212827121001190>.
- [32] G. Bolano, A. Roennau, R. Dillmann, and A. Groz, «Virtual Reality for Offline Programming of Robotic Applications with Online Teaching Methods», in *2020 17th International Conference on Ubiquitous Robots (UR)*, 2020, pp. 625–630. DOI: 10.1109/UR49135.2020.9144806.
- [33] A. Burghardt, D. Szybicki, P. Gierlak, K. Kurc, P. Pietruś, and R. Cygan, «Programming of Industrial Robots Using Virtual Reality and Digital Twins», *Applied Sciences*, vol. 10, no. 2, p. 486, 2020. DOI: 10.3390/app10020486. [Online]. Available: <https://doi.org/10.3390/app10020486>.
- [34] 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>.
- [35] 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>.
- [36] 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.
- [37] D. Puljiz and B. Hein, *Concepts for End-to-end Augmented Reality based Human-Robot Interaction Systems*, 2019. arXiv: 1910.04494 [cs.R0]. [Online]. Available: <https://arxiv.org/abs/1910.04494>.
- [38] 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.

# APPENDIX A

## Additional content