Master Thesis

Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Control Engineering

Virtual hand guiding of industrial robots

Annea Futko

Supervisor: Ing. Pavel Burget, Ph.D.
Field of study: Cybernetics and Robotics
May 2023

MASTER‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name:	**Futko Annea**	Personal ID number:	**498307**
Faculty / Institute:	**Faculty of Electrical Engineering**		
Department / Institute:	**Department of Control Engineering**		
Study program:	**Cybernetics and Robotics**		

II. Master’s thesis details

Master’s thesis title in English:

Virtual hand guiding of industrial robots

Master’s thesis title in Czech:

Virtuální ruční navádění průmyslových robotů

Guidelines:

The goal of this thesis is the development and implementation of virtual hand guiding for an industrial robot using augmented reality (AR), including safety zones and online connection with a physical robot.
The thesis will be conducted through a comprehensive review of existing literature, market analysis, and practical experimentation. The study will start by exploring the capabilities of AR and Microsoft Hololens, followed by an examination of industrial communication protocols and their options for communication with an industrial robotic arm (KUKA).
1. Survey the possibilities of AR, Microsoft Hololens, its capabilities, graphic engine interface and visualizations.
2. Familiarize yourself with the OPC UA communication protocols, KUKA robot control using the KRL language and implement inverse kinematics for a selected type of robot.
3. Develop and implement a virtual hand guide for an industrial robot with special focus to the selected one, including the workspace safety zones, virtual-to-reality vision feature matching, online communication, and coherent and intuitive visualization in AR.

Bibliography / sources:

[1] Burghardt, A., Szybicki, D., Gierlak, P., Kurc, K., Pietruś, P., & Cygan, R. (2020). Programming of Industrial Robots Using Virtual Reality and Digital Twins. Applied Sciences. https://doi.org/10.3390/app10020486
[2] Kuts, V., Otto, T., TÄHEMAA, T., & Bondarenko, Y. (2019). DIGITAL TWIN BASED SYNCHRONISED CONTROL AND SIMULATION OF THE INDUSTRIAL ROBOTIC CELL USING VIRTUAL REALITY. Journal of Machine Engineering, Vol. 19, No. 1, 128–145. 10.5604/01.3001.0013.0464
[3] Laaki, H., Mich, Y., & Tammi, K. (2019). Prototyping a Digital Twin for Real Time Remote Control Over Mobile Networks: Application of Remote Surgery. IEEE Access, Volume:7, 20325-20336. 10.1109/ACCESS.2019.2897018

Name and workplace of master’s thesis supervisor:

Ing. Pavel Burget, Ph.D. Testbed CIIRC

Name and workplace of second master’s thesis supervisor or consultant:

Date of master’s thesis assignment: **16.02.2023** Deadline for master's thesis submission: **26.05.2023**

Assignment valid until:
by the end of summer semester 2023/2024

Ing. Pavel Burget, Ph.D.	prof. Ing. Michael Šebek, DrSc.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the master’s thesis is an individual work. The student must produce her thesis without the assistance of others, with the exception of provided consultations. Within the master’s thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student’s signature

Acknowledgements

Coming up to this point would not be possible without the support of my dearest family whose belief in my dreams was always there. Thank you Nana, Baba & Andi.
To Hera and Vesa, thank you for standing by me and never letting me give up. Your support has meant the world to me. I also want to acknowledge the inspiring women in tech who have motivated me along this path.
Finally, I want to extend my deepest gratitude to my mentor, who has provided me with the invaluable opportunity to explore and develop my diploma topic within the Testbed environment. The work and development within the Testbed have provided me with valuable knowledge and skills, for which I am immensely thankful.

To my family, friends, and mentor, thank you from the bottom of my heart.

Annea

Declaration

I declare that the work presented here was created autonomously, and I have diligently included all references to the sources of information utilized by the prescribed guidelines for upholding ethical principles in the composition of academic dissertations.

Abstract

Implementation of the technology of Augmented Reality in industrial robots, brings a big potential in improving the flexibility of control, efficiency as well as safety in the manufacturing environments. The possibility of remotely guiding, programming, and training the new employees about these types of robots in the industry could be seen as a big advantage for the companies.

This thesis focuses on the development and implementation of a system that uses the power of AR technology, specifically the Microsoft Hololens 2, to facilitate remote guidance and control of a physical Kuka industrial robot.

The design of the virtual robot and other digital content, realized using the Unity platform, are integrated into the real world, allowing the users to visualize and manipulate a digital representation of the robot in real time. The robot's ability to move to a guided position was accomplished through the use of inverse kinematics .

Through the integration of the OPC UA communication protocol, the system ensures reliable and secure data exchange between the virtual and the physical robot. Furthermore, the project addresses safety considerations associated with remote robot guidance and control. By implementing safety measures, it is ensured that the user can interact safely with the system.

Overall, the proposed method of implementation of controlling the position of the industrial robot while hand guiding it via Augmented Reality has potential usage in industry and implementation in other industrial systems too.

Keywords: Augmented Reality, Hololens 2, OPC UA, virtual hand guide

Supervisor: Ing. Pavel Burget, Ph.D.

Abstrakt

Implementace technologie rozšířené reality do průmyslových robotů přináší velký potenciál pro zlepšení flexibility ovládání, efektivity i bezpečnosti ve výrobním prostředí. Možnost dálkového ovládání, programování a školení nových zaměstnanců o těchto typech robotů v průmyslu lze považovat za velkou výhodu pro společnosti.
Tato práce se zaměřuje na vývoj a implementaci systému, který využívá možností technologie rozšířené reality, konkrétně Microsoft Hololens 2, k usnadnění vzdáleného navádění a ovládání fyzického průmyslového robota Kuka.
Návrh virtuálního robota a další digitální obsah, realizovaný pomocí platformy Unity, jsou integrovány do reálného světa, což uživatelům umožňuje vizualizovat a manipulovat s digitální reprezentací robota v reálném čase. Schopnost robota pohybovat se do požadované polohy byla dosažena pomocí inverzní kinematiky.
Díky integraci komunikačního protokolu OPC UA systém zajišťuje spolehlivou a bezpečnou výměnu dat mezi virtuálním a fyzickým robotem.
Kromě toho se projekt zabývá bezpečnostními hledisky spojenými s dálkovým naváděním a řízením robota. Zavedením bezpečnostních opatření je zajištěna bezpečná interakce uživatele se systémem.
Celkově lze říci, že navržený způsob implementace řízení polohy průmyslového robota při jeho ručním navádění prostřednictvím rozšířené reality má potenciální využití v průmyslu a implementaci i v jiných průmyslových systémech.

Klíčová slova: Rozšířená realita, Hololens 2, OPC UA, virtuální průvodce

Contents

Figures

Tables

Chapter 1

INTRODUCTION

In this section there will be a small introduction about the main concept and purpose of this thesis. In other terms, there will be a description of the importance and the reason for this research.

The integration of robotics across various sectors has led to transformative changes and advancements in numerous industries. It started with the performance of the simple repetitive tasks but over the years with the advancements in technology, the industrial robots have become more sophisticated, being able to perform more complex tasks more efficiently and flexibly.
In recent years, with the developments in Industry 4.0, the integration of the technology of Augmented Reality into industrial settings found its way too, this way helping companies improve efficiency and productivity. [32] AR is the technology that overlays digital content into the physical, real world. In the case of industry, it is applicable in guidance and training, maintenance as well as collaboration with robots [40].
The control of robots, particularly industrial robotic arms, traditionally relies on conventional methods such as handheld controllers or programming. However, with the advent of extended reality, innovative approaches have emerged to provide immersive methods for interacting with the environment of industrial robots and controlling them. Using VR and AR allow individuals without specialized programming skills to easily program industrial robots more intuitively. This can lead to significant cost and time savings when integrating industrial robots into manufacturing companies [16].
Furthermore, Augmented Reality (AR) has increasingly emerged as a valuable tool in this regard, offering numerous benefits such as training personnel about the work of the robots, digital twining and virtually collaborating with the robots. [40]

This thesis focuses on developing and implementing a virtual hand-guiding system for industrial robots. To achieve this, we have used the capabilities of the Unity game engine to create a simulation of the virtual robot together with some other virtual content.
The real robot used for this experiment is the KR 10 R1100-2 model of the Kuka industrial robots, located in the Testbed in CIIRC, Prague.
For the communication between real and virtual robots, different studies propose different approaches. Here we have used the OPC UA communication protocol which is also commonly used in industrial automation for data exchange between devices, also being one of the most frequently used ones in industry. Additionally, the Microsoft Hololens 2 glasses were employed to represent the virtual robot in the real world.
From the implementation of the chosen method, the results demonstrate the successful integration of AR technology with an industrial robot, enabling intuitive control through hand guiding while taking into account the safety measures necessary for real-world industrial settings.

Chapter 2

LITERATURE REVIEW

2.1 Related Work

Augmented Reality is the technology that gives its users the visualization of digital content in the real world. This digital content consists of computer-generated 3D objects which are immersed in the real-world scene. [2]
This technology is being used by developers to also create Digital Twins – virtual representation of the physical systems the usage of which can be for a variety of reasons. They can be created to simulate the functionality design and analysis of one system. In the spectrum of industrial robotics, digital twins have been playing a significant role too. They facilitate the simulation and optimization of robotic arm performance before their deployment in real-world scenarios, thus minimizing potential risks and enhancing efficiency. Additionally, digital twins offer real-time monitoring capabilities, allowing for continuous performance evaluation of robotic arms and the ability to predict maintenance needs, ultimately ensuring uninterrupted operations. [9]
Several works and research have been made to create Digital Twins which could be used in either Augmented Reality but also in the Technology of Virtual Reality too. For example in [17] they developed a Digital Twin system for the MotomanGP8Controller industrial robot to be used as a synchronization model of real and virtual industrial robots by using the technology of Virtual Reality. The built system which is based on the Unity game Engine, scripts in C# for controlling both robots and a local network, facilitated by an API provided by the robot manufacturer for communication between the real and virtual environments. A remote guide for the robot to perform a task was made possible by the [19]. In this paper, it was made possible to control a

physical robot using Hololens 1 and Leap Motion controller, a hand motion tracking device system which by tracking the user's finger coordinates and position of the palms [19].
These values with the proposed method in the paper were converted to the robot's Cartesian coordinates. Successfully the robot was able to write a simple Chinese character guided by the user.
Another point of view is presented by the [16] where the application device created would be used in mobile devices. This way the users could simulate the movements of the robot without the need to have proficient programming skills. Their approach is toward making the programming of industrial robots easier, in a more intuitive way without the need for highly qualified programmers. The implementation allows the scanning of Vuforia image targets for the appearance of the virtual robot in the scene and the modification of the joint position and angles using the virtual user interface. In the video link provided in the paper could be seen that the scene created is close to a virtual teaching pendant, allowing one to select speed, save points and modify the value of the joints.
The utilization of Augmented Reality in industrial robotics holds significant promise in guiding the robots into positions, simplifying programming tasks, and optimizing performances. However, despite the interest in applying VR and AR technology for industrial systems and applications, there can also be challenges and limitations. One significant challenge mentioned in the [17] is the lack of developers and professionals who possess the necessary expertise in working with VR technology. The authors point out that creating ready-to-go projects in this domain requires extensive effort, including tasks such as 3D object modeling, rigorous testing, time constraints, and the selection of suitable hardware. These challenges are not exclusive to the development of AR projects but also extend to Mixed Reality endeavors.

2.2 Market Analysis

By offering immersive experiences Augmented Reality is being found to be used in wide areas such as education, training, industry, entertainment as well as healthcare, and day by day the numbers are increasing.
Augmented Reality (AR) is a technology that superimposes digital information such as images, videos, or sounds in the real world. It has been used in various industries such as retail, healthcare, education, manufacturing, and others.
According to Fortune Business Insights, the global Augmented Reality market size was valued at USD 5.92 billion in 2020 and is expected to reach USD 97.76 billion by 2028. [13]
Another reason for the popularity of the usage of this technology is the

COVID 19 pandemic which resulted in restrictions on most face-to-face interactions. In the retail industry, AR applications have been on the rise with the pandemic accelerating the shift to digital shopping by roughly five years according to IBM's 2020 U.S. Retail Index report.[13]
With the advent of the Industry 4.0 revolution, numerous advancements and technologies have emerged in the market. And among these, Augmented Reality (AR) stands out as a prominent addition. The integration of AR in various industrial sectors has brought about significant changes and progress. A comprehensive study conducted by [40] explored the applications and categorization of AR in the industry. The research identified several key areas where AR has been successfully implemented in industrial settings, namely maintenance, assembly, human-robot collaboration, manufacturing, training, and logistics. These areas were found to encompass a substantial portion of the analyzed papers, showcasing the diverse range of AR applications within the realm of Industry 4.0.
By examining the graphical representation provided in this research, it becomes evident that maintenance holds the largest percentage of AR utilization.

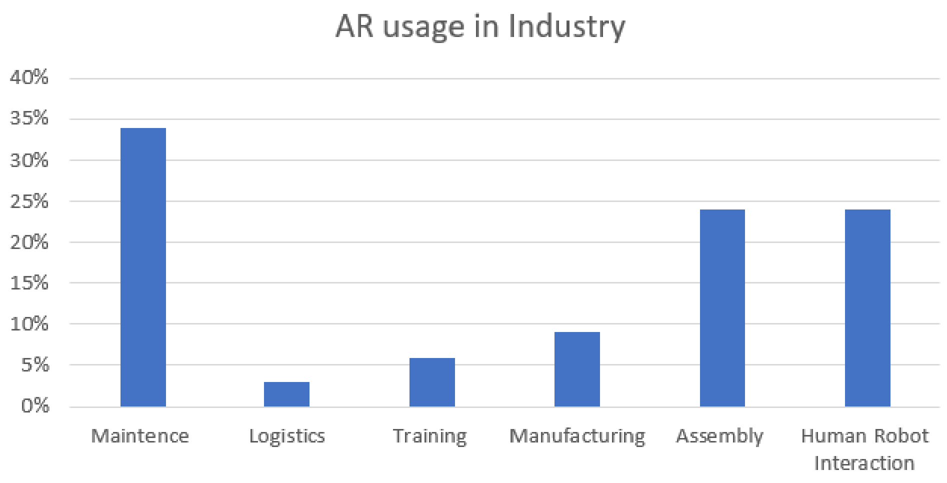


Figure 2.1: Augmented Reality in Industry[40]

From a managerial standpoint, augmented reality proves to be a valuable tool in marketing as well. Before embarking on the application or implementation of AR in different industries, it is imperative to address several key considerations. These include defining the purpose or objective of AR implementation within the company, creating digital content tailored for AR, selecting compatible devices, assembling a proficient development team, and other crucial factors. These preliminary steps must be carefully thought out prior to commencing the application or implementation process.

Looking from a business perspective, it is very useful in marketing the

product as well as saving money by reducing the need for a physical prototype of the product.
In a study conducted by Harvard Business Review, it is explained how customers enjoy shopping which includes AR experiences. For example, 56% of shoppers surveyed by NielsenIQ said that AR gives them more confidence about the quality of a product, and 61% said they prefer to shop with retailers that offer these experiences to them. [11]
This interest of customers is reflected in companies like Microsoft, Google, HTC Hive, Apple and which develop new hardware and software products of this technology.
The implementation part of this work focuses on the AR device offered by Microsoft. HoloLens 2 is a mixed-reality headset that can be used in manufacturing to provide a step-by-step guide to accelerate learning, standardize processes and reduce errors. According to HoloLens 2 TEI study, manufacturers reduced training time by 75 percent, at an average savings of $30 per labor hour. Employees can quickly learn complex tasks and collaborate at the moment from anywhere. [26]
On the official Hololens website, numerous collaborations and companies across various industries have embraced this advanced technology and equipment. Notably, automotive companies have been particularly inclined towards utilizing Hololens for maintenance and training purposes.

A compelling example can be seen in an article published by Toyota car company [27]. The article highlights Toyota's implementation of Hololens 2 specifically for training the employees taking into consideration that the company has a big number of employees.
According to Toyota, the introduction of HoloLens 2 has resulted in a significant boost in training efficiency. Trainers can now supervise multiple trainees concurrently, effectively doubling the training capacity. The hands-free and interactive nature of HoloLens 2 empowers trainees to repeat training sessions as many times as necessary, reinforcing their learning and muscle memory. Furthermore, HoloLens 2 offers invaluable guidance for equipment repairs and planning of the new ones.
According to a press release from Skoda, Storyboard [33], Skoda Auto is testing an augmented reality application known as HoloLens glasses to assist with production line maintenance. The article does not mention if Skoda is using Hololens for any other purpose.

An AR product that uses Hololens is offered by Pocket Virtuality, which we had the opportunity to use and test it. Pocket Virtuality is a tech company based in the Czech Republic that offers AR solutions for businesses and organizations. Their flagship product is called Fata Morgana, which aims to help with training, remote assistance, navigation, and inspection.[39] To achieve this, they use Microsoft's Hololens 2 glasses for the hardware and have created the Fata Morgana Studio software that runs on the Windows 10

operating system.
The Fata Morgana Studio offers an editing mode that allows users to create scenes and scripts for their AR experiences. It also includes the Guided Operation and Guided Inspection features, which make it easier for users to navigate and inspect real-world objects using AR overlays.

Testing the product in the lab, Fata Morgana Studio enables users to create scenes and scripts for AR experiences. It includes Guided Operation and Guided Inspection features for easier navigation and object inspection using AR overlays. In our lab tests, Fata Morgana proved to be a powerful tool for immersive training and guidance with the Hololens 2 headset. Users can connect the headset to their PC and interact with the software's features through a dedicated app. However, limitations include the inability to create complex scenes or import custom objects, limiting its use in certain industries. Despite this, leveraging technology like the Hololens 2 holds great promise for enhancing AR in various industries.
While the application of the technology has a lot of benefits, the [11] potentiates challenges in the adoption of this technology. Firstly, depending on the industry of operating and the technology used, it can face high costs of implementation as well as the human sources for building and implementing the technology because the lack of talent and expertise inside of the company must lead to search buying the app from third-party app builders.
Despite these challenges, the potential of AR in revolutionizing various industries remains significant. As technology continues to advance, addressing the implementation considerations and leveraging AR's capabilities can lead to enhanced customer experiences, increased efficiency, and improved outcomes for businesses across different sectors.

Chapter 3

THEORETICAL FRAMEWORK

In this chapter, we present a theoretical explanation of the core concepts used in this thesis, which lays the foundation for the subsequent analysis and discussion in the following chapters. Our discussion covers key theoretical concepts such as industrial robots, communication protocols, extended reality, and other related technologies, enabling readers to gain a deeper understanding of the topic at hand.

3.1 Industrial Robotics

Gurjeet Singh and V.K. Banga in their paper describe that there are many different types of robots and that they can be classified either by their mechanism, degrees of freedom, type of joints, workspace they operate in, or even the power source that they have[31]
Let's take a look at how the robotic arm or the robot manipulators are classified. These industrial robots vary in different configurations, however, in the big picture, we can say that they can be grouped in robots with Serial Kinematics and Parallel Kinematics. These two groups differ from each other by the configuration of the joints, degrees of freedom, etc .[31]. The joint types of these robots are either Revolute (R) Prismatic (P), Spherical (S), Screw(H), Planar(P), etc.
Robots can be defined by their degrees of freedom (DOF) too. This term refers to the number of independent ways a robot can articulate. The sum total of the independent displacement link is always equal to the number of DOF. It is also well known that 6 DOF is fundamental in executing any task

[H]

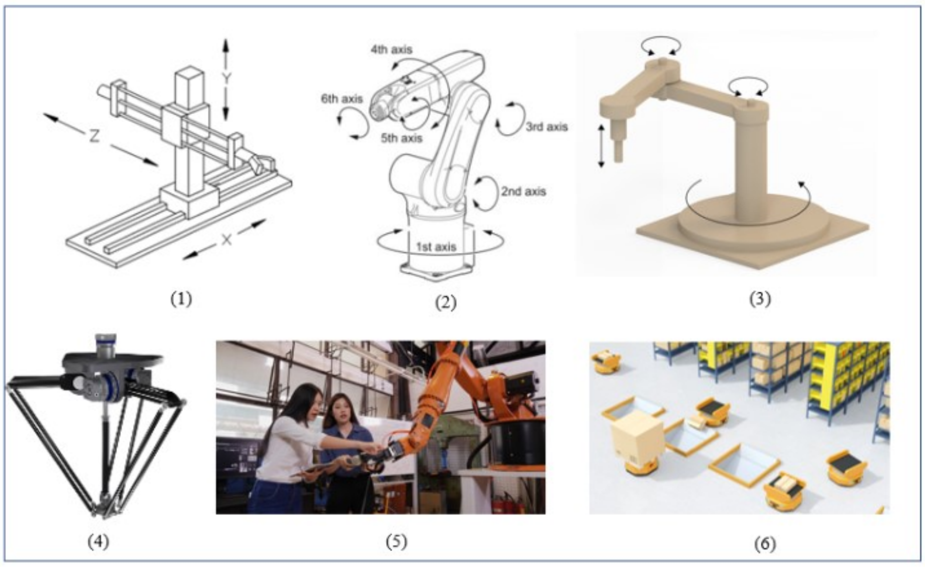


Figure 3.1: Different types of industrial robots[6]

in 3-D [31]. Different types of joints have different numbers of DOF which leads to having different robotic arms depending on their design and joints. The classification of the robots according to their workspace can be found below:

- Cartesian Robot (1)
- Articulated Robotic Arms (2)
- SCARA Robots (3)
- Mixed Reality Toolkit 2.8.2
- Delta Robots (4)
- Collaborative Robots - Cobots) (5)
- Autonomous Robots (6)

It is worth noting that the big technological impact was also the introduction of collaborative robots in the market. For executing different tasks in production and manufacturing these types of robots have started being used. The collaborative robots execute tasks alongside humans while working together in the same workspace with flexibility and safety.
Working with these types of robots it's pretty easy, the programming is not complicated and it is easier to move them from one place to another[30].

As trendy and applicable as it is in other technologies, Extended Reality and AI have had an impact on industrial robots too. The application of Artificial Intelligence allows higher precision and quality, safety, and when we collect them all, it helps to create better automation processes. While with extended reality different training, simulations, design, and planning of the industrial robots and processes are done.

3.1.1 Articulated Robotic Arms

Since in this thesis, the focus is on a specific type of robotic arm that belongs to the type of articulated robotic arms, we will be mentioning some information about them.
These types of robots are the most commonly used ones in industry and manufacturing, have a serial kinematic structure, and usually are made of six degrees of freedom which means that the robot is able to move in three translation directions and three rotational directions.
Referring to Figure 3.1 as it can be seen, the structure consists of 6 Joints that connect together six links. Links are the rigid component that connects the joints, whilst the joints are the connection point between two links allowing them to move.
In the lower part of the robot, we can see the base of it, with which also the first axis is connected allowing the rotation in motion. The axes are connected/supported with one another in a chain and this structure allows the robot to imitate the movements of the human arm. This could also be seen from the design of the robot where axis A2 and A3 form the arm of the robot and the other three axes form the wrist of the robot.
At the end of the robot after the sixth axis is the Tool Center Point (TCP) which is the very last edge of the robot. A 3D point in space, helping to position the End Effector. The End Effector is the last part of the robot where usually different tools are attached, depending on the task that the robot has to perform[3].
The robot and its joints both have reference frames also known as coordinate systems. The frames are attached to the links of the robots and consist of the origin point and orthogonal axes (x,y,z) to describe the position and orientation of the links.
Now let's see how we can move these articulated robotic arms. Kinematics deals with the motion of the body; this plays a big role in the robotic case, allowing the design and control of the movements of the robot.
Robot Kinematics is divided into Forward Kinematics and Inverse Kinematics.

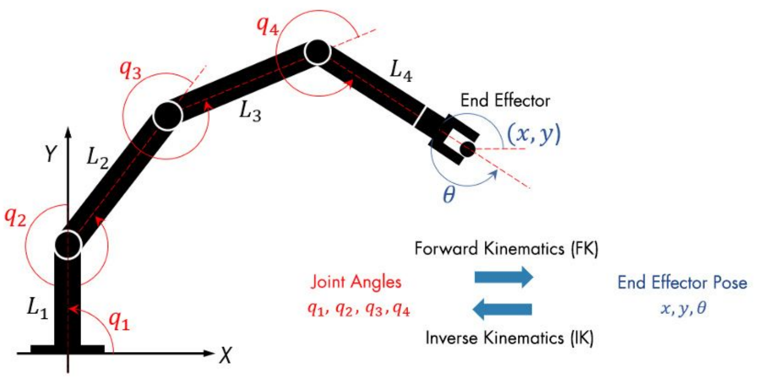


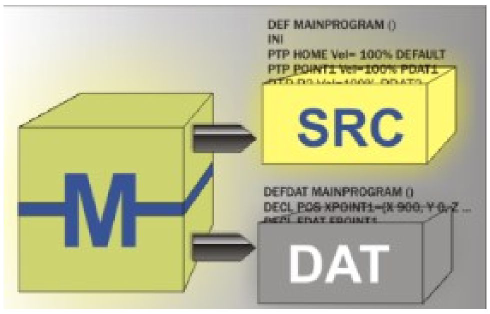
Figure 3.2: Kinematics of Robotic Arms[20]

Forward Kinematics-This method helps to understand the position and orientation of the end-effector if we have all the values of the joints of the robot. So if we have the robot for example in an initial position and we know all the angle values we can determine the location of its end effector[1].
In practical applications, the forward kinematics problem is typically addressed by determining the transformation between a reference frame attached to the end-effector and another reference frame attached to the base or workstation.
This process involves calculating the position and orientation of the end-effector relative to the base. For a serial chain manipulator, solving the forward kinematics is straightforward as it involves concatenating the transformations between frames attached to adjacent links of the chain[1].

For our implementation we need to send the robot to a specific position, this means that we have to change the values of its joints. For this problem, we have the Inverse Kinematics solution. It solves for the joint positions given the end-effector's position and orientation relative to the base. Since the solution of this problem gives nonlinear sets of equations, there are different types of methods used to achieve the results. In the Handbook of Robotics, these methods are classified as closed-from solutions and numerical methods, the details of which can be found in[1].

3.1.2 Kuka Robot Language- KRL

As the name suggests the Kuka Robot language (KRL) is a programming language used for programming the Kuka industrial robots. One program of KRL usually consists of an SRC file and a DAT file which together are called a Module.



```
DEF MAINPROGRAM ()
INI
PTP HOME Vel= 100% DEFAULT
PTP POINT1 Vel=100% PDAT1
```

```
DEFDAT MAINPROGRAM ()
DECL POS XPOINT1={X 900, Y 0, Z ...
```

Figure 3.3: Structure of a Module [14]

The code of the program is in the SRC file, while in the DAT file, the permanent data and point coordinates are found.
The data type is either defined by the user or is a predefined data type. For example, these predefined data types are classified as simple data types and the data types for motion programming[15]. The simple data types that are predefined are:

Predefined Simple Data types	
DATA TYPE	SYNTAX
Integer	INT
Real	REAL
Boolean	BOOL
Character	CHAR

Table 3.1: Predefined Simple Data types

The other predefined type of data as we mentioned are for motion programming and are initialized in different structures.

- STRUC AXIS A1, A2, A3, A4, A5, A6 - where A1 to A6 are the angle values for the axis-specific movement of the robot axes. In this project, we will be using this type of data too.
- STRUC E6AXIS A1, A2, A3, A4, A5, A6, E1, E2, E3, E4, E5, E6 - where A1 to A6 are angle values while E1 to E6 are angle values of external axes.
- STRUC FRAME X, Y, Z, A, B, C - X, Y, and Z are space coordinates, while A, B, and C are the orientation of the coordinate system.

- STRUC POS X, Y, Z, A, B, C, INT S, T - different from struc frame, have also the S(Status) and T(Turn) which define the axis positions unambiguously.
- STRUC E6POS X, Y, Z, A, B, C, E1, E2, E3, E4, E5, E6, INT S, T - has the values of struc pos plus the angle values of the external axes.

The Kuka Robot language has its own way of variable declaration for example in a program are declared as <DECL> Data type Name1 <, ..., NameN>, and in data lists for example are <DECL> <GLOBAL> Data type Name = Value, where Global means, globally valid. Depending on the type of the variable there are some changes and characteristics of the syntax.
Another important thing that is also worth mentioning for our project too, is motion programming and we will stop a little in the PTP. PTP executes a point-to-point motion to the endpoint and it can be specified either in Cartesian or Axis coordinates.
All this information and more can be found in the manual that is offered by Kuka which contains the operating and programming instructions for the usage of the language.

3.2 Extended Reality

Extended Reality (XR) refers to a group of technologies that provide immersive experiences by blending the real and virtual worlds. These technologies engage multiple senses, such as sight, sound, and touch, to make users feel fully immersed in simulated environments.
XR includes Virtual Reality (VR), Augmented Reality (AR), and Mixed Reality (MX), each offering a distinct level of interactivity. XR technologies offer users immersive experiences by merging the physical and digital worlds, allowing them to engage with simulated environments in ways that go beyond traditional reality.

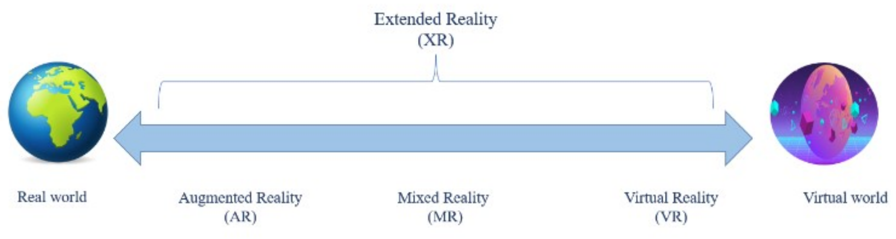


Figure 3.4: Extended Reality Spectrum

3.2.1 Virtual Reality

Virtual Reality (VR) is an immersive technology that offers its users a fully digital environment. The user enters the digital world where he/she gets the impression that it's inside of this world by using sounds, sights, and scenarios from the real world. It's a computer-generated world, providing a 360° view of it with the headset or a set of goggles and its controllers, this of course being the main advantage of it.
This type of technology has found usage in different industries and fields. Leading in gaming and entertainment industries, VR has also been applied in education for training and as teaching methods for subjects such as geography, anatomy, physics, etc.
As in [41] is explained, in healthcare is being used for training surgeons for better performance and practice, in automotive engineering for prototyping, in tourism and architecture for visiting or inspecting places virtually, and so on.
Depending on the manufacturer or the type of VR headsets the disadvantages would be either the limitations because of the mandatory connectivity with a PC, price, or computational power needed.

Figure 3.5: User and the VR experience [7]

3.2.2 Augmented Reality

In Augmented Reality the virtual information is brought into the real, physical world. The user this time is not fully inside of the digital world but has digital content such as images, videos, 3D objects, or interactive data in the real-world environment.
Using these technologies can be possible with either portable devices or AR

headsets. With devices such as smartphones and tablets, and cameras that offer a view of the real world, virtual objects can be seen inside of the real world from their screen.
One good example to understand this could be the famous Pokemon Go, a game for the smartphone or tablet where the Pokemon virtual objects are located in the real world and the player would need to catch them [37].

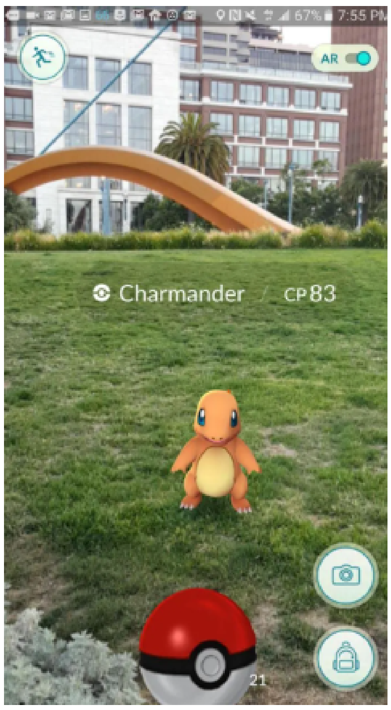


Figure 3.6: AR based Pokemon Go game [37]

On the other hand, the AR headsets give almost the same view, having all these virtual objects, and computer-generated graphics information in a frame of the physical surroundings of the user, but with better performance and for different purposes [10].
With the term smart glasses, in the market now can be found different types of glasses or headsets used for Augmented Reality. Google Glass, Magic Leap, Lenovo Electrooculography glasses, and Hololens are some of examples of them.
What makes these glasses smart are the features they possess such as eye tracking, speech recognition, tracking of the environment, etc. So depending on these features and characteristics some of them offer better applicability in AR Take as an example Google Glass- it features a small screen that sits above the user's right eye and can be used for a variety of applications, including navigation, messaging, and image recognition. But the lack of overlaying the data according to the real environment created blurred environments and acts like a head-up Display. But the device that most bring the augmented

experience to its users is Microsoft's Hololens, for which we will give a detailed overview later on [28].
Depending on the devices mentioned and the field that it is being used in, there are some different ways how we can achieve AR experiences.
For example, Marker-based AR uses predefined markers such as QR codes to identify the virtual objects or images on either head-mounted displays(HMDs)or handheld devices such as smartphones or tablets. The markerless experiences use different algorithms to track, and map the real world and augmented objects. Another example could be using a projector to project information in the real world.
Today, there are so many fields that AR is being applied to. Different industries, companies, and organizations are including augmented reality in the training methods, customer services, guiding, etc.
For example in medicine together with the Internet of Things(IoT) applications are used to monitor patients remotely, for training of new staff, visualize body parts during surgeries or diagnose pathology.
In education, the technology of AR is used to help students better understand different concepts by having a 3D view of them, separating the objects into parts, and learning from each one of them. Navigation is also made a lot more interactive with the guided directions but worth mentioning also the guided tours for museums, smart cities, and so on [10].
As it was also in section 2.2 presented, AR is becoming a strong asset in the industry too. In manufacturing, assisting the workers with AR gadgets, the errors are being reduced and the efficiency is being increased. Maintenance and repair of one machine is becoming much easier with the help of AR too. For example, Skoda Auto uses AR to increase work safety, exchange visual information no matter where the user is located, speed up the maintenance processes, etc.[33].

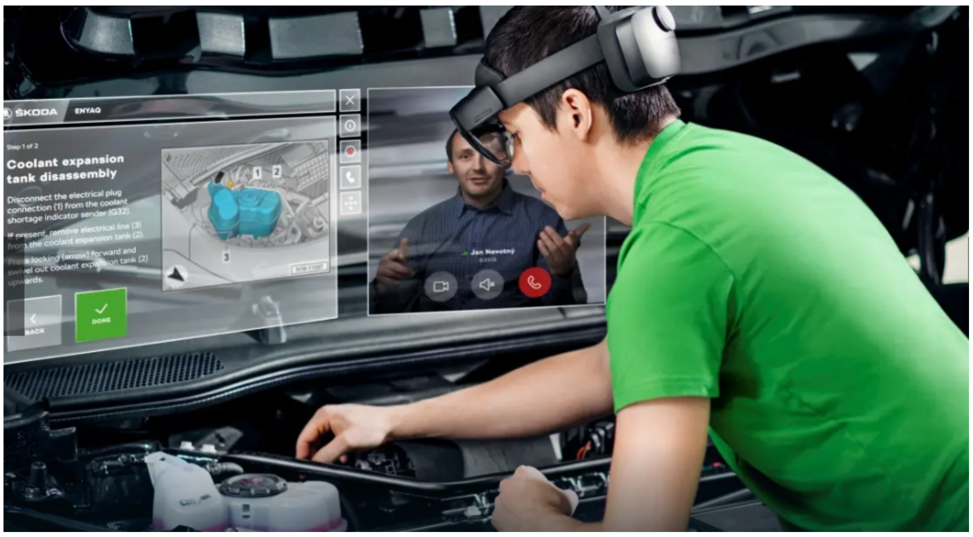


Figure 3.7: Augmented Reality in Skoda citeskoda

3.2.3 Mixed Reality

As the name also suggests, Mixed Reality is the combination of VR and AR. We can say it is a hybrid technology too because by using this technology one can experience the interaction between virtual objects in the real world. It allows the user to remain aware of the surroundings but also provides a high level of immersion.
As well as other technologies, MR too uses a combination of hardware and software. The hardware includes the headset/glasses with sensors and cameras. These components help for tracking the user's position and movement in the physical world. So it needs to understand the environment and the user for better interaction. The position of the user in the physical environment is pretty important for this technology because according to that it can position the precomputed 3D objects and holograms which will blend into the real world and interact with the user.
In order to provide these experiences the mixed reality is based on computer vision, graphical processing, and overall good performance for development.

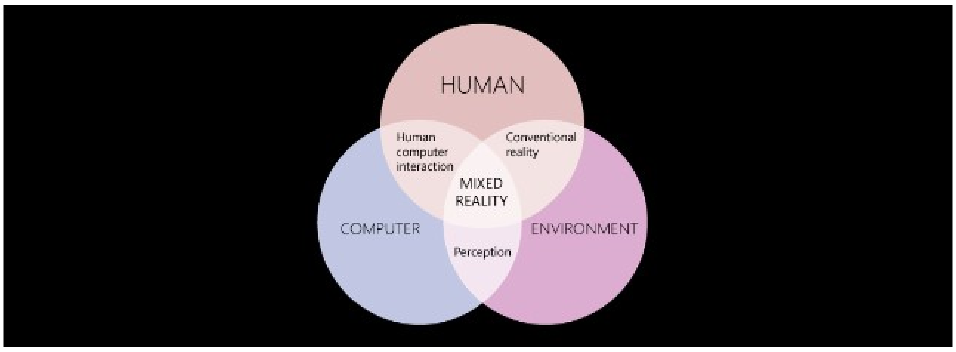


Figure 3.8: Mixed Reality Diagram [22]

As the previously mentioned technologies, the MR also finds applications in different fields and industries. It can be found used in education, medicine, entertainment industry but mostly in manufacturing and production.
In the figure below can be seen some of the frames where this technology has been used [10]. One of the most used products for creating mixed reality experiences nowadays is the Hololens 2 and the Mixed Reality Toolkit offered by Microsoft, a product that will be used in this thesis too.

3.3 Unity - 3D Development Platform

Unity is a real-time development platform that offers developers to build 2D and 3D projects. Not only a game engine but is also an Integrated Development Environment (IDE), an interface that gives developers access to all the tools indeed.
The projects made by Unity are supported in more than 25 major platforms and technologies such as Windows, MacOS, Linux, IOS, Android, etc., with this reaching the largest possible audience. It has been used in different industries and applications such as the Gaming industry, Engineering, film, animation, cinematography, Architecture, Construction, Entertainment, Manufacturing & Production, etc.
Statistics show that 70% of 1000 mobile games were made with unity and it is used in more than 190 countries and territories. [36] .This is because of the user-friendly interface it offers, where simple to major projects can be created. The engine is based on native C++ but the code is written in C# or JavaScript, and it provides a variety of features, libraries, and APIs. It will switch to alternative code editors, depending on the choice but the most common and favorable one is the Visual Studio from Microsoft.

Interface of Unity - Unity has a complex interface with different windows: Hierarchy (left) displays Gameobjects, Scene view (editing in 2D or 3D), Game view (final look), Play (simulation), Inspector (right) for adding/editing properties, and Project (lower left) for Assets, Packages, Scripts, etc.

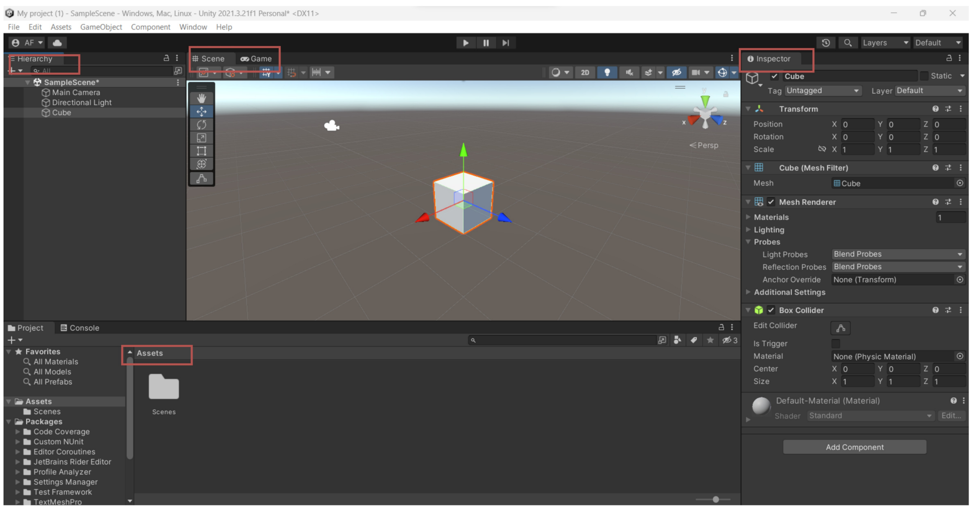


Figure 3.9: Interface of Unity Game Engine

3.3.1 Developing with Unity

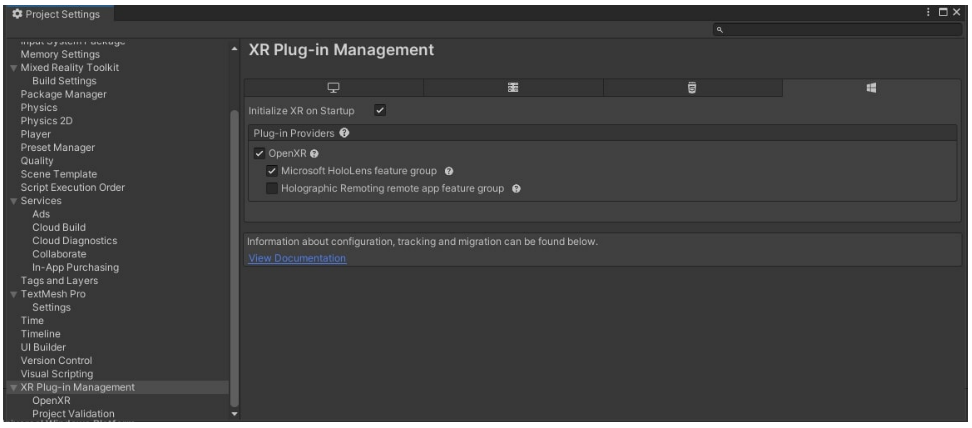
Being the top platform for game and 3D environment development, its success stems not only from being free but also from the robust tools and features it offers. It also provides an Asset store, a marketplace for purchasing a wide range of assets like 3D models, textures, audio clips, and scripts which can be used in any project[36]. Unity provides a wide array of tools for development. Every game or project requires at least one Scene, which can be as simple as a Camera and a Light. The Camera displays the game world, and multiple cameras can be used. The Light defines the environment's color, mood, and realism, with scene-specific adjustments.[35]

GameObjects in Unity - A 3D environment in Unity revolves around GameObjects, which encompass characters, objects, cameras, special effects, and lights. These GameObjects are essential for any game creation. While our focus is on 3D objects, Unity also offers 2D objects and dedicated components for them.
Unity provides basic 3D objects like spheres, cylinders, cubes, and capsules, as well as the ability to import various 3D models from different platforms using formats such as .obj, .dae, .3ds, .dxf, or .fbx.
To create a functional game in Unity, specific components are required for each GameObject. The Transform component determines the position, rotation, and scale of the object. The Renderer component adds visual details such as materials and lighting effects. Colliders are used to detect object collisions and define their shapes and Physics feature enables realistic object movements and interactions.
In addition to these essential components, Unity offers a vast array of tools and features including AI, networking, AR/VR support, optimization tools, and audio/video playback capabilities. For more information on these features, developers can refer to Unity Documentation.

As mentioned before, Unity supports C#, C++, and JavaScript, or more precisely UnityScript - a version of JavaScript. Scripting enables developers to create custom behaviors, object interactions, game logic, AI, user interfaces, shaders, and visual effects—essential for crafting unique and immersive game worlds. Unity offers various tools and features for scripting, including variables, data types, a visual scripting editor, debugging capabilities, script modification options, methods, and numerous libraries. Additionally, Unity allows for the inclusion of external code through plugins, extending project functionality and enabling integration with third-party libraries and services[34].
These integrated plug-ins have benefits like extending the functionalities of the project and integration with third-party libraries and services and this is helpful to our project too, for inserting. The plugin feature in Unity has greatly assisted us in implementing our project, particularly in establishing

the necessary communication protocols, which we will discuss later. By utilizing these plugins, we have gained access to libraries and functionalities that Unity does not offer by default, thereby allowing us to integrate them into our project seamlessly.

Unity and Extended Reality - Unity's support for Extended Reality (XR) development is a major advantage, making it the preferred choice for developers creating immersive experiences. The Unity XR plug-in framework, known as XR SDK, enhances Unity's capabilities on platforms like ARKit, ARCore, Windows Mixed Reality, Microsoft HoloLens, Oculus, OpenXR, and Vuforia Engine.
The XR SDK provides a range of features for AR and VR development, including spatial mapping, object recognition, input management for hand-tracking devices, UI interactions, and rendering capabilities. With these features, developers can create AR and VR applications that run on various devices and platforms, from mobile phones to high-end VR headsets.
Unity's comprehensive XR support, documented in the Unity Manual[38], is the primary reason why we chose this platform for the implementation of our project. Its suitability for XR development aligns perfectly with our needs, ensuring we can leverage its powerful features to create an immersive and impactful experience.



XR Plug-in Management

Initialize XR on Startup ✓

Plug-in Providers

✓ OpenXR
✓ Microsoft HoloLens feature group
☐ Holographic Remoting remote app feature group

Information about configuration, tracking and migration can be found below.
View Documentation

Figure 3.10: Unity's XR setup for Hololens

3.4 Universal Windows Platform

To build and deploy applications into the Hololens device, the Universal Windows Platform(UWP)is needed.
UWP is a development platform created by Microsoft that allows its users to build applications that run on all Windows 10 and 11 devices[25].
It is important to have this platform installed in the Unity version that is being used to create the virtual environment but also it is important to mention that the Windows SDK needs to be installed. This is a Software Development Kit that has different tools, libraries, and other resources needed to build Windows Applications.

3.5 Hololens 2 Technology

Hololens is a technology created by Microsoft that allows its users to create and experience Augmented and Mixed Reality environments.
The first introduced device in 2016 was Hololens 1, and three years later the Hololens 2 was released to the public with more advanced features. The new version of the headset contains better hardware and software specifications starting from the design, performance, and tracking systems as well as more features included[29].

3.5.1 Design and Hardware

In their article, M. Doughty, N.R. Ghugre, and G.A. Wright conduct an analysis of various head-mounted displays, with a specific focus on the Hololens 2. This device has emerged as one of the most widely used and popular options for displaying virtual content, thanks to its impressive capabilities[5].
The lenses utilize advanced technology to project virtual images into the real world, creating a unique and interactive experience. Although it lacks VR capabilities, it is equipped with the ability to understand and interact with its surroundings. The lenses are comprised of three layers with distinct RGB colors, and a light engine located above the lenses projects light into the headset, blending the holograms with the real world seamlessly.

Hololens runs on Windows Holographic Operational System which is based on Windows 10. The device offers a Qualcomm Snapdragon 850 Compute Platform which provides powerful and efficient processing capabilities. Its 4GB of RAM and custom-built Microsoft Holographic Processing Unit work together to create an intuitive and immersive user experience.
The feeling of the experience is supported by a wide range of sensors too.Hololens 2 is equipped with 4 visible light cameras,2 IR cameras, an accelerometer, a gyro, and a magnetometer that allow for highly accurate tracking of eye movements and hand gestures. For audio and speech, it uses a microphone and speakers to allow the user to command and control the device.[21].

Figure 3.11: Hololens 2 Glasses [21]

Figure 3.12: Hololens 2 Hardware Components [21]

The Hololens stands out from other AR devices due to its remarkable hardware features, which collectively contribute to its unique capabilities. While there are numerous noteworthy features, we will mention just a few of them below.

- Eye Tracking - By accurately tracking and calibrating the user's gaze, this feature adapts the visual display to be aligned to their line of sight. It enhances the overall immersive experience and allows for more intuitive interaction with holographic content.
- Voice Enabled - Users can effortlessly command the device through voice commands, making it easier to navigate and control various functions. This allows users to operate the device without the need for physical input.

- Hand Tracking - Interacting with holograms by grasping, touching, moving, or pressing them. The device's sensors accurately detect and interpret hand movements, translating them into virtual interactions.

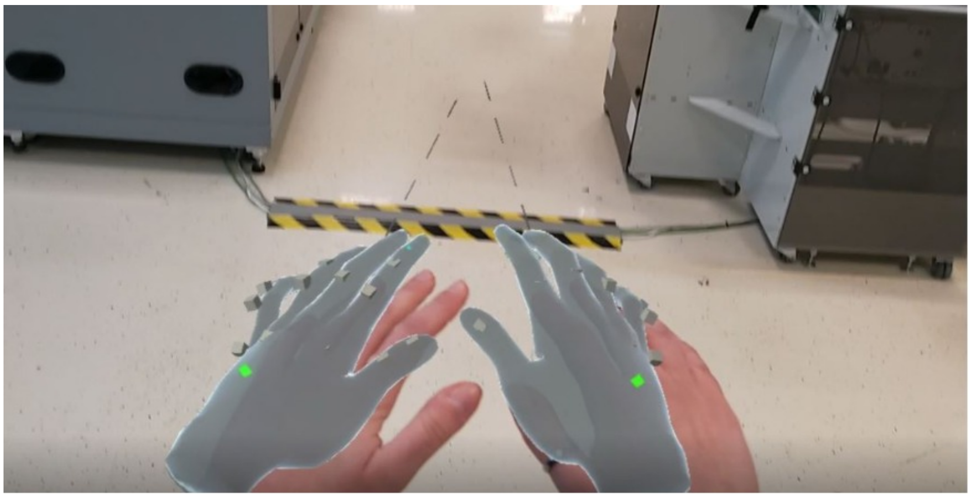


Figure 3.13: Hand Tracking of Hololens 2

- Spatial Mapping - Mapping the physical environment to better blend with the virtual contents & and a lot more. Hololens offers internet browsing and is built on Windows, making it user-friendly and compatible with various apps. It accurately tracks where the user is looking and enables the manipulation of objects through scaling and manual placement.

Comfort - The design of the glasses is comfortable and practical. They are not so heavy and are adjustable for the head. The headband makes it easier to fixate them on the head and having the computer in the back of the glasses makes it much more comfortable.The user can easily flip the lenses up and down without having to remove the entire set of glasses, making it an ideal choice for collaborative experiences.It only has one button and it's easily accessible

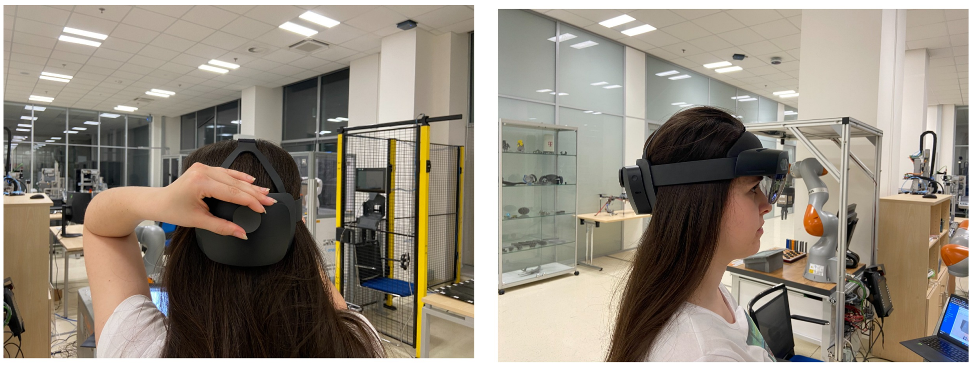


Figure 3.14: Hololens 2 and the user

Limitations - One limitation is that the holographic display of HoloLens is confined to a small area around the user, requiring them to move further to view the entire scene.
Additionally, when attempting to read something, the field of view can be obstructed, necessitating closer proximity and potentially cutting off the window. HoloLens displays are fixed at an optical distance of approximately 2.0 meters from the user, meaning users must always focus at this distance to maintain a clear image.
As an example for example the design and placement of the objects in a Unity scene can be seen in Figure 3.8 whereas in Figure 3.9 can be seen the designed digital objects placed in the real world. The cube is set to be 2m far from the camera and when looking in the direction of the cube the other objects are merely seen.

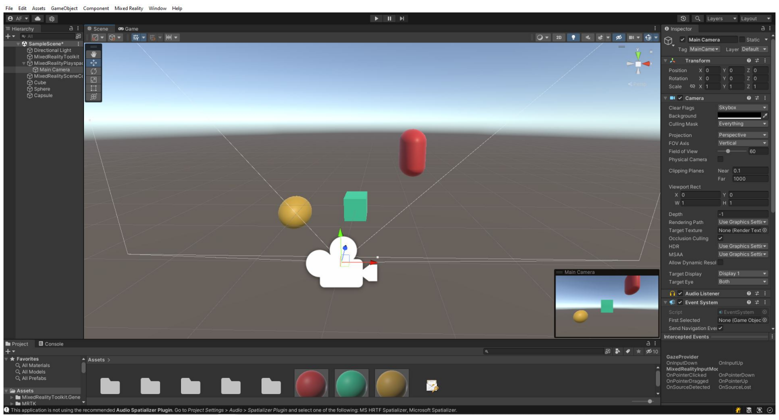


Figure 3.15: Objects placed in the scene in Unity

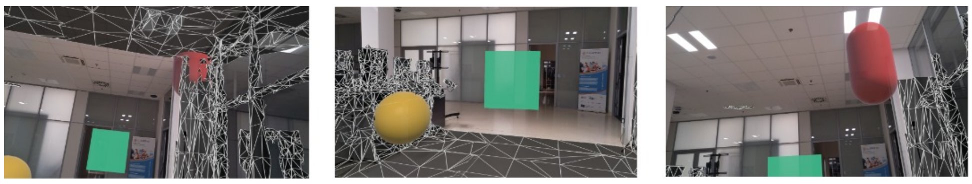


Figure 3.16: View of the Objects via Hololens

Another drawback is that typing on the keyboard is only feasible using the index finger. Furthermore, new users may encounter an issue where certain tabs remain visible in the scene even after shutting down the glasses, floating in the air where they were last used.

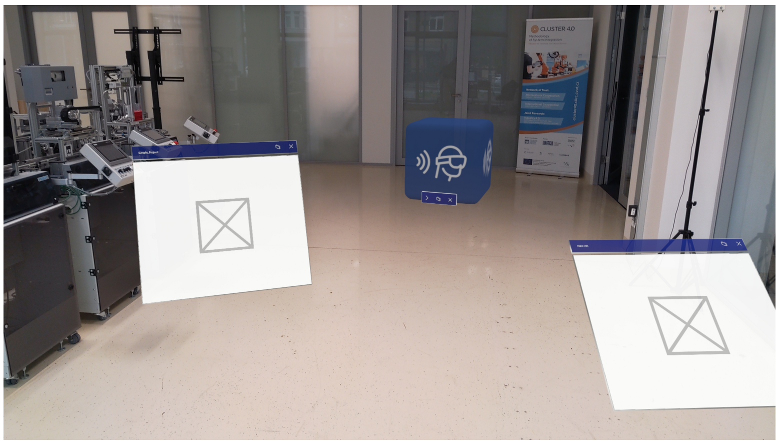


Figure 3.17: Tabs of Previous Usage of Hololens Apps

The battery life of the device is not particularly promising, especially when engaged in complex projects that require a significant amount of time. The battery drains quickly, which poses a drawback considering its intended use for training and guidance processes that are typically lengthy.
Excessive sensitivity to external light is also problematic, as it can interfere with the device's ability to accurately track the environment and effectively display holograms.
These limitations were discovered during the testing of the device in the lab. Despite these flaws, HoloLens remains a powerful tool for augmenting reality and has demonstrated great potential for a variety of applications.

3.5.2 Mixed Reality Toolkit

The Mixed Reality Toolkit (MRTK) is a cross-platform tool offered by Microsoft. It serves as an open-source software development kit (SDK) that assists developers in creating augmented reality (AR) and mixed reality (MR) experiences for various XR projects. MRTK is highly compatible with devices and platforms like Hololens, Windows Mixed Reality, and Oculus, which is why it has gained popularity among users and developers. It can be said that it comes formed by packages, consisting of Foundation, Extension, Tools, and Test Utilities - some of the most frequently used packages in development.
The key advantage of using MRTK for building XR applications is the wide range of features and components which help for a smoothie development of these experiences depending on the chosen platform. provides numerous tools for designing user interfaces in projects.

These include buttons, text windows, keyboards, sliders, tables, graphs, and scripts that allow for interaction with these elements.
Additionally, MRTK offers features such as voice recognition, various gesture controls for interacting with holograms and 3D models, as well as spatial understanding capabilities. Not only this but it also provides pre-built scripts that facilitate interaction with virtual objects, as well as gesture recognition (e.g., capture, change, click) and input handling (e.g., voice commands, gaze interaction)[23]. The architectural overview of this platform is depicted in the figure below.

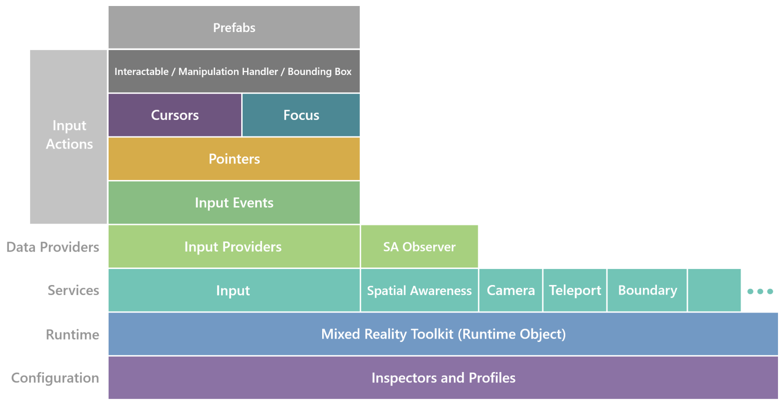


Figure 3.18: Architecture of MRTK [24]

When designing a project for Hololens using Unity, the Mixed Reality Toolkit is an essential tool to consider. By configuring the scene according to this toolkit, various types of profiles, input providers, and camera types can be easily added.

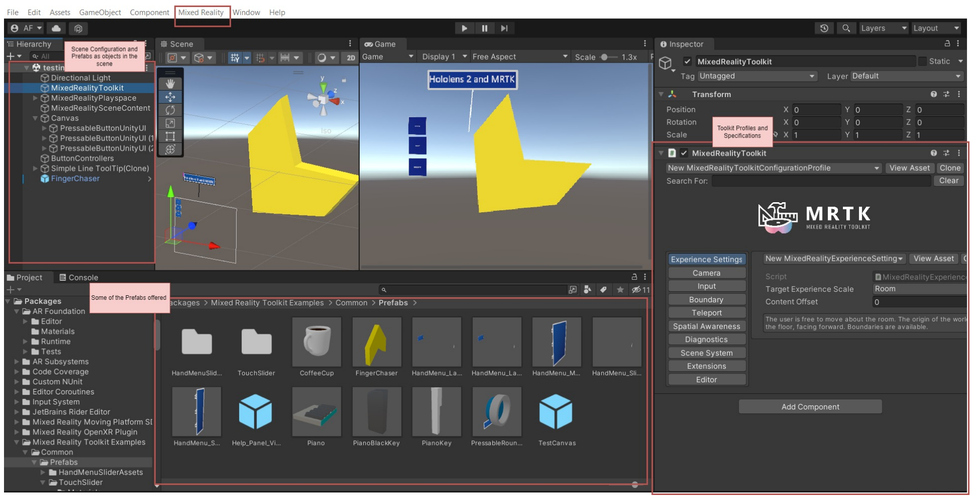

Scene Configuration and Prefabs as objects in the scene
Toolkit Profiles and Specifications
Some of the Prefabs offered
Mixed Reality
Hololens 2 and MRTK
MRTK
MixedRealityToolkit

Figure 3.19: Configured scene with MRTK

Although the toolkit is generally practical, it does have some limitations. For instance, certain features may not be available in certain versions of Unity, resulting in incomplete compatibility.

Additionally, customizing non-toolkit objects to meet specific design requirements can be challenging and the scripts might need to be modified to include namespaces and classes provided by the toolkit. Learning how to effectively use and apply the MRTK requires a significant investment of time and effort, but the benefits are worth it.
Currently, the third version of this platform can be found in the market, which every time comes with more advanced specifications. In this project, we have used the MRTK 2 and some of its offerings, which we will talk about later.

3.6 AR Framework - Vuforia

AR technology has gained widespread popularity and adoption, thanks to the extensive use and development of AR frameworks. These frameworks, which are software-based and provide APIs for developers, play a crucial role in enabling the augmented reality experience.
The primary purpose of these frameworks is to detect images and overlay virtual content onto real-world objects displayed on digital screens. Notable examples of such frameworks include Google's ARCore, Apple's ARKit, PTC's Vuforia, and Kudan. However, each framework has its own distinct features and performance characteristics. Such differences may cause differences in the function or performance of the applications developed using the AR frameworks [18]. In our study, we will specifically focus on Vuforia, a widely utilized platform for AR development that is compatible with other popular platforms.

Vuforia offers a range of features to its users, including object recognition, which allows for tracking 2D and 3D objects in the real world. With model targets, Vuforia recognizes objects based on their 3D or CAD models [12]. Additionally, Vuforia's image targets feature is particularly useful for flat images or objects. Using Vuforia, virtual or digital objects can be seamlessly placed on these objects in the real world. In our research, we have used this feature of Vuforia, the implementation of which can be found in 5.5

Figure 3.20: Vuforia image recognition feature [12]

3.7 Communication Protocols

Communication Protocols are some regulated standards that help the devices exchange information with each other. These protocols are important because they help the different types of devices communicate with each other without errors or conflicts.
They can be found in computer networks, telecommunication, and industrial automation. Depending on the application in the market can be found different types of these protocols. But what is important to pay attention to, is the performance and reliability of it.
In this thesis, we will be dealing with the industrial device which is our robotic arm so the focus from now on will be on the communication protocols which are used in the industry.
For example, some of the used protocols are such as EtherNet, MQTT, Modbus, OPC UA.

The Message Queuing Telemetry Transport (MQTT) protocol is a lightweight and efficient communication protocol that operates based on the concept of a broker. The broker receives published messages and subsequently dispatches them to subscribe clients, employing a publish/subscribe model. MQTT relies on the transmission control protocol (TCP), which is a fundamental communication protocol [4]. MQTT messages consist of a topic that clients can subscribe to. Among the popular MQTT brokers, Mosquitto is widely utilized. It supports three different levels of quality service which means giving different priorities to different users or applications. It is commonly used in IoT but also in automation and industry too.

3.7.1 OPC UA - Open Platform Communication UA

As we mentioned there are different types of communication protocols but we will stop at the OPC UA(Open Platform Communication Unified Architecture). OPC UA is an open standard that defines the connectivity, interoperability, security, and reliability of industrial devices and systems. The OPC Foundation provides a detailed guide on OPC UA on its website. What makes this type of protocol useful is its platform independence. It enables interoperability between different devices and systems. It can handle communication between individual devices, local networks, and even enterprise-wide systems. It is suitable for small and large-scale deployments and another feature of it that is important to mention is security. OPC UA provides authorization and access control mechanisms, ensuring that only authorized entities can access and manipulate the data.
It is built on a client-server architecture, where both the server and client can either read data or send data to one another.
OPC UA represents data as objects or variables that can be browsed by clients. It uses a structured information model consisting of nodes, to define the exchanged data The objects are organized in a hierarchical structure called the address space. The address space is a tree-like structure that organizes the node hierarchy and contains all the objects that can be accessed by clients. The data can be read, written, and browsed by the address to provide effective methods for better communication. [8].
It supports three different levels of quality service which means giving different priorities to different users or applications. It is commonly used in IoT but also in automation and industry too.
As we mentioned there are different types of communication protocols but we will stop at the OPC UA(Open Platform Communication Unified Architecture). OPC UA is an open standard that defines the connectivity, interoperability, security, and reliability of industrial devices and systems. The OPC Foundation provides a detailed guide on OPC UA on its website. What makes this type of protocol useful is its platform independence. It enables interoperability between different devices and systems. It can handle communication between individual devices, local networks, and even enterprise-wide systems. It is suitable for small and large-scale deployments and another feature of it that is important to mention is security. OPC UA provides authorization and access control mechanisms, ensuring that only authorized entities can access and manipulate the data.

It is built on a client-server architecture, where both the server and client can either read data or send data to one another. OPC UA represents data as objects or variables that can be browsed by clients. It uses a structured information model consisting of nodes, to define the exchanged data The objects are organized in a hierarchical structure called the address space. The address space is a tree-like structure that organizes the node hierarchy and contains all the objects that clients can access. The data can be read, written, and browsed by the address to provide effective methods for better communication [8].

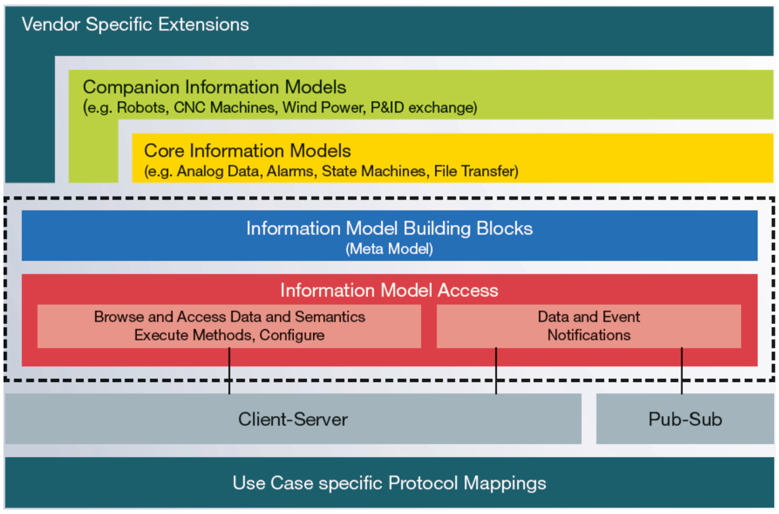

Vendor Specific Extensions
Companion Information Models
(e.g. Robots, CNC Machines, Wind Power, P&ID exchange)
Core Information Models
(e.g. Analog Data, Alarms, State Machines, File Transfer)
Information Model Building Blocks
(Meta Model)
Information Model Access
Browse and Access Data and Semantics
Execute Methods, Configure
Data and Event
Notifications
Client-Server
Pub-Sub
Use Case specific Protocol Mappings

Figure 3.21: OPC UA Information Modeling

Chapter 4

METHODOLOGY

In this chapter, we will provide an overview of the methodology chosen for the development and implementation of the practical part of the thesis. Selected hardware and software, as well as the workflow for the realization of the project, will be presented.

To facilitate the practical implementation, the project is structured into multiple distinct work stages, each encompassing its unique set of challenges and significance.
The work begins with the design and creation of the virtual content such as the digital robotic arm and other 3D objects, continuing with the motion of the robot by applying mathematical solutions to guide the robot to a specified position. Setting up client-server communication with the real and virtual world and finally visualizing and application of the project using the technology of Augmented Reality, are the key steps proposed to implement. A visual representation illustrating the flow of the process is provided below.

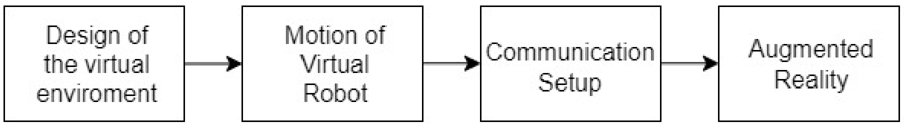


Figure 4.1: Development Process

4.1 Hardware and Software

The practical part of this project aims to develop and implement a system that utilizes Augmented Reality technology to hand-guide an industrial robot. In the preceding Theory section, we provided a general description of the technologies employed. Now, we will go into their specifications and the applications within this project.

Regarding the hardware components, we will be utilizing Microsoft's cutting-edge Hololens 2 glasses, a PC for design and programming, as well as the Kuka KR 10 R1100-2 robotic arm which is situated in the Testbed, CIIRC.

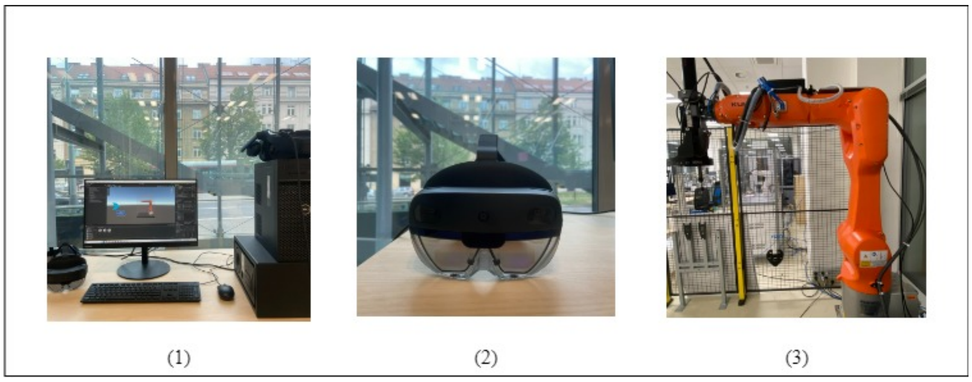


Figure 4.2: Hardware used in the implementation of the object.(1)PC,(2)Hololens 2 Glasses,(3)Kuka Robot

The decision to adopt and leverage this technology stems from its practical benefits. Hololens glasses, a leading augmented reality (AR) solution in the market, possess a range of distinctive features. Moreover, the high-performance computer we employed played a crucial role in executing the digital aspect of the project. Furthermore, the chosen model for the robotic arm is a commonly used robotic arm model within the industry.

Great attention has been given to the selection of software versions in order to ensure compatibility and smooth operation. Specifically, certain versions were carefully chosen due to compatibility issues that existed between them. For instance, certain versions of Unity lacked support or had limited functionality with the Mixed Reality toolkit, which is crucial for deploying the project on Hololens. Therefore, it was imperative to select appropriate versions that enabled seamless integration.

Below are presented the software components and their corresponding versions employed in this project:

Figure 4.3: Software of the project.

- Unity - 2021.3.16 LTS
- Visual Studio 2019
- Mixed Reality Toolkit 2.8.2
- OPC UA .NET Library Standard 1.4.367.42
- Microsoft Windows 10
- Vuforia Engine 10.15.3

4.2 Architecture of the work

In order to implement the architecture depicted in the figure below, a user is required to carry out the project. The user interacts with the designed user interface within the Hololens application to command the virtual robot to move to a specified position. Once the virtual robot reaches the new position, the angle values of the joints need to be transmitted to the real robot.
This communication and data transmission between the virtual and real robots are facilitated through OPC UA communication.

By sending the angle values, the real industrial robot is instructed to move to the corresponding position, mirroring the actions of the virtual robot accurately.

Thus, whenever the user directs the digital robot to specific positions, the same values are related to the real industrial robot.
More details about the procedure implementation will be discussed in the following section.

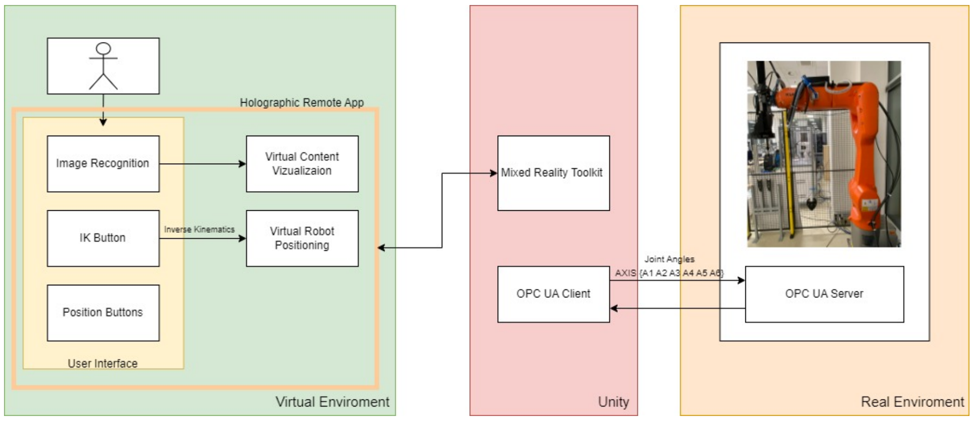


Figure 4.4: Proposed Project Architecture

Chapter 5

IMPLEMENTATION

The implementation section provides details of the project's development and execution. It presents how the project was carried out, focusing on each component of the framework that was employed.

5.1 Design of the 3D Model

During the design phase, we focus on enhancing the 3D model of the selected robot and generating a digital twin based on its physical specifications. This crucial stage involves tackling various challenges and implementing processes that enable the development of the digital robot using the powerful Unity 3D modeling engine. We make use of the extensive features offered by this platform to adapt and remodel the digital robot, ensuring its accuracy and functionality.

5.1.1 The Physical Robot

To begin designing the virtual environment and creating a digital twin, it was essential to first select a physical robot. As previously mentioned, the model of our physical robot is KUKA KR 10 R1100-2. Understanding the details of the actual robot's design was crucial for our project's advancement.

Below, we will outline several specifications and characteristics of this robot, primarily sourced from its Datasheet.
The KR 10 R1100-2 is an industrial manipulator model manufactured by Kuka, a renowned German company specializing in industrial robots and automation machinery. This particular model boasts a maximum payload capacity of 11.1kg and a reach of 1101mm. It offers the flexibility to be mounted on the floor, ceiling, or wall.

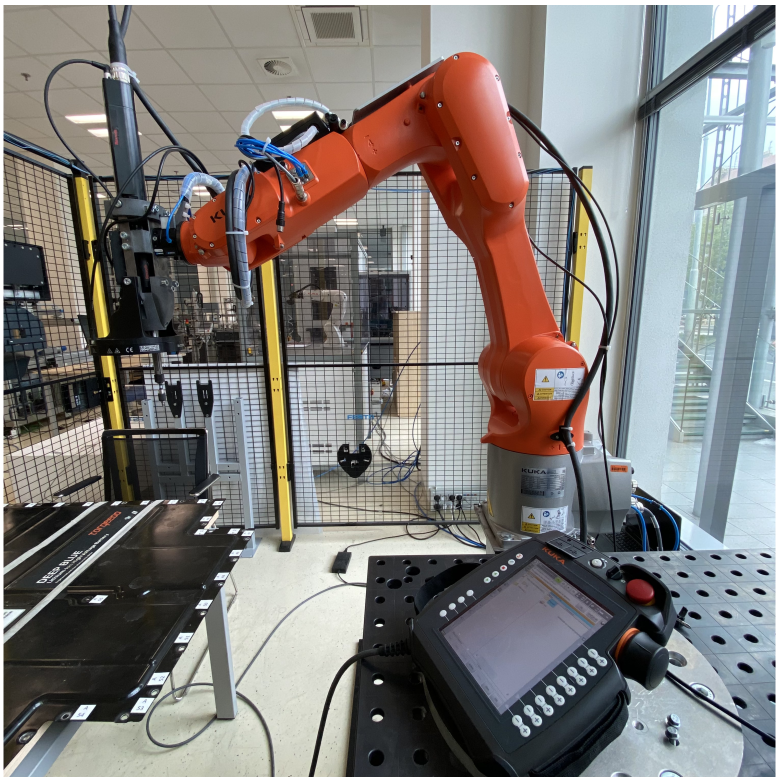


Figure 5.1: The Physical Kuka Robot

The KR 10 R1100-2 is an industrial manipulator model manufactured by Kuka, a renowned German company specializing in industrial robots and automation machinery. This particular model boasts a maximum payload capacity of 11.1kg and a reach of 1101mm. It offers the flexibility to be mounted on the floor, ceiling, or wall.
For our purposes, we have chosen to mount the robot on the floor within a work cell. With six axes or degrees of freedom, this robot provides a wide range of positions and orientations.
The table below, extracted from the Kuka KR robot documentation file, presents detailed information about the motion range for each joint (A1 to A6).

Motion Range	
A1	+-170°
A2	-190°/ 45°
A3	-120°/ 156°
A4	+-185°
A5	+-120°
A6	+-350°

Table 5.1: Motion range of Joints

It is worth noting that this manipulator is composed of various assemblies, with each axis possessing its own distinct rotation direction. The figure below visually illustrates the joints and six degrees of freedom of this robot, indicating the rotational direction of each axis.
Familiarizing oneself with the location and movement of each axis is crucial for programming precise robot movements but also keeping in mind that the precise positioning of each joint can differ from one another too.

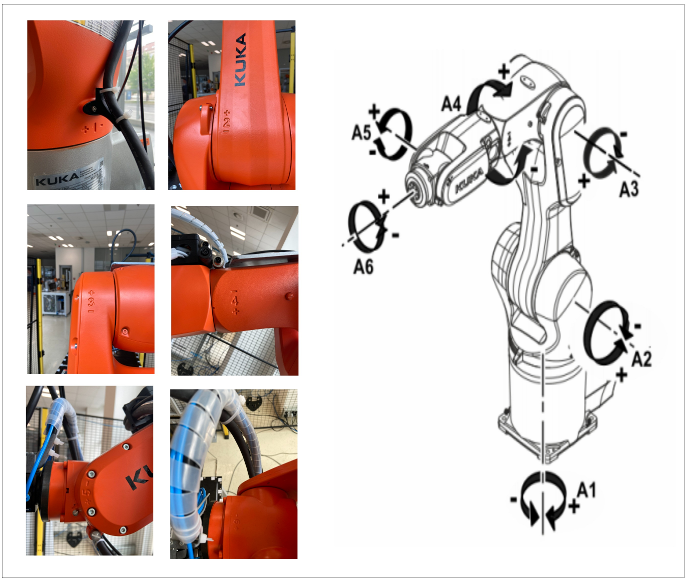


Figure 5.2: Joints and Axis of the Robot.

Mastering Position	
A1	0°
A2	-90°
A3	90°
A4	90°
A5	0°
A6	0°

Table 5.2: Position of Robot Joints

The robot's coordinate system can be described using Cartesian Coordinates, represented by the x, y, and z axes, following the right-hand rule. The orientation of the robot is typically represented using Euler angles or quaternions. Based on this description, the coordinate system of the robot can be classified into two types: the Cartesian coordinate system and the Joint coordinate system.

The Cartesian coordinate system is utilized to define the position of the robot and its end effector in space, using the X, Y, and Z axes. On the other hand, the Joint coordinate system is based on the angle data of each joint in the robot. Each joint has its own coordinate system, which is dependent on the previous parent joint in the kinematic chain. For this project, our focus will be on the Joint coordinate system.

5.1.2 The Digital Robot

To create a digital replica of our physical robot, obtaining a 3D model was essential. Fortunately, like many other robotic companies, Kuka provides a comprehensive selection of CAD models for its products. We were able to locate the CAD file of our specific model, the Kuka KR10 R1100-2, in the Kuka Download Center.

The 3D model of the Kuka KR10 R1100-2 is available in a .obj file format, allowing us to accurately represent its physical structure. The visual representation of the raw 3D object can be observed in the figure below.

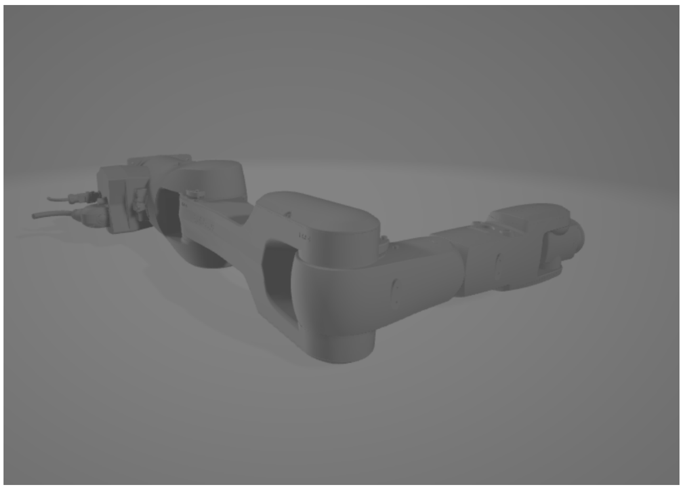


Figure 5.3: 3D Model of the Robot from Kuka Download Center

In our project, the next step involved importing our 3D model into the Unity game engine. By successfully importing the model, we gained the ability to explore various aspects such as other parts, materials, and configurations associated with it within the Unity environment.
This allowed us to have a more comprehensive understanding of the complete model and its components.

Due to inefficiencies and lack of clarity in the initial configuration, we decided to redesign and restructure the robot model to better suit our project's requirements. Using the Unity game engine, we implemented several core edits, as mentioned in the third chapter.
During this phase of the experiment, we had to carefully consider and apply appropriate methods to ensure that the virtual robot could accurately mimic the movements and appearance of the physical robot. We aimed to identify and address any potential issues that might arise, aiming to prevent or minimize them throughout the process.

Transform of the Robot - When we imported the 3D model into Unity, its initial position didn't align correctly with the main coordinate view in Unity's scene. To fix this, we rotated the robot model by -90° toward the X-axis. This adjustment improved our view of the robot in the Unity scene and made our development process smoother.

Scale of the Robot - To optimize the visualization of the 3D model in Hololens 2, we resized the model to match the dimensions of the physical robot. In Unity, the scaling factor is represented by (1, 1, 1), where 1 unit equals 1 meter in Hololens. To achieve the desired size, we adjusted the virtual object's size by decreasing it, making it appear closer to the actual robot's size when viewed through Hololens. This resizing ensured a more realistic representation of the physical robot in the Hololens environment.

Parts of the Model - During this phase, we divided the 3D model into separate parts for better structure and ease of work. Unity's parent-child relationships were used, where the Kuka robot is the main object containing grouped parts to represent it digitally.
This relationship is important as it affects the position, rotation, and scale of child objects based on their parent. By organizing the parts in a hierarchy, we accurately represent the robot's joints and enable efficient management and manipulation of its components. The Base Obj and Joint1 to Joint6 are individual 3D parts that form the core of the model, and organizing them in the hierarchy ensures proper connection and manipulation. We also added a tool to match the physical robot's end effector, enhancing its realism.

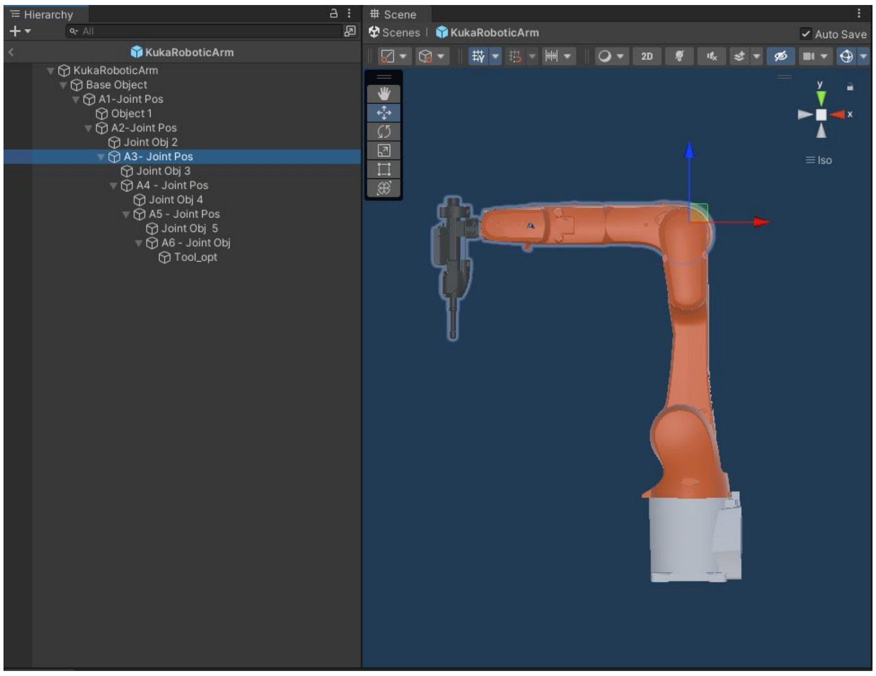


Figure 5.4: Hierarchical Structure of the Digital Robot

To align and define the axis of all the joints, we created new empty GameObjects named A1-Joint Pos, A2-Joint Pos, up to A6-Pos This step was crucial as it allowed us to move all the robot joints simultaneously. However, establishing the parent-child relationship alone was not enough. We had to accurately position, rotate, and scale these axis objects based on the robot's configuration and the central point of each joint.

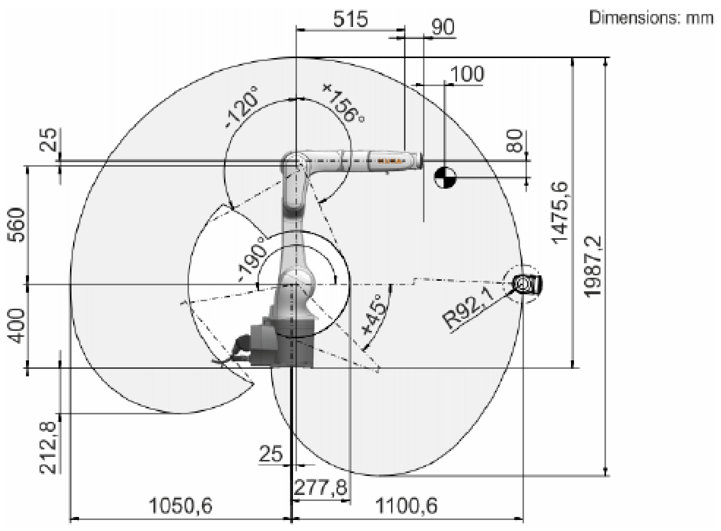


Figure 5.5: Dimensions of the Physical Robot

To adhere to the dimensions specified in the real robot's datasheet, we carefully positioned these axes accordingly. This ensured proper alignment and facilitated the accurate movement of each joint. The presented Figure 5.5 is taken from the physical robots' datasheet, where the dimensions of the robot are presented.

The updated version of the robot, shown in Figure 5.5, features color materials applied to the tool robot to closely resemble the physical robot. A new 3D object, a holder or table, has also been introduced into the scene. Additionally, there is a visible sphere object called "TargetBall." This object serves as the positioning target for the robot, which we will discuss in more detail soon.

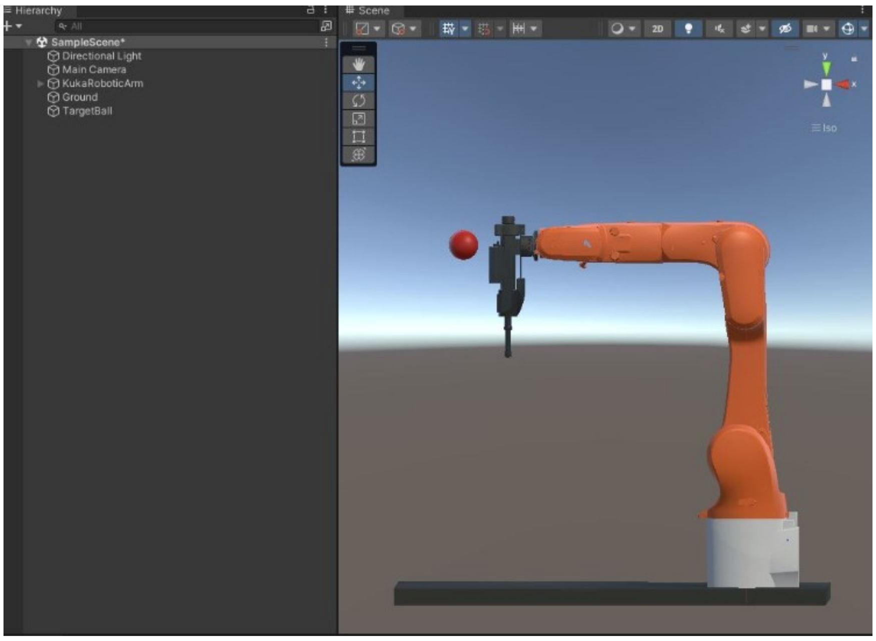


Figure 5.6: Final Look of the 3D Model

The main program utilizes a loop to execute commands based on a command number. It initializes robot settings and velocity values. Upon receiving a command request, it uses a switch statement to determine the appropriate action based on the command number. If no matching command is found, it proceeds with a default action. The accompanying figure provides more details on the data exchange process.

5.2 Motion of the Robot

This section provides an overview of the technical details involved in enabling the movement of the virtual robot, including the optimization of the process. It explores the concept of inverse kinematics and highlights the necessary modifications and related concepts.

To control the movement of the robotic arm, we need to provide it with instructions on where it should go. This is achieved through the implementation of the inverse kinematics method.
As mentioned earlier, the purpose of this method is to determine the position and angles of robot joints in order to reach a desired target point.
For the application of Inverse Kinematics, a lot of different approaches can be found, since there may be multiple paths for the robot to reach the target, or even no solutions at all since the target may be in an unreachable position for the robot.

There are numerous online tools and solutions available for this purpose, the Unity Asset Store also offers options for both simple and complex objects or scenarios, although some of them may require to be purchased and the price may vary too. For the development of this project, we have referred to the solution proposed by [42] The idea is that we have the 6DOF robotic arm and we want it to reach a specific destination point where our target is. But there is a distance between its end effector and the target. Let's call this distance the Error.

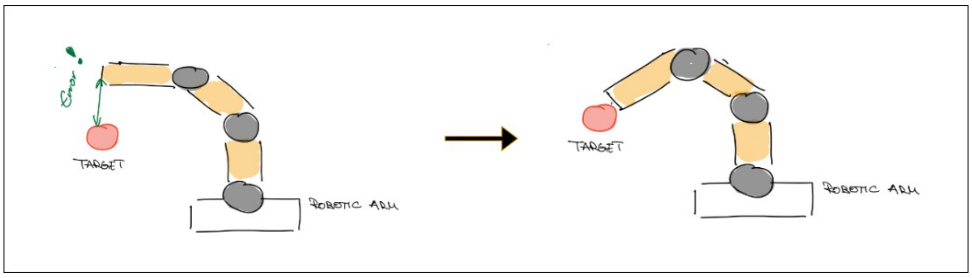


Figure 5.7: Inverse Kinematics Approach

Utilizing Forward Kinematics, we can determine the initial joint angles and derive the position (x, y, z) of the end effector. Furthermore, we possess knowledge of the target's position through its transformation in Unity. However, what remains uncertain is the appropriate values to assign to the joint angles in order to achieve the desired position and effectively minimize the distance between them.

So let's initialize a function calling it an Error function or the cost function which defines the distance between the target and the end effector.

$$E = E(T, A) \tag{5.1}$$

This formula calculates the distance between the position of the end effector (obtained through forward kinematics using the joint angles A) and the target position T. It represents the error or mismatch between the current position and the desired target position.

Having identified the error, our objective is to minimize it. To achieve this, we will be using the Gradient Descent method which is an optimization technique that aims to minimize the error or the cost function.
Conceptually, we can imagine this method as finding the lowest point on a hill territory. Beginning from an initial position and measuring the slope of the territory and taking small steps downhill in the direction of the steepest slope.
By calculating the slope and adjusting the position all the time we eventually approach the goal. In our case, the "territory" represents the error, and the application of Gradient Descent enables us to iteratively update the joint angles, ultimately reaching the target.

Let's introduce the method used for the calculation of the gradient.
Firstly we use a delta value as a perturbation to estimate the direction of change of joint angles, required to reduce error and converge towards the target position.

$$A_{\text{perturbed}} = A + \delta \tag{5.2}$$

Calculate the error with the perturbed solution:

$$Eperturbed = E(T, A_{\text{perturbed}}) \tag{5.3}$$

Calculate the gradient error :

$$\nabla G = \frac{E_{\text{perturbed}} - E}{\delta} \tag{5.4}$$

And finally, we can update the joint angles for a new position.

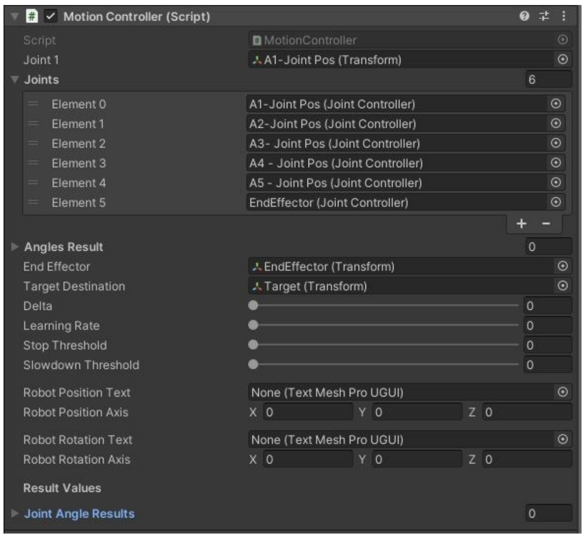
$$A_{\text{new}} = A_{\text{old}} - \alpha \nabla G \epsilon \tag{5.5}$$

- α is the learning rate or step size, controlling the size of the update.
- ∇G is the gradient of the error.
- ϵ is a predefined threshold value that determines when the robot is considered to reach the target or be very close to it.

As a result of the update, we obtain an array consisting of six elements representing the joint angles with their new values.
After updating the joint angles, the algorithm proceeds to the convergence check stage. In a created function the algorithm checks whether the current error function is below a predefined stop threshold, which can be adjusted and fine-tuned.
If the error is smaller than the stop threshold, it means that the robot arm has reached the target position or is very close to it, and the algorithm can terminate.
The whole inverse kinematics runs in the Update function of Unity. It means that it is called for every frame and when we change the position of the target the algorithm will start to perform again, updating the Joint angles until it reaches the target destination.

The Learning Rate α, stop threshold and the slowdown threshold ϵ are public variables that are configurable to be adjusted to tune them based on our robotic arm.

The “MotionController” script is written in C# and is attached to a Gameobject in Unity. To move each Joint separately another script is created. The “JointController” Script helps in controlling the rotation and movement of each individual joint. This happened by assigning this script to each joint link of the robot in Unity.
Here we define the axes of the movement of the joint and limit the angles of the joints by giving the minimum and the maximum of it. This is a necessary step since each Robotic Arm has a specific motion range of Joints as we have also presented them in Table 5.2


Motion Controller (Script)

Script	MotionController
Joint 1	A1-Joint Pos (Transform)
Joints	6
Element 0	A1-Joint Pos (Joint Controller)
Element 1	A2-Joint Pos (Joint Controller)
Element 2	A3- Joint Pos (Joint Controller)
Element 3	A4 - Joint Pos (Joint Controller)
Element 4	A5 - Joint Pos (Joint Controller)
Element 5	EndEffector (Joint Controller)
Angles Result	0
End Effector	EndEffector (Transform)
Target Destination	Target (Transform)
Delta	0
Learning Rate	0
Stop Threshold	0
Slowdown Threshold	0
Robot Position Text	None (Text Mesh Pro UGUI)
Robot Position Axis	X 0 Y 0 Z 0
Robot Rotation Text	None (Text Mesh Pro UGUI)
Robot Rotation Axis	X 0 Y 0 Z 0
Result Values	
Joint Angle Results	0

Figure 5.8: Motion Controller Script

5.3 Communication Setup

After the configuration and functionality of the virtual robot is finalized, communication with the physical robot had to be created.
As we have emphasized earlier, for communication we have chosen the OPC UA communication protocol. In Section 3.7.1 the theory and explanation of this protocol can be found.
The reason we have chosen the OPC UA is because of the possibilities, the efficiency, and the easy use in any platform that it offers and Kuka robots support this type of communication too. Since OPC UA is based on the Client-Server architecture we had to configure the communication according to this.

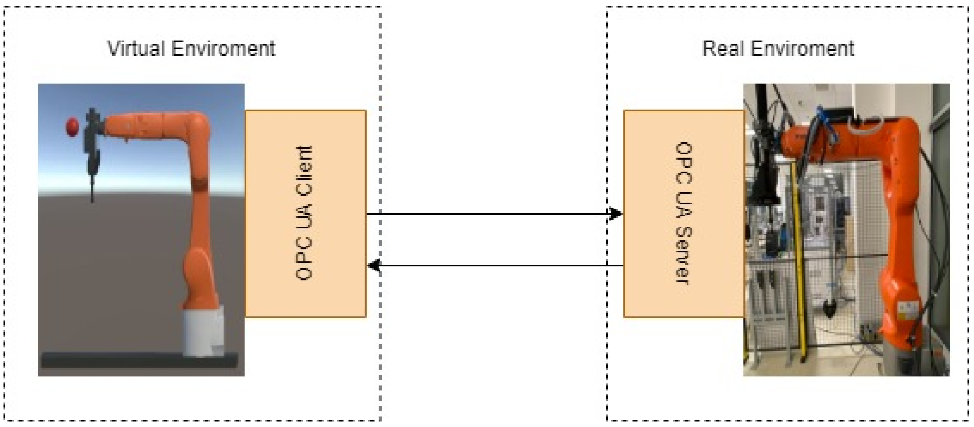
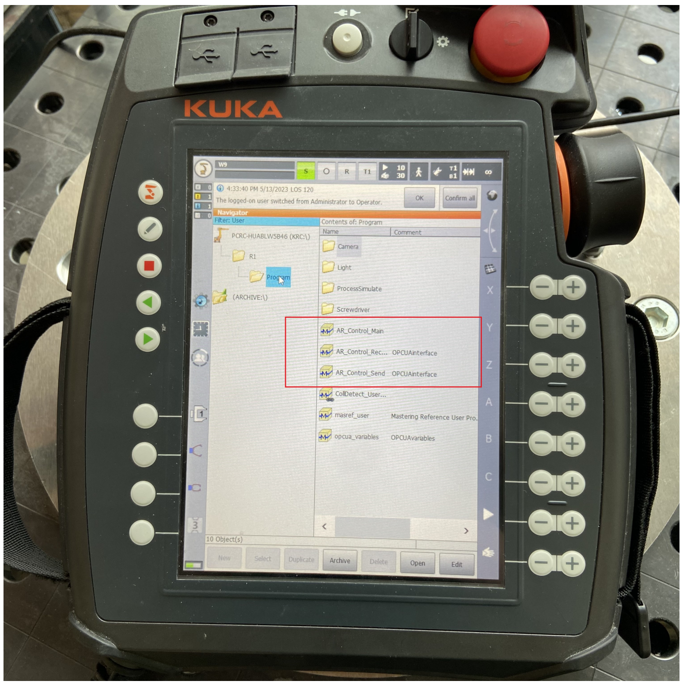


Figure 5.9: Fundamental Communication Connection

5.3.1 Server

The server utilized in this project, the Kuka Robot, has successfully incorporated OPC UA communication. To facilitate this communication, we used the KRL language for programming the robot. Three distinct files have been generated, consisting of one SRC file and two DAT files.
The SRC file, named "AR_Control_Main", encompasses the core implementation of the program, dictating the logical flow and execution of the communication process.
The DAT file, "AR_Control_Receive", includes the declaration of all the data that our server is expected to receive from the client. On the other hand, in the "AR_Control_Send" DAT file, we declare all the values that will be transmitted to the client.



4:33:40 PM 5/13/2023 LOS 120
The logged-on user switched from Administrator to Operator.

Name	Comment
Camera	
Light	
ProcessSimulate	
Screwdriver	
AR_Control_Main	
AR_Control_Rec...	OPCUAinterface
AR_Control_Send	OPCUAinterface
CollDetect_User...	
masref_user	Mastering Reference User Pro.
opcua_variables	OPCUAvariables

10 Object(s)

Figure 5.10: View of Initialized Server Files

The main program utilizes a loop to execute commands based on a command number. It initializes robot settings and velocity values. Upon receiving a command request, it uses a switch statement to determine the appropriate action based on the command number.
For example, If the command number matches 1, the program executes the Homing command, which moves the robot to a predefined homing position using Point-to-Point (PTP) motion. If the command number is 2, the program performs a PTP motion to a specified position. The coordinates of this position are stored in a variable, which is an AXIS data type capable of

storing joint angles the server receives from the client based on the position of the digital robot.
If no matching command is found, it proceeds with a default action. The accompanying figure 5.11provides more details on the data exchange process.

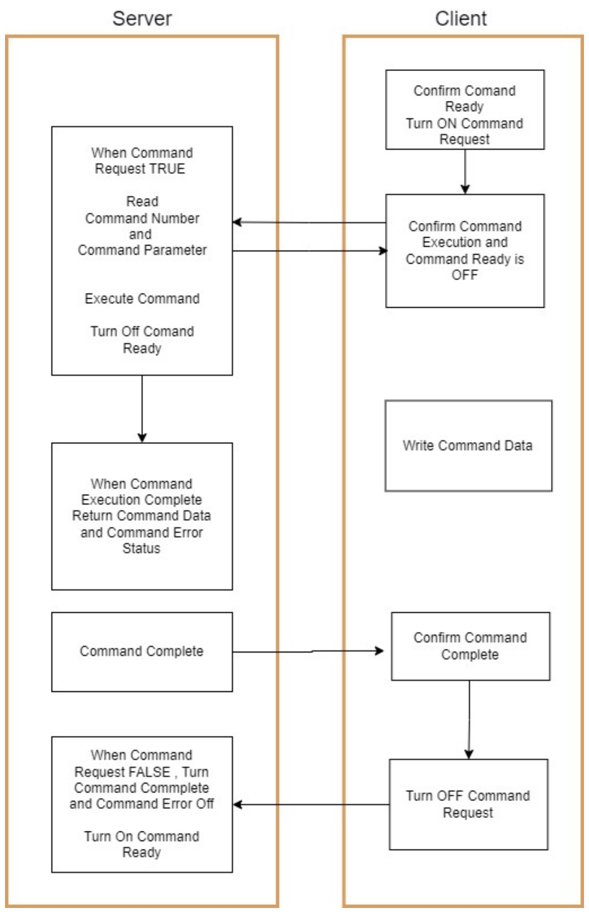

Server
Client
Confirm Comand Ready Turn ON Command Request
When Command Request TRUE
Read Command Number and Command Parameter
Execute Command
Turn Off Comand Ready
Confirm Command Execution and Command Ready is OFF
Write Command Data
When Command Execution Complete Return Command Data and Command Error Status
Command Complete
Confirm Command Complete
When Command Request FALSE , Turn Command Commplete and Command Error Off
Turn On Command Ready
Turn OFF Command Request

Figure 5.11: Communication Diagram between Server and the Client

During the initial server connection phase, we relied on UAExpert, dedicated software for visualizing and troubleshooting OPC UA server connections. This tool played a vital role in various aspects of the connection process.
We used it to search for address spaces and NodeIDs, which were then declared to the client for smooth communication. UAExpert also enabled us to monitor the values we set on the server from the mentioned file, ensuring precise data transmission.
By utilizing this software, we validated our server's functionality, promptly addressing any changes or issues that emerged during development.

5.3.2 Client

The Unity-based client for our project allows us to access data and interact with the OPC UA server. During the research, we found that the Unity Asset store offers an existing OPC UA connection solution for digital twinning and industrial processes. However, considering the high cost of the asset provided by RealVirtual.io, we decided to develop our own communication system instead.
To ensure proper organization, we divided our communication code into two scripts programmed in C#. The first script, "OPCUAConfiguration," handles all the necessary client configuration actions. On the other hand, the second script, "OPCUAClient," manages the methods for reading and writing variables in accordance with server communication.

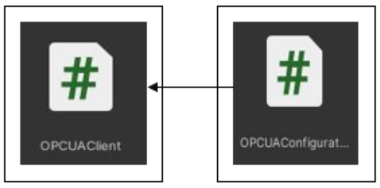


Figure 5.12: OPC UA Client Scripts

As Unity lacks built-in support for OPC UA communication, we needed to utilize a plugin to enable OPC UA client functionality within the project. To achieve this, we incorporated the OPC Foundation's .NET Standard library. Since Unity also operates in the .NET Framework, we added the required DLL files to the Plugins folder of the project, ensuring compatibility with Unity.
In the configuration script, the "InitializeOPCUAClient" method is crucial in creating the OPC UA client application configuration and establishing a session with the server. It leverages the ApplicationConfiguration class to configure the client application and the Session class to initiate a session with the server. To enable the configuration process, several steps must be followed:

-Security Settings: Establishing secure communication to ensure an appropriate level of encryption and authentication.
-Endpoint Selection: Identifying the desired endpoint for designing the transport protocol and security mode.
-Certificate Management: Importing trusted server certificates, generating and managing client certificates, and configuring certificate revocation checks.
- Session Management: Creating a session, which serves as a logical connection between the client and allows data exchange.

Additionally, there are noteworthy sections in the script. The "TagList" dictionary contains the tags (OPC UA nodes) we intend to capture from the OPC UA server. Each tag is associated with a name and its address (NodeID).
The "ChangeType" method is responsible for converting a client value to the appropriate data type based on the data type of the OPC tag. It handles various built-in types and converts the client value to an array if the OPC tag has a value rank greater than 0.
The "ReadValue" method is utilized to retrieve the value of a specific tag from the server, while the "WriteValue" method is employed to write a value to a specific tag on the server.

Within the "OPCUAClient" script, we have included a section where we configure the OPC UA functionality. This script has been specifically designed to facilitate the reading of values received from the server and the writing of values to the variables expected by the server.
The server is capable of accepting various types of values, including but not limited to "Command_Request," "Command_Number," and "Command_Parameter." These values are essential for the server's operation and are written accordingly based on the activation of buttons within the scene.

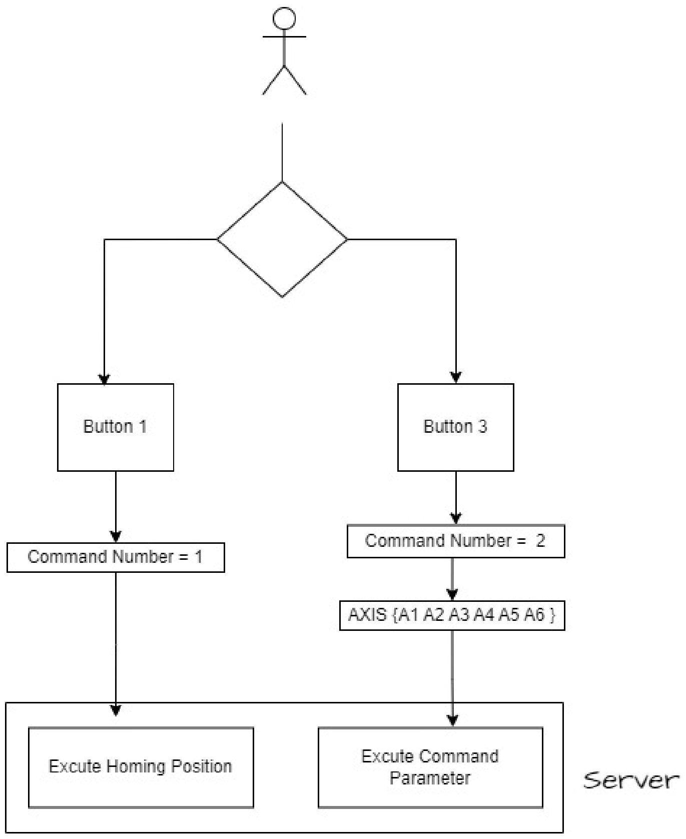


Figure 5.13: Writing Values to Sever Approach

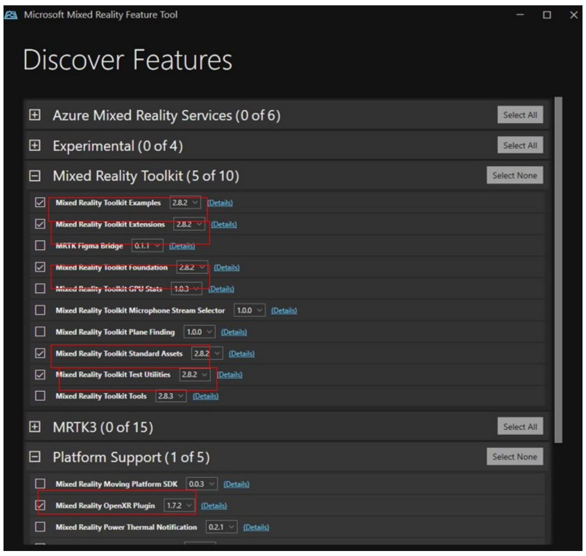
To activate the robot's initial position, the HomingModeClick() method is employed. By clicking the first button, this method sends the necessary values to trigger the robot's homing process. Conversely, the AxisModeClick() method, associated with the third button, is responsible for transmitting angle values to the nodes called by the InverseKinematics script. The Diagram 5.13 presents the workflow of these methods when used by the click of the user. To establish a connection with the server, it is necessary to add the OPCUA-Client script to a GameObject within the Unity scene. Once the scene is in Play mode, the connection to the server is initiated, allowing seamless communication between the client and the server.

5.4 Configuration for Augmented Reality

The next phase of the project involves configuring it for Augmented Reality (AR) technology, specifically adapting it for use with Hololens glasses. The initial crucial step is to configure our scene in Unity to accommodate the requirements of Hololens.

To effectively develop an AR project in Unity, the recommended approach is to utilize the Mixed Reality Toolkit, which provides comprehensive support. Here are the steps we followed:

1. **Setting up the Mixed Reality Toolkit in the Project:** We carefully selected the components from the toolkit to ensure our project incorporates as many features as possible. We made specific choices based on the versions and options depicted in the figure below.



Microsoft Mixed Reality Feature Tool

Discover Features

⊞ Azure Mixed Reality Services (0 of 6) — Select All
⊞ Experimental (0 of 4) — Select All
⊟ Mixed Reality Toolkit (5 of 10) — Select None

- [✓] Mixed Reality Toolkit Examples 2.8.2 (Details)
- [✓] Mixed Reality Toolkit Extensions 2.8.2 (Details)
- [ ] MRTK Figma Bridge 0.1.1 (Details)
- [✓] Mixed Reality Toolkit Foundation 2.8.2 (Details)
- [ ] Mixed Reality Toolkit GPU Stats 1.0.3 (Details)
- [ ] Mixed Reality Toolkit Microphone Stream Selector 1.0.0 (Details)
- [ ] Mixed Reality Toolkit Plane Finding 1.0.0 (Details)
- [✓] Mixed Reality Toolkit Standard Assets 2.8.2 (Details)
- [✓] Mixed Reality Toolkit Test Utilities 2.8.2 (Details)
- [ ] Mixed Reality Toolkit Tools 2.8.3 (Details)

⊞ MRTK3 (0 of 15) — Select All
⊟ Platform Support (1 of 5) — Select None

- [ ] Mixed Reality Moving Platform SDK 0.0.3 (Details)
- [✓] Mixed Reality OpenXR Plugin 1.7.2 (Details)
- [ ] Mixed Reality Power Thermal Notification 0.2.1 (Details)

Figure 5.14: MRTK Packages in the Project

2. **Configuring the Building Target:**It is essential to build our project for the Universal Windows Platform and select the appropriate target platform, which, in our case, is Hololens. We also specified the ARM64 architecture and other relevant data.
3. **Project configuration for Hololens 2:** using Unity's Plug-in provider, OpenXR with all the correct settings for UWP build.
4. **Configuring the scene design:**We integrated MRTK into our scene, resulting in its configuration for Hololens 2. This adjustment is made by

making the Hololens camera the main Unity camera and incorporating additional features such as various inputs, spatial mapping, camera types, and more.

5. **Building and Deployment of the project in Hololens:** This is a crucial step to have our project run independently from Unity.

To avoid the need for frequent project deployments on the Hololens to visualize the scene after every minor change, we utilized the holographic control app.
This convenient application can be obtained from the Microsoft Store for Hololens, enabling us to establish a direct connection between Unity and the device using the wireless connection's IP address.
Upon activating play mode in Unity, the game scene is instantly presented within this application.

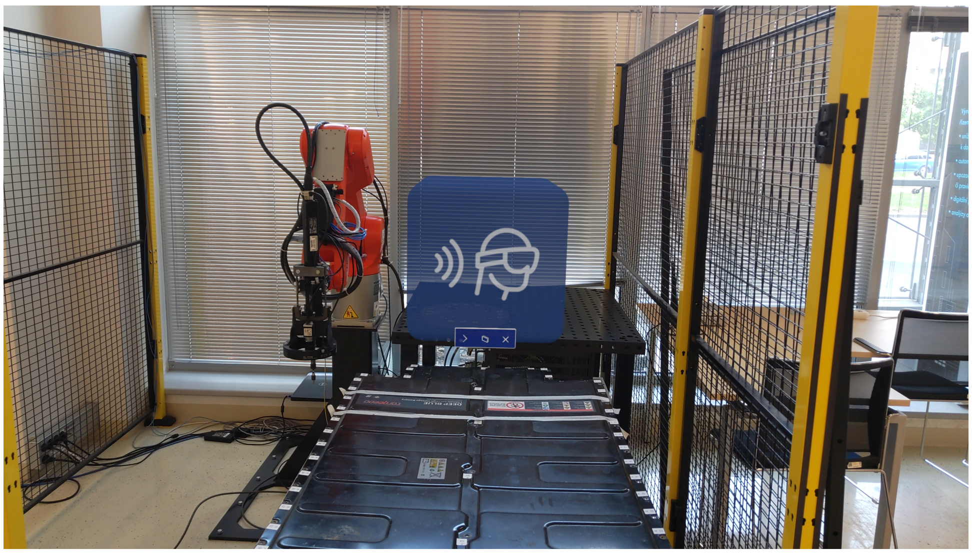
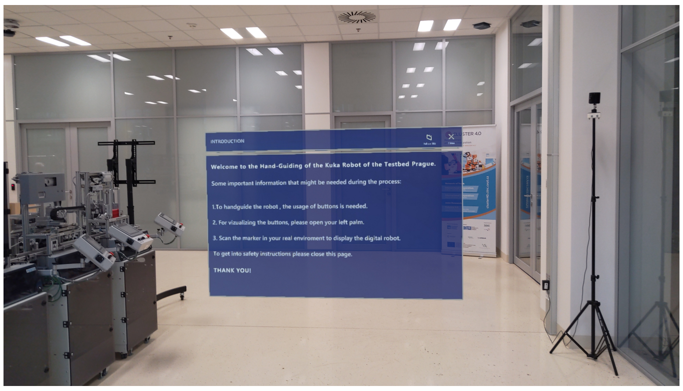


Figure 5.15: Holographic Remote Player App

5.5 User Interface and Design

Due to the potential application of this project in the industrial sector for guiding industrial robots or training users, we have placed significant emphasis on the project's design and user interface. Our goal is to ensure that the application is as clear and user-friendly as possible.

Let's get into the design choices we have implemented:
-Tutorial Screens: These informative windows guide the user through the application, providing step-by-step instructions and addressing user security concerns.



Welcome to the Hand-Guiding of the Kuka Robot of the Testbed Prague.

Some important information that might be needed during the process:

1.To handguide the robot , the usage of buttons is needed.

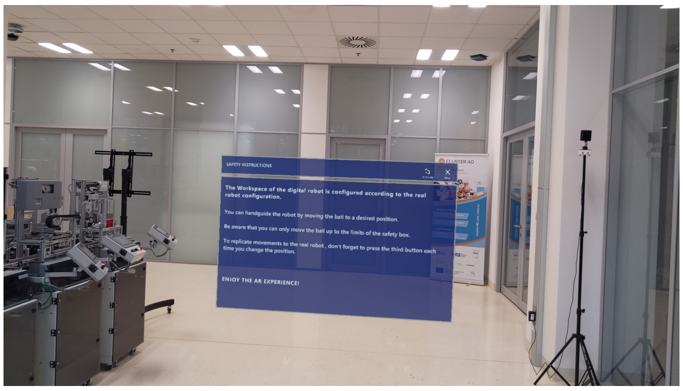
2. For vizualizing the buttons, please open your left palm.

3. Scan the marker in your real enviroment to display the digital robot.

To get into safety instructions please close this page.

THANK YOU!

Figure 5.16: Introduction Window



The Workspace of the digital robot is configured according to the real robot configuration.

You can handguide the robot by moving the ball to a desired position.

Be aware that you can only move the ball up to the limits of the safety box.

To replicate movements to the real robot , don't forget to press the third button each time you change the position.

ENJOY THE AR EXPERIENCE!

Figure 5.17: Safety Instructions Window

-Hand Menu Prefab of MRTK features: We have opted to utilize this prefab, which displays buttons necessary for interacting with the robot whenever the user opens their left or right palm. This placement allows easy access to the buttons without the need for the user to search for them and also to react quickly in case of an emergency or safety reasons.

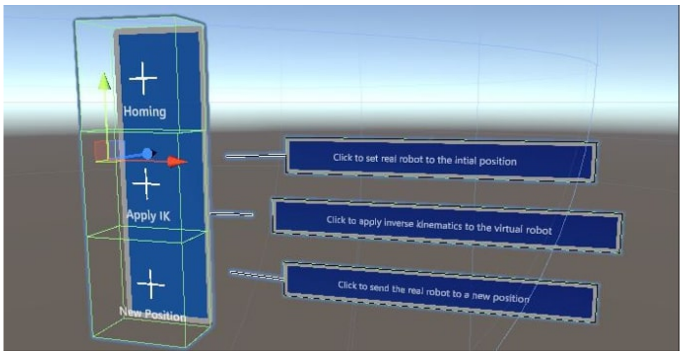

Homing
Apply IK
New Position
Click to set real robot to the intial position
Click to apply inverse kinematics to the virtual robot
Click to send the real robot to a new position

Figure 5.18: Safety buttons in the scene

-SimpleLine ToolTip: This prefab, imported from MRTK, is utilized in our scene to provide specific details for each button using three instances of it as can be seen in Figure 5.18

-Image Target: To position the digital robot in any desired location within the real environment, we have incorporated Vuforia.
Since the origin of the Hololens device depends on the starting point of the application, to address this, we have implemented a method that allows the user to determine the placement of the virtual robot in the real environment. By utilizing the Hololens camera, the user can scan the Aruco Marker, which is positioned in the physical environment. This scanning process enables the user to choose the desired location for the virtual robot within the real environment.

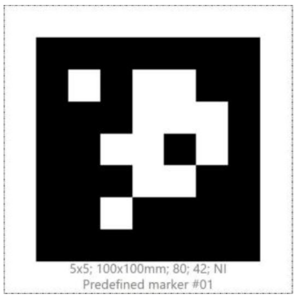


Figure 5.19: Marker for positioning the digital content

-Solver Handler This script is attached to a chevron and its functionality is to track the position of the digital robot. So for example, if the user is far from the position of the virtual component in the real world, it directs and orientates toward that object to find it easier on the screen. We configured it to track the target by the head movements but there are also other features offered by MRTK.

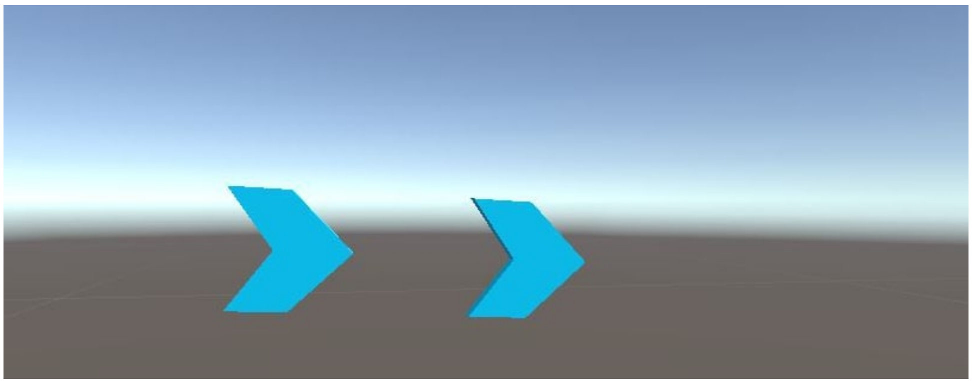


Figure 5.20: Object Follower

-ColorChanger: This script is applied to the target object and is designed to alter the color of the virtual ball in the scene each time the user catches it. This functionality enhances detection and facilitates maintaining its position within the scene.

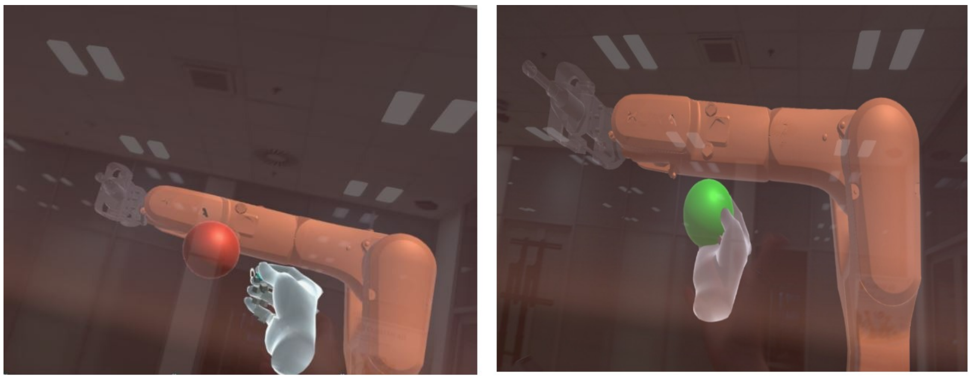


Figure 5.21: Color Change of the Ball when in interaction

5.6 Safety Consideration

Safety is a crucial aspect that we have prioritized in this project, considering that the movements will be replicated by the real robot. Therefore, we have taken special precautions to ensure safety.
Up until this point in the project, the security measures primarily involve continuous control of movement transmission. In addition to facilitating program usage, the buttons on the interface also contribute to safety.
To apply the inverse kinematics method to the digital robot, we need to press the second button. Only after pressing this button, the robot can move to the desired position that we have specified so that the robot does not immediately go to the target since the beginning of the application.
The third button serves the most important function as it transmits the angle values obtained by the digital robot to map the movements onto the real robot. Let's go through the steps of how this process works:

1. The user moves the target to a desired position.
2. The virtual robot moves to that position.
3. The user presses the "New Position" button (the third button in the user interface).
4. The angle values are sent to the real robot.
5. The real robot moves to the position corresponding to the virtual one.

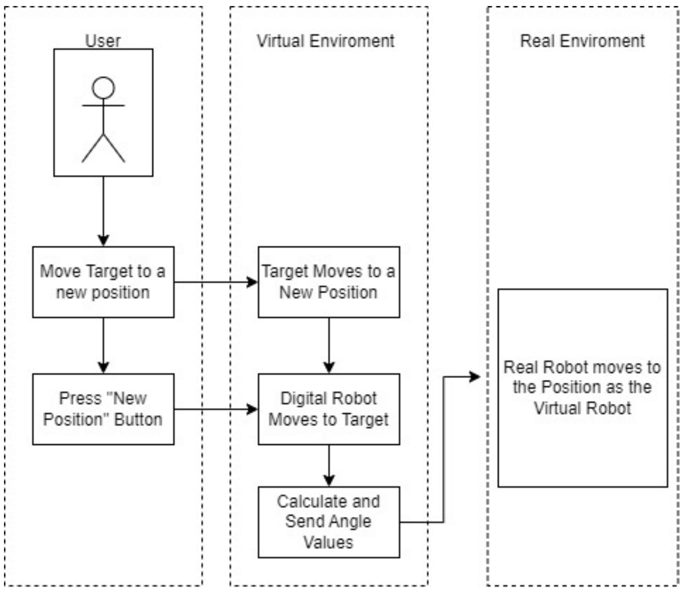

User
Virtual Enviroment
Real Enviroment
Move Target to a new position
Target Moves to a New Position
Press "New Position" Button
Digital Robot Moves to Target
Calculate and Send Angle Values
Real Robot moves to the Position as the Virtual Robot

Figure 5.22: "New Position" Button Functionality

By employing this method, we ensure a clear observation and prevent the real robot from reaching any positions that may cause singularities or pose any risks in the real environment.
To enhance safety, we have implemented an additional measure known as the Safetybox. Its purpose is to prevent the transmission of target positions that are beyond the reach of the robot, as these positions can lead to potential issues. So the user can maneuver with the position of the Target Ball only inside of the limits of the box.
This measure is particularly important considering that the real robot is located within the Testbed workspace, which contains other objects and is situated near windows. These environmental factors must be carefully considered while guiding the digital robot. And if for example in considering the hand guiding of the robot not only from the lab but in another environment where we cannot see the surroundings of the real robot, the Safetybox does not allow us to pass to an inaccessible position.

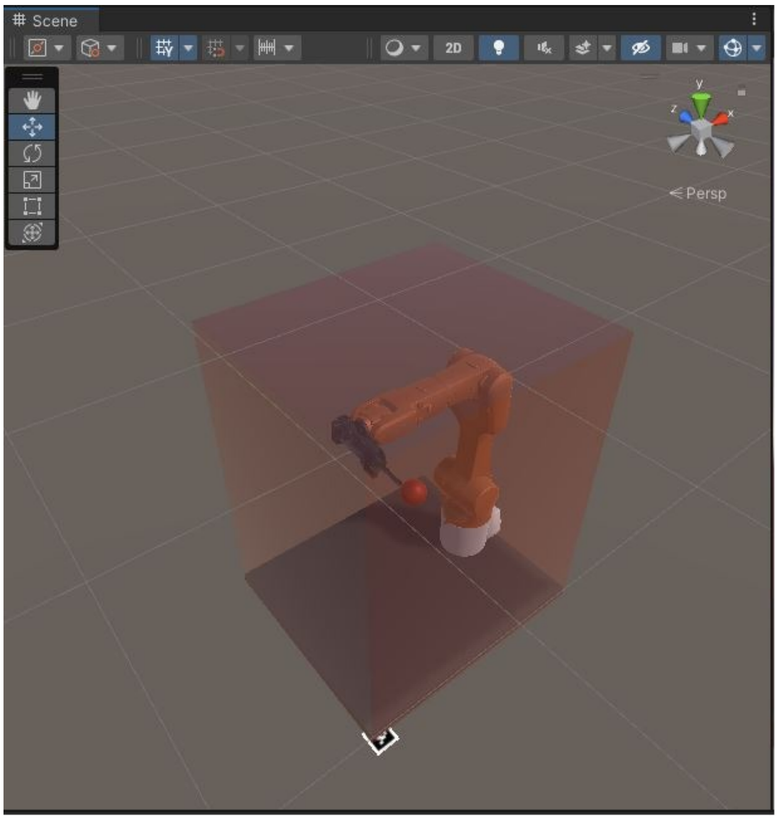


Figure 5.23: Safety Box

Last but not least, we have imposed restrictions on the range of motion for each joint of the virtual robot by defining maximum and minimum limits. This additional security measure helps to prevent self-collisions and singularities, ensuring safe and reliable operation.

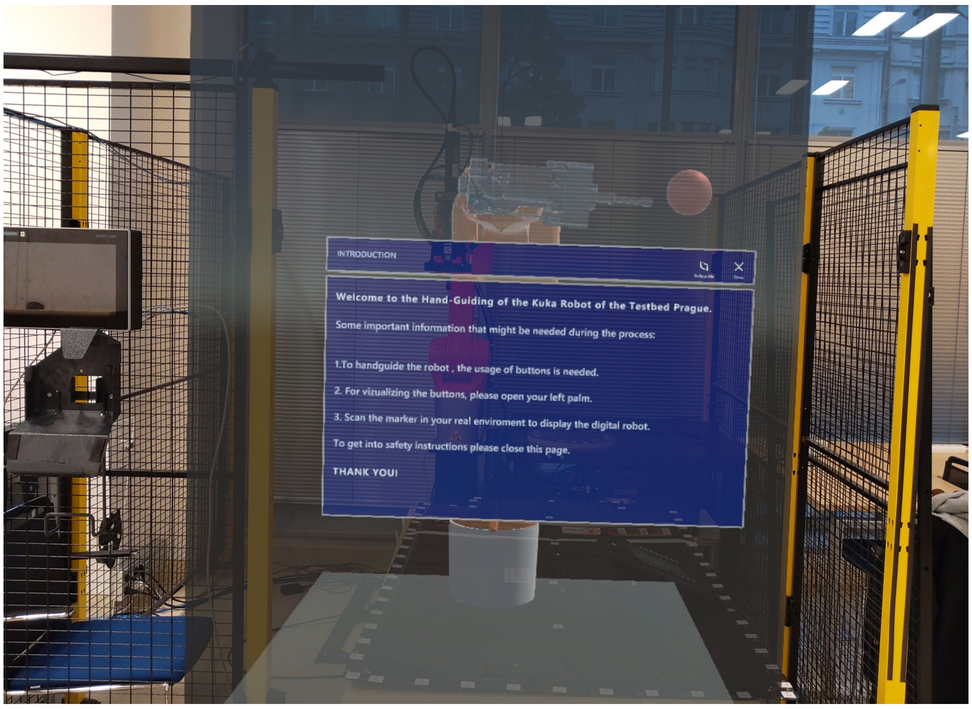
Chapter 6

RESULTS

Here, the focus will be on presenting the results obtained after the implementation of the process together with the outcomes of the applied methods and techniques.

The project successfully accomplished its main objective of creating a virtual hand guide for an industrial robot. By exploring communication methods between the real and virtual worlds and leveraging Microsoft Hololens' Augmented Reality technology, the following outcomes were obtained.

The designed virtual content, including the digital model of the Kuka KR 10 R1100-2, the target object for the robot to follow while being guided by the user of the headset, along with intuitive User Interface visualizations to enhance usability, were integrated into the physical environment.
The features from the virtual world accurately matched their counterparts in reality, resulting in a faithful digital representation of the physical robot.
At the beginning of the scene, the user had the opportunity to read about the usage of the system as well as the safety measures that were taken.


INTRODUCTION

Welcome to the Hand-Guiding of the Kuka Robot of the Testbed Prague.

Some important information that might be needed during the process:

1.To handguide the robot , the usage of buttons is needed.

2. For vizualizing the buttons, please open your left palm.

3. Scan the marker in your real enviroment to display the digital robot.

To get into safety instructions please close this page.

THANK YOU!

Figure 6.1: Virtual Content in Real Environment - Introduction Window

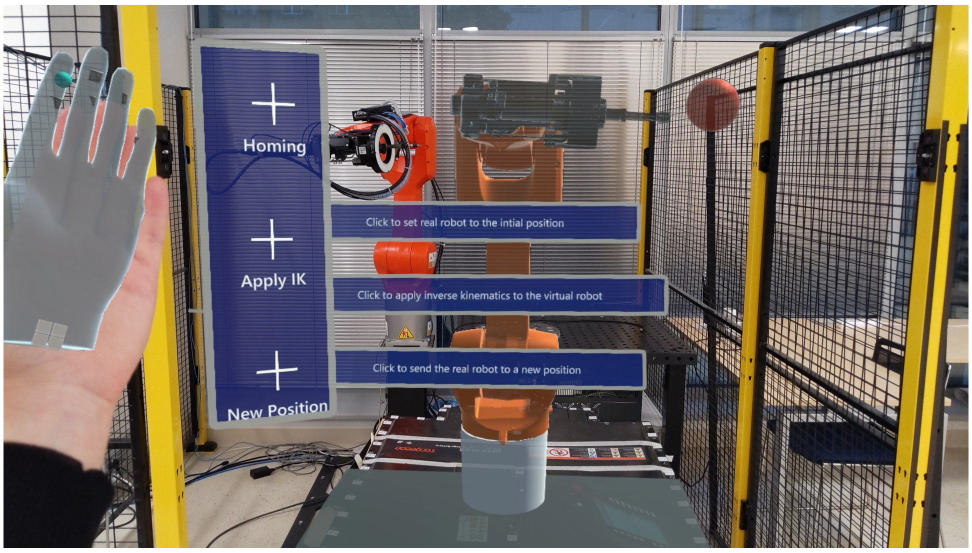

Homing
Apply IK
New Position
Click to set real robot to the intial position
Click to apply inverse kinematics to the virtual robot
Click to send the real robot to a new position

Figure 6.2: Virtual Content in Real Environment - Buttons

Positioning of the digital robot in the real world was accomplished using Vuforia's image recognition capabilities. The Marker used as the image target was placed next to the workspace of the Kuka robot. Using the Hololens camera, the marker was scanned and the virtual content successfully appeared and was placed in the real environment.

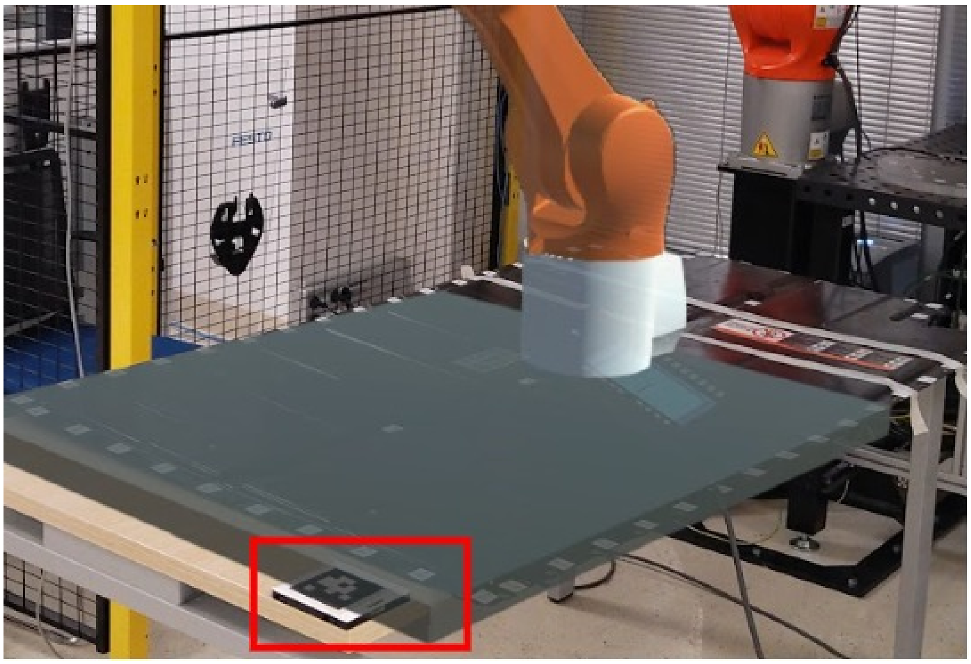


Figure 6.3: Scanned Marker in the Scene

After following the guidance from the virtual tutorials in the scene, the user was able to guide the robot. This was made possible by first sending the physical robot to the homing position.
The next step involved the manipulation of a target ball, allowing the user to designate a desired position for the robot to move towards. By moving the target ball, the user could effectively guide the robot toward the designated location.
When the user found herself in close proximity to the target, she could employ the grabbing feature by performing an air tap and holding it while moving. In cases where the target is out of reach, the user could employ the hand ray to direct their attention towards the target and perform an air tap to move to another position.
With the introduction of the IK button, users gained the ability to apply inverse kinematics to the virtual robot, resulting in its movement toward the target position. By pressing the IK button, the system initiated the calculation of the appropriate joint angles for the robot to reach the designated target. Whenever the user made adjustments to the position, the IK functionality promptly updated the joint angles of the robot to accommodate these changes. This dynamic update ensured that the virtual robot followed the user's intended target position.

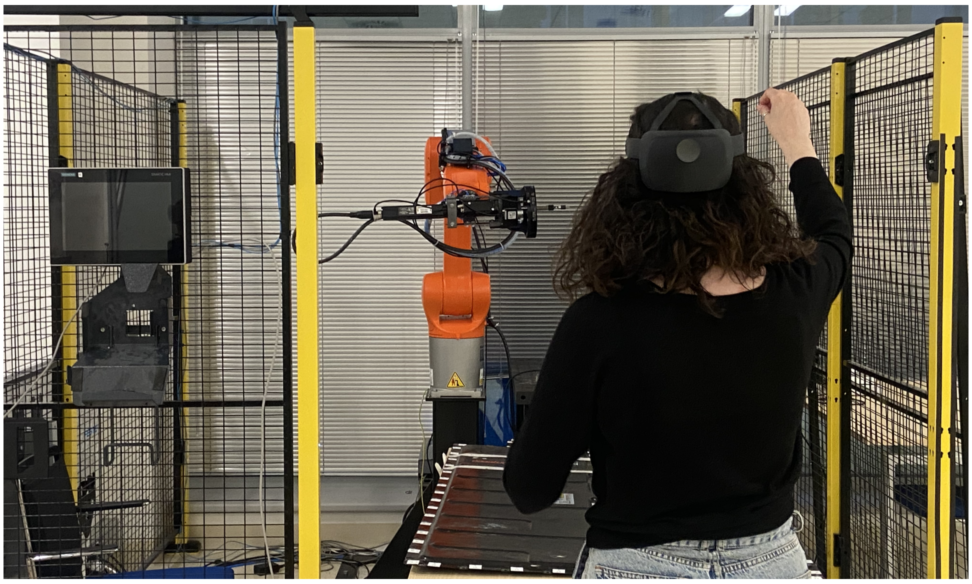


Figure 6.4: User moving the target to a position

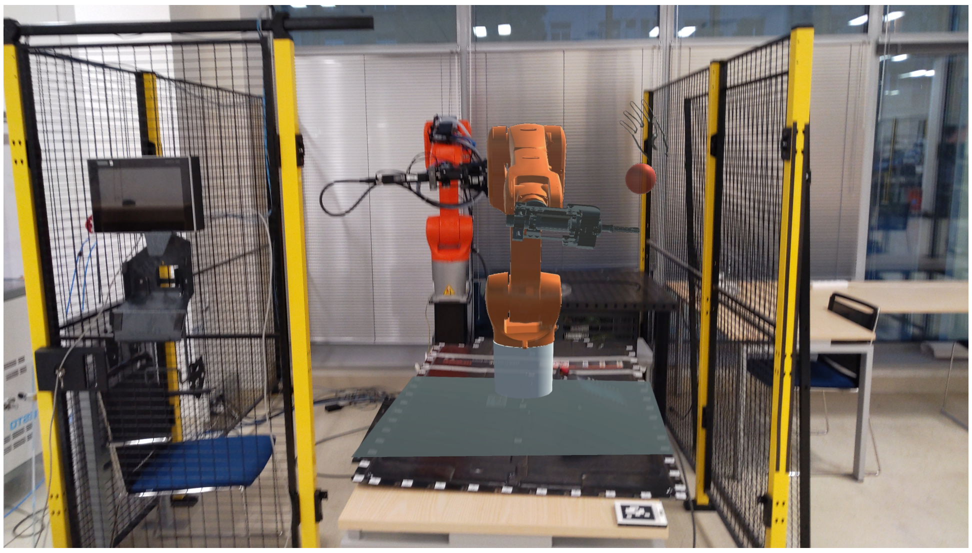


Figure 6.5: Guided position of the virtual and physical robot

For communication between the digital and physical robots, we employed OPC UA communication to send joint angle values. This established a stable and secure connection between the digital project and the Kuka Robot.
In order to run the project, it was necessary to execute it from Unity. During the application on Hololens, we encountered an issue where there was no response in the communication between the client and the server when the application was running only from Hololens independently. The client we have created could only communicate with the server when run from Unity.

To address this challenge, we utilized the Holographic Remote Player App, which makes possible the connection between Hololens and compatible apps, including Unity.
By practicing this solution, a successful connection by specifying the IP address of the Hololens device within Unity was established keeping in mind that the communication was made possible when both the PC where the project runs and Hololens is connected to WIFi.
This method enabled smooth communication, allowing us to read the values on the client that were sent from the server and write values on it.
As a result, it became essential to run the project from Unity in order to ensure proper functionality and achieve successful communication between the digital and the physical entities. The application of the project which was deployed to Hololens can independently run, without the communication of OPC UA. The app, which could be found in the Hololens is in Figure 6.6, could potentially be used as a demo of the project as well as for the functionality of the industrial robotic arm.

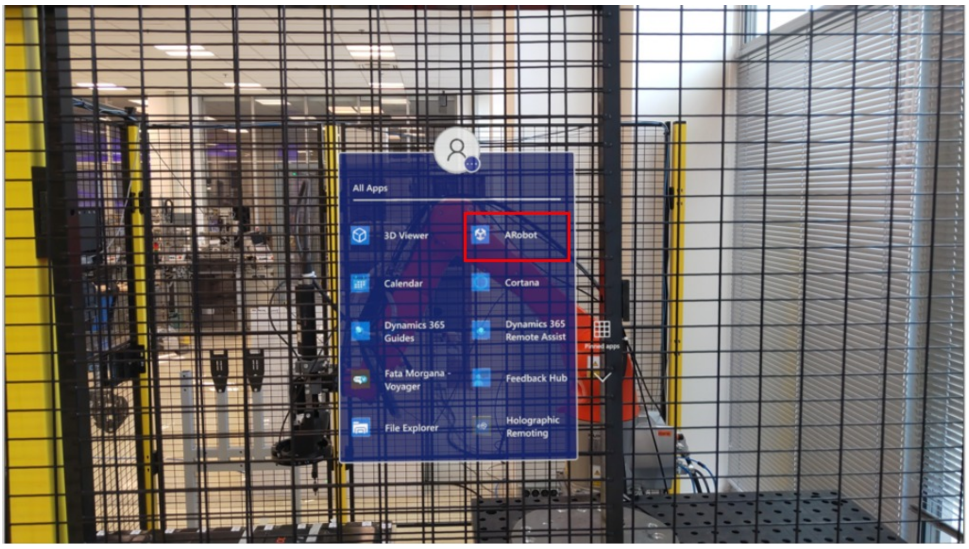
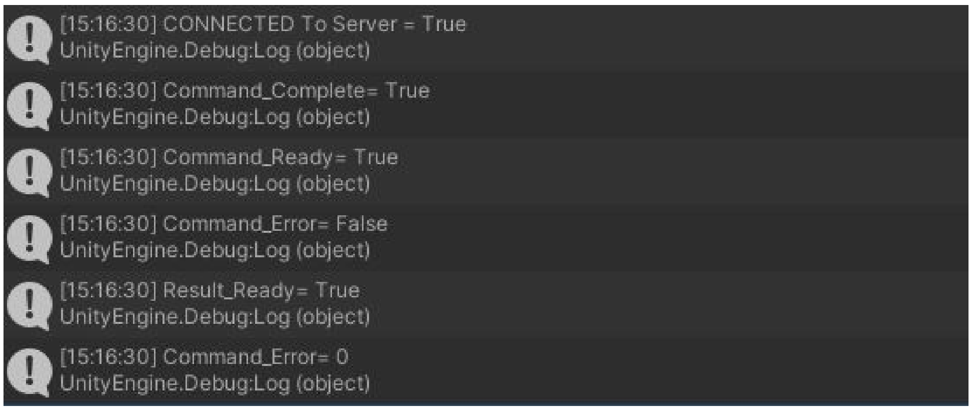


Figure 6.6: ARobot App in Hololens

After the start of the project, in the Console window of Unity, the communication state was observed. The values in the Console can be seen in Figure 6.7.
It was possible to read the values sent by the server while also sending the joint angles to the robot, we observed the changes in the position of the robot as well as validated the correctness of the results from the UA Expert.



```
[15:16:30] CONNECTED To Server = True
UnityEngine.Debug:Log (object)
[15:16:30] Command_Complete= True
UnityEngine.Debug:Log (object)
[15:16:30] Command_Ready= True
UnityEngine.Debug:Log (object)
[15:16:30] Command_Error= False
UnityEngine.Debug:Log (object)
[15:16:30] Result_Ready= True
UnityEngine.Debug:Log (object)
[15:16:30] Command_Error= 0
UnityEngine.Debug:Log (object)
```

Figure 6.7: Connection Values Read in Unity

While the movements from the virtual world were mapped to the real world correctly safety considerations were taken into account, as users were restricted from guiding the robot to unreachable positions.
A safety box was implemented to confine the target ball within its limitations, ensuring secure operation and considering the workspace of the physical robot. In the following figures, the different guided positions of the robot are shown. Also, the Marker was scanned in different places in order to have a better visualization of the position mapping.

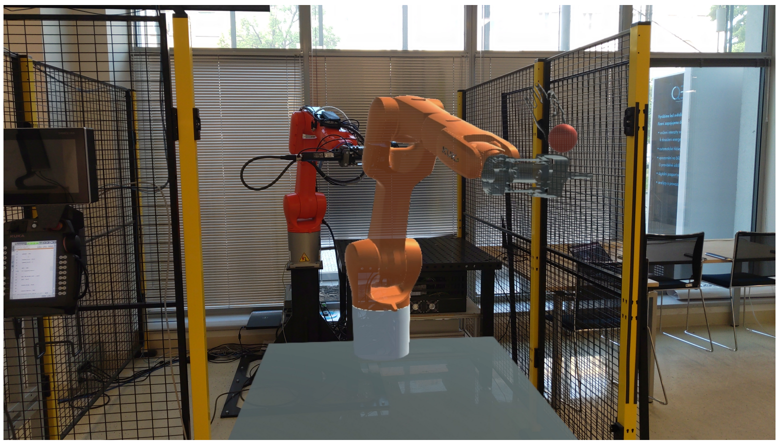


Figure 6.8: Guided Position of the Robot - 1

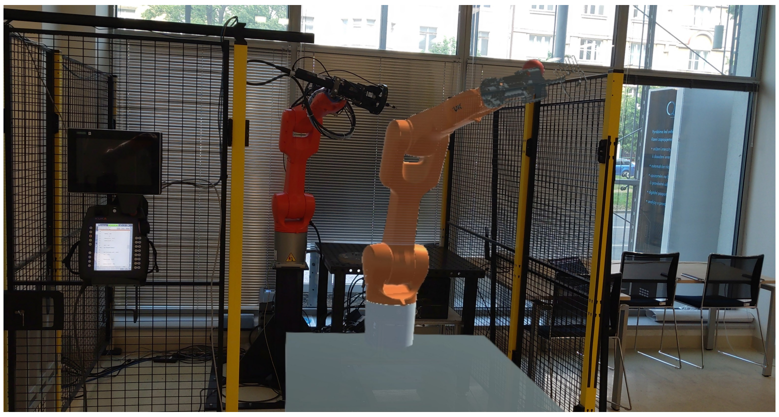


Figure 6.9: Guided Position of the Robot - 2

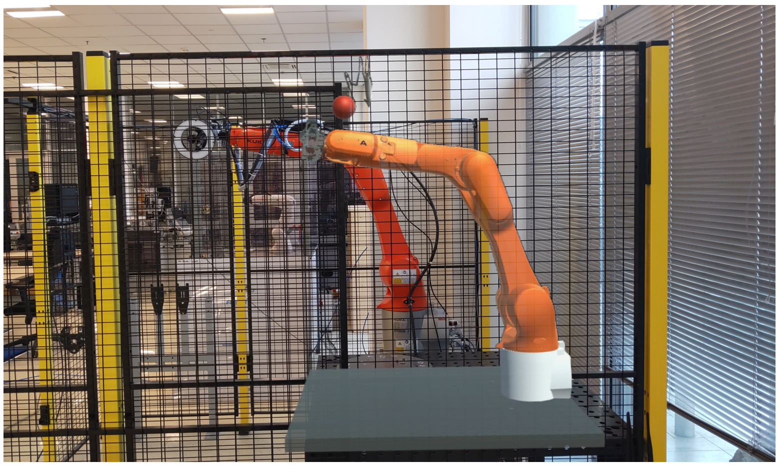


Figure 6.10: Guided Position of the Robot - 3

Chapter 7

CONCLUSION

In conclusion, this diploma project focuses on the practical application of Augmented Reality technology to facilitate the guidance of physical industrial robots by providing virtual world guidance.
The system utilizes OPC UA Communication to map the digital robot's joint angle values to the selected industrial robot, ensuring efficient operation. Emphasizing user and environmental safety, the project has successfully demonstrated the robot's ability to follow targeted points.
The implemented system serves multiple purposes within the existing context. Primarily, this system eliminates the need for traditional programming techniques by enabling the control of the industrial robot through the application itself. With virtual control, users could modify the robot's position remotely, ensuring the safety of workers and program users alike.
Additionally, the developed virtual app could be offered as an effective training tool for educating young workers about industrial robots. It would comprehensive explanations of the robot's functionality, safety considerations, and proper utilization, all without the requirement of using a physical robot. This approach maximizes training efficiency while ensuring optimal learning outcomes.
The program developed for robot movement is versatile and adaptable to different robot models, requiring only minor modifications depending on the model and configuration of the chosen robot.
With stable OPC UA communication obtained, the system can be extended to apply Augmented Reality technology to any industrial device or machine which uses this type of communication protocol, where additionally the configuration of the server would need to be adjusted accordingly.
Notably, the project's most challenging aspect lies in the Unity-based adaptation and redesigning of the model, necessitating specific 3D modeling skills.
In summary, the system stands as a robust and secure solution, suitable

for current usage and future advancements. Its potential lies in facilitating industrial processes while incorporating Augmented Reality, offering a solid foundation for further development.

Chapter 8

FUTURE WORK

Currently, the system we have created depends on the connection to the computer, while it runs via Unity. The future objective is for the system to operate autonomously, without relying on Unity and the Holographic Remote App.
This would involve deploying the application on the Hololens device, enabling it to establish OPC communication with the server and transmit data regardless of the user's location.
An alternative development possibility can be considered, which involves eliminating the need to manually press a button each time to send angle values to the physical robot for a new movement. Instead, continuous transmission of values without interruptions can be achieved.
To accomplish this, we think that it is necessary to explore options for rewriting the stored values in the server space while ensuring safety considerations are met.
In terms of security, it is important to note that this is the initial version of the system. However, for future iterations, additional steps can be implemented to enhance security measures.
For instance, precautions can be taken to prevent self-collision or singularity issues. Users can be alerted if the robot is directed toward a position where such issues may occur.
Overall, the project has significant potential to become a valuable product in various industries, particularly in applications involving industrial robots.

Bibliography

[1] O.Khatib B.Siciliano. *Springer Handbook of Robotics*. Springer US, Boston, MA, 2005.

[2] Andrzej Burghardt, Dariusz Szybicki, Piotr Gierlak, Krzysztof Kurc, Paulina Pietruś, and Rafał Cygan. Programming of industrial robots using virtual reality and digital twins. *Applied Sciences*, 10(2), 2020.

[3] Yudhisteer Chintaram. Digital twin of anthropomorphic robotic arm, 2022.

[4] IBM Developer. Getting to know mqtt. *IBM Developer*, 2017.

[5] Mitchell Doughty, Nilesh R. Ghugre, and Graham A. Wright. Augmenting performance: A systematic review of optical see-through head-mounted displays in surgery. *Journal of Imaging*, 8, 2022.

[6] Mark Fairchild. Industrial robot types and their different uses, 2021.

[7] Flex. Manufacturing Goods in a Virtual and Augmented World. `https://flex.com/resources/manufacturing-goods-in-a-virtual-and-augmented-world`, 2021.

[8] OPC Foundation. OPC Foundation Reference. `https://reference.opcfoundation.org/`.

[9] Gaurav Garg, Vladimir Kuts, and Gholamreza Anbarjafari. Digital twin for fanuc robots: Industrial robot programming and simulation using virtual reality. *Sustainability*, 13(18), 2021.

[10] Gleb B. VR vs AR vs MR: Differences and Real-Life Applications. `https://rubygarage.org/blog/difference-between-ar-vr-mr`, 2020.

[11] HBR. How augmented reality can and can't help your brand, 2022.

[12] PTC Inc. Vuforia Library. `https://library.vuforia.com/`.

[13] Fortune Business Insights. Augmented reality [ar] market size, share & forecast [2028], 2021.

[14] KUKA. *KUKA Robot Programming 1*. KUKA Roboter GmbH, 2015.

[15] KUKA. *KUKA System Software 8.3 - Operating and Programming Instructions for System Integrators*. KUKA Roboter GmbH, 2015.

[16] Vitalii Kutia, Filip Lukasz Ruchel, and Krzysztof Chrapek. Simulation and programming of an industrial robot based on augmented reality. *Modeling, Control and Information Technologies: Proceedings of International scientific and practical conference*, page 184–186, Nov. 2019.

[17] Vladimir Kuts, Tauno Otto, Toivo Tähemaa, and Yevhen Bondarenko. Digital twin based synchronised control and simulation of the industrial robotic cell using virtual reality. *Journal of Machine Engineering*, 19:128–144, 02 2019.

[18] Juhwan Lee, Sangwon Hwang, Jisun Lee, and Seungwoo Kang. Comparative performance characterization of mobile ar frameworks in the context of ar-based grocery shopping applications. *Applied Sciences*, 10(4), 2020.

[19] Congyuan Liang, Chao Liu, Xiaofeng Liu, Long Cheng, and Chenguang Yang. Robot teleoperation system based on mixed reality. In *2019 IEEE 4th International Conference on Advanced Robotics and Mechatronics (ICARM)*, pages 384–389, 2019.

[20] MathWorks. Inverse Kinematics. `https://www.mathworks.com/discovery/inverse-kinematics.html`.

[21] Microsoft. Microsoft HoloLens Hardware. `https://www.microsoft.com/en-us/hololens/hardware`.

[22] Microsoft. Microsoft Learn: Mixed Reality. `https://learn.microsoft.com/en-us/windows/mixed-reality/discover/mixed-reality`.

[23] Microsoft. Microsoft Learn: Mixed Reality Toolkit for Unity (MRTK). `https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/?view=mrtkunity-2022-05`.

[24] Microsoft. Microsoft Learn: Mixed Reality Toolkit for Unity (MRTK) Architecture Overview. `https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/architecture/overview?view=mrtkunity-2022-05`.

[25] Microsoft. Microsoft Learn: Universal Application Platform Guide. https://learn.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide.

[26] Microsoft. Microsoft mixed reality - manufacturing, 2023.

[27] Microsoft. Toyota motor north america: Dynamics 365, 2023.

[28] N.Zuidhof, B.A.Somaya, O.Peters, and P.P. Veerbek. Defining smart glasses: A rapid review of state-of-the-art perspectives and future challenges from a social sciences' perspective. *Augmented Human Researc*, 6(1), 2021.

[29] Arrigo Palumbo. Microsoft hololens 2 in medical and healthcare context: State of the art and future prospects. *Sensors*, 22(20), 2022.

[30] F. Sherwani, Muhammad Mujtaba Asad, and B.S.K.K. Ibrahim. Collaborative robots and industrial revolution 4.0 (ir 4.0). In *2020 International Conference on Emerging Trends in Smart Technologies (ICETST)*, pages 1–5, 2020.

[31] Gurjeet Singh and V.K. Banga. Robots and its types for industrial applications. *Materials Today: Proceedings*, 60:1779–1786, 2022.

[32] SparkCognition. Augmented Reality in Industry 4.0: A Comprehensive Guide. https://www.spark.tc/augmented-reality-in-industry-4-0-a-comprehensive-guide/, 2023.

[33] ŠKODA Storyboard. Škoda auto tests augmented reality glasses for production line maintenance and technical training, 2023.

[34] Unity Technologies. Plugins. https://docs.unity3d.com/Manual/Plugins.html.

[35] Unity Technologies. Scripting. https://docs.unity3d.com/Manual/ScriptingSection.html, 2023.

[36] Unity Technologies. Welcome to unity, Mar 2023.

[37] David Thier. What Is Pokémon Go And Why Is Everybody Talking About It? https://www.forbes.com/sites/davidthier/2016/07/11/facebook-twitter-social-what-is-pokemon-go-and-why-is-everybody-talking-about-it/?sh=72b1de921758, 2016.

[38] Unity Technologies. Unity Documentation: XR. https://docs.unity3d.com/Manual/XR.html.

[39] Pocket Virtuality. Pocket virtuality.

[40] S.Dudic J.Sulc B.Bajci V.Reljic I.Milenkovi. Augmented reality applications in industry 4.0 environment. *APPLIED SCIENCES-BASEL*, 11(12):5592, 2021.

[41] I . Wohlgenannt, A.Simons, and S. Stieglitz. Virtual reality. *Business & Information Systems Engineering*, 62(5), 2020.

[42] Alan Zucconi. Robotic Arms: An Introduction. `https://www.alanzucconi.com/2017/04/10/robotic-arms/`, 2017.