João Pedro Silva Neves

Application of Mixed Reality Devices for Robot Manipulator Programming: Aspects Related to Graphical Animations

Dissertação de Mestrado em Engenharia Mecânica
na Especialidade de Produção e Projeto

July/2018

· U C ·

UNIVERSIDADE DE COIMBRA

· U C ·

FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

DEPARTAMENTO DE
ENGENHARIA MECÂNICA

Application of Mixed Reality Devices for Robot Manipulator Programming: Aspects related to Graphical Animations

Submitted in Partial Fulfilment of the Requirements for the Degree of Master in Mechanical Engineering in the speciality of Production and Project.

Utilização de Dispositivos de Realidade Mista para Programação de Robôs Manipuladores: Aspetos relacionados com Animação Gráfica.

Author
João Pedro Silva Neves

Advisor[s]
Professor Doutor Joaquim Norberto Cardoso Pires da Silva
Engenheiro Doutor Theo Doll

Jury

President	**Professor Doutor Altino de Jesus Roque Loureiro** **Professor Associado c/ Agregação da Universidade de Coimbra**
Vowel[s]	**Professor Doutor António Fernando Macedo Ribeiro** **Professor Associado c/ Agregação da Universidade do Minho** **Professor Doutor Carlos Xavier Pais Viegas** **Professor Auxiliar da Universidade de Coimbra**
Advisor	**Professor Doutor Joaquim Norberto Cardoso Pires da Silva** **Professor Associado c/ Agregação da Universidade de Coimbra** **Engenheiro Doutor Theo Doll** **CEO na empresa Söhner Kunststofftechnik Gmbh**

Institutional Collaboration

Söhner Kunststofftechnik Gmbh

Koris Robotics Gmbh

Coimbra, July, 2018

ACKNOWLEDGEMENTS

This dissertation is the climax of all the hard work fulfilled during this five years as a student of Mechanical Engineering at the University of Coimbra. I cannot start without acknowledging all the people that supported and guided me throughout these years, whom I am truly grateful to.

To Prof. Dr J. Norberto Pires, advisor, for all the advice and guidance given throughout this Master Thesis, and for providing us all the tools and means to develop this project.

To Dr Theo Doll, advisor, for the internship provided that gave us the opportunity to start this project, and for the *Microsoft Hololens* lent for development.

To Diogo Serrario, my friend and partner in this project, for his collaboration and relentless work given to this project. The outcome of this project was only possible with our teamwork.

To my friends and colleagues in the Robotics Lab, for their friendship and all the advices given that were essential to assure the outcome of this Dissertation.

To all my friends that have been by my side in this 5 years of study, with whom I have unforgettable memories, and whose friendship I could always count.

To my parents and my brother, for all the unconditional support given, for being the greatest role models in my life and for always believing in me

To my girlfriend, that has been by my side during the development of this Master Thesis, and whom I must thank for all the support and courage given.

Abstract

The main goal of this Master Thesis' dissertation consists in describing a new project developed with the purpose to create a completely new interface, which provides a generic implementation of Mixed Reality to monitor, optimize and program robot manipulators' tasks. This interface will hopefully expand the range of people with capabilities to program industrial robots, since the advanced knowledge of programming languages will be an optional requirement. The motivation for this project begins with the will to improve Human-Machine Interfaces (HMI) and to demonstrate that an approach with Mixed Reality can lead to huge improvements in monitoring and programming advanced robotic tasks, such as Additive-manufacturing, collaborative robots, amongst others. Perceiving information as direct visual content, in real time, anticipate trajectories, or even apply modifications to a robot motion with gesture or sound inputs, are some of the great advantages of this technology.

The project developed in this master thesis resulted in two apps for a *Microsoft HoloLens* each with different purposes. The first (*Path Visualization*) is a Mixed Reality interface that allows users to control a robotic manipulator, visualize its movement and fully monitor the task assigned to the robot. The second app (*Graphical Path Manipulation*) is a new Mixed Reality environment that allows users to create a completely new path or adjust one, as well as enabling the possibility to change motion parameters and teach the new/adjusted path trajectory to the robot. This Master Thesis will focus on all the development, made by the writer, to create the Mixed Reality applications for the *Microsoft HoloLens*. This was possible using the *Unity3D*, *Vuforia* and *Visual Studio* software interfaces.

Although this project's purpose is not to focus on a specific industrial task, it can be easily adapted posteriorly to any kind of robotic task desired. The concept however is open for further studies.

Keywords Mixed Reality (MR), Augmented Reality (AR), Robotic Manipulators, Microsoft *HoloLens*, Human-Machine Interfaces (HMI), Task Optimization, Advanced Programming.

Resumo

O objetivo desta dissertação consiste na descrição do projeto desenvolvido com o propósito de criar uma interface completamente nova, que implementa tecnologias de Realidade Mista para a programação e monitorização de robôs manipuladores. Esta interface tem como intuito principal expandir o número de pessoas com capacidades para programar e monitorizar robôs manipuladores, ao tornar o conhecimento avançado de linguagens de programação um requerimento opcional e não fundamental. A motivação para este projeto começa pela vontade de proporcionar avanços em interfaces de comunicação Homem-Máquina (IHM), e na demonstração das vantagens proporcionadas pelas tecnologias de Realidade Mista na monitorização e programação de processos robóticos avançados (Produção aditiva, robôs colaborativos, entre outros). Ter acesso direto a informação como conteúdo visual aplicado no espaço e em tempo real, antecipar trajetórias, ou até modificar o movimento de um robô através de som ou gestos manuais, são algumas das grandes vantagens proporcionadas por esta tecnologia.

O projeto desenvolvido durante a realização desta tese de mestrado resultou na criação de duas aplicações para um *Microsoft HoloLens*, ambas com propósitos diferentes. A primeira aplicação (*Path Visualization*) consiste numa interface de realidade mista que permite aos utilizadores o total controlo sobre um robô manipulador e ao mesmo tempo uma completa monitorização das trajetórias realizadas pelo mesmo. A segunda aplicação (*Graphical Path Manipulation*) consiste num novo ambiente de realidade mista que permite que os utilizadores criem uma trajetória nova para o robô, facilmente ajustável sempre que necessário, e também proporciona a possibilidade de alterar os parâmetros inerentes ao movimento do robô. Todas as trajetórias são depois enviadas pela aplicação para o robô real que as executa como definido. Esta dissertação apenas terá foco no desenvolvimento realizado pelo escritor, que consiste nas aplicações de realidade mista para o Microsoft HoloLens. As aplicações foram realizadas utilizando os softwares *Unity3D*, *Vuforia* e *Visual Studio*.

Apesar do projeto não ser direcionado para uma tarefa especifica, pode ser facilmente adaptado para uma, estando o conceito disponível para avanços futuros.

Palavras-chave: Realidade Mista, Realidade Aumentada, Robôs Manipuladores, *Microsoft Hololens*, Interfaces Homem-Máquina, Otimização de Tarefas, Programação Avançada.

Contents

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AI - Artificial Intelligence
AGV- Automated Guided Vehicle
API - Application Programming Interface
AR - Augmented Reality
CAD – Computer-Aided Design
CAVE - Cave Automatic Virtual Environments
HMD - Head-Mounted Display
HMI - Human-Machine Interface
IDE - Integrated Development Environment
IMU - Inertia Measurement Unit
IGS - Image Guided Surgery
MR - Mixed Reality
OHMD - Optical Head-Mounted Display
RPAR - Robot Programming using Augmented Reality
RPMR - Robot Programming using Mixed Reality
SDK - System Development Kit
SLAM - Simultaneous Localization and Mapping
TCP - Transmission Control Protocol
UAV- Unmanned Aerial Vehicle
GUI - Graphical User Interface
UWP - Universal Windows Platform
VR - Virtual Reality

1. INTRODUCTION

1.1. Motivation

Nowadays technology is evolving at a large scale, and with it human life has drastically changed. We can look back at the 20th century where people did not have access to internet or smartphones, and life would go on peacefully. But imagining a hypothetical situation, where all smartphones would disappear and internet would be shut down in a present instance, human life would not be indifferent which would surely result in a big setback on our daily lives. Therefore, it is safe to assume that humans are dependent of technology, and all the new technologies that start to appear but are not yet indispensable to us, will certainly become indispensable in the future. Emerging nowadays, Virtual Reality (VR), Augmented Reality (AR) and Mixed Reality (MR) are starting to make a big impact on the market, with an increasing tendency to grow. Between these three, Augmented Reality and Mixed Reality are probably the reality changing technologies, that can be applied in a bigger diversity of fields for a greater variety of purposes. In the state of art presented on Chapter 2, it will be possible to see that these technologies are not only applied in the fields of entertainment and gaming, but in fact, a major development is being made on applying these concepts for Marketing, Education, Medicine, Engineering and Robotics.

In the field of robotics these technologies can start a small revolution in Human-Machine Interfaces (HMI). Making an introspective thought on this issue, we can easily understand that improving HMIs is only possible with advances at the interactional level. For example, in humanoid robots we can achieve it with methods of programming by demonstration, even with Artificial Intelligence (AI) and Machine Learning techniques to acquire robot learning capabilities (Schaal, 1999), but what about robot manipulators?

If we think about this, improving HMIs for robot manipulators is not an easy task, and direct programming by demonstration is not always the best solution. The human morphology is different from the robot one, and even if it is possible to demonstrate some actions, there are industrial tasks, more complicated, which are inviable to be demonstrated by a human being. Solving this issue is possible with Mixed Reality technologies, as they can give an unprecedent experience, with a much higher level of interaction at every task designed for a robotic manipulator. The possibilities for displaying different virtual elements on the real world, and to interact with them as if they were real, makes MR technology flexible and easily adaptable by the user.

The main goal for this project consists in applying a new interface for optimization and advanced programming for robot manipulators. There is a lack of development in advanced programming solutions, for robot manipulators, with Augmented and Mixed Reality technologies, and it is still not a major focus of study, which makes the project described in this dissertation a completely new approach on this subject. This dissertation will prove that the features and capabilities provided by these reality bending technologies, not only improve methods for advanced programming, but also provide a new approach on robot programming methods, completely different from the traditional ones, namely, lead-through, walk-through and offline programming (OSHA Technical Manual, 1990).

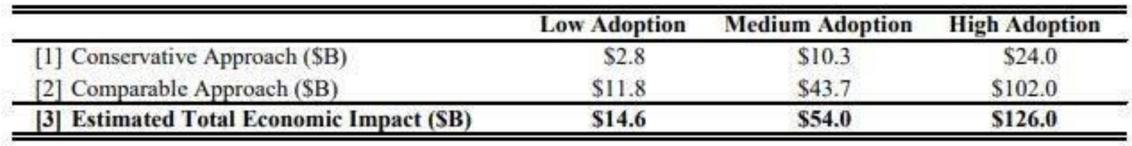
The MR environment used is displayed using the most valuable MR tech device currently on the market, the Microsoft HoloLens. The project will be explained with detail on Chapter 4 and 5, and all the software used to develop it will be described on Chapter 3.

MR, AR and VR environments are still being developed nowadays, and the room for progress is still quite large. In fact, there are still improvements required for a near perfect experience, and recommending this concept to be addressed in future research is only viable if these technologies are truly going to grow, and if the stake on the world economy is sustainable. An economic analysis on the importance of these technologies will be addressed on the next subchapter.

1.2. MR, AR and VR economic analysis

It is important to understand that these technologies, which bend the perspective of reality will have a big impact on the future. Resembling all types of technologies, there are always ups and downs during its development stage, with impact on society and the market economy. As already explained, these technologies cannot be replaced, since the level of interaction and flexibility provided is quite high nowadays, with lots of improvements in the way. Looking at the prospects, on the work by (Christensen et al, 2016), an estimative is made for the projected global economic impact of VR and AR based on the spread of this technologies in the next five years. The analysis is made considering three levels of adoption to overcome the underlying difficulty of predicting these technologies success in the future. The results can be seen on Table 1.1.

Table 1.1- Economic analysis to AR and VR between 2016-2020. Table from (Christensen, 2016)



	Low Adoption	Medium Adoption	High Adoption
[1] Conservative Approach ($B)	$2.8	$10.3	$24.0
[2] Comparable Approach ($B)	$11.8	$43.7	$102.0
[3] Estimated Total Economic Impact ($B)	**$14.6**	**$54.0**	**$126.0**

This analysis traces back to 2016 and it shows that even with a low adoption of these technologies, the total economic impact estimated is still very large. This already gives the notion of the growth proportion that these technologies can endure on the market. On Figure 1.1, a more recent graphic from a 2018 study indicates that the growth of AR and MR market size until 2022 will be exponentially bigger than the increase on the last two years. We can expect for the next four years a greater upsurge on the adoption of these technologies, and without a doubt betting on them is a good strategy.

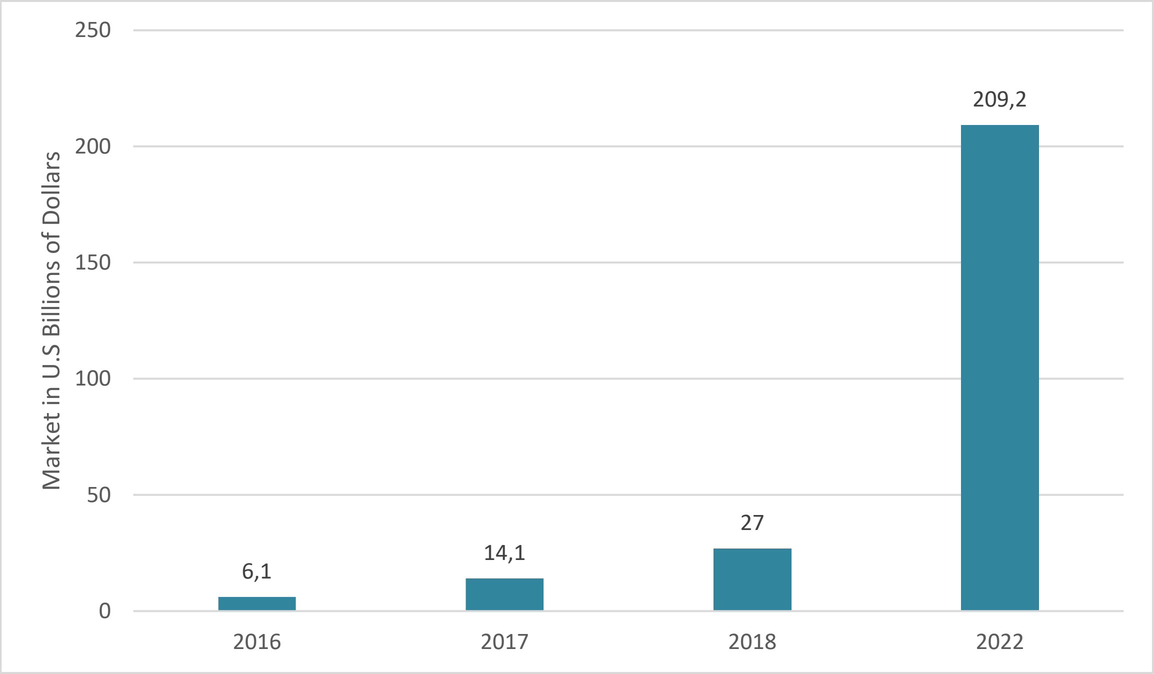


Figure 1.1- Forecast of AR and MR market size worldwide from 2016 to 2022. (Values obtained from https://www.statista.com/statistics/591181/global-augmented-virtual-reality-market-size/ at July 13, 2016)

However, it is also important to understand where these technologies will have a greater impact, and in which fields will undergo a higher focus on development. Figure 1.2 estimates the billions of dollars in the market for these technologies in 2020 and 2025 for different fields of application. Through the assessment of the bar chart it is possible to verify, as expected, that video games represent the biggest impact on the market for these technologies. Predicting an increase of 4,7 billion dollars in the market for AR and VR on video games is quite expectable, as they fit perfectly in this field.

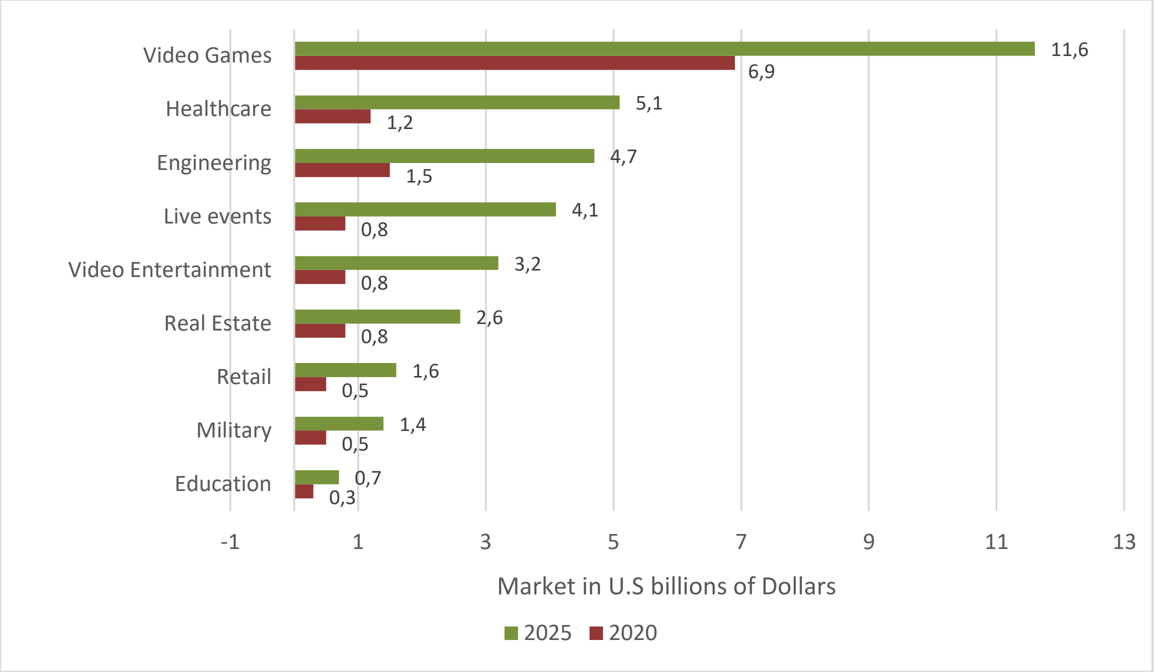


Figure 1.2 – Analysis of VR and AR on the market in billions of dollars from 2020-2025, in multiple fields of work. (Values obtained from https://www.statista.com/statistics/610112/worldwide-forecast-augmented-and-mixed-reality-software-market-by-segment/ at July 13, 2016)

While the major influence of VR and AR on video games is expected, it is worth mentioning that even though VR and AR in Healthcare and Engineering economics stats are vastly inferior for 2020 compared to video games, it is expected a much bigger percentage increase in these fields, growing in 3,9 and 3,2 billion dollars respectively on the market until 2025. This represents a quite large investment, which verifies our assumptions that these technologies can be an upgrade for industrial and engineering tasks, as well in aiding humans on delicate tasks in healthcare fields. Chapter 2 will also describe some examples of developments and apps created resorting to these technologies, for multiple fields of work.

2. STATE OF ART

The definition of reality, and the differences between what is real and an illusion have been well debated, based on the different perspectives and personal experiences that can develop unique ways to see reality. This Chapter will explore the “reality benders” created by men, currently used in different fields of work, with a wide range of applications and devices on the market for different purposes, and vastly recognized as a technology of the future.

2.1. Augmented Reality (AR)

Augmented Reality represents the connection of the real world with a virtual one. This technology allows users to see and create new three-dimensional objects, through a digital display, which are detected by our visual and auditive senses and mixed with the real world to improve it.

An example of this technology is “*Pokemon GO*”, an AR game developed for mobile devices illustrated in Figure 2.1, that allowed the user to see a virtual *Pokemon* in the real world using the live feed of the camera.

Displaying virtual content in the real world is only possible by using a digital display to somehow “manipulate” our senses. In the market, there are different types of displays to implement augmented applications, each with different technological approaches which differ according to the intended visual effect on the outcome. The next sections will introduce some of the displays used for Augmented Reality apps with examples.

Figure 2.1 - AR environment on “Pokemon GO” App

2.1.1. Monitor-based AR displays

The monitor-based AR displays are the simplest and the first type of displays to be used for Augmented Reality experiences. They allow users to see the augmentation on a screen, such as a TV, smartphone or tablet, that normally works as a window, displaying the augmented content based on a capture of the real world, as illustrated on Figure 2.2 (a) and Figure 2.2 (b). This type of displays reflects the world indirectly from the user’s perspective, giving him a non-immersing experience, because the augmentation is based on a real-world capture from a different perspective, like a TV, that displays the augmentation based on a capture from an external Camera. In some cases, such as smartphones or tablets, the device contains an incorporated camera, making it possible to display an augmentation based on the live or stored feed of the real world captured by it.

(a) (b)

Figure 2.2 – Examples of Monitor based AR displays: (a) Augmented Reality App on tablet.; (b) Augmented Reality on TV News broadcasting.

2.1.2. See-through AR displays

The see-through displays allow users to visualize augmentations obtained from a reflection of the real world identical to the user's own perspective of reality. This allows a totally different experience, immersing users in the augmented world, making it feel more realistic and increasing the interactivity. These displays are also called Head Mounted Displays (HMD) since they fit the user heads.

There are two types of see-through displays, video-see through and optical-see through, visualized on Figure 2.3 (a) and Figure 2.3 (b) respectively.

Video-see through displays are usually HMDs with a system of cameras attached, that capture the world in the same way as the user's eyes, implementing virtual content onto that recorded image, creating the augmentation.

Optical-see through devices are HMDs with a semi-transparent glass that project the virtual elements directly in the real world, also called Optical Head Mounted Devices (OHMD). This type of devices requires advanced technologies, but the experiences provided by them are unmatched by any other type of displays. One example of this type of displays is the Microsoft *HoloLens*, which was used to develop the project further described in this master thesis. This device will be presented in detail on subchapter 2.2.1

(a) Acer Windows Mixed Reality AH101 headset

(b) Microsoft *HoloLens*

Figure 2.3 – Examples of See-through AR devices: (a) Video-see through; (b) Optical-see through.

2.2. Virtual Reality

Virtual Reality is a computer-generated experience that fully immerses users into a three-dimensional virtual world, that can be a virtual representation resembling our reality or a completely new environment impossible to see in the physical world. When using this type of technologies, users interact with this environment via auditory and visual feedback, and have the advantage of acquiring a haptic perception of the environment, i.e., a possibility to interact with the environment by touching virtual objects.

This technology is mostly used on virtual reality headsets, that basically are immersive HMDs combined with a controller, which allows users to be fully immersed on the virtual worlds, being able to move inside this environment, “look around” and even interact with virtual features. An example of these virtual reality headsets is Oculus Rift, shown on Figure 2.4.

Virtual Reality is also applied on Cave Automatic Virtual Environments (CAVE) that consists in projecting images and videos into three to six walls of a cubic room. This allows users to acquire a deeper immersive experience, with a wider field of view, although its price is not as affordable as the virtual reality headsets.

Figure 2.4 – Oculus Rift, virtual reality headset

2.3. Mixed Reality

To fully understand the concept of Mixed Reality, (Coppens, 2017) commented and updated the approach by (Milgram et al, 1994) on the reality-virtuality continuum, that represents in one-dimension the range of visual technologies that shape reality between the real and virtual environment. With the RV continuum represented on Figure 2.5, we can understand that Mixed Reality includes features from all the spectrum of these technologies. Mixed Reality enables the creation and visualization of virtual objects in the real world as an Augmented Reality application, and at the same time allows the manipulation of those same objects as a Virtual reality application, allowing a unique interaction between the user, the virtual and the real world. One example of a Mixed Reality device is *Microsoft HoloLens*, which is presented in the next section.

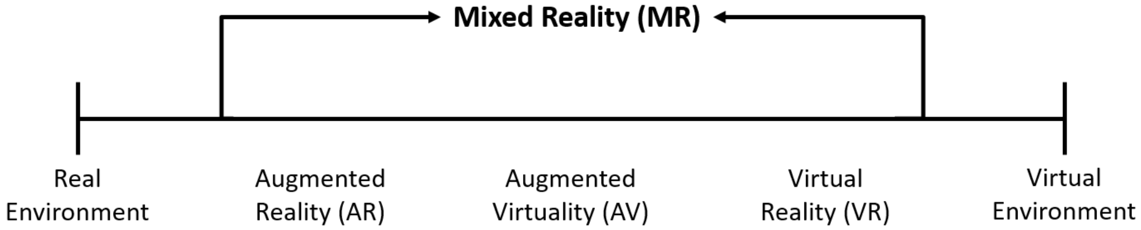


Figure 2.5 – Reality-virtuality continuum. Adapted from (Coppens, 2017).

2.3.1. Microsoft HoloLens

Microsoft *HoloLens* is an HMD device created by Microsoft and launched in March 30, 2016 still being considered as one of the best Mixed Reality smart glasses on the market, with the advantage of working as a computer, running the Microsoft Windows 10 operating system.

This device enables a Mixed Reality experience with a high level of precision and quality, only achievable with the advanced hardware of this device. With the full set of cameras on the device, and the 3 Inert Measurement Units (IMU) attached, *HoloLens* was the first Mixed Reality device to fully create spatial perception capabilities, allowing holograms to stay fixed in a predefined position in real space. The 3 IMUs available, enable the possibility for *HoloLens* to track the position and rotation of the user's head in space, stablishing always a 3D positional connection between the user and the Hologram. The set of environment understanding cameras (4 cameras, 2 on each side) detect the real environment with precision giving the device all the data required to create a mesh of the room where the user is working, by usage of Simultaneous Localization and Mapping (SLAM) techniques, resulting on a holographic model of the real world work environment. With this model, a positional relation between the physical environment and the holograms is stablished. This spatial relationship between the user, hologram and physical environment give *HoloLens* the key for a unique experience.

HoloLens also allows the user to interact with holograms and manipulate them with predefined hand gesture inputs or via sound inputs, changing their position, size and orientation. *HoloLens* can also be considered as a Holographic computer that display every computer functionalities as holograms, possible to interact by gesture or sound inputs

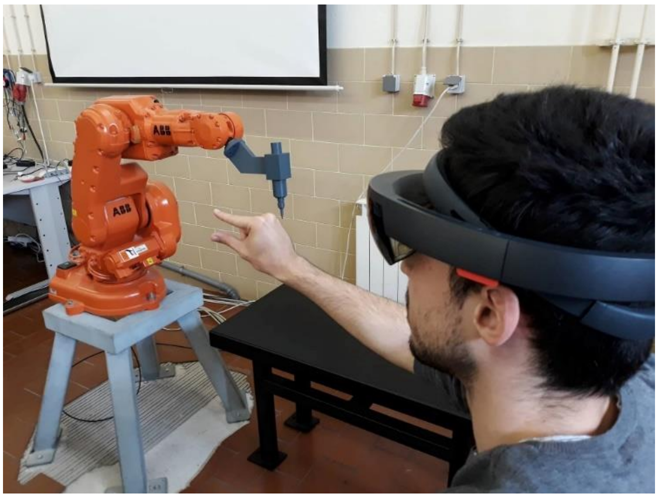


Figure 2.6 – Programmer using a *Microsoft HoloLens*

To achieve the best results for our project, we decided that *HoloLens* would be the most adequate device to use, based on its characteristics and since it's Mixed Reality effects can be quite useful for people with less programming knowledge, to work on task programming and optimization in robotics.

2.4. Advances on Mixed and Augmented Reality

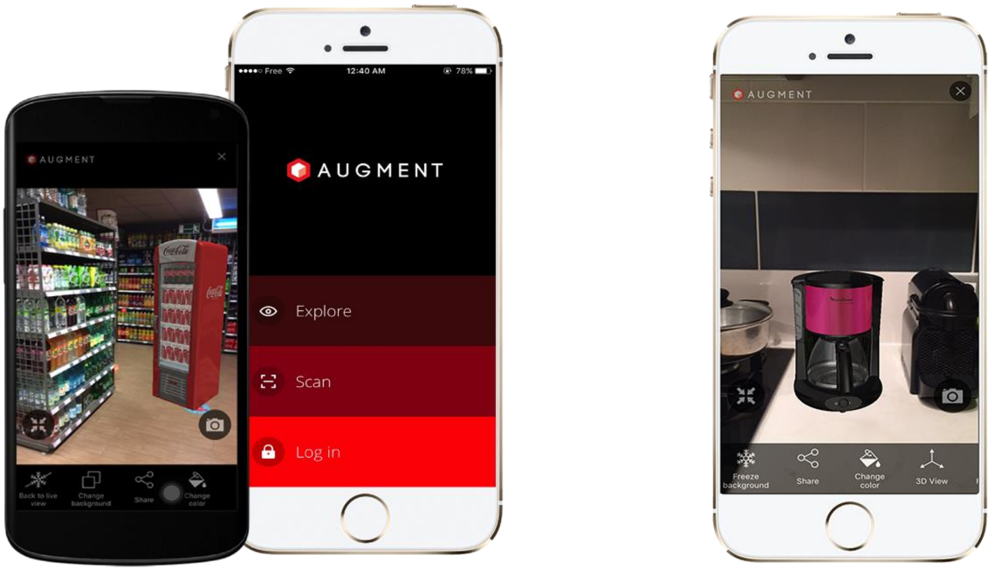
This subchapter will introduce some advances made on the Mixed and Augmented Reality fields in the last years, in several industry sectors, showing how this technology can be applied in numerous fields for different purposes, and why their evolution is surpassing expectations with no limit in sight. We will focus on areas that are going through a big development with AR and MR technologies, such as Sales and Marketing, Education and Learning, Medicine, Engineering and Robotics.

2.4.1. Sales and Marketing

For this type of industry, it is easy to affirm that using Augmented and Mixed Reality can be a huge improvement to simplify the interaction between the client and the product which the company intends to sell. Some examples of apps, more connected to Augmented Reality, will be presented to contextualize this.

1. **Augment**

Augment is an app developed by the *Augment Enterprise* that enables the possibility to see products in 3D in the real environment, using devices with Augmented Reality capabilities and running on IOS or Android operating systems, which provides an interactive way to sell a product. This app is commonly used for business to business (B2B) sales solutions or even for Retail and E-Commerce. In B2B solutions, companies can have access to a database containing all their products, and improve the way the sales representatives' pitch to the client through a more realistic and interactive presentation with enhanced visual impact (and therefore increasing the likelihood of closing deals). Retail and E-Commerce can also be targeted by this app providing them access to the augmented product, thus allowing the client to obtain a detailed view of the product before the purchase.



(a) “

(b)

Figure 2.7 – “Augment” app on mobile devices: (a) “Augment” app being used for defining store logistics; (b) “Augment” app for product visualization before buying.

2. IKEA: Augmented Reality furniture

IKEA is a well-known multinational furniture retail company, that recently created an Augmented Reality app to promote their products to the client. With this app the client can select amongst a database of virtual products and augment them into the real world. This is a great improvement in agility for the client to choose a product and prevent regrets after the purchase.

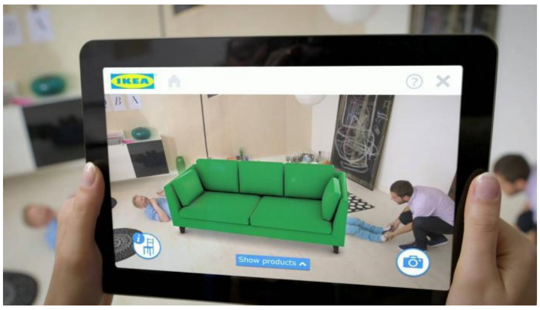


Figure 2.8 – “IKEA: Augmented Reality furniture” app on a tablet device

2.4.2. Education and Learning

In the education field, Augmented and Mixed Reality can work well on improving the way information is transmitted, making it look more “fun” and easy to understand. It was already proven that these technologies can enhance the learning capabilities of a person in a much faster way. In the following topics some examples of Mixed and Augmented Reality on the education field are shown.

1. Mixed Reality on Engineering Education and training

As an introduction to using Mixed Reality for the educational role (Müller et al., 2007) presented a research on using this capability to facilitate the collaborative experimentation, through a set off different environments. In this paper a conceptual approach is made focusing on the different type of possible lab environments based on the user access to them, and the nature of the equipment and their advantages and disadvantages to the learning experience. At last an interesting approach is made to a collaborative learning environment containing virtual and real lab tools, for a complete exploitation of their advantages.

2. Quinta da Regaleira 4.0

“Quinta da Regaleira” is a palace situated in Sintra, Portugal, and a known touristic spot in that region. To promote the interactivity of the visiting experience and to improve the way information about the place is given to the visitor, “Quinta da Regaleira 4.0” app was created. This app uses Augmented Reality capabilities, and gives users the possibility to see augmented interactable personas of the Portuguese culture, placed on specific parts of the palace, from whom is possible to obtain information about the palace as a tourist guide.

Figure 2.9 – Historical figure on real world on “Quinta da Reagaleira 4.0” app. (Image from http://imagensdemarca.sapo.pt/atualidade/uma-experiencia-4-0-na-quinta-da-regaleira/)

3. Anatomy 4D

Anatomy 4D is an app that uses Augmented Reality capabilities. The user has access to a wide range of organs and body parts from the human anatomy, identical to the real ones, providing an in-depth experience on human anatomy. In fact, this app gives students an entirely different learning process for this subject in a more pleasant way, hence improving the learning curve by giving the student a closer experience to reality.

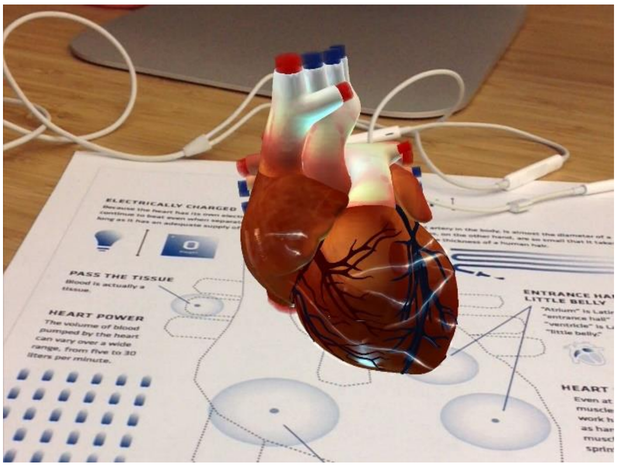


Figure 2.10 – Augmented heart on “Anatomy 4D” app

2.4.3. Medicine

In medicine, doctors and nurses perform tasks that need to be perfectly executed, since a mistake, most of the time, can affect the patient's health, or in some cases even create a life threat situation. Therefore, supplying the professionals with a virtual system that can provide instructions to a specific task, or aid with visual input regarding something occluded to the human eye can be a great mean of improving their effectiveness.

1. Mixed Reality Image Guided Surgery (IGS)

IGS systems allow surgeons to perform surgeries with visual guide instructions using tracked surgical instruments and a collection of preoperative and intraoperative images. In fact, (Kersten-Oertel et al, 2013) presented a detailed state of art on the different applications of Mixed Reality into IGS systems. It is easy to understand from the capabilities of interactivity and the way it blends with the real world, that Mixed Reality has great utility in this area. Figure 2.11 illustrates how Mixed Reality is already being applied for IGS, with an under-development application using *Augmedics xvision* Head-Mounted Device (HMD), which allows users to see augmented x-ray images of the patience, guiding the surgeon through the process.

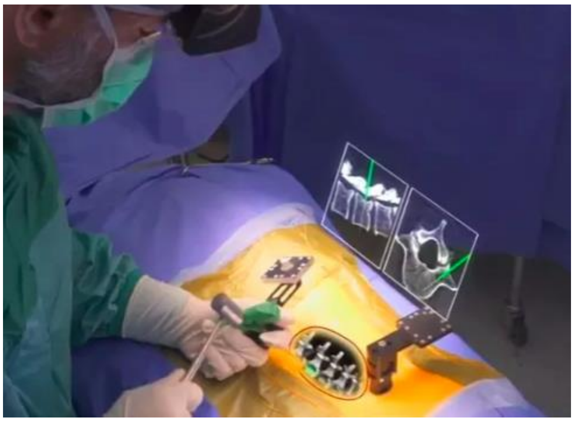


Figure 2.11 – Mixed Reality example on IGS.
Augmedics xvision HMD

2. AccuVein

Accuvein is a great example of a simple device on the medical field that can be very helpful for nurses or doctors when applying vaccines or withdrawing blood, since sometimes it is hard to successfully find veins on people. This device projects onto the patient arm, an image containing the highlighted veins on the correct location, which represent a simple solution for a common problem in this field.

Figure 2.12 – Augmented veins on patient using *Accuvein* app

2.4.4. Engineering

Engineering is one of the fields where AR and MR are more focused right now as a tool for improving different subjects. A lot of processes require a thorough execution by the operator, where Augmented and Mixed Reality can enter and facilitate processes preventing errors and misconceptions. Therefore, this technology can be useful to instruct engineers, for maintenance purposes, prototyping or even for collaborative work in a shared experience.

1. **Mixed Reality on Learning Factories**.

Learning factories are now being introduced in several companies and in partnerships between the industry and Universities to maximize the practical knowledge for engineers. In the 4.0 industry era, manufacturing requires the knowledge of more complex subjects such as advanced analytics, internet of things (IoT) and advanced monitoring processes, which most of the time are not taught in an ordinary university class. Learning factories provide a way for companies to teach trainees by demonstration, which is responsible for better teaching results, without breaking or altering the production timeline. This is an advantage to both parts, and as explained on the work by (Jurascheck et al., 2018), Mixed Reality can be implemented in this field. As previously mentioned, Mixed Reality can be a great tool for increasing the effectiveness of learning methods, and in this case, it can improve the learning process within those factories while it simplifies and helps in the manufacturing and producing processes. Some applications for Mixed Reality on learning factories are well illustrated on Figure 2.13.

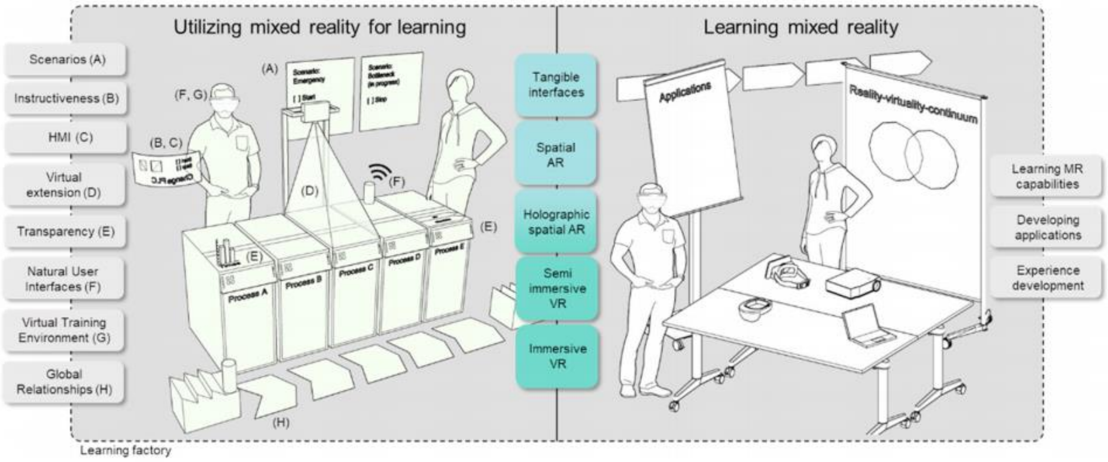


Figure 2.13 – The potential field of applications of Mixed Reality in learning factories.
Image from (Jurascheck et al, 2018)

2. I-Mechanic – AR Car Repair App

I-Mechanic” is an app developed by AR Media that guides the user through several tasks related to car maintenance and repair. It allows users to see augmented instructions for common maintenance tasks and even points out for near mechanics or auto part stores if needed. A major advantage of this application is its compatibility with a wide variety of car models, and despite the concept’s simplicity it can be improved for harder tasks. The app is shown on Figure 2.14

Figure 2.14 – “I-Mechanic Car Repair” app

3. Microsoft HoloLens on Engineering prototyping and construction

As already described on chapter 2.2.1, *Microsoft HoloLens* is the leader on the Mixed Reality technology and nowadays it is being used on engineering projects in several companies. With software’s that allow the usage of CAD models and the deployment of apps for these devices, it is possible to create ways to visualize prototypes and models in 3D space. (Kosowatz, 2017) describes two applications developed for engineering purposes, using the *Microsoft HoloLens*, and made by Trimble and the University of Cambridge. The first app is called *Project Monitoring* and places holographic models within the real structure while is being built, allowing the user to monitor the process and find errors during the construction.

The second app is used for bridge inspections. This app extracts images from photos taken by workers at site which are then uploaded to a cloud service, that uses them to render a holographic representation of the bridge onto reality, where the user can walk around it and detect any structural anomalies.

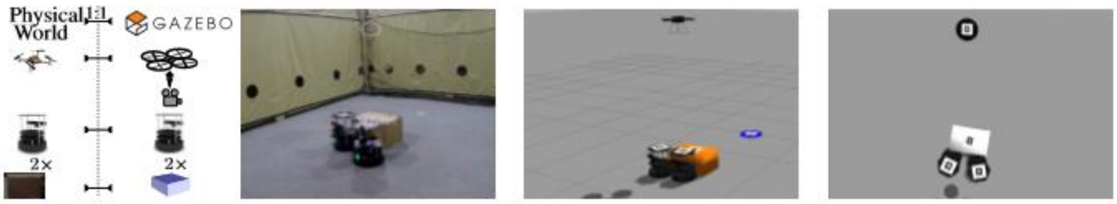
Figure 2.15 – “Project Monitoring” App by Trimble being used to aid in the construction of a system of pipes

2.4.5. Robotics

At last, Robotics is another field where augmented and Mixed Reality can play an important role. Interfaces for Human-machine communications are something that engineers and enthusiasts try to improve every day, continuously simplifying and optimizing them to apply on this field. In robotics, Mixed Reality can be the asset responsible for a completely new type of HMIs, thus simplifying how to see, in real time, important information regarding the robot and the process it is running, for a more precise monitoring. However, improving HMIs is not the only achievement that AR and MR can bring to this field. These technologies can facilitate the programming of industrial robots’ tasks, and add better options to apply corrections on them. In this topic, some work made in this area will be exposed and it will be understood why Mixed Reality can be a tool to further development in this area, considering it still lacks influence on it.

1. Mixed Reality on Mobile Robots

Nowadays mobile robots are presented in several shapes and for different purposes, as an improvement to replace AGV’s for industrial transport, in room scanning and behavioural detection with advanced AI and computer vision technologies, quadcopters for recreational purposes or even drones in the military field. When thinking about robots, it is easy to see how technologies like AI and Computer Vision are being applied in this field, to give decision making and environment scanning capabilities to robots., however, Mixed Reality can be amongst them as exposed in the work of (Chen et al, 2009) and (Hönig et al, 2015). The first paper presented a simulation model using Mixed Reality technologies for mobile robots, as well as its advantages well explained in an easy comprehendible manner. In fact, it is easy to understand that by adding virtual elements on the physical world, imposing physical restrictions and treating virtual elements as real ones, it is possible to realize different tests and provide learning examples to a robot. This can be of great use to simplify the development stage of a robot, minimize testing costs and even to reduce the required time to achieve a final product. The second paper is recent and demonstrates how Mixed Reality can be used to reduce gaps between simulation and implementation, or even be used as additional capabilities to apply on robots. Figure 2.16 illustrates one of the demonstrations presented on the paper, describing how Mixed Reality can be an additional step between simulation and implementation. While maintaining a relationship between the virtual and the real environment, after a complete virtual simulation, additional simulations can be realized replacing virtual elements with real elements, dividing the implementation process in more flexible stages, easier to analyse and to detect failures.



a) MR Components **b)** Physical World **c)** Virtual World (Gazebo) **c)** Virtual Camera

Figure 2.16 – Using Mixed Reality on simulation techniques. “Two TurtleBots collaboratively move a box using the guidance of a physical UAV with a virtual downward facing camera simulated with GAZEBO. The pose of the robots and the box are estimated using AR tag detection”. Image from (Hönig et al, 2015)

2. Mixed and Augmented Reality on Robotic manipulators

Advancements on Human-Machine Interfaces for robotic manipulators are not easily accomplished, and direct programming by demonstration is not the ideal solution. Nevertheless, it is possible nowadays, where MR and AR can be a solution to it. In fact, there are already some advances on this subject as shown in the work of (Michalos et al., 2015), where it is described an AR System, that enables users to monitor every process and task in a robotic system with visual feedback about the processes, the robot state and displaying visual instructions. Systems like these simplify the work of the operator in monitoring processes for industrial robots. On the other hand, (Guhl et al., 2017) introduced the concept of applying Mixed Reality on robotic manipulator systems, using devices like *HoloLens*, and connecting the AR device with the robot system through an intermediary server, illustrated on Figure 2.17.

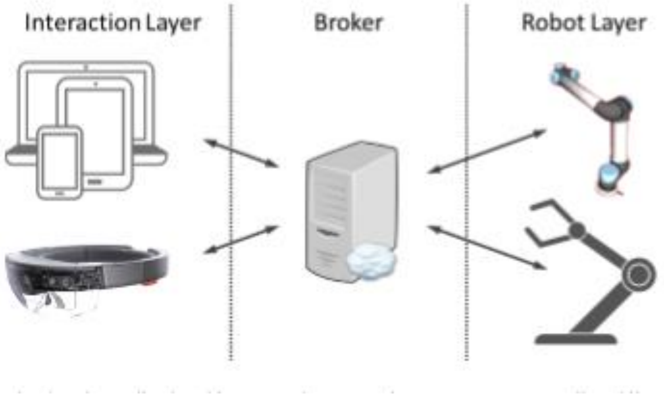


Figure 2.17 –Communication Architecture on (Guhl, J et al 2017)

However, programming robot manipulators with Mixed Reality still lacks new developments, and there is not enough research considering its usefulness. In fact, (Chong et al, 2009) already introduced an approach to the usage of Augmented Reality technologies for Robot programming, where users have access to a virtual model of the Robot, and can teach it a trajectory by defining the last position using marker-based techniques, i.e. visual markers detectable with computer vision methods. The concept its only applied on a virtual model however, providing limited features, which can be more useful for simple testing and simulation than for real time programming of industrial robots.

Nevertheless, this dissertation will introduce an interface to monitor robot manipulator's tasks in a Mixed Reality environment, explained on Chapter 4, and a new method for Robot Programming using Mixed Reality (RPMR) described on Chapter 5, that allows users to directly program a real robot motion, in a Mixed Reality environment where users interact with virtual elements using gesture inputs. This is a completely new approach that provides users with a programming method far more intuitive and much more useful on an industrial level. Table 2.1 summarises the benefits for Robot Programming using Mixed Reality (RPMR) comparing it with the concept of Robot programming using Augmented Reality referred on the work by (Chong et al, 2009) and other traditional methods. In fact, RPMR safety issues are inexistent as there is no direct contact with the real robot or no need to maintain proximity to it, and the interaction level provided is surpassed by no other.

Table 2.1- Comparison between RPMR, RPAR and other traditional methods. Adapted from Table 1 on (Chong et al, 2009)

	Robot Type	**Safety issue**	**Working Environment**	**Interaction Level**	**Intuitiveness**
RPMR	Real	No	Real	Very High	High
RPAR	Virtual	No	Real	High	High
Lead-Through	Real	Yes	Real	Medium	Low
Walk-Through	Real	Yes	Real	High	High
Offline	Virtual	No	Virtual	Low	Low

3. SOFTWARE

As initially presented in this dissertation's abstract, the project developed during this semester resulted in two different apps in an initial approach to optimize and program robotic manipulator's tasks, promoting the interaction between human and manipulator, using Mixed Reality. This project was developed as a partnership with my colleague Diogo Serrario, and the project description will be equally divided in our master thesis reports. This chapter will present and explain all the software used, being them Unity3D, *Vuforia, Visual Studio* and *RobotStudio*. However, for the *Microsoft HoloLens* apps only Unity3D, *Vuforia* and *Visual Studio* were required, and by that reason this dissertation focuses only on the development made using these three programs. *RobotStudio* took part in the implementation system for this project, described on my colleague's Master Thesis.

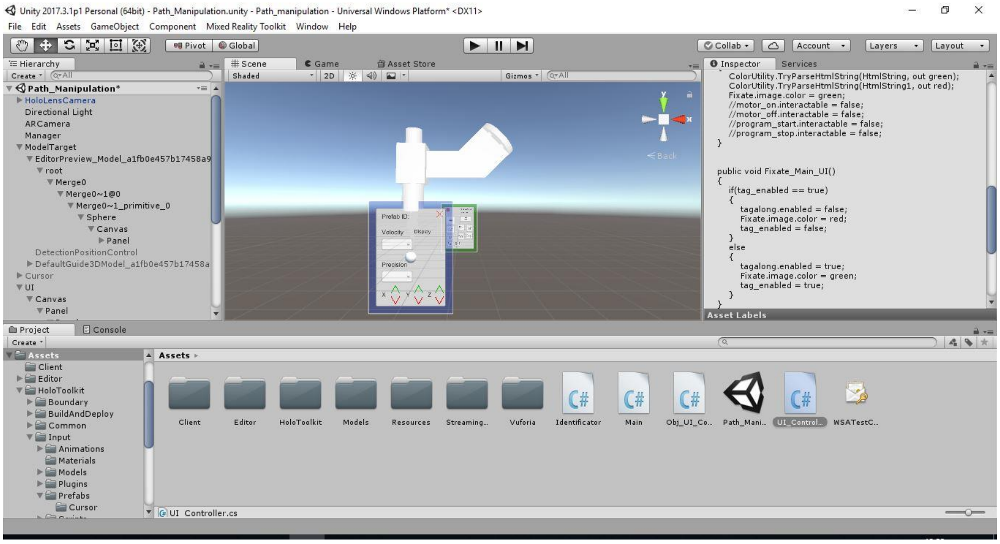
3.1. Unity3D

Unity3D is a cross-platform game engine and Integrated Development Enironment (IDE) created by *Unity* technologies. It uses a graphical interface easy to comprehend and it is a widely used software in the gaming industry, especially for mobile apps. *Unity3D* is also used to make VR and AR apps for different platforms, and since version 2017.1 *Vuforia* SDK is already integrated on the program.

The reason why it was chosen to develop this project, besides its effectiveness, is because it is recommended by Microsoft to create Mixed Reality apps for *HoloLens*,

Regarding *HoloLens'* development, C# is the programming language used in the software, with *Visual Studio* working as the programming platform. When a project is completed, *Unity*3D creates an intermediary application, that posteriorly needs to be recompiled by *Visual Studio* to be ready to deploy on *HoloLens*.

Unity3D stays as the main platform to develop our "Virtual World".



```
    ColorUtility.TryParseHtmlString(HtmlString, out green);
    ColorUtility.TryParseHtmlString(HtmlString1, out red);
    Fixate.image.color = green;
    //motor_on.interactable = false;
    //motor_off.interactable = false;
    //program_start.interactable = false;
    //program_stop.interactable = false;
}

public void Fixate_Main_UI()
{
    if(tag_enabled == true)
    {
        tagalong.enabled = false;
        Fixate.image.color = red;
        tag_enabled = false;
    }
    else
    {
        tagalong.enabled = true;
        Fixate.image.color = green;
        tag_enabled = true;
    }
}
```

Figure 3.1 – *Unity3D* interface

3.2. Vuforia

Vuforia is a System Development Kit (SDK), that allows the creation of Augmented Reality apps for mobile devices. Using Computer vision technology, enables *Vuforia* functionalities to recognize and track planar images or even 3D objects. The latest feature developed by *Vuforia* is called “Model target” and allows the recognition and tracking of a 3D object, based on the CAD model of the object.

Vuforia provides APIs in C++, Java, Objective-C++ and to the .NET languages with an extension for Unity3D.

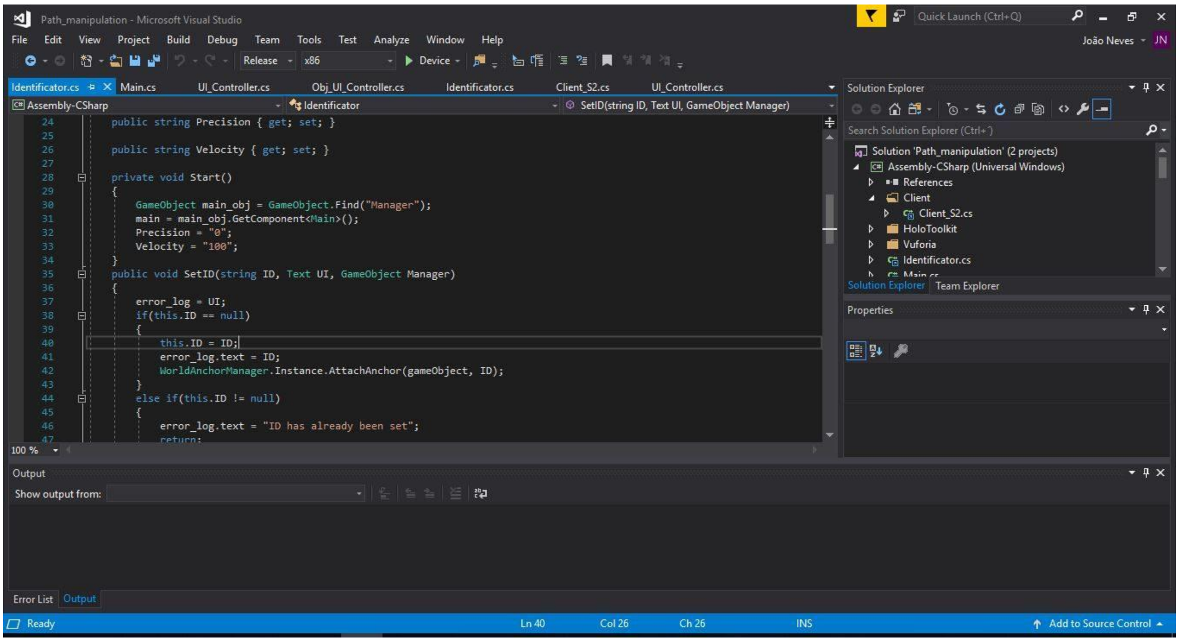
This SDK is used in our project because it is implemented as an extension for *Unity3D*, and it allows the detection of 3D models based on CAD files.

3.3. Visual Studio

Microsoft Visual Studio is an IDE developed by Microsoft, created to develop software dedicated to the .NET framework and its languages. It is used to develop computer programs, web sites, web apps, web services or even mobile apps.

It is also possible to install a plugin that enables the connection between *Visual Studio* and *Unity3D*, making *Visual Studio* the programming interface for the game engine.

In our project, *Visual Studio* is also used to deploy the final app for *HoloLens*, built by *Unity3D* as a Universal Windows Platform (UWP) application.



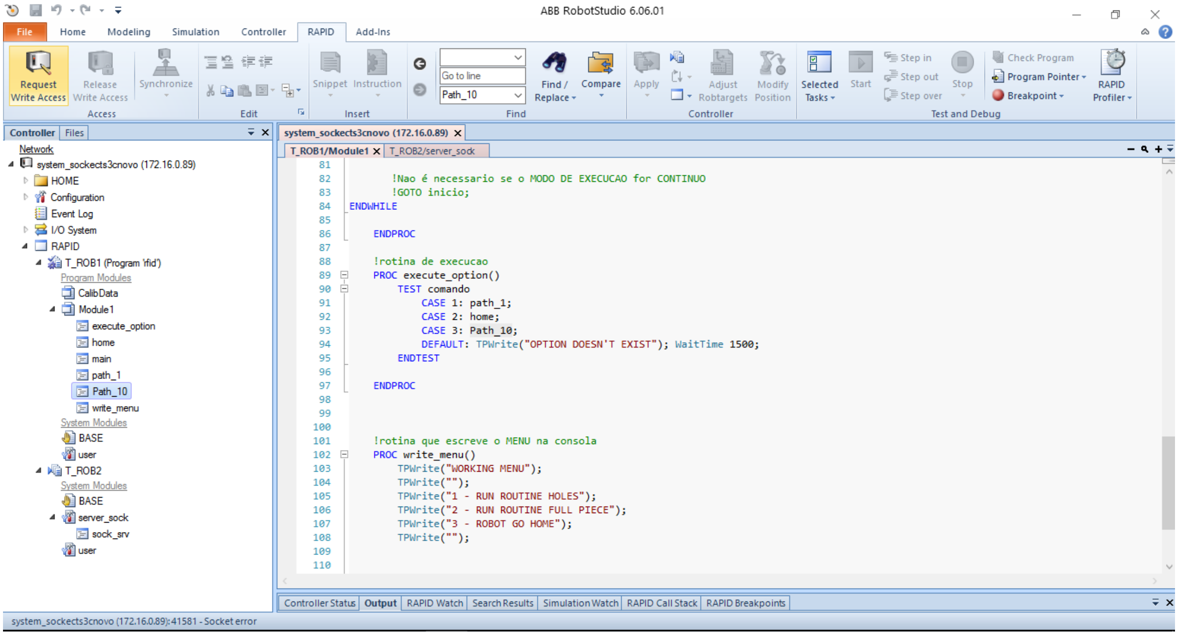
```
    public string Precision { get; set; }

    public string Velocity { get; set; }

    private void Start()
    {
        GameObject main_obj = GameObject.Find("Manager");
        main = main_obj.GetComponent<Main>();
        Precision = "0";
        Velocity = "100";
    }
    public void SetID(string ID, Text UI, GameObject Manager)
    {
        error_log = UI;
        if(this.ID == null)
        {
            this.ID = ID;
            error_log.text = ID;
            WorldAnchorManager.Instance.AttachAnchor(gameObject, ID);
        }
        else if(this.ID != null)
        {
            error_log.text = "ID has already been set";
            return;
```

Figure 3.2 – *Visual Studio* Interface

3.4. RobotStudio

RoboStudio is a software of graphical simulation and programming from *ABB Robotics*, which enables users to fully design robotic cells. This software uses virtual controllers that run the exact same operational software of the real controllers, which makes it particularly useful to develop solutions, using the PC. The *RobotStudio* environment has the advantage of running a simulation environment that supports CAD files, thus it is intuitive to work with, and at the same time, allows a reliable programming for industrial tasks. It also allows to stablish communication protocols, with servers and clients to share and receive data from other platforms.

RobotStudio is the software used due to its usefulness on robot programming and accepting communications with other devices, as well as being the recommended software for programming the robot manipulator used (*ABB IRB140*).



```

        !Nao é necessario se o MODO DE EXECUCAO for CONTINUO
        !GOTO inicio;
ENDWHILE

    ENDPROC

    !rotina de execucao
    PROC execute_option()
        TEST comando
            CASE 1: path_1;
            CASE 2: home;
            CASE 3: Path_10;
            DEFAULT: TPWrite("OPTION DOESN'T EXIST"); WaitTime 1500;
        ENDTEST

    ENDPROC



    !rotina que escreve o MENU na consola
    PROC write_menu()
        TPWrite("WORKING MENU");
        TPWrite("");
        TPWrite("1 - RUN ROUTINE HOLES");
        TPWrite("2 - RUN ROUTINE FULL PIECE");
        TPWrite("3 - ROBOT GO HOME");
        TPWrite("");


```

system_sockects3cnovo (172.16.0.89):41581 - Socket error

Figure 3.3 - “ABB RobotStudio” programming environment

4. PATH VISUALIZATION

In this chapter, the first application developed in this Master Thesis project will be introduced. This app uses HoloLens Mixed Reality capabilities to visualize a robot trajectory for real time monitoring of the process, thus improving the analysis of errors and misconceptions by users. This is a first approach on a generic visual interface that can be adapted for different processes such as material deposition, additive manufacturing or machining.

Firstly, a theoretical description of the application and the communication protocol between the Robot and the *HoloLens* will show the way this app was devised and constructed. Secondly, technical descriptions will present important features developed and required for the successful functioning of the app.

4.1. App description

Path visualization was the first approach made in this project to implement Mixed Reality features for a robotic manipulator. However, it underwent several changes until reaching the final system. The greatest challenge addressed was how to create a holographic line to follow the robot movement in real time, constantly being able to control it without losing quality on the line and maintaining a correct spatial alignment. The plan needed to guarantee these conditions, is divided in several tasks with different purposes, that combined made our goal achievable. Those tasks are the following:

1) **Merging two worlds** -To ensure that the robot trajectory displayed on *HoloLens* is correctly adjusted to the movement in 3D, assuring that the three-dimensional world axis from both platforms are connected is critical. Unfortunately, as predicted, *RobotStudio* and *Unity3D* representations of the real world do not connect with each other, i.e. a point with the same 3D coordinates in both platforms will stay represented in different places on the real world. Overcoming

this issue requires the definition of a local set of axes, equal between both “worlds”, and matching the starting positions of the robot (real world) with the holographic world. This was possible through the object tracking capabilities used, that are explained on subchapter 4.3.

2) **Transferring Data between Robot and *HoloLens* -** Another task necessary to be performed is a way to transfer all the required information between the Robot and the *HoloLens* and *vice versa*. The solution to this issue demanded a stable and quick way of communication to achieve the best results. Therefore, the communication protocol developed to this task will be described on subchapter 4.2.1.
3) **Commanding process using Mixed Reality -** At last, the app also required a protocol to communicate with the Robot controller and control its functionalities, in this case using Mixed Reality. This was possible due to the creation of a Graphical User interface (GUI), with available options to ensure the good functioning of the app and start communications with the Robot controller. This GUI and commanding process will be thoroughly described on subchapter 4.2.2.
4) **Adaptive Holographic Line -** Another requirement for this app consists in a good quality holographic line, adapted in real time, that follows the movement of the real robot, without gaps. A technical description with some key lines of code regarding this achievement is exposed on subchapter 4.4.

4.2. Communication mechanism

To attend every option required for a correct usage of this app, two different protocols were devised to attend two different needs at the same time, both represented on Figure 4.1. The first protocol (represented in a green path on Figure 4.1) works from the robot controller to the *HoloLens,* and it is responsible for transferring the required amount of positional data in real time, to display the holographic path line. The second protocol (represented in a red path on Figure 4.1) works from the *HoloLens* to the robot controller, enabling the possibility to send "commands" that oblige the robot to do the tasks required by the user.

This master thesis only addresses technical terms relative to the developed server and client in C# for the *HoloLens* device.

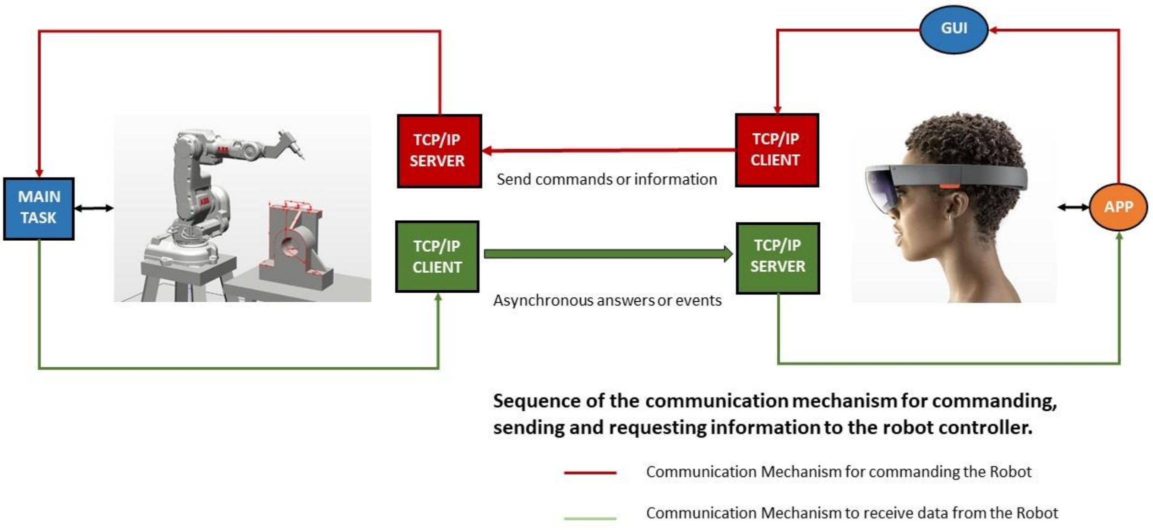


Figure 4.1- Scheme of the Communication Mechanism used in the "Path Visualization" app

4.2.1. Protocol to send Robot Positions

To represent the path of the robot in a holographic line, requires a large amount of positions to update it smoothly, and represent it perfectly. In fact, the number of positions and the time interval between communications required to produce a good path representation are dependent of the robot velocity. For that reason, it was necessary to develop a system where data could be transferred in small amounts of time, without blocking the app. To ensure this, a Transmission Control Protocol (TCP) asynchronous server was created for this app, which is initialized by the user, and once it starts, waits on a specific port, for data packages to be sent by the robot. Considering that many communications will occur during a robot task, only a non-blocking asynchronous approach for the server can execute this without the app losing performance. The reason to choose a TCP/IP protocol, relies on the advantage of securing that every data package is received and reaches its destination.

The protocol to send robot positions is easily explained on Figure 4.1, following the green path. It works with asynchronous events, started with a command sent by the *HoloLens* App, following the red path. Firstly, the cadence of asynchronous events is established by setting the time interval between trajectories to be sent by the robot, which is defined on a command sent to the robot controller and started on the *HoloLens* app's GUI. Afterwards, trajectories are cyclically fetched from the robot, respecting the time interval defined, and sent through the TCP robot client created on *RobotStudio*, to the *HoloLens* TCP Server, one by one, in asynchronous events. Every time the server receives this data, it is passed on and treated in the App's main code.

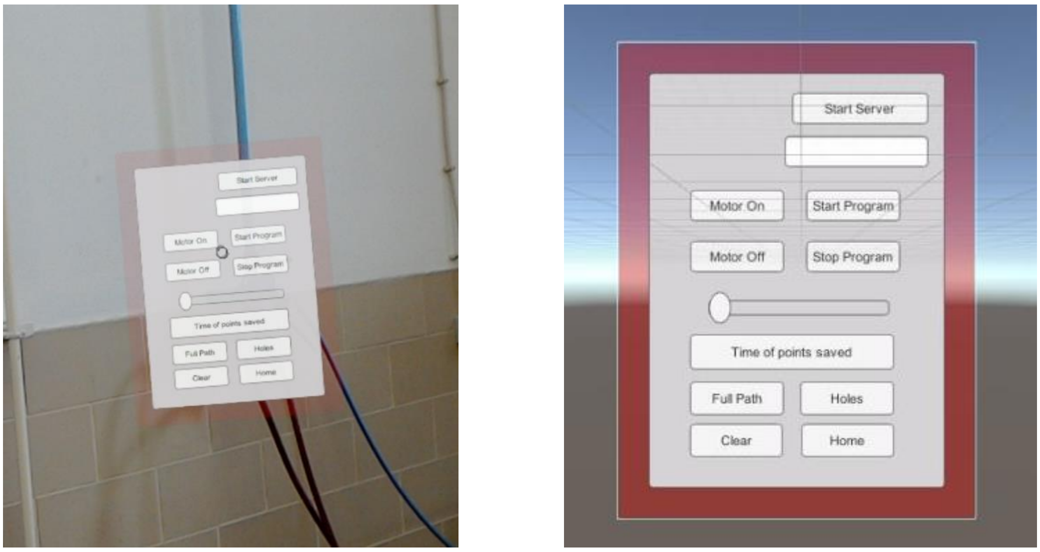
The *HoloLens* TCP server was created on *Unity*, consequently being written in C#. When building an app for UWP on *Unity*, most of the *.Net.Sockets* APIs are not available, because *.Net* for UWP apps include only a subset of the types provided by the *.Net* Framework. This created numerous issues when trying to develop this server. However, overcoming these issues was possible by using the *SocketAsyncEventArgs* class, which is part of a set of enhancements to the *System.Net.Sockets* namespace, and provides an asynchronous approach to develop high performance socket applications. The server was developed using an updated approach to this app, from the work developed by (Schueltz, 2010).

4.2.2. Protocol to command the Robot

The protocol to command the robot is easily understood, following the red path on Figure 4.1. The protocol is established from the *HoloLens* to the robot, with a TCP client placed on the *HoloLens* that sends commanding messages to a robot TCP server, based on the option selected by users, in a main GUI to control the entire system. The GUI, shown on **Erro! A origem da referência não foi encontrada.** (a) and (b), gives users the possibility t o control the robot's motors, the program state, the cadence of points to be sent by the robot TCP client, and options regarding the MR App, such as starting the server or even clear the visible line.

This TCP client was also developed using the same class referred on subchapter 4.2.1, and it answers to click events from the GUI predefined buttons, sending a specific command string that triggers the required event on the *RobotStudio's RAPID* code. After data is successively sent, the TCP client is closed.

This protocol can work while the *HoloLens* is receiving robot positions without blocking the app, which is essential for this application.



(a) (b)

Figure 4.2- *Path Visualization* App's GUI; (a) GUI visualized on the *HoloLens*; (b) GUI visualized on the Unity Editor.

4.3. Object tracking

This subchapter will not focus on how the object tracking was developed for this project, but only on how it solves the issues of different 3D representations between both systems. The tracking method by *Vuforia*, enables the possibility to detect a physical object based on its CAD file, and when detected, augments a white virtual object equal to the physical one, and places it on the physical object's position, as illustrated on Figure 4.3 (b).

In this app, a detection of the robot's tool is made, and the virtual representation of this tool is the key element to merge the 3D representations. Basically, the virtual tool that is augmented, is treated as an object in *Unity*, with several components attached that allow the correct rendering process of the object. Within the components inherent to these objects, there is always one called "Transform". This component basically assigns a local referential to the object, that can be positioned in relation to the world referential or to the transform of the object's parent. If the real tool is well placed, after being detected by the *HoloLens*, the virtual object is settled in its position and it is guaranteed that the local referential of the virtual object stays calibrated with the local referential of the tool inside the robot's simulation environment. After calibrating, and knowing the position of the first point (robot tool's tip) we can work with positional data from the robot, by applying some vectoral work relative to the local referential. The object tracking procedure is illustrated on Figure 4.3 (a) and (b). After the tracking is done, the app starts as visualized on Figure 4.4.

The technical procedure made to create the object tracking used in this app will be explained in Diogo Serrario's Master Thesis.

(a) (b)

Figure 4.3 - Different stages of the Object tracking process in "Path Visualization" app: (a) Before tracking. Matching the guide lines with the real tool; (b) After tracking. Virtual object rendered on the real tool's position

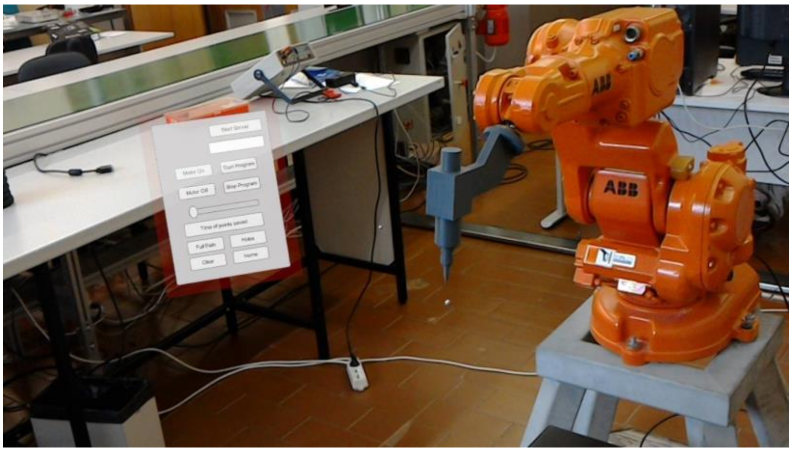


Figure 4.4 - App stage after tracking

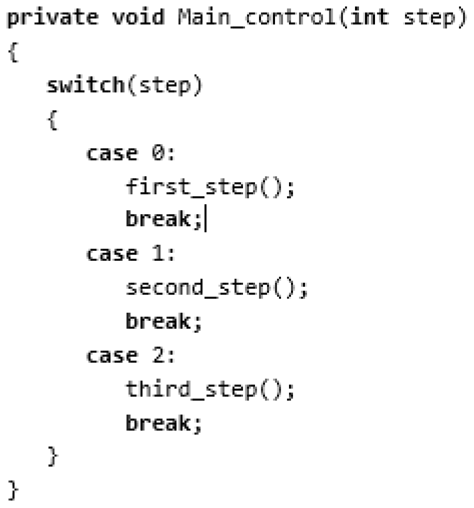
-

4.4. Holographic Path

The Holographic Path visualized in this app allows users to clearly see the trajectory made by the robot tool since the beginning of the process. In fact, the Holographic line is a simple trailing of the robot's tooltip, but if needed, it can be easily adapted to any kind of process the user wants to monitor. The main code on this app is organized in three main step functions, each representing different stages within the app, as described on Figure 4.5, and runs in in the Update function (Looping every frame) as shown on Figure 4.6. The app starts at the first stage ("*first_step()"* on Figure 4.5), where the object tracking features are on, and the user needs to match the virtual object guide lines with the real tool as shown on Figure 4.3 (a). After the tracking is done, the app enters the second stage. In this stage all the components in *Unity*, relative to the object tracking feature and to the rendering of the Virtual tool, are destroyed or disabled, and a virtual sphere is placed in the robot tool's tip as exposed on Figure 4.4. This virtual sphere represents the first position of the tooltip and starting point for the holographic line, treated as a child of the *Model Target CAD tool object.*

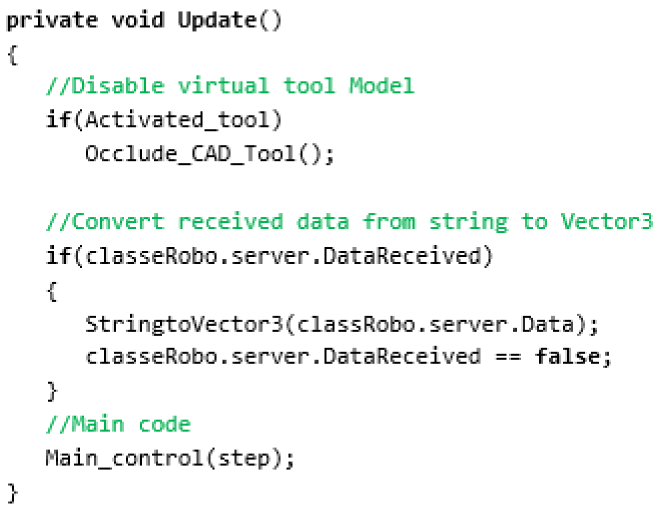
At last the app enters the final stage that works until the end of the running state, and executes all the code required to create and update the holographic line visualized on the

Mixed Reality device. To implement this, the *LineRenderer* feature from *Unity3D* was used, which allows the creation of a virtual (holographic) line that goes through specified vertex positions given.



```
private void Main_control(int step)
{
    switch(step)
    {
        case 0:
            first_step();
            break;
        case 1:
            second_step();
            break;
        case 2:
            third_step();
            break;
    }
}
```

Figure 4.5- Code Snippet from *Unity* C# Main code. The code is divides in three main stages.



```
private void Update()
{
    //Disable virtual tool Model
    if(Activated_tool)
        Occlude_CAD_Tool();

    //Convert received data from string to Vector3
    if(classeRobo.server.DataReceived)
    {
        StringtoVector3(classRobo.server.Data);
        classeRobo.server.DataReceived == false;
    }
    //Main code
    Main_control(step);
}
```

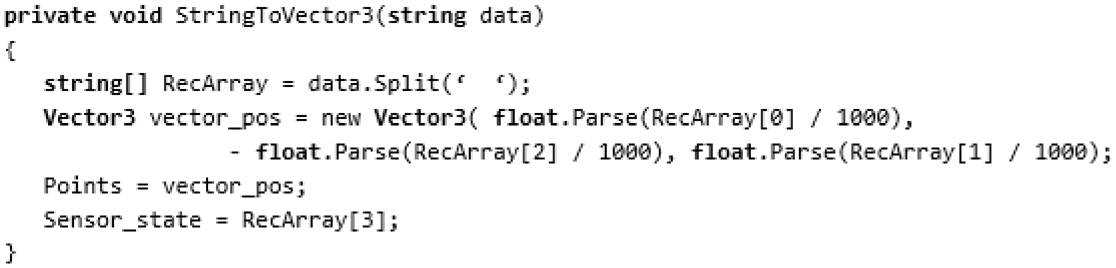
Figure 4.6- Code Snippet from *Unity* C# Update function. The update function is responsible for running every frame the method to convert positional data received in the server, and the main code.

4.4.1. Converting Robot Positional Data

Every time the *HoloLens* receives a string with positional data from the robot, it needs to be stored assuring it does not get lost. Therefore, the main code in this app is prepared to pick-up data every time it reaches the server and convert it into the *Unity3D Vector3* struct, that is used as the struct in 3D to store positional data for virtual objects, as shown on Figure 4.7.

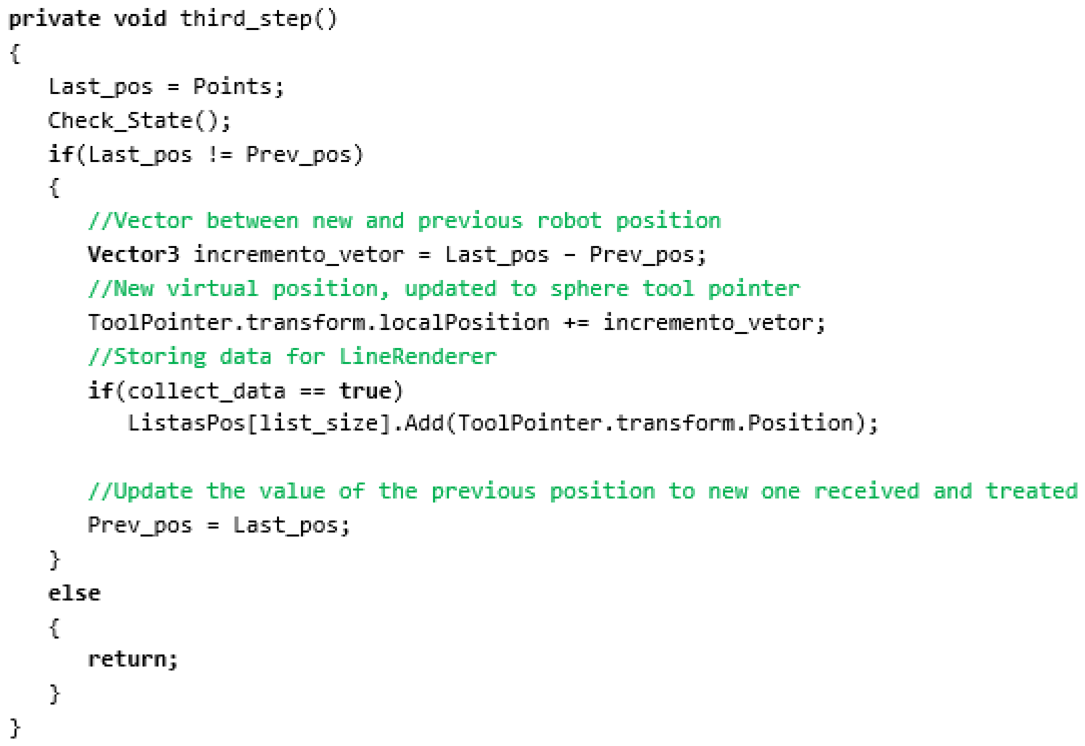
In fact, the conversion method is not only a change of structures. After converting from string to Vector3, the values obtained represent the positions in the robot environment and not the virtual ones. This issue was solved working with vectors instead of positions. Every time a new robot position is received, the vector between the last point and the new one is calculated. This new vector represents the linear translation between the two robot positions. Adding it to the last virtual position saved, the new virtual position of the robot tool, in the MR environment, is obtained. The code created for this method is exposed on Figure 4.8.

All these virtual positions are stored in a dynamic array, using the *List <T>* class, making them ready to be used to set the *LineRenderer* vertex positions. If by mistake the robot sends a position equal to the one sent before, the List is prepared to reject it, avoiding storage of useless data.



```
private void StringToVector3(string data)
{
   string[] RecArray = data.Split(‘  ‘);
   Vector3 vector_pos = new Vector3( float.Parse(RecArray[0] / 1000),
                 - float.Parse(RecArray[2] / 1000), float.Parse(RecArray[1] / 1000);
   Points = vector_pos;
   Sensor_state = RecArray[3];
}
```

Figure 4.7- Function to Convert positional data received in the Server from *string* to *Vector3.*



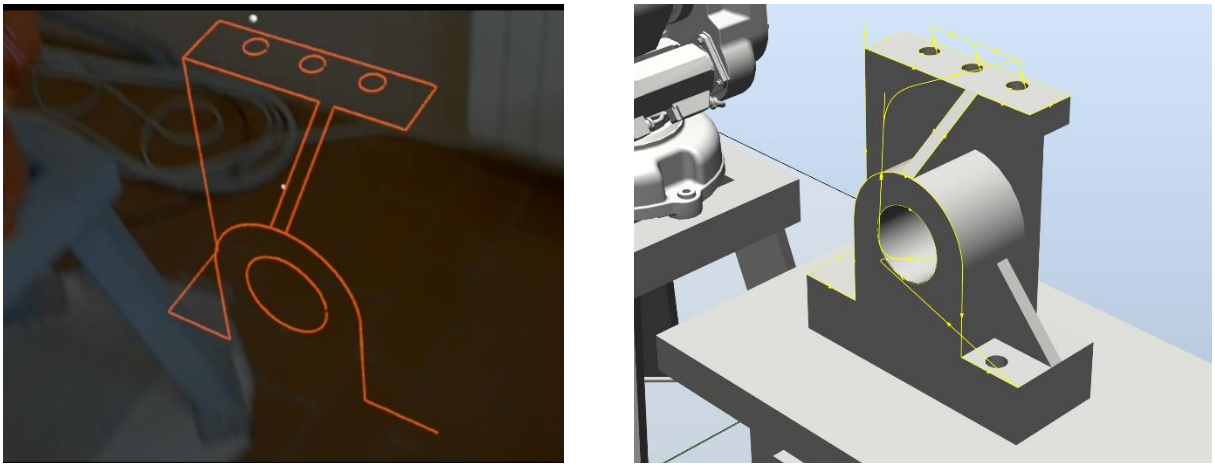
```
private void third_step()
{
    Last_pos = Points;
    Check_State();
    if(Last_pos != Prev_pos)
    {
        //Vector between new and previous robot position
        Vector3 incremento_vetor = Last_pos - Prev_pos;
        //New virtual position, updated to sphere tool pointer
        ToolPointer.transform.localPosition += incremento_vetor;
        //Storing data for LineRenderer
        if(collect_data == true)
            ListasPos[list_size].Add(ToolPointer.transform.Position);

        //Update the value of the previous position to new one received and treated
        Prev_pos = Last_pos;
    }
    else
    {
        return;
    }
}
```

Figure 4.8- Code snippet from the third stage on the Unity C# main code Responsible for transforming robot positions into virtual positions, updating the position at the pointer (sphere at the tool's tip), and saving the new vertex.

4.4.2. Adaptive Virtual Line

The code is also prepared to use the positional data stored in the *List<Vector3>*, and set it as vertices for the *LineRenderer* object, following the order within the *List<Vector3>*.

This process is done at every frame update of the application, where it checks the List for recent additions, and adds the new positions to the *LineRenderer* on the *List<Vector3>*. Following this process in a fast-repeating cadence, the visualized line quickly updates with high quality, as shown on Figure 4.9 (a), that demonstrates how the path displayed on the *HoloLens* is identical to the path on the robot simulation environment, visualized on Figure 4.9 (b).



(a)

(b)

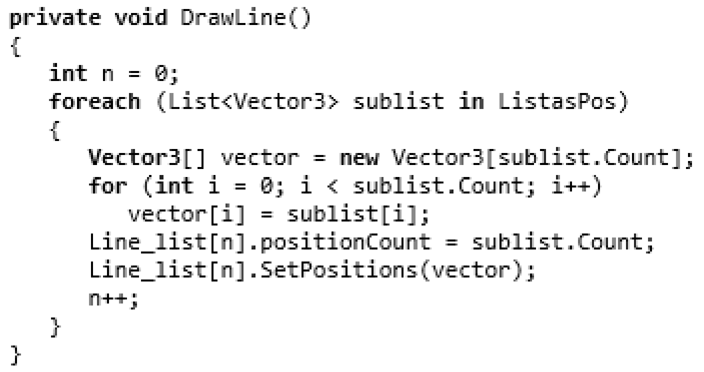
Figure 4.9- Comparison between the Robot trajectories visualized on the *HoloLens* and in the *RobotStudio* simulation environment: (a) Holographic path on *HoloLens*; (b) Virtual path on *RobotStudio*.

4.4.3. Sensor's State Analysis

Another functionality available is responsible for analysing a sensor's state that detects the contact with the working piece. This is useful for users to only visualize the process being made in the piece, without showing robot movements with no interest for the monitoring of the process.

To achieve this, the data sent by the robot contains at the end the sensor binary value (1 if it is in contact with the piece, or 0 if not). Having access to this information, a virtual line is only displayed when in contact with the piece and despises all the positions when the sensor has value 0.

Unfortunately, the *LineRenderer* does not allow interruptions within its length, requiring a new *LineRenderer* to be instantiated every time the sensor changes from value 1 to 0. In this way, the holographic path is separated by multiple lines in contact with the piece. Figure 4.10 represents the coding method to render the trajectories over the working piece.



```
private void DrawLine()
{
   int n = 0;
   foreach (List<Vector3> sublist in ListasPos)
   {
      Vector3[] vector = new Vector3[sublist.Count];
      for (int i = 0; i < sublist.Count; i++)
         vector[i] = sublist[i];
      Line_list[n].positionCount = sublist.Count;
      Line_list[n].SetPositions(vector);
      n++;
   }
}
```

Figure 4.10- Function to Draw and update holographic lines considering sensor states. All Data has been divided in sub lists to create multiple *LineRenderers,* also stored in a *List* of lines

5. GRAPHICAL PATH MANIPULATION

Being able to visualize and control a robotic process with MR technologies is useful, however, using the fullest of MR capabilities is only possible by enabling the possibilities to manipulate the virtual environment. To achieve this, *Graphical Path Manipulation*, was the second application created within this project, providing a way for users to create a robot path using the Mixed Reality environment on *HoloLens*, and applying the concept of robot programming using Mixed Reality (RPMR) presented on Chapter 2.

This app is introduced as something completely new to apply on industry, which can bring benefits to it, by providing simpler interfaces to program a robot, without requiring advanced knowledge and programming skills.

This chapter will focus on describing the virtual environment of this app, where all the significant methods for its construction and best performance will be thoroughly explained.

5.1. App description

Graphical Path Manipulation is an app created to apply the *HoloLens* capabilities of interaction with virtual elements, in a Mixed Reality environment, into the programming of a robot manipulator. With this app, the user will be able to create a robot path adding points (line’s vertex), as virtual spheres, choose the type of movement between adjacent points, visualized as an easily adjustable line, and even specifying the motion type and its parameters.

To achieve this, several tasks needed to be performed by the app, to ensure a smooth running and the best performance to the user experience.

1) **Editing Environment** - An essential characteristic of this app is the Editing environment, which is easily comprehended by the user, and is designed to

display all the required options to create, delete and execute a complete path. On subchapter 5.2 the editing environment created will be explained along with the first steps required to initialize it.

2) **Linear and circular movements** – The main task for this app consists in giving users the opportunity to define the robot movements. To do this, the user will be able to select on the app's main GUI the type of movement to be use, and see it on the three-dimensional space, as a virtual line that describes the motion to be realized by the robot. To provide the user with this holographic line of the path, adjustable at any time, some methods needed to be applied as described on subchapter 5.3.
3) **Robot parameter definition and data processing** - After creating a new path on the editing environment, the user will be able to define the velocity and precision of the robot on specific points, and the resulting data must be processed, to be "understood" by the robot controller. To achieve this, all data is objectified and divided, leading to its proper organization. The concept developed to overcome this issue will be explained on subchapter 5.4.
4) **Main App and GUI** - To create paths, control the robot and send the path to the robot controller, the main app provides a main GUI resembling the GUI used in the "Path Visualization" app described on Chapter 4. This GUI is responsible for several tasks, all explained on subchapter 5.5.

5.2. Editing Environment

The Editing Environment was designed to be simple framework, allowing the user to place robot positions on the virtual environment, defining the motion between them. Resembling the first application, applying some initial calibration between the robot and the Mixed Reality environment is required. To ensure this, the app also starts with the object tracking detection of the tool as a process of merging two worlds, using the same *Vuforia* technology explained on subchapter 4.3. In this case, when the tracking is made, the first robot position on the virtual environment is obtained and placed under the robot tooltip, as a virtual sphere, and it represents the starting point of the path to be created by the user. Knowing both starting points in the two environments is a first step to guarantee the correct programming of the robot

After the tracking is successfully completed, the editing environment starts and the user can start creating the path, as visualized on Figure 5.1 (a) an (b). It is composed by a GUI with all the options required to create the path, and send it to the robot.

Whenever the user sends a new path to the robot, the vectors between the virtual sphere's created (line vertex) are calculated, and the positional information obtained is sent. Afterwards, the RAPID program is prepared to convert the received data, ensuring that the robot will move exactly as the user specifies.

To allow the user to manipulate and click on virtual elements, the app subscribes to the IInputClickHandler and IManipulationHandler events from the 2017.2.1 HoloToolKit *Unity3D* package.

(a)

(b)

Figure 5.1 – Editing environment on *Graphical Path Manipulation* app for *HoloLens*. (a) Before path creation; (b) After path creation.

5.3. Linear and Circular movements

One important task for this application is to provide a condition for the user to pre-visualize how the robot will move based on the defined instructions. To concretize this, a holographic line must be displayed (linear or circular) every time a new movement is selected, and updated when any adjustment is made by the user. There are two types of movements available for users to create, each requiring different approaches for displaying its virtual representation. All movements are prepared to start from the latest virtual sphere created, to ensure a continuous path.

5.3.1. Linear Movement

The first type of movement, and the simplest to visualize with a virtual line, is a linear movement. Every time the user selects this type of movement, a virtual sphere is added to the scene, representing the last point of the linear movement. Displaying this movement is easy, as the movement will be performed between the last and the new sphere created.

The line is created resorting to the *LineRenderer* class from *Unity3D*, setting these two spheres as the first and last vertex of the line and constantly verifying its positions at frame rate (typically 60 frames per second). With this, the user can manipulate any of the spheres in the three-dimensional space, and the line will automatically adapt correctly. Figure 5.2 shows this type of movement visualized through the Mixed Reality device, and how it changes when a sphere is manipulated on the scene.

(a) (b)

Figure 5.2 – Representation of linear movements as Holographic lines on the *Graphical Path Manipulation* app: (a) Holographic representation of a linear movement after its creation, without manipulation; (b) Holographic line after being manipulated by the user

5.3.2. Circular movement

Creating a circular movement that can be transferred to the robot is easy, however, visualizing it on the *HoloLens* device is more complicated, and requires a viable method to allow the creation of an adaptive circular line of the path.

Firstly, when the user selects this type of movement, instead of creating one virtual sphere, as in the linear movement case described on subchapter 5.3.1, two virtual spheres are created and displayed in different positions in space. These two spheres represent the middle and the last position of the robot in this movement, being essential to define a *Robotstudio RAPID* circular movement. The three virtual sphere positions (initial, middle and last vertex), where the first point of the circle is the last virtual sphere created in the previous programmed line, are enough to define the circular movement on the robot, but not for a pre-visualization of the movement on the MR environment.

Unfortunately, *Unity* does not include a class to create a circular line visible in the *HoloLens*, therefore a method was created to adapt the *LineRenderer* class to display a line in a circular form, doing a perfect arc between the three virtual spheres of the respective movement.

The method used applies the geometric approach, illustrated on Figure 5.3, to find a centre circle based on three points, and it is programmed to find that centre on the three-dimensional space. As shown on Figure 5.3, the centre is obtained based on the bisectors intersection, and it is being continuously updated at frame rate, leading to its recalculation whenever the user changes any of the three spheres' positions. Acquiring the centre and the radius of the circle, the arc is easily created by finding multiple positions of the arc between the two spheres that limit it, and adding them as vertices for the *LineRenderer*. Ensuring a high number of vertices to the *LineRenderer* guarantees the visualization of a perfect circular line.

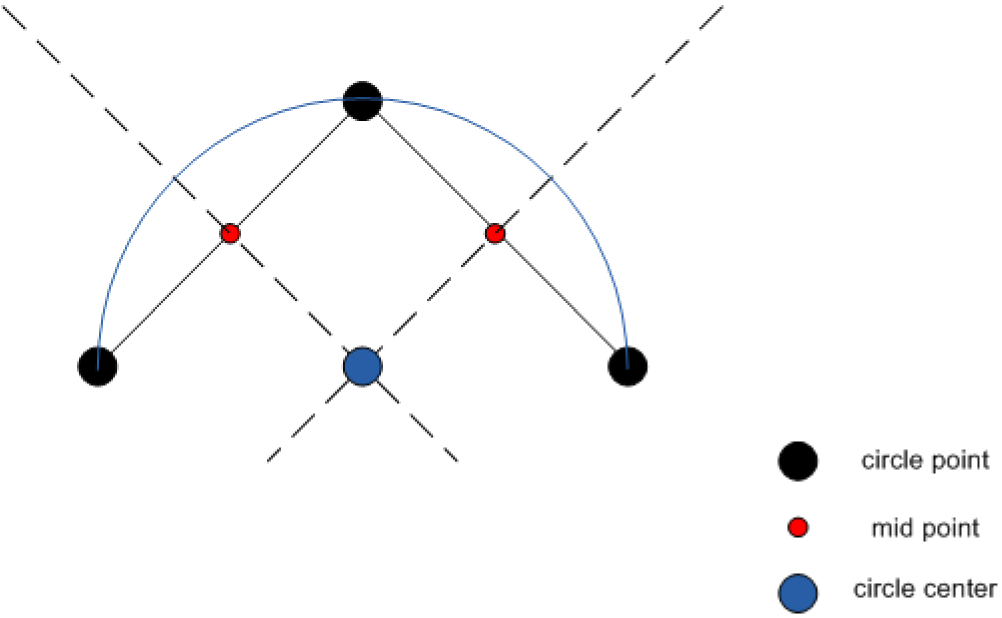


Figure 5.3 – Geometric approach to find the centre of a circle based on three circle points

5.4. Robot parameter definition and Data processing

Another functionality available in this app, is the possibility to define the robot's velocity and precision for each movement. In the robot simulation environment (*RobotStudio*), movements are defined by *robtargets,* which are robot targets that establish the motion's length, position in space, velocity and precision. This *robtargets* are placed in the first and last position of a linear movement and in the first, middle and last position of circular movement, equal to the disposition of virtual spheres in the Holographic lines. In fact, the virtual spheres represent each *robtarget* of the complete path, each one with a specific position and velocity.

To organize this data and define it properly, it was initially divided for each object (virtual sphere) through an identity script created, called *Identificator*. This script is inherent to each one of the sphere objects, working as an identity and data storage. This code script is described on subchapter 5.4.1.

Also, it was required a way for the user to change, at any time, the velocity and precision of the robot. For this purpose, every virtual sphere contains a GUI, as a child, that allows the possibility to change velocity and precision parameters. This GUI is described on subchapter 5.4.2.

5.4.1. Object Identity

As already described, every virtual sphere on the scene contains an identity. When a virtual sphere is instantiated ("created"), the *Identificator* script is attached as a component for this object. This *Identificator* was designed to accomplish several tasks within the app, listed below.

1) After the *Identificator* script is attached to an object, a method is programmed to run and automatically give an identity to the object. After the "name" is defined, it is protected making it impossible to be altered while the application runs.

2) The “name” created is basically a number followed by a space character and a letter. This number represents the sequence of birth (creation) number of the sphere on the scene, i.e., when a sphere is created the application keeps track and updates the number of created spheres and uses that number to identify the sphere. This enables the possibility to organize every sphere in the path.
3) The letters associated to a sphere identity can be of three types: “L”, if the sphere is a vertex of a linear movement; “C” if the sphere is a vertex of a circular movement and is not the last vertex; and “CL” if it is the last vertex of a circular motion. This guarantees that every sphere’s role in the path is identified
4) The name will be sent to the robot controller, and will be crucial to sort and define every *robtarget* correctly.
5) Also, an Identificator for an object *n* automatically calculates the positional vector from the object (*n-1*) to *n*, which is the positional information designated to send to the robot.
6) Since a sphere represents a *robtarget* structure, it is required that the *Identificator* stores information as well about the velocity and precision associated with the motion to which the sphere belongs to. Consequently, that information can be easily sent to the robot controller to fully define the robot’s path.

The *Identificator* script subscribes to the *IInputClickHandler* Event, i.e., responsible for opening or closing the object GUI presented on the next topic.

5.4.2. Object GUI

All the objects with an *Identificator* script attached, will have an inherent GUI. This GUI is activated when a click event is realized on the respective object, and is locally positioned, i.e., the object is positioned relative to its parent’s position.

This GUI is represented on Figure 5.4 and Figure 5.5, and gives the user the possibility to change the velocity and precision of the robot, in any virtual object sphere containing the *Identificator* script.

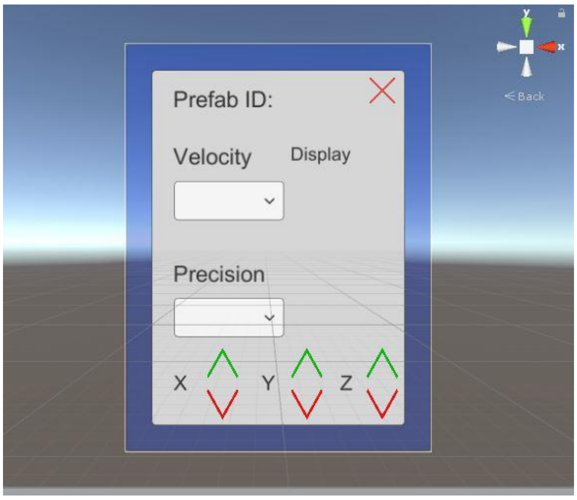


Figure 5.4 – Object GUI on the *Unity* editor

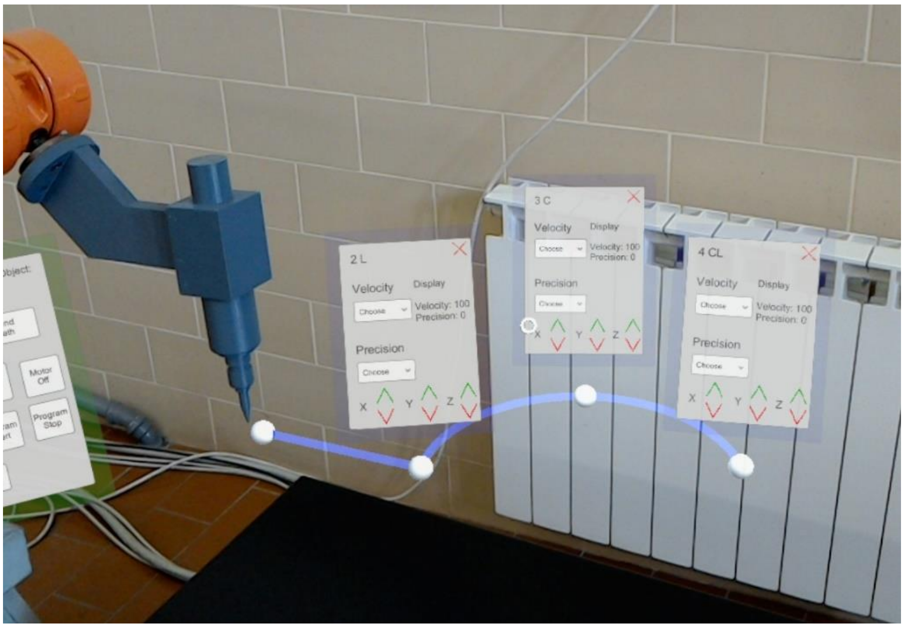
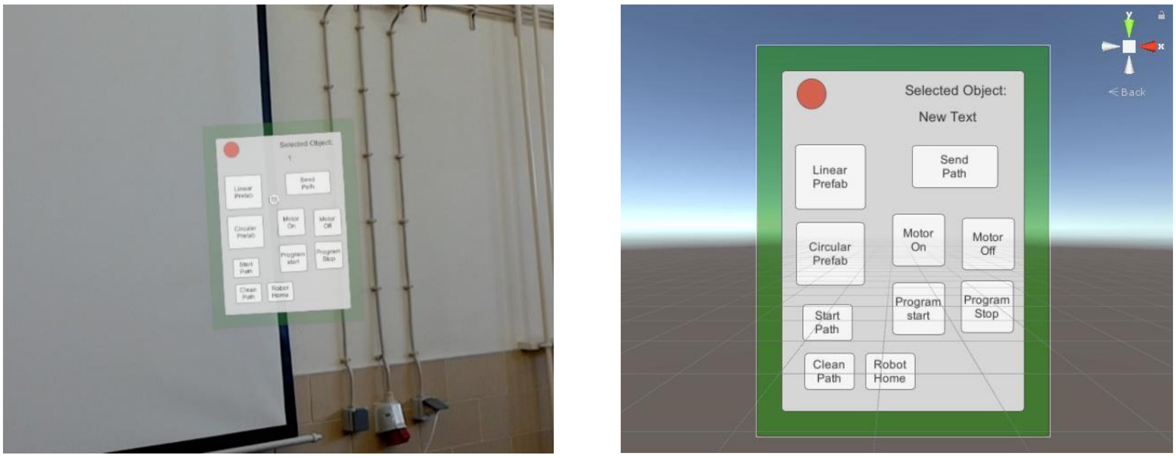


Figure 5.5 – Object GUI on the Mixed Reality Environment

5.5. Main App and GUI

The main App is responsible for updating, at each frame, the *holographic* lines that exist in the Mixed Reality scene, for controlling every process guaranteeing a correct running of the app, and for sending the complete information to the robot controller when the advanced user commands it from the main GUI, shown on Figure 5.6 (a) and (b). Consequently, with the main GUI the user can perform a variety of tasks, that will be listed and explained.

1) **Send commands to the robot to start/stop motors** - This allows users to engage or disengage the robot motors. The robot program can only be activated when motors are engaged.
2) **Start/stop robot program execution** - Commands the robot to enter “running mode” or “standby mode”. To use the *HoloLens* device programming features, the robot must be in “running mode”;
3) **Send the robot to home position** - The robot is commanded to go to the home (or reference position). As previously explained, this is the position where the robot should be when the calibration or object tracking procedure is executed;
4) **Select and/or create a linear or circular movement** - Allows users to add a new movement to an existing path or create a new one;
5) **Send the path to the robot** - The application just takes the information stored on each sphere’s *Identificator* script of the created path, packs the information of position, velocity, precision and type of motion and sends it, one by one (with confirmation), to the robot controller. In detail, the system packs the information in a string, with values separated by spaces, sends the information through the available asynchronous non-blocking socket connection and waits for the robot to confirm that the message was received. Only when confirmation is received the application proceeds to the next sphere.



(a) (b)

Figure 5.6 – Main GUI for the *Graphical Path Manipulation app*: (a) Main GUI in the Holographic environment; (b) Main GUI in the *Unity* editor.

6. FINAL DISCUSSION

This project, which produced two apps and a scientific paper submitted and accepted in *Industrial Robot* journal, was an innovative approach that obtained satisfying results. Both applications realized the tasks designed, with good performance, and even though they are still under development, the operator might find it already quite useful. However, perfection was not achieved and a considerable amount of development and code optimization is still required, to be applied directly on the industry. This Chapter will discuss how future work should be applied on this project and concept, and will highlight some issues that could not be solve, due to the limited time available to conclude this master thesis dissertation. At last, subchapter 6.3 will present a comparison with a recent project by the *WizLink* company.

6.1. Path Visualization

The first app enabled the possibility to visualize a robot trajectory in a MR environment, to improve task monitoring. The trajectory obtained, matches perfectly with the real path represented by the robot, and the overall performance of all tasks within the app is very good. Nevertheless, there are some issues regarding coordinate calibration that could not be fixed yet, and some errors needed to be fixed on the object tracking method used.

The calibration is not perfect yet, and the reason lies on the method that was used for merging the two worlds, described on subchapters 4.1 and 4.3. The method used to guarantee that all the robot positions where correctly placed in the Mixed Reality environment had the inconvenience of being highly sensitive to the real tool position. As the tool tracking represents the local matching of the real and virtual robot coordinates, ensuring that the tool is placed correctly is crucial, and a small deviance on it is enough to the rendering of a decalibrated trajectory. Surpassing this shortcoming would be possible by tracking the tool and the working piece for a more accurate matching. However, this was not possible because we did not have access to that working piece.

Another incorrection was detected on the object tracking method from *Vuforia*, that was still not entirely prepared for *HoloLens*. In fact, it is very sensitive to lightning, which means that if the lightning was not strong enough in the environment, the virtual tool rendered would not be positioned on the exact position of the real one. This required a method of trial and error to correctly position the virtual sphere tool's tip under the real tool's tip.

This app presents workers with a new interface to monitor industrial tasks, that provides useful advantages on the displaying of visual information about the robot task, assuring a better response for users on analysis and detecting errors at real time.

As future work, these inconveniences must be addressed, and some additional features need to be applied. The user must be able to, after visualizing a trajectory, alter the path by manually updating the virtual line, if needed, and to enable the same environment to be accessed by multiple robots and operators.

6.2. Graphical Path Manipulation

The second app developed in this project gives users the possibility to program a robot to a desired task, without writing code, and using an intuitive MR environment. The app works quite well and no major issues can be pointed. The problems relative to the object tracking method are also applied in this app, since both applications use the same method. On this app, only sporadically, we verified that circular lines visualized on the *HoloLens* did not represent perfectly the circular movement of the one sent to the robot. The calibration to this aspect could not be performed due to the limited time, but it is not a major concern as it is easily corrected.

The concept of this app should be continued for further work, since its benefits are highly visible. The app provides a generic new interface that uses HoloLens capabilities to allow operators to fully program an industrial robot for a specified task, resorting to an interactive environment where all options are quite intuitive to the user. The robot can be programmed by users to execute any task only with hand gesture inputs, without requiring a single line of code. Thus, it is easily adaptable to any kind of industrial task, and can be posteriorly used by industrial workers for programming robots on tasks such as machining, additive manufacturing, deburring operations, amongst others, applying the required adjustments on the app for each specific task.

The final goal proposed for future work consists in mixing all these achievements in one app, that will provide users with visual feedback of the robot motion, and at the same time giving users programming features. This would enable the possibility of real time optimization of trajectories on complex industrial tasks.

6.3. Comparison with WizLink's project

WyzLink is a company focused on developing AR/VR solutions, that recently developed a project for robot programming with similar goals to our *Graphical Path Manipulation* app. This project is shown in (Zhou, 2017) and (WyzLink., 2018), which are references to two videos published on YouTube. These videos were visualized on July 1, 2018, after the development of the two apps described in this dissertation, and did not influence this project, despite the similarities with *Graphical Path Manipulation* app. However, it is important to compare both applications, with an analysis of advantages and disadvantages.

Comparing the YouTube videos referred with our project´s video in (J.Neves et al., 2018), it is possible to see that the approaches are quite different. While *WyzLink's* project allows users to place only one 3D point in space at a time, and teaches a robot manipulator to move linearly to that point, *Graphical Path Manipulation* has the advantage of allowing users to define linear and circular motions, change velocities and precision settings on specific points, and pre-visualize the movement in holographic lines, which makes this dissertation's approach an improved version of the *Wyzlink´s* one. The only advantage from *Wyzlink's HoloLens* app is the possibility to visualize a virtual model of the real robot, that moves exactly like the real one, which can be useful for pre-visualization of the real robot joints movement.

Nevertheless, it is possible to see that Mixed Reality is being used and new developments are being made by researchers in this field, which corroborates the importance of this dissertation in the future of robotics

7. CONCLUSION

This dissertation presented a concept to use Mixed Reality on the robotics field, and described two *HoloLens* apps, that provided a MR environment for monitoring and programming robot manipulators. The first app was designed to allow a complete monitoring of a robot trajectory, displayed on the *HoloLens* Mixed Reality environment. The second app provides users with a graphical way of programming a real robot manipulator, without writing code, thus being fully immersed in the Mixed Reality environment and its capabilities. A posterior analysis was made regarding the shortcomings on both apps, to overcome them, and where future work should be applied to successfully improve this project.

As final conclusions, with the assessment of all the achievements obtained, it was proven that Augmented and Mixed Reality technologies are a useful upgrade on robotic systems. It was verified that, applying a graphical environment fully immersed on our real perception, improves the user’s response and capability to monitor and control robotic manipulator’s tasks at real time, and even allows new programming methods far more intuitive and with higher reliability in comparison to the traditional methods used nowadays.

The future of these technologies is still in its early stages, and the prospects are that a large technological advance will be seen quite soon. This dissertation not only described the project developed, but also presents a starting point to simplify future applications in this area.

BIBLIOGRAPHY

Chong, J. W. S., Ong, S. K., Nee, A. Y. C., & Youcef-Youmi, K. (2009). “Robot programming using Augmented Reality: An interactive method for planning collision-free paths”. In: Robotics and Computer-Integrated Manufacturing, 25(3), 689–701.

Christensen, L. R., Marcik, W., Rafert, G., & Wong, C. (2016). In: The Global Economic Impacts Associated with Virtual and Augmented Reality, 1–21.

Coppens, A. (2017), “Merging real and virtual worlds: An analysis of the state of the art and practical evaluation of Microsoft HoloLens”. Master Thesis in Computer Science. Department of Computer Science – University of Mons

Guhl, J., Tung, S., & Kruger, J. (2017). “Concept and Architecture for Programming Industrial Robots using Augmented Reality with Mobile Devices like Microsoft HoloLens”. In: Conference: 2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation. Sept 12-15. Limassol, Cyprus

Honig,W., Milanes, C., Scaria, L., Phan, T., Bolas, M., & Ayanian, N. (2015), “Mixed Reality for Robotics”, In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Congress Center Hamburg. Sept 28 - Oct 2, 2015. Hamburg, Germany.

Jurascheck, M., Büth, L., Posselt, G., & Hermman, C. (2018), “Mixed Reality in Learning Factories”, In: Procedia Manufacturing, ISSN: 2351-9789, Vol: 23, Page: 153-158.

Neves, J., Serrario, D., & Pires, J, N. (2018). “Application of Mixed Reality in Robot Manipulator Programming”. Acessed in June 12, 2018. on: https://www.youtube.com/watch?v=EWVPy2d39II

Kersten-Oertel, M., Jannin, P., & Louis Collins, D. (2013). “The state of the art of visualization in Mixed Reality image guided surgery”, In: Computerized medical imaging and graphics: : the official journal of the Computerized Medical Imaging Society, ISSN: 1879-0771, Vol: 37, Issue: 2, Page: 98-112

Kosowatz, J. (2017),”Mixed Reality Changes How Engineers Work”, Accessed in: June, 02 2018 on: https://www.asme.org/engineering-topics/articles/technology-and-society/mixed-reality-changes-engineers-work

Michalos, G., Karagiannis, P., Makris, S., Tokçalar, Ö., & Chryssolouris, G. (2015) "Augmented Reality (AR) Applications for Supporting Human-Robot Interactive Cooperation", In: 48th CIRP Conference on Manufacturing Systems. June 24-26. Ischia, Italy.

Milgram, P., Takemura, H., Utsumi, A., & Kishino F. (1994). “Augmented Reality: a class of displays on the reality-virtuality continuum”. In: Proceedings of SPIE -

The International Society for Optical Engineering. Telemanipulator and Telepresence Technologies. 2351. DOI: 10.1117/12.197321.

Müller, D., Whilhelm Bruns, F., Heinz-Hermann, E., Robben, B., & Yong-Ho, Yoo. (2007), “Mixed Reality Learning Spaces for Collaborative Experimentation: A Challenge for Engineering Education and Training”, In: International Journal of Engineering (iJOE), Vol 3, No.4, pp. 1-5.

Roni, S (2010). “Async Socket Server Sample C#”. Accessed in: March,10 2018 on: http://blog.ronischuetz.com/2010/05/async-socket-server-sample-in-c.html

Schaal, S. (1999)." Is imitation learning the route to humanoid robots?" in*: Trends* in *Cognitive Sciences*, *3*(6), 233–242. https://doi.org/10.1016/S1364-6613(99)01327-3

U.S Department of Labor. (1990) “OSHA technical manual, industrial robots and robot system safety”, section IV: chapter 4. Available from: http://www.osha.gov/dts/osta/otm/otm_iv/otm_iv_4.html, accessed on June 26, 2018.

WyzLink. (2018). “WyzLink MR Robotic control_short version”. Accessed in: July 10, 2018 on: https://www.youtube.com/watch?v=QtFvLXbJrwA

Yeng-Hung Chen, I., MacDonald, B., & Wünsche, B. (2009), “Mixed Reality Simulation for Mobile Robots”, In: 2009 IEEE International Conference on Robotics and Automation. May 12-17, 2009. Kobe, Japan. ISSN: 1050-4729

Zhou, B (2017). “hololens robotics - wyzlink”. Accessed in: July 10, 2018 on: https://www.youtube.com/watch?v=dBpqZ6x9rok