Tampere University

Antonios Sylari

HAND-GESTURE BASED PROGRAMMING OF INDUSTRIAL ROBOT MANIPULATORS

Master of Science Thesis
Faculty of Engineering and
Natural Sciences
January 2020

ABSTRACT

Antonios Sylari: Hand-Gesture Based Programming of Industrial Robot Manipulators
Master of Science Thesis
Tampere University
Automation Engineering, Factory Automation and Industrial Informatics
January 2020

Nowadays, industrial robot manipulators and manufacturing processes are associated as never before. Robot manipulators execute repetitive tasks with increased accuracy and speed, features necessary for industries with needs for manufacturing of products in large quantities by reducing the production time. Although robot manipulators have a significant role for the enhancement of productivity within industries, the programming process of the robot manipulators is an important drawback. Traditional programming methodologies requires robot programming experts and are time consuming.

This thesis work aims to develop an application for programming industrial robot manipulators excluding the need of traditional programing methodologies exploiting the intuitiveness of humans' hands' gestures. The development of input devices for intuitive Human-Machine Interactions provides the possibility to capture such gestures. Hence, the need of the need of robot manipulator programming experts can be replaced by task experts. In addition, the integration of intuitive means of interaction can reduce be also reduced. The components to capture the hands' operators' gestures are a data glove and a precise hand-tracking device. The robot manipulator imitates the motion that human operator performs with the hand, in terms of position. Inverse kinematics are applied to enhance the programming of robot manipulators independently of their structure and manufacturer and researching the possibility for optimizing the programmed robot paths. Finally, a Human-Machine Interface contributes in the programming process by offering important information for the programming process and the status of the integrated components.

Keywords: industrial robot manipulators, human-robot interaction, online robot programming, hand-gestures, industrial automation, manufacturing processes, sensors, human-machine interactions, human-machine interface

PREFACE

First and foremost, I would like to thank and give my gratitude to my parents, Ferdinant Sylari and Mimoza Sylari, for their sacrifices, efforts and support throughout the difficulties to achieve my goal and complete the Master of Science degree in Automation Engineering in Tampere, Finland. I would like to dedicate my Thesis Work in the memory of my father Ferdinant Sylari, who was also brother and best friend, and left from my life too soon (1963 - 2019).

Secondly, I would like to thank Prof. Jose Luis Martinez Lastra and Dr. Borja Ramis Ferrer for the opportunity to work as a researcher at facilities of Future Automation Systems and Technologies Laboratory at Tampere University. Their contribution, support and guidance helped me through the completion of my Thesis. I would like also to thank the members of FAST-Lab, Anne Korhonen, Luis Gonzalez Moctezuma, Wael Mohammed for their support.

Thirdly, I would like to thank my family in Turku, Zoe-Pamela Topalli, Aristomenis Berdesis and my nephew Ioannis Berdesis for their pleasant breaks we had through the two and a half year of my Master's degree. In addition, I want to thank my friends and colleagues at Tampere, Ronal Bejarano and Saigopal Vasudevan.

Finally, my gratitude to my second family in my hometown Thessaloniki, Greece, Stefan-Eros Celai, Anastasia Kanavou, Stelios Akritidis, George Thomas, Panagiotis Papadopoulos and Rafaela-Maria Mikiktsi for their support within the difficult moment I faced through the last six months.

Thessaloniki, 12 January 2020

Antonios Sylari

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

HCI	Human-Computer Interaction	
HMI	Human-Machine Interface	
HRI	Human-Robot Interaction	
IKC	Inverse Kinematics Calculator	
IR	Infrared	
JSON	JavaScript Object Notation	
PDA	Personal Digital Assistant	
SDK	Software Development Kit	
TCP	Tool Centre Point	
TCP/IP	Transmission Control Protocol / Internet Protocol	
α	Link length,	millimetres
Δ	Distance	millimetres
θ	Joint angle,	radians
π	Perimeter of a circle	
a	Link twist,	radians
d	Link offset,	millimetres
k	Angle	degree
mm	millimetres	
ms	milliseconds	
R_{3x3}	Rotation matrix	
P_{3x1}	Translation matrix	
${}^{i-n}_{i}T$	Homogenous transformation matrix	
q_i	Prismatic joint displacement	

1. INTRODUCTION

This chapter presents the motivation, justification for the problem statement and the research questions of this Master thesis. In addition, this chapter introduces the scope, the limitations and the objectives to consider during the implementation of this Master thesis work.

1.1 Motivation

Nowadays, robots are becoming more and more an integral part in the lives of humans. From the simplest robot vacuum cleaner to robotics in the field of healthcare and complex industrial robotic systems even mobile robots to planet Mars, robots have a direct or indirect impact to the humans' lives. Humans takes advantage of the benefits that robots offer, especially in manufacturing processes. According to "*International Federation of Robotics*" (IFR), there is a progressively increase on investments for robots by industries, [1], [2].

Industrial robot manipulators are commonly employed for executing various manufacturing processes. The programming of an industrial robot manipulator requires people with expertise on the field of robot programming. The wide-spread techniques to program an industrial robot are offline and online.

Programming of an industrial robot manipulator with the offline technique requires a software in which the programmer can visualize the cell of the robot, including potential obstacles, along with the robot manipulator model. Then, follows the definition of targets and paths for the completion of a task. The software can simulate the robot movements and detect collisions along the paths. The online technique demands from the programmer to be in short distance from the physical robot. A teach pendant is the mean to control the robot manipulator and lead it on the desired targets. Once the definition of the paths is completed, the robot programmer creates the final robot code through the teach pendant.

Although, both techniques have their own advantages and disadvantages. The required time to stop the operation of the robot manipulator in order to add the new code and test its less on the offline compared to online. On the other hand, the time to create the visualization of the cell and program the robot is higher than the online.

It is clear that in order to enable the HRI, there is a need for a software or some intermediate specialized input devices. However, the manipulation of such means requires professionals with knowledge in the field of robotics.

As a result, the process of robot programming consumes time either for the creation of the robot's work cell or the suspension of the manufacturing process, which in turn leads increases the cost. To reduce the impact of these factors, some approaches at-tempt to develop alternative ways for enabling HRI. In this scope, voice recognition [3], human's movements [4] and hand-gestures are used to control robots [5]. In this way, the programming process can be achieved by non-robot professional intuitively. However, an important factor to consider is how accurately can such approaches control and precisely navigate the robot manipulator to desired target.

This thesis focuses on developing an application for programming robot manipulators, deployed on the factory floor, with hand-gestures to control robot's actions and movements independently of the robot's structure or manufacturer. The robot manipulator must be able to imitate the movement of the hand in order to achieve robot Programming by Imitation (PbI). In addition, the programming process will be performed by non-robot professionals, but with the required training on how to operate this application.

1.2 Justification

Over the past years many research works have been conducted in developing approaches for intuitively program industrial robot manipulators, [6]–[8].

The emergence of new input devices, such as data gloves [9] and cameras [10], mostly designed and developed for gaming purposes, created an opportunity to utilize them if a wider range of applications. The reason is that such input devices, eliminate or sometimes reduce the need for using buttons, and use the natural and intuitive human motion to interact with a computer, machine and robot manipulator.

As a result, the need of offline programming software and non-intuitive device, such as the teach pendant, are eliminated. Consequently, programming experts might be substituted by experts in tasks' execution.

1.3 Problem Statement and Research Questions

As the market and customers' needs change day by day, industries attempt to maintain flexible manufacturing processes. Therefore, re-programming of industrial assets is necessary. In this context, robot manipulators have a significant role, but the programming process is time-consuming and costly. Input devices that enable an intuitive interaction

with the robot can be beneficial in such approaches. there is a need to develop approaches for intuitively programming robot manipulators quickly and without the need of programming experts.

So the following research questions are raised:

- How to allow non-programming experts to program industrial robot manipulators?
- What are the appropriate devices to recognize gestures and precisely control the robot manipulator?
- How to program industrial robots independently the robot's structure and manufacturer?
- How to optimize the robot's task, after the hand-gesture based programming?

1.4 Objectives

In order to achieve and successfully implement the goal of the thesis work, there is a need to define the objectives.

The aim of this thesis work is the development of an application which will allow a non-programming expert to program intuitively an industrial robot manipulator. The human operator will be to control the actions and movements of the robot manipulator using hand-gestures of both right and left hand. The recognition of left and right-hand gestures requires two devices to capture the pose and motion of the left and right hand respectively. The provided data from those devices will be transmitted to the main application in which the next action will be decided.

The purpose of the left-hand gestures is to control the robot actions. Those actions are the enable and disable of interaction with the robot, allow the movement of it, enable and disable the teaching phase, grasp or release a work object and allow the robot to execute the previously taught path.

Human operator will move the right hand above or below the hand tracking device. The centre of the hand tracking device will correspond to a fixed position for the robot. This is pre-declared, so that the human operator can have a reference position for beginning the control of the robot.

Moreover, depending on the robot's structure different, the inverse kinematics have to be calculated to control the robot's position using joint angles and not Cartesian coordinates.

The solution of the inverse kinematic must then be sent to the robot's controller. In addition, the application's main controller must also receive this solution. The reason is that robot programming code will contain robot tasks in the Joint space.

1.5 Limitations

This chapter presents the intended limitations for developing this thesis. Those limitations have been set prior the implementation. However, some limitations can change or extend.

- The application to be developed is only applicable for single industrial robot manipulators.
- Industrial robot manipulators with two robotic arms are not possible to be programmed within this thesis.
- This implementation focuses on programming the ABB IRB120 6 DOF industrial robot [11] and an OMRON Adept eCobra 600 pro 4 DOF/Scara robot from [12]. The robot manufacturer and type can be extended as long as it supports Web Socket communication.
- The robot manipulator cannot be programmed using only one hand.
- The generated final robot code includes only move robot commands. Using I/Os or sockets for any purpose cannot be generated automatically but programmed manually by the user.

1.6 Outline

The thesis structure is the following: Chapter 2 reviews research work in the field of HRI concerning the control and programming of robots. In Chapter 3 the approach for implementing this thesis is presented. In addition, the kinematic analysis for a robot manipulator is given. Chapter 4 presents the implementation of the approach. Finally, Chapter 5 concludes the implemented approach and suggests future work in order to extend this thesis.

Part of the Thesis Work was presented and published on the IEEE International Conference on Industrial Informatics (INDIN) in 2019 at Espoo, Finland. The title of the paper is "*Hand Gesture-Based On-Line Programming of Industrial Robot Manipulators*" and the authors are Antonios Sylari, Dr. Borja Ramis Ferrer, Prof. José Luis Martinez Lastra. [13]

2. LITERATURE REVIEW

This chapter reviews and summarizes related research works related to HRI for robot control and robot programming. Moreover, this section presents research works for programming robots utilizing different input devices. As input devices, they can be considered devices that connect to a computer for Human-Computer Interaction such as keyboard, joystick, data gloves or vision-based devices. Furthermore, the chapter overviews on-line robot programming approaches and applications.

2.1 Human-Robot Interaction

HRI is a field to research various approaches regarding the communication of humans and robots. Goodrich and Schultz describe HRI as a research devoted to the development of interfaces which will enable the communication of human and robots [14].

The HRI can be classified as physical and remote interactions. The differentiation is not necessarily on the grounds of distance that separate human and robot but how humans and robots interact. [14]

In the paper from Michalos et al. [15], and as a part of the European Union Project, ROBO-PARTNER [16], describe the physical HRI that might appear in an industrial environment on the factory floor, as those examined within the project. The first category presents the co-existence of human and robot in the robot's work-envelop. Sensors attached to the robot, detect the presence of human for safety reasons. On the second category, a mobile robot feeds the human with necessary tools and parts for the completions of hu-man's task. The required time to execute the task is decreased as the human does not get distracted. The last category describes a direct physical HRI. The human is responsible for the task completion. However, the robot provides assistance as it holds parts and human through lead control moves it to the appropriate target. [15]

From another point of view, Bdiwi et al. [17] in 2017, classify the physical HRI in four levels. Compared to [15], they propose on the first level a common work-envelop without a common task. The second level allows the cooperation of human and robot but without a direct interaction; the robot operates as an assistant within a pre-defined path. The third and fourth levels from Bdiwi et al. [17] can be matched with the second level of Michalos et al. [15], in which robot feeds the human with tools or parts and the robot and human work together with direct interaction respectively.

Apart from physical interactions, human can also interact remotely with robots in cases which the human presence is not a feasible option; distance and safety form the main motives for such interactions.

To overcome the barrier of distance, Marescaux et. al [18] attempt a remote surgery between two continents, [18]. Another case is the hazardous environment in which operations have serious consequences on human's health, [19]. Robots have also been employed for rescuing purposes, [20]. Experts independently on their field is not possible always to travel, especially in great distances and thus, remote interactions provide the ease for accomplishing the required tasks. Indeed, for such interactions, robots must provide feedback to the human through sensors [18] such as cameras [21].

However, remote interactions occur also within the industrial environments. More specifically, the online programming process of industrial robot manipulators requires an input device, which in most of the cases is the teach pendant. The ABB teach pendant is shown in Figure 1. In addition, gesture-based systems allow intuitive control of robots [22]. Even the human can be in close proximity with the robot manipulator, the interaction is not necessarily direct. The human utilizes an input device for the control and visually observes the robot.

Figure 1. *ABB teach pendant [23]*

Overall, this section presents the HRI and its classification as physical and remote interactions. Physical interactions commonly appear on the factory floor, particularly with the growth of the quantity of robots in the factories and the need of human-robot cooperation. The research works show that with physical HRI it is possible to reduce the time for a task completion with the least possible human effort. Moreover, remote interaction requires human to utilize a device able to manipulate the robot. Such devices can be defined as input to a computer device, which in turn create the base for establishing the HRI. However, the interface to enable the remote HRI, must be human-friendly and allow an intuitive robot control.

2.2 Human-Robot Interaction for Robot Control

Over the past few years, different devices are developed, especially within the scope of electronic games. However, speech is another option for and intuitive HRI. The objective is to allow the user to experience an interaction as intuitive as possible. As a result, researchers attempt to benefit from the possibilities of such devices for remotely control robots.

A web-based human robot collaboration is introduced by Wang [24]. In this paper, the objectives are the safety and the network. Yet, it worth to mention the remote control and monitor of the robot manipulator that are part of the paper and are examined. The remote control of the robot manipulator is achieved through a web-based human-robot interface in which human can monitor the robot and jog individually each robot joint. [24] This approach can successfully manipulate a robot but the interface, as a mean for HRI, does not indicate an intuitive approach.

A different approach for HRI is presented by Nilas et al. [25]. The authors designed a PDA (Personal Digital Assistant) to send "*high-level task*" commands to a robot within the scope of path planning. These commands are related to manipulating the robot and controlling the end-effector. Similar approaches can be found in [26]–[28]. Such a device provides options for controlling a robot but are considered less intuitive compared to other devices for HRI.

Joystick is one of the primary devices for remote interactions. In 2008, Yang et al. [21], attempt to control remotely a field robot. The system integrates two different methods of control with visual feedback. In the first method, manual mode, a user manipulates the displacement of the robot. Meanwhile, on the second method, auto mode, the user controls the robot's end-effector and utilizes inverse kinematics for the robot's displacement. [21] Even with such a device, human can control a robot, it is not recommended for intuitive robot control especially with the advances of input devices.

Control panel, kinesthetics guidance and a data glove are compared by Fischer et al. [29] as means for HRI. The most appropriate approach for remote robot control is the control panel and the data glove, while the kinesthetics guidance necessitates human to be in direct interaction. Within this research work an experiment with 51 users was conducted for controlling a robot manipulator. The grade of familiarity with robots was approximately 2 on a scale 1-4. A video with the robot's task execution and errors was presented to the participants of the experiment in order to familiarize with the system's operation. 15 participants were asked to interact with the data glove and their statements regarding it varied. Some users commented it as a natural feeling regarding the robot

control and quite simple for manipulating the robot at big distances. However, other users reported that the robot's control was not easy enough and it felt sensitive. For the tests of the control panel 18 participants were recruited. On the positive side of comments for this experiment, users stated that it is a convenient mean for users that are do not have any knowledge with motor skills and felt sure about their actions without the fear of damaging. However, other users found difficult to learn and understand the functionality of each button, latency problems were observed, and this option does not allow and intuitive interaction. The last mean of interaction, the kinesthetics guidance, the control of the robot felt easy and by moving the hand the robot was following exactly the same path. But, the weight of the robot does not always allow every user to use the kinesthetics guidance. In addition, the workspace of the robot is not always suitable for such a control as each human has different height and thus more difficulties on reaching different targets. Finally, users indicated that is not appropriate for high precision robot control. [29] Aspects such as ease of use and intuitive feeling are crucial for operations in which user must manipulate a robot.

Another approach for controlling a humanoid robot has been implemented by Song et al. [30]. This approach attempts to allow human to intuitively control the robot. This is achieved by capturing the human's body motion through a Kinect device. Figure 2 illustrates the Kinect device. The conducted experiments regard the human motion detection, the robot's response in real-time as for human's motion and robot's obstacle avoidance. The authors state that the results indicate a "*correct and feasible*" robot control [30]. However, no results regarding the ease of this approach and the motion recognition rate are provided. Nevertheless, this paper presents an alternate option for intuitive robot control.

Figure 2. *Kinect device [31]*

Nowadays, the research on HRI focuses on approaches, in which humans use their body and hand for an intuitive HRI. However, another approach for intuitive and natural HRI is speech recognition. Mubin et al. [32], present an approach for interacting with robots through speech. The authors used ROILA "*a speech recognition friendly artificial language*" that could perform better compared to English. ROILA is a developed language, designed to be simple to understand by the robot and easy to learn, [32], [33]. To validate

the operation of their approach 15 participants with native language Australian English participate. This restriction was set in order minimize the error due to various dialects. The accuracy of speech recognition was about 70%. [32] This accuracy is considered low as stated by the authors as it can be proven to be critical in various applications of HRI. In addition, the accuracy rate could vary in case the participants had different native languages. In [34], where an approach for offline robot programming is presented, it is mentioned that if the pronunciation is not the correct, the chance for a false speech recognition is increased. This issue could be resolved by implementing an approach which integrated artificial intelligence algorithms but training such an approach can require a huge amount of data from people with various native speaking languages, which in turn, is a time-consuming process.

Collectively, this section reviews research works that utilize a variety of input devices as mean for establishing HRI. Each device has its own advantage but also drawback. In terms of intuitiveness kinesthetics guidance, data glove, vision-based systems along with speech recognition allow humans to control robots by instinct without much effort. Indeed, training for the operation of the system is required.

2.3 Gesture-Based HRI for Robot Control

The manner humans manage their body, head and hands to express themselves or point to an object, raises through their natural instincts. Nowadays it is possible to capture those natural expressions mostly by developing approaches which integrate cameras [35], depth sensors [36] and data gloves [37]. The use of input devices in the field of HRI for robot control was introduces in the previous section. Nevertheless, this section provides a more in-depth emphasis in related work conducted on gesture-based approaches for intuitive robot control.

Park et al. [38], develop a vision-based gesture interface for controlling two humanoid robots. The system includes two fixed cameras for capturing head and hand gestures from the two users. The 13 available gestures allow the users to move the robot in all directions (forward, backwards, right and left) and move the arms of the humanoid robots. In a similar approach Nickel and Stiefelhagen [39] utilize a stereo camera for capturing pointing gestures. The gesture recognition depends on visually capturing the head, hands and head orientation. The experiments were conducted on the humanoid robot ARMAR, [40]. The experiments show that without the feature of head orientation the gesture recognition accuracy was 74% and with it, the accuracy could improve up to 87%.

The introduction of devices which integrate depth sensors [4] and/or infrared supported cameras (IR) [41] replaced the traditional cameras for developing body and/or hand gesture recognition for HRI. The Kinect sensor was deployed by Yavşan and Uçar [42] for capturing upper-body human gestures and later a humanoid robot imitates human's motion. Qian et al. [22] control a dual arm robot with hand gestures. The user can move the robotic arms up, down, left and right depending on the captured gestures through a depth sensor. The gesture recognition precision through Hidden Markov Model (HMM) classification within this research work was 85%.

Depth cameras are also utilized to detect the position of the hand. In [43], a depth camera detects the user's hand position and uses it as a reference point for initiating the robot control. The user's hand displacements will be then translated to robot manipulation. Participants in the experiments indicate that this approach is easy to learn and to control the robot. In addition, the authors state that this approach is intuitive and accurate, but no numerical results were provided. [43] Although, an analysis regarding the depth accuracy of the Kinect sensor shows a standard deviation about 1.5cm, [44].

However, the available in the market devices are not the only option. Neto et al. [45], utilize two accelerometers, one for each hand, in order to capture gestures and postures of right and left hand. One arm enables and disables the control of the robot and the other handles the manipulation of a 6 DOF robot in X, Y and Z-axes and the rotation about X, Y and Z-axes. The manipulation for translation and rotation is achieved in one axis at the time. To boost the recognition ratio, the authors put in practice Artificial Neural Networks. The average accuracy of the system is 92%. [45]

Accelerometers were also deployed for the development of data gloves, [46]. In [47], five accelerometer were installed in each finger and one in the centre of the palm. The recognition rate varies relying on the number of samples and the rate varies from 30% (for one sample) to 98% (for 25 samples). [47] Additionally, inertia and magnetic measurement units (IMMUs) can replace the accelerometers [9], [48], [49]. Flex sensors is another choice for data gloves. They offer accurate tracking for each finger detection with-out the need of additional back systems, [50].

This section reviewed approaches for gesture based HRI for robot control. For most of the gesture recognition approaches, vision-based devices were utilized for capturing gestures but with lower gesture recognition rates compared to wearable devices.

2.4 Online Robot Programming

Programming of industrial robot manipulators demands robot programming experts, halt of the manufacturing lines for long period of time and as a result the financial expenses are increased. Offline and online robot programming methods are those used throughout the industry. On the one hand, offline programming is based on a computer software, in which the robot's work envelop along with any obstacles must be designed and positioned accurately as the real factory floor. On the other hand, online robot programming is achieved with the teach pendant, through which the industrial robot is manipulated to the positions of the desired path. The use of these methods require time and are not user-friendly for the people without knowledge on the field of robotics. Various approached, systems and frameworks have been developed by researchers, having as main goal the simplicity of the industrial robot programming process.

Kohrt et al. [51], introduce a supportive system for programming a 5-axis industrial robot manipulator online, along with a plan planner for generating the final robot's path. The proposed system is composed of a camera, joystick, teach pendant and the industrial robot manipulator. The purpose of the camera is to capture images from the robot's cell to acquire information related with the position and orientation of possible objects and generate the CAD model of those objects. More information regarding the robot's work envelope are acquired through manual manipulation of the robot (joystick and teach pendant), non-specified robot movements and previously created robot programs. Different targets can be manually defined by the user. A path planner along with Voronoi roadmap generation and a mission planner, define the robot's trajectory. [51]

On the other hand, Schou et al. [52] develop an interface for programming "*Autonomous Industrial Mobile Manipulator*" (AIMM), [52]. The concept of AIMM consists of an industrial robot manipulator attached to a mobile robot, [52]. The human-robot interface is built upon a graphical user interface and physical HRI through kinesthetics guidance. The process of robot programming is achieved into two steps. The first step is called the "*Specification Phase*", [52]. On this step the user declares the sequence of the skills, such as move and pick, and the skills' parameters. Then the user proceeds to the "*Teaching Phase*", [52]. The user through kinesthetics guidance, leads the robotic arm to desired positions in skills sequence, as those were declared in the previous step. Kinesthetics guidance is achieved through a force sensor which in turn, enables the direct physical HRI. The authors conducted two experiments; one simple (pick and place) and an advanced task (assembly). Within this work, the user graphical interface and the kinesthetics guidance are the means for the robot programming. [52] This research work, describes an interesting technique for programming which can be also operated by non-

programming experts. However, it is restricted as the kinesthetics guidance requires force sensors attached to the robot manipulator, which in most of the cases is not a built-in feature.

In [53], Zoliewski and Pioskowik propose a "*concept and implementation*" for online robot programming. This implementation consists of body gestures and a user interface for enabling HRI. Human can interact with the robot via body gestures captured by a depth sensor. The target of the user interface is to manually manipulate the robot. A user manipulates the robot in X-Y-Z axes and rotate the end-effector. [53]

A similar approach is presented in [54] by Pedersen et al. In this research work, a depth camera detects human body gestures along with the integration of a user interface. Within the user interface, the user defines the sequence of the robot's skills to perform, similar to [53]. Such a skill is a grab an object; the object is indicated to the robot from the user by performing pointing gesture. Programming the robot is a process completed within two phases; sequence and teaching. In the initial phase, the user deals only with the user interface to define the sequence of the robot's skills. Then, in the second phase user by performing body gestures can instruct the mobile robot manipulator to follow him/her, and later point out the object to grasp. The authors integrated QR codes in order for the robot to recognize which location or object the user points out. [54]

On the other hand, body and hand-gestures are combined for programming an industrial robot, [5]. Body gesture are capture with a Kinect device, while hand gestures with Leap Motion Controller, as shown in Figure 3. The user can manipulate the robot upwards, downwards, right, left, forward and backward. The validation of this framework has been achieved after applying on an automotive industry for assembly. The experiments of this framework from five users indicate that body gestures are more appropriate for programming the robot only if the robot's accuracy is not required. Hand-gestures were preferred when the users were near the robot. Moreover, the participants mentioned that both hand and body-gestures are less time-consuming and are easier for the programming process compared to other ways of programming. [5]

Figure 3. *Leap Motion Controller*

Within this section, related works to the online robot programming have been presented. Different approaches with the use of different input devices such as force sensors, cameras and voice commands attempt to simplify the programming process. However, the vision-based systems provide an intuitive way for establishing the HRI.

2.5 Summary

This section summarizes the related works carried out for the field of HRI regarding the robot control and robot programming along with diverse input devices.

The field of HRI studies the communication of the human and robot under different conditions. The growth of the robots' usage initiates the need for physical interaction for humans and robots, especially within the factory floor. The research works show that with physical HRI it is possible to reduce the time for a task completion with the least possible human effort. Moreover, the emergence of sensors, allow humans access harsh environments where human presence is not possible, by remotely controlling robots. Remote interactions are not defined only due to the distinct distance among the human and the robot. Such a case is the online programming of an industrial robot manipulator through a teach-pendant.

This thesis work is classified as a remote HRI as the human operator will manipulate the industrial robot for programming purposes in close proximity but without any direct interaction.

Researchers developed various approaches for controlling industrial robot manipulators. The emergence of different gaming input devices which offer a more natural and intuitive experience, lead to the integration within the field of HRI for robot control. Vision-based devices and data glove capture the motions and movements of the human body. The users feel more comfortable as there is no need for in-depth knowledge for the operation of such devices. On the other hand, joysticks can equally control a robot manipulator, but this device as mean of interaction is not characterized as intuitive device. Similarly, hu-man-computer interfaces allow joint by joint control, but it is a rather complicated

mean. However, human-computer interfaces are helpful when their purpose is to observe the status of the operation.

Speech is another approach for interacting with robots. However, the results show that such a mean of interaction has a low recognition rate, compared to gesture recognition, and it is highly affected by the pronunciation of the people. On the other, PDA as a device for HRI allows a successful interaction but such a device lacks intuitiveness. From one perspective this device could be considered as a replacement of a teach pendant as humans defines targets to be reached and robots' end-effectors actions.

The humans use their body, head and hands to express themselves or point to an object. Such expressions raise through the humans' natural instincts. Vision-based and wearable devices capture those expressions, but with different success rates which may be critical depending the applied application. Indeed, sensors, integrated software and developed applications can increase the recognition rate.

The gesture-based robot control is extended to programming industrial robots online. Having in mind that the target is to allow task experts program industrial robots, the approach must enable an intuitive human-robot interface. From the literature, the most appropriate method to achieve this is by utilizing vision-based and wearable devices. Other approaches, such as kinesthetics guidance, require sensors equipped on the industrial robot. Collaborative robots offer this option, but they are not as widely used as the known industrial robot manipulators.

3. PROPOSAL FOR HAND-GESTURE BASED PROGRAMMING

The focus of this thesis is the development of an application that allow human operators to program industrial robot manipulators intuitively using hand-gestures as an attempt to reduce the need of traditional programming methods. Furthermore, human operator will manipulate the motion of the industrial robot with hand gestures. The use of hand gestures by the humans, comes out naturally and by instinct. Thus, hand gestures as a mean of communication, might allow humans without robot programming skills to interact and program industrial robot manipulators. Concurrently, the industrial robot manipulator, will imitate the motion that the human operator performs with the hand. Within this thesis work, the human operator must program industrial robot independently of its manufacturer and structure.

3.1 Proposal description

In order to achieve the hand-gesture based programming of the industrial robot manipulator, human operator will use both hands in order to proceed with the programming. One hand will control the robot's action and the second hand will move on the space, so that the robot will imitate this movement. As a result, two devices are required. To select the most suitable devices it must be considered the ease of use, intuitiveness of handling such devices and precision and accuracy.

The first device must be able to detect the static gestures performed with the left hand with the least possible errors. The second device must have hand-tracking possibilities to detect the motion of the hand and yield accurately the position of human operator's hand as a reference within the Cartesian space. The two devices purchased for this thesis is a data glove from CaptoGlove Company[1] and a hand-tracking device form Leap Motion Company[2].

In the industries the industrial robot manipulators that dominate are two types, 6 DOF and SCARA robots. FAST-Lab possesses both these types in its facilities. The ABB IRB120 [11] and the Adept eCobra 600 Pro from OMRON [12] were chosen to be the robot manipulators for this thesis.

---

[1] https://www.captoglove.com/
[2] https://www.leapmotion.com/

To compute the inverse kinematics, for creating the joint target, the MATLAB[3] software will be used. Apart from this, within the MATLAB software, inverse kinematic solutions that are not feasible for the industrial robot manipulators must be discarded. The final joint target must be sent to robot's controller and at the AC.

A graphical web-based HMI (Human-Machine Interface) is required to be developed. The purpose of the HMI is not related to the robot's programming process rather to define a set of settings for establishing the HRI. The HMI must allow the human operator to select the industrial robot manipulator that is going to be programmed. The importance of this selection is to redirect the Application Controller to the choose the appropriate kinematic calculator to solve the inverse kinematics. Moreover, the human operator must be able to declare what will be the available workspace limits to manipulate the industrial robot. The limits must define the upper and lower possible reachable coordinate in X-Y-Z axes. As each robot manipulator might have different IP and port settings in order to establish a connection, human operator must be able to define those settings. Then the human operator must connect the data glove with the Application Controller. The same option does not exist for the hand-tracking device as there is no need for such an action. After the completion of the robot's programming process, the human operator must be able to generate the final robot's code. The HMI provides also to the user the possibility to observe I which Cartesian coordinate position is currently the TCP of the robot and the connection status of all the components within the system must be shown in order to recognize any failure in the connection process. Finally, instructions regarding the operation of the whole application and the predefined hand gestures must be easily accessible.

Human operator must be able to determine by hand gestures the actions of the robot. In another approach all of the controlling commands regarding the robot's action could be added in the HMI. However, such an approach reduces the intuitiveness are distracts the human operator from the main goal, the robot programming. The main actions of the robot must be to move in the workspace or hold its current position. Two more gestures are required to control the end-effector of the gripper. For this thesis as an end-effector is selected a gripper, and thus the human operator must be able to send the command to grasp or release the work-object. The initiation and termination of storing the targets must be marked with two different hand gestures. Moreover, the distinction of different task within one robot code can be achieved. Last but not least, is the determination of whether the gestures are enabled or disabled. This feature ensures that any accidental

[3] https://www.mathworks.com/products/matlab.html

attempt to perform a gesture by the human operator will not cause any undesirable action.

To establish the HRI it is vital to develop a controller that will handle all the communication of all the aforementioned components. The controller for this thesis has been named Application Controller (AC) and precedes that all the components must be able to communicate using Sockets for exchanging all the data. The AC must receive the readings from the sensors of the data glove and recognize the currently performed gesture by the human operator's left hand. Meanwhile, the right hand must be in the interaction area of the hand-tracking device, so that the AC receives the position. In case that one these requirements are not fulfilled the AC should not order any action to the robot.

After the AC receives from the HMI the robot to be programmed, the robot's network settings and the workspace limits, it establishes connection with the MATLAB software. Then the position of the tracked by the hand-tracking device, is sent to MATLAB as a target in the Cartesian space, where the inverse kinematics will be computed and then sent to the robot's controller the new joint target. This new joint target must be received also by the AC in order to be stored for creating the robot's code.

The human operator indicates the termination of the programming process by performing the appropriate gesture. The AC analyses the programmed paths and discards unnecessary targets. The human operator orders the final robot code from the HMI, the AC must generate the appropriate code, depending the robot manufacturer.

Finally, task optimization techniques will be examined in order to optimize the created path by the human operator. The purpose of such algorithms is to attempt to smooth the created path and reduce the cycle time of the task.

3.2 Components

This section presents a description and the technical characteristics of the selected components to proceed with this thesis work. The first component is the data glove. Various data gloves are available in the market with different integrated sensors. Most common are the bending sensors and IMUs (Inertial Measurement Units). Wireless communication technologies allow the users to operate such a device without the restriction of being in close proximity with the computer. Regarding the gesture recognition, data gloves offer higher successful recognition rates compared to vision-based devices.

The hand-tracking device chosen, the Leap Motion Controller, tracks the displacement of the hand and have the possibility to have a reference point from which the human

operator can always start to manipulate the robot manipulator. Devices with depth cameras can also detect the position of the hand. However, the Leap Motion Controller provides data with high accuracy, [55].

The selected industrial robot manipulators are the ABB IRB120 and OMRON Adept eCobra 600 PRO. The IRB120 is a 6 DOF industrial robot and eCobra 600 PRO a SCARA type of robot manipulator with 4 DOF. The selection of these robot manipulators is based on the validation of hand-gesture programming robot manipulators independently of their manufacturer and structure.

The calculator of inverse kinematics is software component. The development is achieved through the MATLAB software.

The integration and communication of the all the hardware and software components will be achieved through a main controller, the Application Controller. AC will connect to the data glove, hand-tracking device and the Inverse Kinematics Calculator (IKC) to recognize the performed gestures, receive the hand-position information and transmit them to the IKC. Finally, the industrial robot manipulators, will communicate only with the IKC.

3.2.1 Data glove

The pair of data glove chosen for this thesis is the CaptoGlove. Figure 4 depicts the right-hand glove of CaptoGlove utilized within this thesis. Table 1 shows the technical specifications of the data glove. They offer individual finger and hand-tracking features. In order to capture the static gestures five bending sensors are integrated on the glove, one on each finger. The glove offers wireless connection with the computer using BTLE (Bluetooth Low Energy)[4] technology, which allows the user to move freely without being restricted from cables.

The CaptoGlove Company provides offers different SDK (Software Development Kit) packages (Unreal 4.0, .NET and Unity, C++) [56] for developing applications in order to retrieve data from the data glove's sensors. For this thesis, the C++ SDK will be used in order to acquire data from the finger's sensors and then transmit them to the controller.

[4] https://www.bluetooth.com/

Figure 4. *CaptoGlove [57]*

Table 1. *CaptoGlove technical specifications [57]*

Characteristics	Attributes
Sensors	• Gyroscope (X, Y and Z-axes) • Accelerometer (X, Y and Z-axes) • Magnetometer (X, Y and Z-axes) • Barometer • Five bending sensors • One pressure sensor
Battery	Ten hours rechargeable Li-ion Polymer Battery (3.7V USB cable)
Connectivity Technology	BLTE

This pair of data glove was chosen for this thesis due to the ease of data transmission, the wireless connectivity, the battery's capacity and the availability to replace the bending sensors in case of breaking down. Another data glove was previously purchased with similar characteristics, but the replacement of sensors and the support were insufficient.

3.2.2 Hand-tracking device

As the robot manipulator must follow the position of the human operator's hand, a hand-tracking device is required. The Leap Motion Controller, is presented in Figure 5 (left figure), can track the palms and the fingers of both hands and offers accurate date of the palm and the fingers' pinpoint with low latency. More technical specifications of the device are shown in Table 2. Compared to other tracking devices which integrate depth sensors, Leap Motion Controller integrates cameras and IR (Infrared) LEDs. As a result, the tracking of the hand can occur also in low lighting conditions, but the interaction area is significantly smaller compared to depth sensor devices. The interaction of the Leap Motion Controller is shown in Figure 5 (right figure). The device is able to detect motions

of a human operators' hands 80 cm above the device, 80 cm wide and deep with 150º and 120º angle respectively [58].

Figure 5. *Leap Motion Controller (left figure) and the interaction area (right figure) [58]*

Table 2. *Leap Motion Controller technical specifications [59]*

Characteristics	Attributes
Sensors	• Two IR cameras • Three LEDs
Frames	200 per second
Dimensions	• Width: 80 mm • Depth: 30 mm • Height: 13 mm
Connectivity Technology	USB cable

3.2.3 Industrial Robot Manipulators

Two robots were chosen for this thesis work, the ABB IRB 120 and the Adept eCobra 600 Pro from OMRON. The FAST-Lab has is equipped with different robot manufacturers and robot types. For this thesis, the two robots were chosen to test the outcome of this thesis not only in one specific robot type and manufacturer. The following subsections briefly describe the specifications of the two industrial robot manipulators.

3.2.4 OMRON Adept eCobra 600 PRO

The first industrial robot manipulator is the Adept eCobra 600 Pro from OMRON [60]. Figure 6 shows the eCobra 600 Pro robot. This is a SCARA robot with 4 DOF. The maximum reach is 600 mm and it can carry out payload up to 5.5 kg. the robot's controller and amplifiers are fully integrated in back side of the robot. Table 3 shows the eCobra 600 Pro specifications [12]. The robot is integrated in the FASTory line at the FAST-Lab

of Tampere University in Hervanta. In this case the robot is already equipped with an end-effector for painting mobile phones on paper.

Figure 6. *OMRON Adept eCobra 600 PRO [60]*

Table 3. *Adept eCobra 600 PRO specifications [12]*

Feature	Value
Degrees of Freedom	4
Handling capacity (kg)	5.5 kg
Reach	600 mm
Weight (kg)	41 kg

3.2.5 ABB IRB120

The second industrial robot manipulator to be programmed is the ABB IRB120, as shown in Figure 7 (left figure). This is a 6 DOF robot with maximum reach at 580 mm and can carry workpieces up to 3 kg taking into consideration the weight of the robot's end-effector. The robot has as an end-effector the ABB Smart Gripper.

The robot's controller is the IRC5 Compact [61], as shown in Figure 7 (right figure), with the control software RobotWare in charge of robot's motion control, development and external communication. Table 4 show the specification of ABB IRB120 robot manipulator.

Figure 7. *ABB IRB120 industrial robot (left figure) [11], and IRC5 Compact robot controller (right figure) [61]*

Table 4. *ABB IRB120 specifications [62]*

Feature	Value
Degrees of Freedom	6
Handling capacity (kg)	3 kg
Reach	580 mm
Weight (kg)	25 kg

3.3 Robot manipulator kinematics

The robot manipulator kinematics are classified into two kinematics problems, the forward and inverse kinematics, as shown in Figure 8. Forward kinematics regards the calculation of position and orientation for the end-effector, given the angles of each robot's joint. On the other hand, the inverse kinematics problem, computes the robot's joint angles given the desired position and orientation to be reached by the industrial robot manipulator.

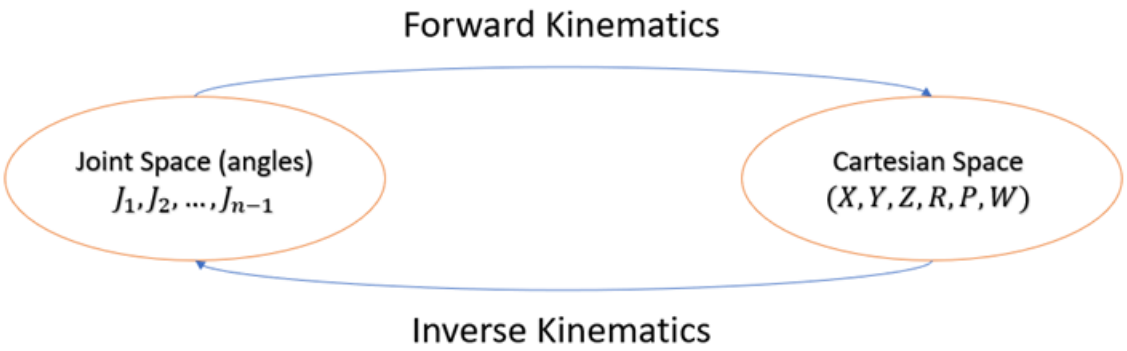

Figure 8. *Transition from forward to inverse kinematics*

3.3.1 Forward kinematics

Forward kinematics concern the derivation of the position (in Cartesian space) and orientation of the robot manipulator's end-effector considering the given angle of each robot's joint ($J_1 \dots J_n$), as shown in Figure 9.

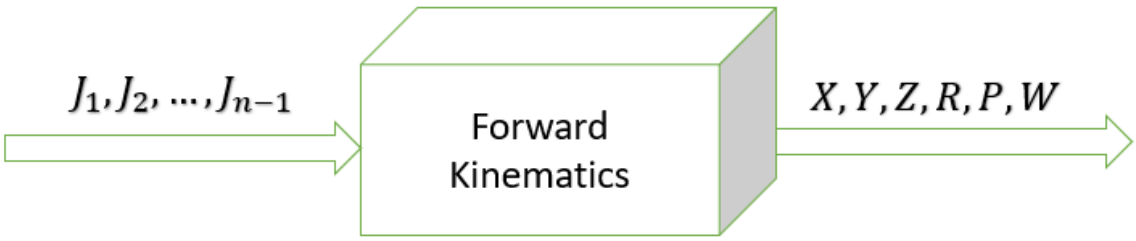

Figure 9. *Outcome of forward kinematics*

In 1955, Denavit and Hartenberg [63], [64] presented a convention for attaching coordinate frames to spatial linkages. Paul [65], showed the ease of DH convention to describe the geometry of a robot manipulator and from it derive the kinematic equations of the robot manipulator which later leads to the calculation of the forward kinematics. The DH table along with the description of each feature is presented in Table 5.

Table 5. *Denavit-Hartenberg convention [66], [67]*

Feature	Symbol	Description
Joint angle	θ	The angle from $x_{i-1}$to x_i axes about z_{i-1} axis.
Link offset	d	The distance from the origin of frame $i-1$ to x_i along z_{i-1}.
Link twist	a	The angle from $z_{i-1}$to z_i axis about x_i.
Link length	α	The offset distance from the z_{i-1} to z_i along the x_i.

This convention, or as it is usually called DH convention, describes the rotation and translation between two frames along Z-axis and X-axis, and it represented by a 4x4 homogenous transformation matrix. This matrix is a representation of the rotation and the translation that occurs on X and Z-axes, and it is described using the following matrix representation, as shown in [67]:

$$ {}^{i-1}_{i}T = Rot_{z_i}(d_i)\, Trans_{z_i}(d_i)\, Rot_{x_i}(\alpha_i)\, Trans_{x_i}(a_i) \tag{1} $$

Each of the individual matrices to construct the homogenous transformation matrix are presented below:

$$ \begin{aligned} Trans_{z_i}(d_i) &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ Rot_{z_i}(\theta_i) &= \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ Trans_{x_i}(a_{i,\,i+1}) &= \begin{bmatrix} 1 & 0 & 0 & a_{i,\,i+1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ Rot_{x_i}(\alpha_{i,\,i+1}) &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_{i,\,i+1} & -\sin\alpha_{i,\,i+1} & 0 \\ 0 & \sin\alpha_{i,\,i+1} & \cos\alpha_{i,\,i+1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{2} $$

The complete transformation matrix is:

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \sin\theta_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3} $$

where θ_i, d_i, a_i and α_ι are the geometric parameters of the ith joint as those were defined in DH convention [67], [68].

The homogenous transformation matrix consists of two sub-matrices, a 3x3 rotation matrix and a 3x1 translational vector. The homogenous transformation matrix can be represented, as shown in [67], as:

$$ {}^{i-1}_{i}T = \begin{bmatrix} R_{3x3} & P_{3x1} \\ 0 & 1 \end{bmatrix} \tag{4} $$

3.3.2 Inverse kinematics

Inverse kinematics concern the calculation of the robot manipulator's joint angles $J_1 \ldots J_n$, given the position and orientation of the end-effector as it can be seen through Figure 10.

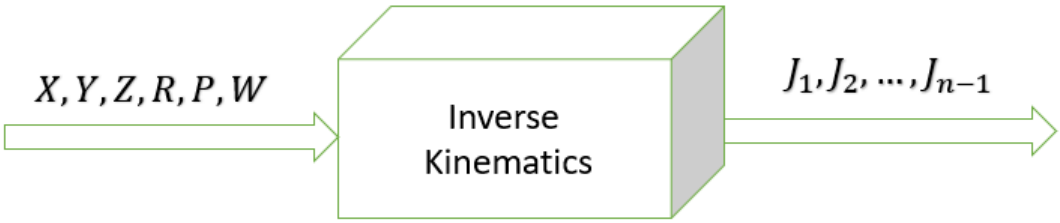

Figure 10. *Outcome of inverse kinematics*

To solve the problem of inverse kinematics different approaches can be selected, depending the robot's structure. The following sub-sections describe the kinematic analysis of the two industrial robot manipulators and the calculation of the inverse kinematics for each individual robot of this thesis.

3.3.3 Adept eCobra 600 PRO kinematics analysis

First the DH convention is defined for the OMRON Adept eCobra 600 PRO according to the Denavit-Hartenberg rules, as those defined on Table 5. Table 6 shows the D-H convention for the Adept eCobra 600 PRO industrial robot.

Table 6. *D-H convention of Adept eCobra 600 PRO*

Joint i	θ	d	a	α
1	θ_1	387	325	0
2	θ_2	0	275	π
3	0	q_3	0	0
4	θ_4	0	0	0

Then the homogenous transformation matrices between two joints from the robot's base to the robot's TCP are defined using the Equation (1). In total 4 matrices will be created.

$$
\begin{aligned}
{}^{0}_{1}T &= \begin{bmatrix} c_1 & -s_1 & 0 & 325 * c_1 \\ s_1 & c_1 & 0 & 325 * s_1 \\ 0 & 0 & 1 & 387 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^{1}_{2}T &= \begin{bmatrix} c_2 & s_2 & 0 & 275 * c_2 \\ s_2 & -c_2 & 0 & 275 * s_2 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^{2}_{3}T &= \begin{bmatrix} c_3 & -s_3 & 0 & 0 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & q_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
\tag{5}
$$

$$ {}^{3}_{4}T = \begin{bmatrix} c_4 & -s_4 & 0 & 0 \\ s_4 & c_4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

where $c_i = \cos(\theta_i)$ and $s_i = \sin(\theta_i)$.

To obtain the homogenous transformation matrix from the base of the industrial manipulator till robot's TCP, the previous matrices must be multiplied as:

$$ {}^{0}_{4}T = {}^{0}_{1}T * {}^{1}_{2}T * {}^{2}_{3}T * {}^{3}_{4}T \tag{6} $$

This matrix from Equation (6) will have the following form:

$$ {}^{0}_{4}T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7} $$

where:

$$ r_{11} = c_4 * \big(c_3 * (c_1 * c_2 - s_1 * s_2) + s_3 * (c_1 * s_2 + c_2 * s_1)\big) + s_4 * (c_3 * (c_1 * s_2 + c_2 * s_1) - s_3 * (c_1 * c_2 - s_1 * s_2)) $$

$$ r_{12} = c_4 * \big(c_3 * (c_1 * s_2 + c_2 * s_1) + s_3 * (c_1 * c_2 - s_1 * s_2)\big) - s_4 * (c_3 * (c_1 * c_2 - s_1 * s_2) - s_3 * (c_1 * s_2 + c_2 * s_1)) $$

$$ r_{13} = 0 $$

$$ r_{21} = c_4 * \big(c_3 * (c_1 * s_2 + c_2 * s_1) - s_3 * (c_1 * c_2 - s_1 * s_2)\big) - s_4 * (c_3 * (c_1 * c_2 - s_1 * s_2) - s_3 * (c_1 * s_2 + c_2 * s_1)) $$

$$ r_{22} = -c_4 * \big(c_3 * (c_1 * c_2 - s_1 * s_2) + s_3 * (c_1 * s_2 + c_2 * s_1)\big) - s_4 * (c_3 * (c_1 * s_2 + c_2 * s_1) - s_3 * (c_1 * c_2 - s_1 * s_2)) $$

$$ r_{23} = 0 \tag{8} $$

$$ r_{31} = 0 $$

$$ r_{32} = 0 $$

$$ r_{33} = -1 $$

$$ p_x = 325 * c_1 + 275 * c_1 * c_2 - 275 * s_1 * s_2 $$

$$ p_y = 325 * s_1 + 275 * c_1 * s_2 - 275 * c_2 * s_1 $$

$$ p_z = 387 - q_3 $$

and $c_i = \cos(\theta_i)$ and $s_i = \sin(\theta_i)$.

3.3.4 Adept eCobra 600 PRO inverse kinematics

The structure of the Adept eCobra 600 PRO consists of three revolute ($\theta_1, \theta_2, \theta_4$) and one prismatic ($q_3$) joints, so the robot has a configuration of RRPR. The calculation of the inverse kinematics for the OMRON Adept eCobra 600 pro is achieved using the geometric approach, similarly to [69]. In order to proceed with solving the problem of inverse kinematics of the Adept eCobra 600 PRO industrial the schematic diagram of a SCARA robot in 2D from the top side must be considered, as shown in Figure 11.

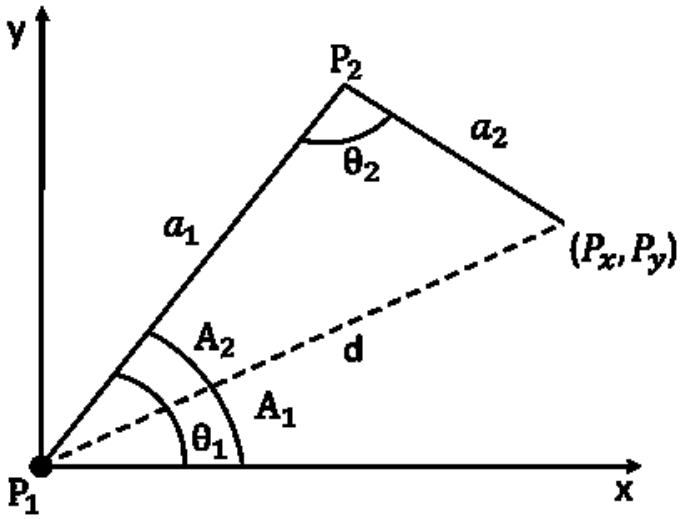

Figure 11. *Schematic diagram of SCARA robot in 2D*

From the above diagram a_1 and a_2 represent the length of the segment P_1 to P_2 and P_2 to (P_x, P_y) respectively. θ_1 and θ_2 are the angles of joint 1 and joint 2. The values A_1 and A_2 are the divided angles of θ_1.

The points P_1, P_2 and P_x, P_y, form a triangle with three sides, a_1, a_2 and d. Applying the law of cosines in this triangle it is possible to calculate the angle of θ_2. The law of cosines comes from the Pythagorean theorem which is applied only on right-angle triangles. The following Equation (9) express the law of cosines, from which the angle θ_2 is calculated:

$$d^2 = a_1^2 + a_2^2 - 2 * a_1 * a_2 \cos(\theta_2) \quad (9)$$

and therefore:

$$\theta_2 = \pm \cos^{-1}\left(\frac{a_1^2 + a_2^2 - d^2}{2 * a_1 * a_2}\right) \quad (10)$$

The next step is to calculate the angles of A_1 and A_2 in order to find the θ_1. From the schematic diagram of the Figure 11, it can be observed that a right-angle triangle is created, with points P_1, P_2 and P_x, P_y.

Applying the arctangent of y and x the angle of A_1 is calculated as shown in the following equation:

$$A_1 = \tan^{-1}(\frac{y}{x}) \quad (11)$$

On the other hand, in order to calculate A_2 the law of cosines must be applied again.

$$A_2 = \cos^{-1}(\frac{d^2 + a_1^2 - a_2^2}{2 * d * a_1}) \quad (12)$$

Then by adding or subtracting the angles of A_1 and A_2 it is possible calculate angle of θ_1 depending the required configuration of the robot (right or left configuration).

To solve the displacement of q_3:

$$q_3 = d_1 - P_z \quad (13)$$

Finally, the calculation of θ_4 is achieved by:

$$\theta_4 = \theta_1 + \theta_2 - tan_2^{-1}(r_{11}, r_{12}) \quad (14)$$

3.3.5 ABB IRB120 kinematics analysis

The DH convention is constructed following the Denavit-Hartenberg rules as shown in Table 5. Therefore, the DH convention for the ABB IRB120 robot is shown in Table 7:

Table 7. *DH convention for the ABB IRB120*

Joint *i*	$\boldsymbol{\theta}$	$\boldsymbol{d}$	$\boldsymbol{a}$	$\boldsymbol{\alpha}$
1	θ_1	290	0	$-\pi/2$
2	$\theta_2 - \pi/2$	0	270	0
3	θ_3	0	70	$-\pi/2$
4	θ_4	302	0	$\pi/2$
5	θ_5	0	0	$-\pi/2$
6	$\theta_6 + \pi$	72	0	0

Then the homogenous transformation matrices between two joints from the robot's base to the TCP are defined using the Equation (1). In total 6 matrices will be created.

$$
\begin{aligned}
{}^{0}_{1}T &= \begin{bmatrix} c_1 & 0 & -s_1 & 0 \\ s_1 & 0 & c_1 & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^{1}_{2}T &= \begin{bmatrix} c_2 & -s_2 & 0 & 270 * c_2 \\ s_2 & c_2 & 0 & 270 * s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^{2}_{3}T &= \begin{bmatrix} c_3 & 0 & -s_3 & 70 * c_3 \\ s_3 & 0 & c_3 & 70 * s_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^{3}_{4}T &= \begin{bmatrix} c_4 & 0 & s_4 & 0 \\ s_4 & 0 & -c_4 & 0 \\ 0 & 1 & 0 & 302 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^{4}_{5}T &= \begin{bmatrix} c_5 & 0 & -s_5 & 0 \\ s_5 & 0 & c_5 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
{}^{5}_{6}T &= \begin{bmatrix} c_6 & -s_6 & 0 & 0 \\ s_6 & c_6 & 0 & 0 \\ 0 & 0 & 1 & 72 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
\tag{15}
$$

where $c_i = \cos(\theta_i)$ and $s_i = \sin(\theta_i)$.

To obtain the homogenous transformation matrix from the robot manipulator's base to the robot's TCP, the previous matrices must be multiplied as:

$$
{}^{0}_{6}T = {}^{0}_{1}T * {}^{1}_{2}T * {}^{2}_{3}T * {}^{3}_{4}T * {}^{4}_{5}T * {}^{5}_{6}T \tag{16}
$$

This matrix will have the following form:

$$
{}^{0}_{6}T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{17}
$$

where:

$$
\begin{aligned}
r_{11} = {} & s_6 * \big(c_4 * s_1 - s_4 * (c_1 * c_2 * c_3 - c_1 * s_2 * s_3)\big) - c_6 * (s_5(c_1 * c_2 * s_3 + c_1 * c_3 * s_2) \\
& - c_5 * (s_1 * s_4 + c_4 * (c_1 * c_2 * c_3 - c_1 * s_2 * s_3)) \\
r_{12} = {} & s_6 * \Big(s_5 * (c_1 * c_2 * s_3 + c_1 * c_3 * s_2) - c_5 \\
& * \big(s_1 * s_4 + c_4 * (c_1 * c_2 * c_3 - c_1 * s_2 * s_3)\big)\Big) + c_6 \\
& * \big(c_4 * s_1 - s_4 * (c_1 * c_2 * c_3 - c_1 * s_2 * s_3)\big) \\
r_{13} = {} & -c_5 * (c_1 * c_2 * s_3 + c_1 * c_3 * s_2) - s_5 * (s_1 * s_4 + c_4 * (c_1 * c_2 * c_3 - c_1 * s_2 * s_3))
\end{aligned}
\tag{18}
$$

$$r_{21} = -c_6 * \Big(s_5 * (c_2 * s_1 * s_3 + c_3 * s_1 * s_2) + c_5 * \big(c_1 * s_4 - c_4(c_2 * c_3 * s_1 - s_1 * s_2 * s_3)\big)\Big) - s_6 * (c_1 * c_4 + s_4 * (c_2 * c_3 * s_1 - s_1 * s_2 * s_3))$$

$$r_{22} = s_6 * \Big(s_5 * (c_2 * s_1 * s_3 + c_3 * s_1 * s_2) + c_5 * \big(c_1 * s_4 - c_4 * (c_2 * c_3 * s_1 - s_1 * s_2 * s_3)\big)\Big) - c_6 * (c_1 * c_4 + s_4 * (c_2 * c_3 * s_1 - s_1 * s_2 * s_3))$$

$$r_{23} = s_5 * \big(c_1 * s_4 - c_4 * (c_2 * c_3 * s_1 - s_1 * s_2 * s_3)\big) - c_5 * (c_2 * s_1 * s_3 + c_3 * s_1 * s_2)$$

$$r_{31} = s_4 * s_6 * (c_2 * s_3 + c_3 * s_2) - c_6 * (s_5 * (c_2 * c_3 - s_2 * s_3) + c_4 * c_5 * (c_2 * s_3 + c_3 * s_2))$$

$$r_{32} = s_6 * \big(s_5 * (c_2 * c_3 - s_2 * s_3) + c_4 * c_5(c_2 * s_3 + c_3 * s_2)\big) + c_6 * s_4 * (c_2 * s_3 + c_3 * s_2)$$

$$r_{33} = c_4 * s_5 * (c_2 * s_3 + c_3 * s_2) - c_5 * (c_2 * c_3 - s_2 * s_3)$$

$$p_x = 270 * c_1 * c_2 - 72 * c_5 * (c_1 * c_2 * s_3 + c_1 * c_3 * s_2) - 72 * s_5 * \big(s_1 * s_4 + c_4 * (c_1 * c_2 * c_3 - c_1 * s_2 * s_3)\big) + 70 * c_1 * c_2 * c_3 - 302 * c_1 * c_2 * s_3 - 302 * c_1 * c_3 * s_2 - 70 * c_1 * s_2 * s_3$$

$$p_y = 270 * c_2 * s_1 - 72 * c_5 * (c_2 * s_1 * s_3 + c_3 * s_1 * s_2) + 72 * s_5 * \big(c_1 * s_4 - c_4 * (c_2 * c_3 * s_1 - s_1 * s_2 * s_3)\big) + 70 * c_2 * c_3 * s_1 - 302 * c_2 * s_1 * s_3 - 302 * c_3 * s_1 * s_2 - 70 * s_1 * s_2 * s_3$$

$$p_z = 302 * s_2 * s_3 - 302 * c_2 * c_3 - 70 * c_2 * s_3 - 70 * c_3 * s_2 - 270 * s_2 - 72 * c_5 * (c_2 * c_3 - s_2 * s_3) + 72 * c_4 * s_5 * (c_2 * s_3 + c_3 * s_2) + 290$$

and $c_i = \cos(\theta_i)$ and $s_i = \sin(\theta_i)$.

3.3.6 ABB IRB120 inverse kinematics

The ABB IRB 120 is an industrial robot manipulator with a spherical wrist. This can be justified due to the fact that the axes of joints 4, 5 and 6 of the robot intersect. Hence, to solve the inverse kinematics for the IRB120, a closed-form solution will be chosen to obtain all the joint angles. The problem of inverse kinematics will be splitted into two distinct parts. The first problem concerns the calculation of the position of θ_1, θ_2 and θ_3, similarly to [67], [70], [71]. Then, the second problem regards the orientation to be achieved by the end-effector. This concerns the calculation of the last three joint angles (θ_4, θ_5 and θ_6), as J. J. Craig, [72].

On such a manipulator, the first step is to define the V point, which is the point in which the last three joints intersect. The V can be calculated using the following equation similar to [67]:

$$P_V = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} - d_6 * \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \tag{19}$$

Then the angle θ_1 can be computed as:

$$\theta_1 = tan_2^{-1}(P_{Vx}, P_{Vy}) \tag{20}$$

If there is a solution for the θ_1, then there is also a solution for $\theta_1 + \pi$, [67]. If θ_1 has two possible solutions, then angles θ_2 and θ_3 can also have two possible solutions. Up to this point, there are in total four possible solutions.

The next step is to calculate the θ_2. To achieve this, a schematic diagram of link 2 and 3 of the robot's manipulator will be considered, as shown in the following Figure 12.

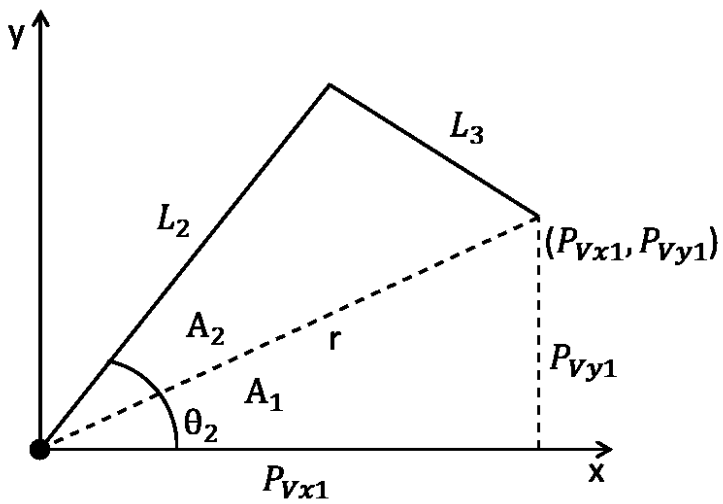

Figure 12. *Schematic diagram of link 2 and link 3*

To calculate the θ_2 angle, the law of cosines is applied.

The points P_{Vx} and P_{Vy} are the robot's TCP in terms of the previously calculated θ_1. The matrix ${}^0_1T$ is calculated according to the solved θ_1 angle and then multiplied by the previously calculated P_V matrix.

$$P_{V1} = \begin{bmatrix} P_{Vx} \\ P_{Vy} \\ P_{Vz} \end{bmatrix} * {}^0_1T \tag{21}$$

Now, the angle θ_2 is calculated according to the following equations.

$$r = \sqrt{P_{Vx1}^2 + P_{Vy1}^2} \tag{22}$$

$$A_1 = atan_2^{-1}(P_{Vy1}, P_{Vx1})) \tag{23}$$

$$A_2 = \cos^{-1}((L_2^2 + r^2 - L_3^2)/(2 * r * L_2)) \quad (24)$$

Now we calculate θ_2 for two possible configurations, elbow up and elbow down [70], [71].

$$\theta_2 = \frac{\pi}{2} - A_1 - A_2 \quad (25)$$

and,

$$\theta_2 = \frac{\pi}{2} - A_1 + A_2 \quad (26)$$

The angle of θ_3 is calculated similarly to θ_2, as in [70], [71] with the following equations:

$$r = \sqrt{P_{Vx1}^2 + P_{Vy1}^2} \quad (27)$$

$$A_3 = \cos^{-1}((L_2^2 + L_3^2 - r^2)/(2 * L_2 * L_3)) \quad (28)$$

Finally, the two solutions as for the robot's configuration with the elbow up and elbow down for θ_3 are calculated similar to [70], [71]:

$$\theta_3 = \pi - (\pi - A_3) - A_3 \quad (29)$$

$$\theta_3 = \pi - (\pi - A_3) + A_3 \quad (30)$$

Now the calculation of the first subproblem (calculation of the position of θ_1, θ_2 and θ_3) is completed. Then, the second problem regards the orientation of the end-effector will be calculated (angles of θ_4, θ_5 and θ_6). To achieve this, the Z-Y-Z Euler solution will be utilized to determine the θ_4, θ_5 and θ_6, as those explained by J. J. Craig's book, [72].

The rotation matrix of the Z-Y-Z Euler solution will be the following, [72]:

$$R_{zyz}(\varphi, \theta, \psi) = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (31)$$

$$R_{zyz}(\varphi, \theta, \psi) = \begin{bmatrix} c_\varphi * c_\psi * c_\psi - s_\varphi * s_\psi & -c_\varphi * c_\psi * c_\psi - s_\varphi * s_\psi & c_\varphi * s_\theta \\ s_\varphi * c_\theta * c_\psi - c_\varphi * s_\psi & -s_\varphi * c_\theta * c_\psi - c_\varphi * s_\psi & s_\varphi * s_\theta \\ -s_\theta * c_\psi & s_\theta * s_\psi & c_\theta \end{bmatrix} \quad (32)$$

In order to calculate the angles of the last three joints, their rotation matrix must be derived. With the already calculated θ_1, θ_2 and θ_3, it is possible to calculate the rotation matrix ${}^0_3R$. Given the rotation matrix from the robot's base to the robot's TCP, the ${}^3_6R$ can be derived as, [72]:

$${}^3_6R = ({}^0_3R)^T * {}^0_6R \quad (33)$$

Then,

$$ {}^{3}_{6}R_{Z'Y'Z'}(\varphi,\theta,\psi) = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (34) $$

Now the angles of θ_4, θ_5 and θ_6 can be calculated by applying Z-Y-Z Euler's formula, [72].

If $\theta_5 \neq 0$ then for elbow up:

$$ \theta_5 = tan_2^{-1}(\sqrt{r_{31}^2 + r_{32}^2}, r_{33}) \quad (35) $$

$$ \theta_4 = tan_2^{-1}(\frac{r_{32}}{s_\beta}, \frac{-r_{31}}{s_\beta}) \quad (36) $$

$$ \theta_6 = tan_2^{-1}(\frac{r_{23}}{s_\beta}, \frac{r_{13}}{s_\beta}) \quad (37) $$

and for elbow down:

$$ \theta_5 = tan_2^{-1}(-\sqrt{r_{31}^2 + r_{32}^2}, r_{33}) \quad (38) $$

$$ \theta_4 = tan_2^{-1}(-\frac{r_{32}}{s_\beta}, -\frac{-r_{31}}{s_\beta}) \quad (39) $$

$$ \theta_6 = tan_2^{-1}(-\frac{r_{23}}{s_\beta}, -\frac{r_{13}}{s_\beta}) \quad (40) $$

In case that $\theta_5 = 0$, then:

$$ \theta_4 = 0 \quad (41) $$

$$ \theta_6 = tan_2^{-1}(-r_{12}, r_{11}) \quad (42) $$

And if $\theta_5 = 180$, then:

$$ \theta_4 = 0 \quad (43) $$

$$ \theta_6 = tan_2^{-1}(r_{12}, -r_{11}) \quad (44) $$

3.4 Interactions and communication of integrated components

To enable the HRI for programming industrial robot manipulators, the abovementioned components must communicate to exchange the necessary data for achieving the goal of programming the industrial robot manipulators with hand-gestures. The main mean of

communication among the integrated components is the Sockets. All the hardware devices (data glove, hand-tracking device and industrial robots) allow the external communication with Sockets.

The data glove transmits via Bluetooth the data from its sensors, to the AC through a C++ application. This application is developed through the provided SDK by the data glove's company. On the other hand, the hand-tracking device connects to a computer via USB cable. However, a built-in application from the hand-tracking device's company receives the data extracted through the device's cameras and sends them to the AC through Sockets. Those two hardware components enable the HRI to manipulate and program the industrial robot.

The industrial robot manipulators connect to a computer through an Ethernet cable, and communicate only with the IKC. The IKC receives the target that contains the coordinate in terms of the Cartesian space, to be reached by the robot manipulator. Afterwards, it calculates the robot configuration and transmits it to the industrial robot.

The web-based HMI has an active role on the programming process for the initialization of the programming process, the duration, as long as the termination of the process. The HMI is accessible by the human operator through a web browser.

The Figure 13, depicts the communication and interactions of the integrated hardware and software components. Each bidirectional connection represents the communication among each component.

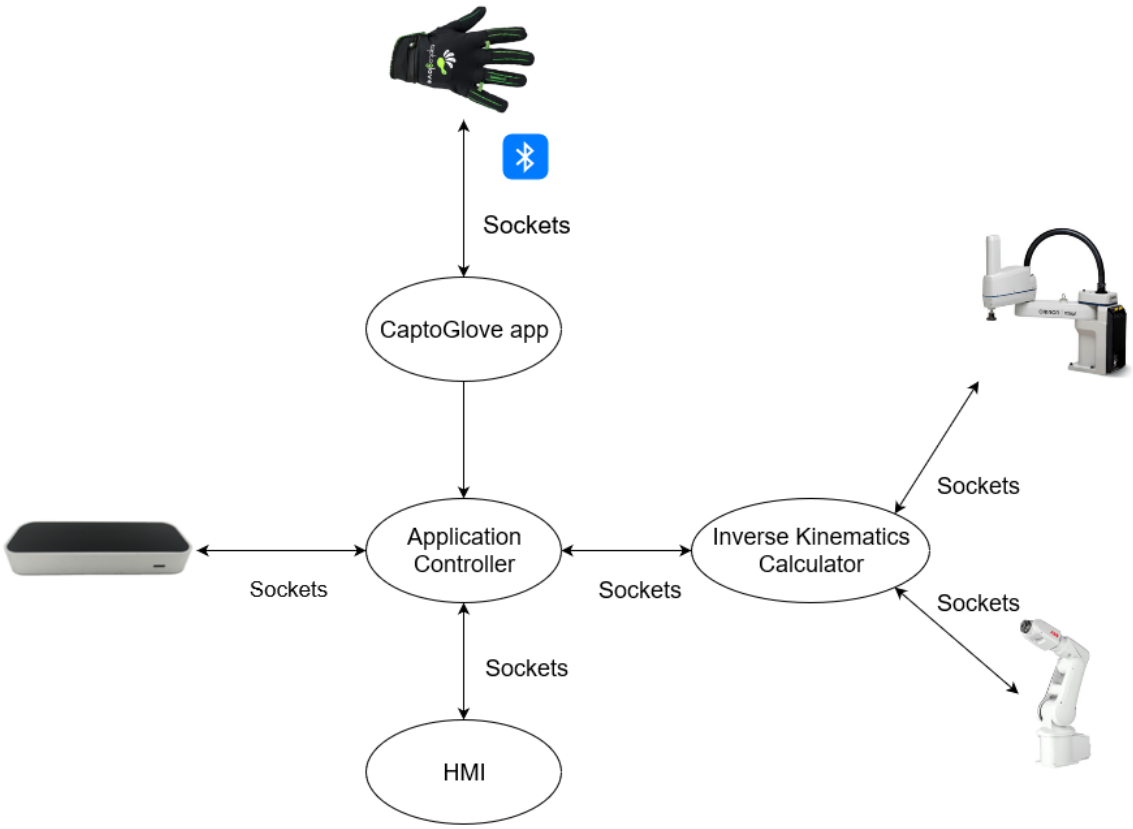

Figure 13. *Integration and communication of hardware and software components*

4. IMPLEMENTATION

The previous chapter introduces the description of the proposal for the programming of industrial robot manipulators using hand-gestures. In addition, a description of the components utilized within the Thesis Work is provided. Finally, the kinematics analysis and inverse kinematics of two industrial robots are also presented along with the integration and communication of the selected components.

This chapter presents the functionality of the application (Application Controller) developed to handle the interactions among human and robot. This includes the communication of the components with the AC, the defined static hand-gestures and the functions of the HMI. Moreover, a description is provided related to the path improvement functions, the creation of the target and the generation of the final's rapid code, along with the path improvement functions and the limitations set on the AC. Last are described the approaches to optimize the robot's task.

4.1 Application Controller

AC is the main controller of the Thesis Work. Its purpose is to enable HRI for programming the industrial robot manipulators by recognizing hand-gestures and hand displacements. So, AC must handle all the required communications with the components. The communication between AC and the components is achieved with Sockets. For this Thesis work, each of the component is determined as a server, while AC hosts three clients. More details regarding these communications are provided in the sections, 4.1.1, 4.1.3 and 4.1.4. Figure 14 depicts the general sequence diagram with the exchanged messages among the components.

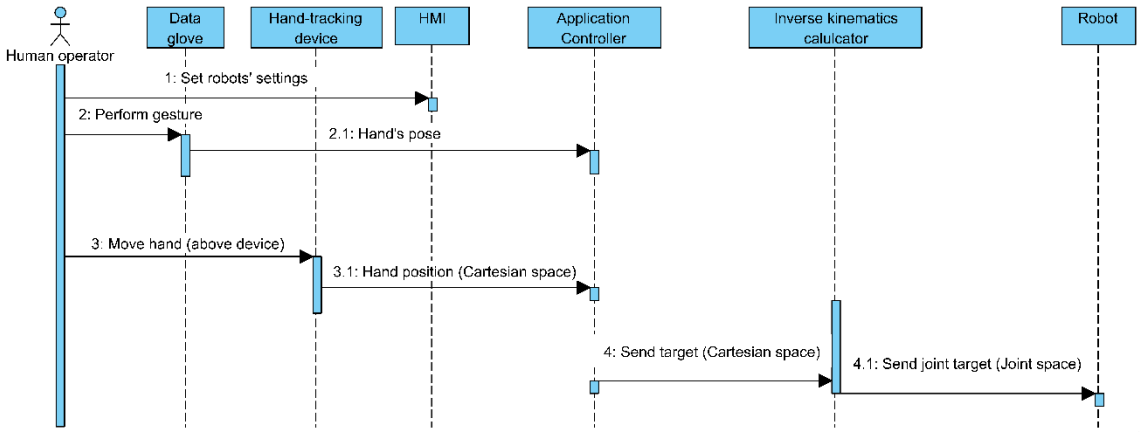

Figure 14. *Sequence diagram with the components' communications*

The programming language of the AC is JavaScript[5] using Node.Js[6], a JavaScript environment, based on the Chrome's V8 Engine[7], developed by Ryan Dahl in 2009.

4.1.1 Data glove communication

The AC communicates with the data glove through a developed application. The CaptoGlove Company provides a C++ SDK in order to develop applications for various purposes. For the Thesis Work, the data of the flex sensors from each finger are needed. The communication is achieved using Sockets with IP address "*127.0.0.1*" and port number 20000.

The interaction with the data glove is enabled by the human operator executing the aforementioned application on the HMI, as described in section 4.3.3. The application subscribes to the data glove's controller to initiate the retrieval of the data from the integrated sensors. AC requests for the data and the CaptoGlove application responds with a message containing all the values of the fingers' sensors. Finally, AC recognizes the performed by the human operator gesture and informs the CaptoGlove app that the data are received, and it is waiting for the next data. Figure 15 depicts the sequence diagram with the exchanged messages for establishing the data glove communication and retrieving the necessary data.

---

[5] https://www.w3schools.com/js/
[6] https://nodejs.org/en/
[7] https://v8.dev/

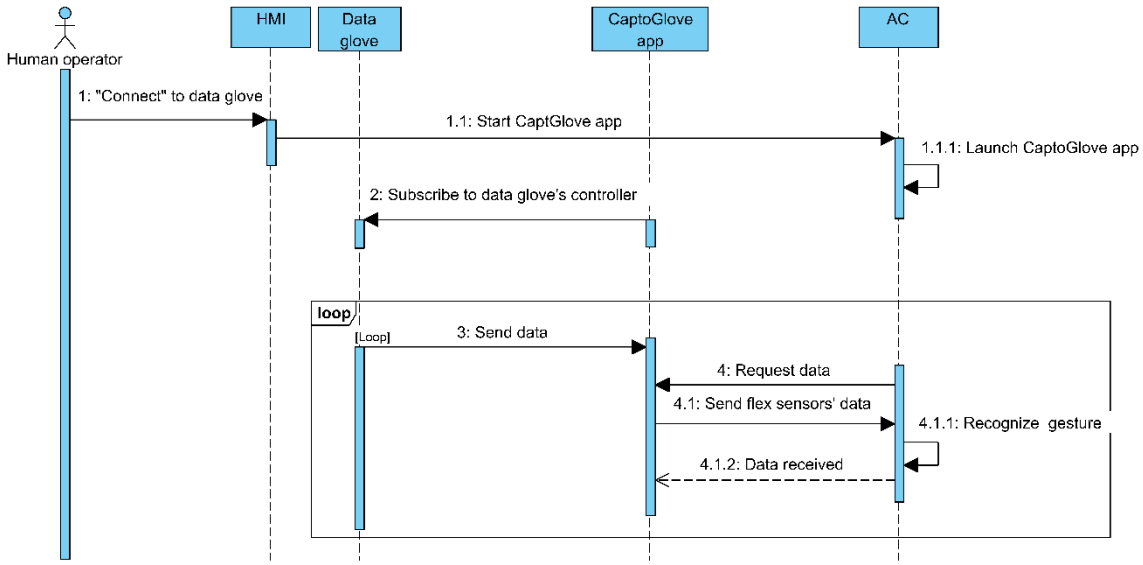

Figure 15. *Data glove communication, sequence diagram*

4.1.2 Hand-tracking device communication

The Leap Motion Company offers various SDKs for developer. As the programming language of the AC is JavaScript the corresponding SDK has been used to retrieve the data related to the position of hand. To establish the communication, AC hosts a client that connects to the Leap Motion server. To launch this server, human operator launches the Orion software from Leap Motion Company. The connection is established using the IP address “127.0.0.1” and port number 6437. Then using the function “*frame*” from the Leap Motion SDK it is possible to retrieve the data.

4.1.3 HMI communication

The communication of HMI and AC is established using Sockets with the server-client model. The data on HMI are refreshed from AC every 50 ms. All the necessary settings to establish the HRI are defined from the human operator through the HMI. Those settings are sent to AC. The generation of the final code is initiated through the HMI. The connection status of the components is sent from AC to the HMI. More information related to the Human-Robot interface are provided in the section 4.3. Figure 16 depicts the messages that are exchanged between AC and HMI.

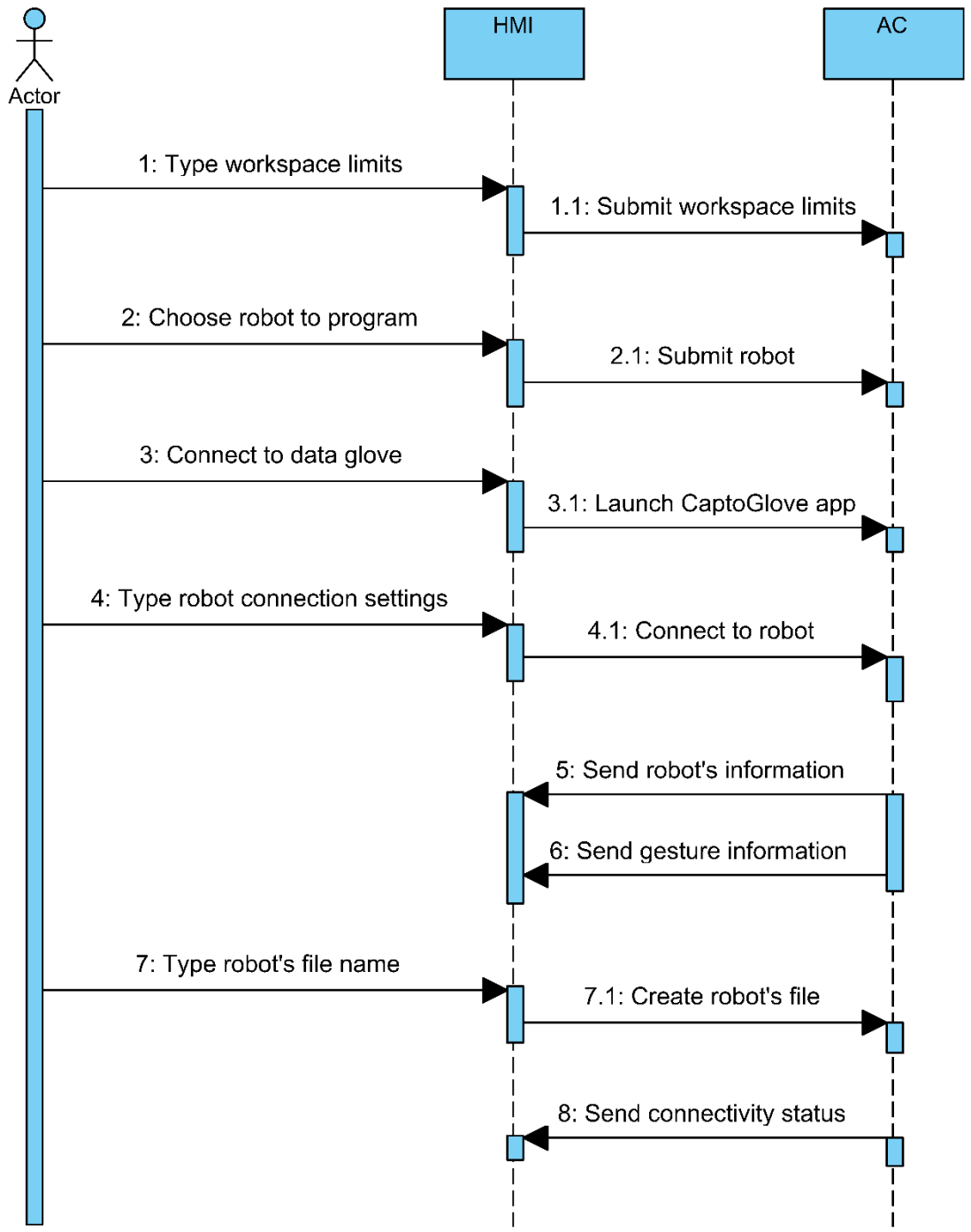

Figure 16. *HMI communication*

4.1.4 Inverse Kinematics calculator communication

The IKC is in control for solving the inverse kinematics for each target received from the AC, and the dispatch of the calculated joint angles to the industrial robot manipulator and

the AC. For the development of IKC, the MATLAB software was used. In addition, the ARTE[8] robotics Toolbox has been utilized.

The communication of AC with IKC and the industrial robots is achieved with Sockets. AC transfers to the IKC all the necessary settings to connect to the industrial robot. Thereafter, the robot's controller requests the joint angles to move the robot. AC having already received the hand position in the Cartesian space, sends a target to be reached to IKC. Then depending the chosen robot (eCobra 600 PRO or IRB120) the joints' angle are calculated. Robot's controller receives the angles and leads the robot. Whereas robot achieves the movement a confirmation is sent to IKC and in turn, the angles are sent to AC for storing and use within the final's robot code. Figure 17 depicts the exchanged messages among AC, IKC and the industrial robots.

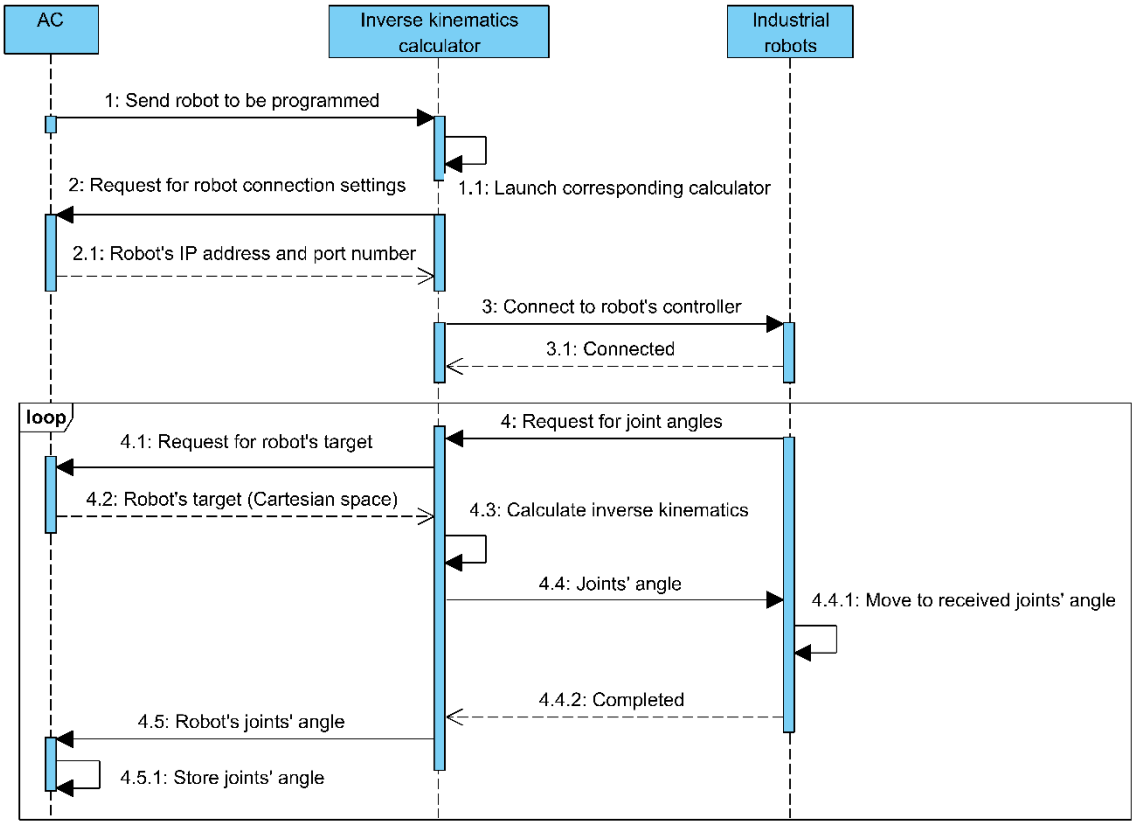

Figure 17. *Inverse kinematics calculator communication*

4.2 Hand-gestures

This section introduces the static hand-gestures that enable the interaction among human and robot. Table 8, presents the hand pose for executing each one of the nine

[8] http://arvc.umh.es/ /index_en.html

gesture. The definition of each static gesture is not unique, and it can be changed through the AC depending on the individual ease and comfort.

Table 8. *Hand gestures [13]*

#	Gestures	How to	When to use	Pose
1	Enable Gestures	Bend thumb and pinky finger	Enable gestures	
2	Disable Gestures	Bend thumb and ring finger and extend the other 3.	Disable gestures.	
3	Move Robot	Fist and thumbs up	Manipulate robot	
4	Hold Position	Extend all fingers	Keep current position	
5	Grasp	Bend thumb and index finger and extend the other 3	Close gripper	
6	Release	Bend thumb and middle finger and extend the other 3	Open gripper	
7	Start Recording Path	Bend thumb, ring and pinky finger and extend the other 2	Save desired targets to the path	
8	Stop Recording Path	Bend thumb, index and middle finger and extend the other 2	Indicate the completion of a path	
9	Run Path	Thumb up, index finger extended and the other 3 bend	Execute recorded path	

Two gestures are devoted for indicating whether the use of gestures is enabled or disabled (gestures 1 and 2). The purpose of these static gestures is to ensure that in case human operator performs by accident a control gesture, there will be no response from the industrial robot manipulator. Similarly, in case human operator removes the other hand from the interaction area of the Leap Motion Controller, the interaction with the robot is instantly aborted. [13]

Moreover, four gestures (gestures 3, 4, 5 and 6) are defined for controlling the robot's action. Human operator must place one hand on top of the Leap Motion Controller and perform Gesture 3 with the other hand, in order to start manipulating the industrial robot. Human operator must keep the hand with the corresponding pose of Gesture 3 to continue manipulating the industrial robot. Gesture 4 halts the manipulation and the robot manipulator maintain its current position until Gesture 3 is re-performed. Gestures 5 and

6 define the action of the robot's end-effector, for grasping or releasing a work-object. [13]

Two gestures are assigned to indicate the initiation and the termination of the robot's programming process. While the Gesture 7 is once performed, any action that human operator denotes to the industrial robot (manipulate robot, grasp and release a work-object), is stored by the AC. Gesture 8 terminates the programming process and one path is completed. Each time, human operator performs this sequence a new robot's path is created.

Finally, Gesture 9 retrieves and executes only the latest stored path. This is gesture cannot be performed while human operator manipulates the robot for recording a new path. In case human operator desired to delete the path and teach it again, it is possible through the HMI (section 4.3.7). [13]

4.3 Human-Machine Interface

As it was mentioned in 3.1, HMI is used to define the necessary settings to establish the HRI. The robot's programming process is strictly relied on the hand-gestures. The development of HMI was based on four programming languages, JavaScript, HTML[9] (Hypertext Markup Language), CSS[10] (Cascading Style Sheets) and W3.SCC[11]. The developed HMI is web-based, and it can run in a web browser. To access the HMI, the AC must be already executed, and the "localhost:8000/UserInterface.html" opens the web-page.

Three different pages were developed. The first page shows the interface that human operator will use through the process, one page with the required instructions for connecting all the components and the last page describes the use and purpose of the already mentioned (4.2) hand-gestures.

The purpose of the Human-Machine Interface is to define the necessary settings to establish the communication of the AC, the IKC and the industrial robot. Moreover, the execution of the data glove's application, which establishes the connection with the AC, is manually activated from the human operator through the HMI. The workspace in which the human operator will manipulate the industrial robot and the name of the final robot's code are declared on the HMI. Finally, human operator can use the HMI for observing

---

[9] https://www.w3schools.com/html/default.asp
[10] https://www.w3schools.com/css/default.asp
[11] https://www.w3schools.com/w3css/default.asp

the connection status with all the connected components and robot related information. Figure 18 depicts the develop HMI.

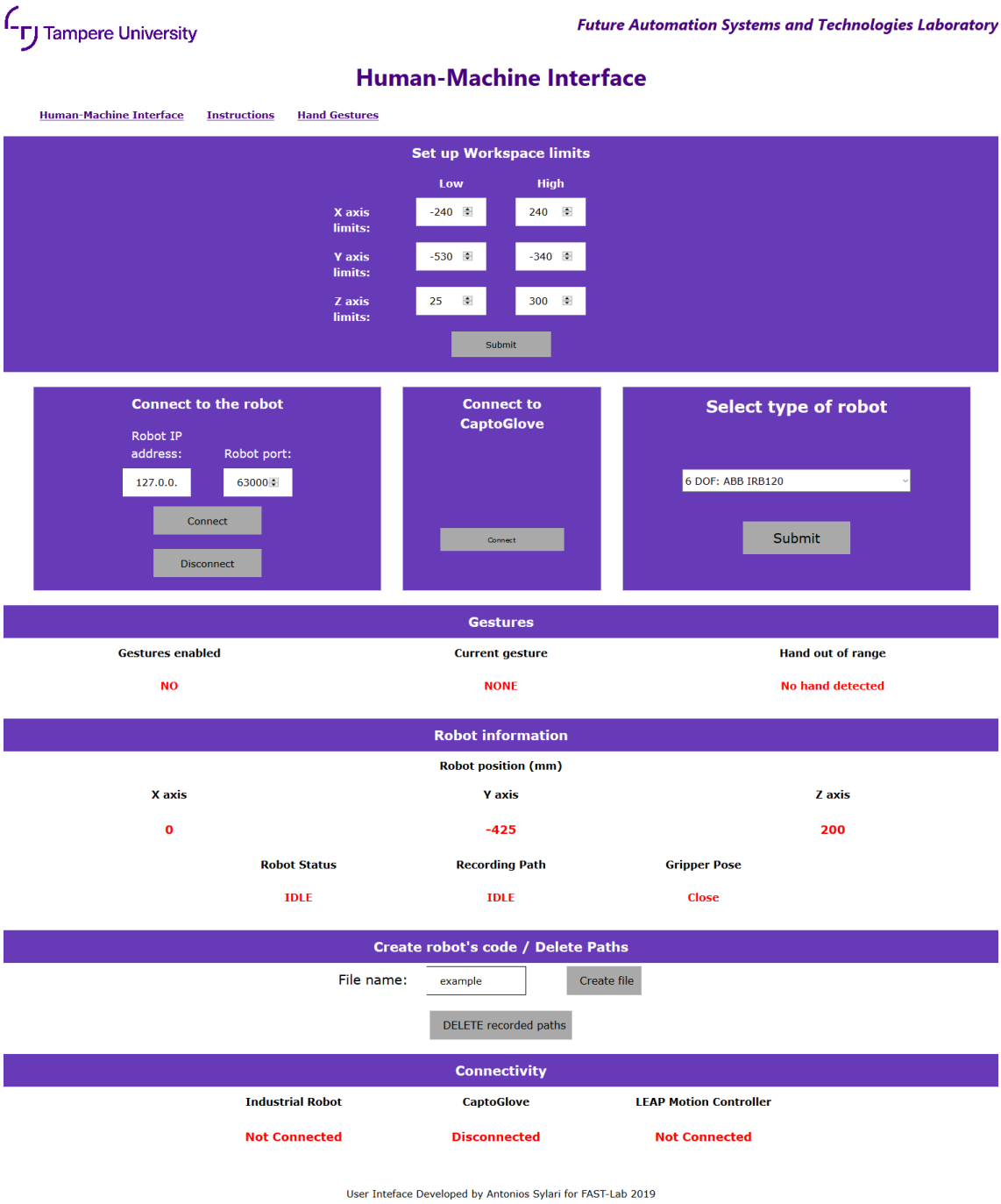

Figure 18. *Human-Machine Interface*

4.3.1 Define workspace limits

The industrial robot manipulators are located within cells in order to ensure safety. The dimensions of a cell are not fixed for every robot and depends on the application. This can affect the reachability of the industrial robot. Within the field "*Setup Workspace limits*", human operator defines the limits that are possible to manipulate the industrial robot.

The connection to the robot manipulator will not be successful until the human operator fills in the "*Low*" and "*High*" limits and press the "*Submit*" button. Figure 19 depicts the field to insert the workspace limits.

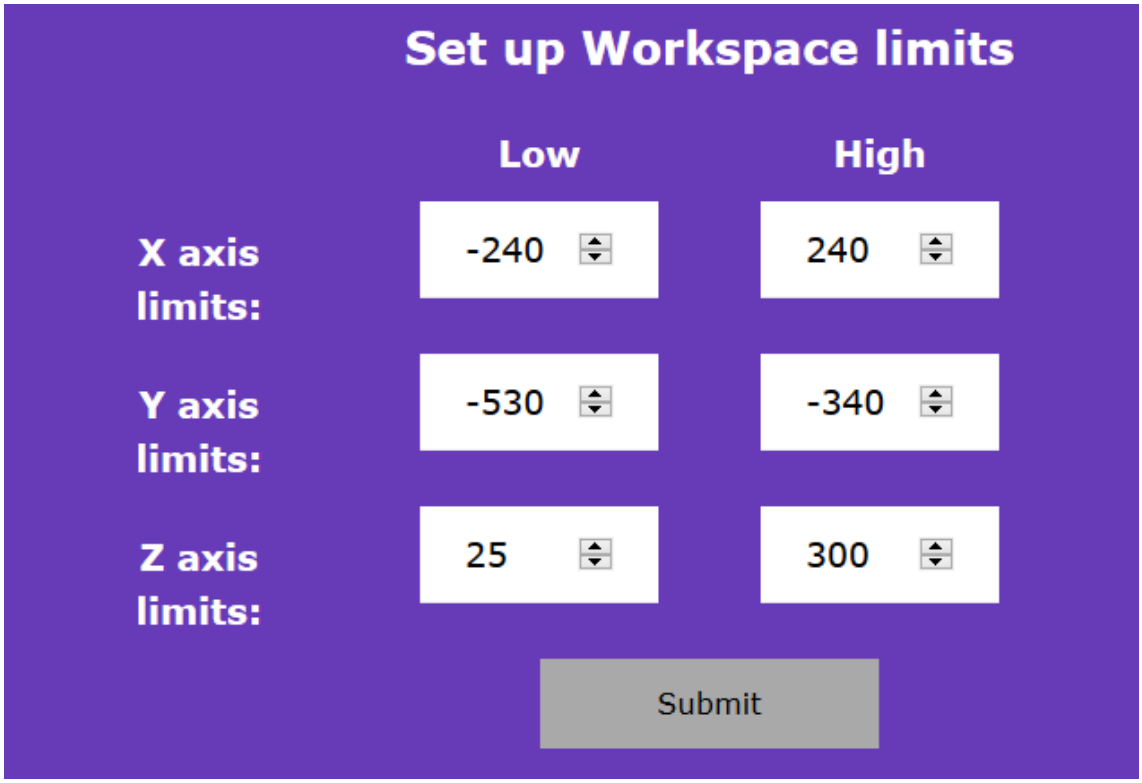

Figure 19. *Setup Workspace limits*

4.3.2 Connect to robot

The field "*Connect to the robot*" requires from human operator to insert the IP address and robot port in order to connect with the robot's controller. Figure 20 depicts the aforementioned field. This field aims to allow human operator to connect to different robot controllers from various manufacturers. For this Thesis Work, human operator can connect to the Adept eCobra 600 PRO robot by typing "*127.0.0.1*", IP address, and 50000 port number. For the ABB IRB120 robot the following settings connect the AC to the IRC5 controller, "*192.168.101.100*" and 32100.

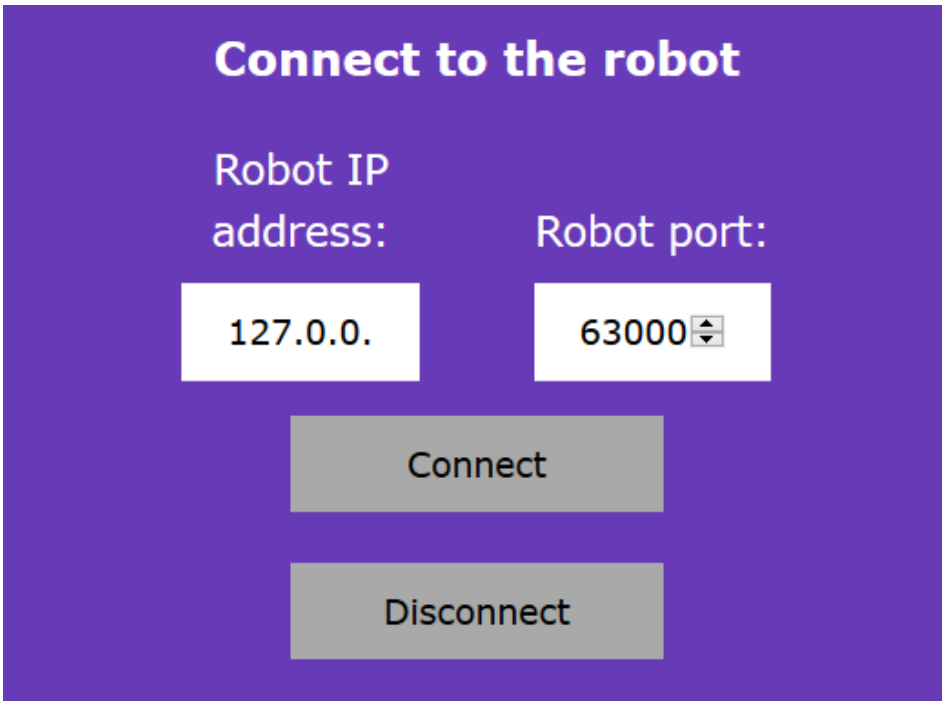

Figure 20. *Connect to the robot*

4.3.3 Connect to data glove

As it was described on subsection 4.1.1, the data glove as a device cannot directly transmit the data to the AC. A C++ application was developed to retrieve all the necessary data. Human operator can execute this application through the field "*Connect to CaptoGlove*" as shown in Figure 21.

Figure 21. *Connect to CaptoGlove*

4.3.4 Select type of robot

On this field, human operator chooses the industrial robot manipulator to be programmed, as shown in Figure 22. The necessity of this field is to declare the robot, so

AC knows beforehand which IKC to choose and which type of robot's code suffix to choose on the generated robot's code.

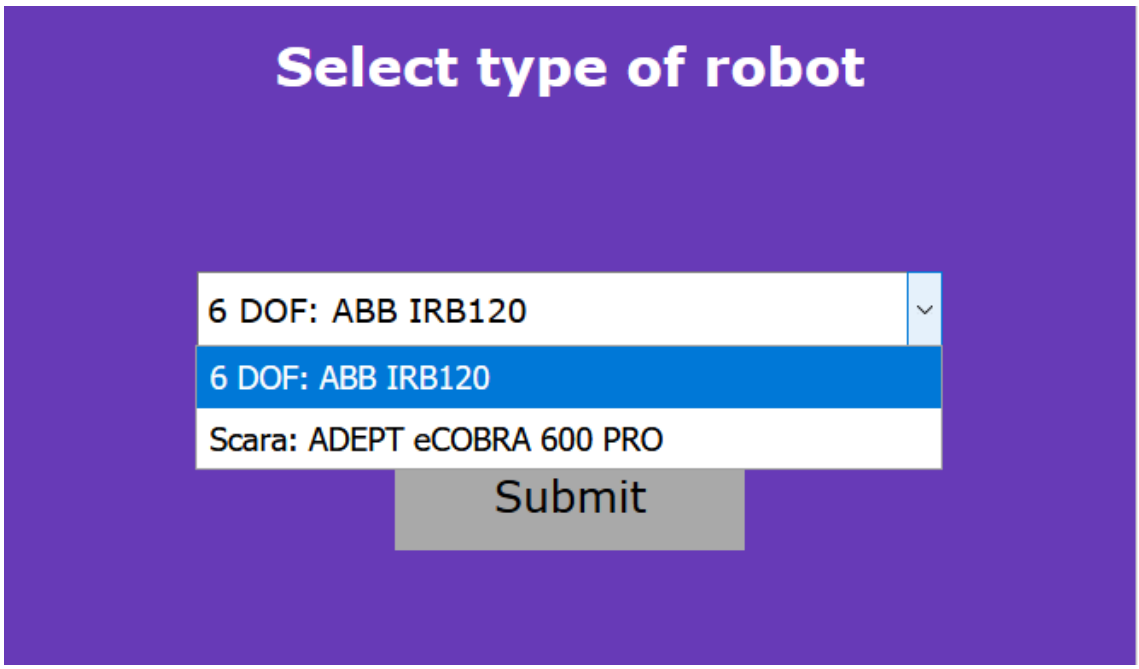

Figure 22. *Select type of robot*

4.3.5 Gestures

The field "*Gestures*" contains three sections, as shown in Figure 23. Section "*Gestures Enabled*", shows whether human operator enabled the recognition for the performed gestures. "*Current gesture*" describes the currently performed gesture. In case of data glove's malfunction, a wrong reading will appear. The last section, "*Hand out of range*", displays that the hand is detected by the Leap Motion Controller. These sections are dedicated only for ensuring the human operator that the AC is recognizing correctly the performed gestures and the hand is correctly detected.

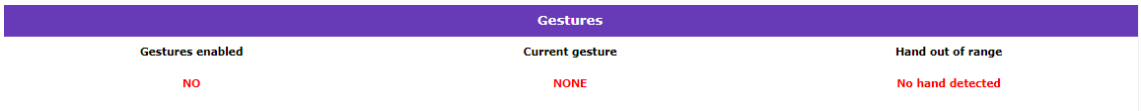

Figure 23. *Gestures field*

4.3.6 Robot information

The "*Robot Information*" field is divided in four sections, as shown in Figure 24. The first field provides information related to the robot's TCP position in the Cartesian Space. The shown values are measured in millimetres (mm).

The second field shows the "*Robot Status*". Three different status have been defined, "*IDLE*", "*BUSY*", and "*IN OPERATION*". "*IDLE*" appears when the AC is disconnected

from the robot's controller. The robot is in "*BUSY*" status while human operator performs "*Move Robot*", "Hold Position", "*Grasp*" and "Release". The "*IN OPERATION*" status defines the process of performing "*Run Path*" and the robot is re-executing the path.

The "*Recording Path*" section depicts whether the human operator is recording a path ("*RECORDING*" status), the completion of a recording ("*COMPLETED*") or "*IDLE*" when human operator is only manipulating the robot.

Finally, the "*Gripper Pose*", changes its value between "*OPEN*" and "*CLOSE*" depending the pose of the end-effector. Even human operator can visually observe the status of the actual end-effector, this section is used by the human operator to detect any malfunction on the end-effector

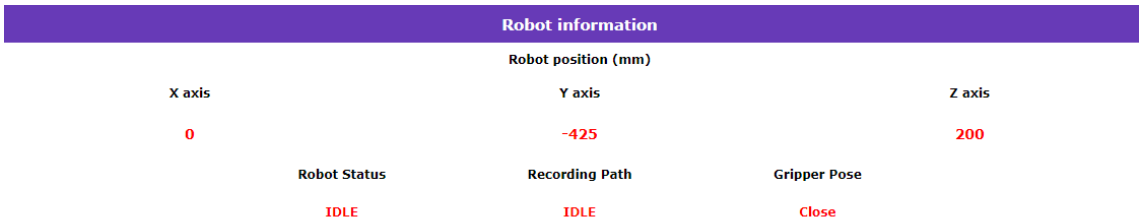

Figure 24. *Robot information*

4.3.7 Create robot's code / Delete Paths

After the completion of the recording all the needed paths for the industrial robot manipulator, human operator can request from the AC to generate the final robot's code. Human operator types only the name for the file to be generated, without adding a suffix with the type of the file to be generated, as human operator has already chosen the robot to be programmed (4.3.4), AC generates the appropriate file. For the Adept robot the suffix is "*.pg*" and for the ABB is "*.mod*". Last, any recorded path that needs to be discarded, can be deleted with the "*DELETE recorded paths*" button. Figure 25 shows the aforementioned field.

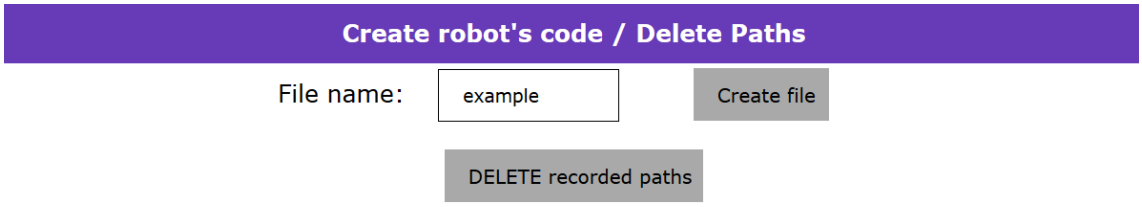

Figure 25. *Create robot's code*

4.3.8 Connectivity

Within the "*Connectivity*" field, human operator can observe the connection status between the AC, the robot manipulators, the data glove and the Leap Motion Controller, as shown in Figure 26. There are three possible statuses, "*Connected*", "*Disconnected*" and "*Not Connected*". "*Disconnected*" status appears when a device has been disconnected from the AC, while "*Not Connected*" appears only before any connection was attempted to be established.

Figure 26. *Connectivity field*

4.4 Frame and trembling correction

Within the implementation, two issues were discovered related to the difference in the axes of the Leap Motion device and the industrial robots. The first issue regards the frames between the device and the robots, and the second issue is related to the sub-millimeter accuracy of the Leap Motion device.

As it can be seen from the Figure 27 and Figure 28 the axes on the robots' base and the Leap Motion device does not correspond to each other. In turn, any motion of the human operator's hand above the Leap Motion device, results in a different motion of the industrial robot. As a result, the data that AC receives from the Leap Motion Controller, must be mapped to correspond with the robots. So, the X-axis of Leap Motion, will be translated to Y-axis for the robot, Leap Motion's Y-axis will correspond to the axis of Z of the robot's frame and last, Z-axis of the Leap Motion, will conform the X-axis of the robots. [13]

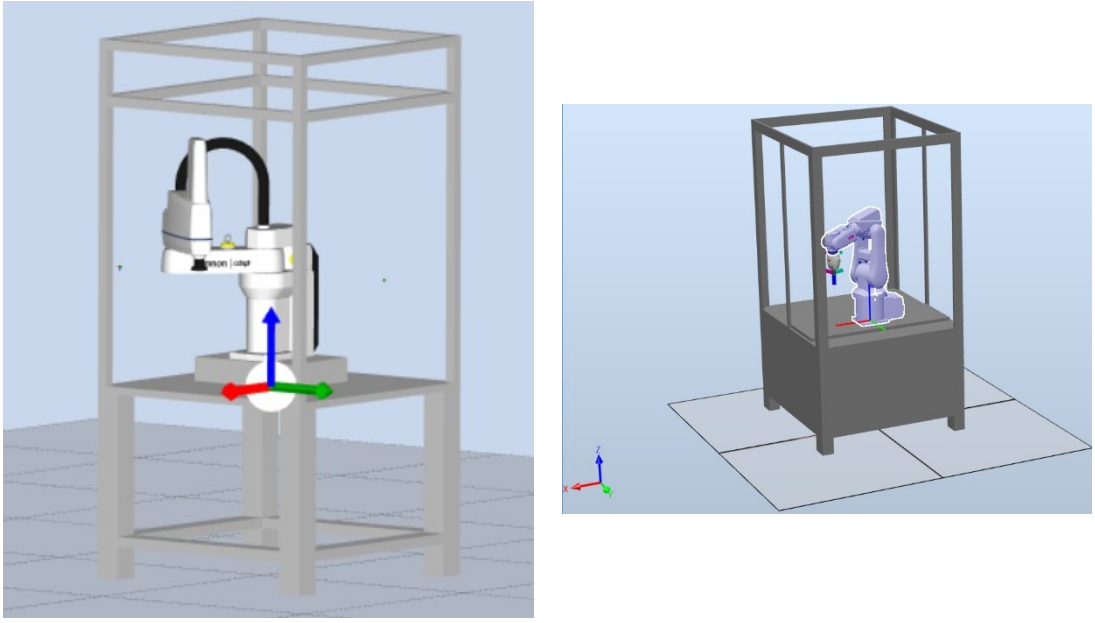

Figure 27. *Base frames of the two industrial robots [13]*

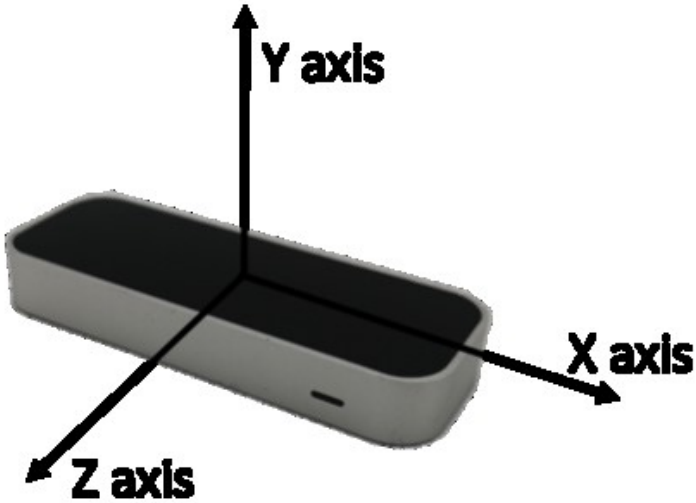

Figure 28. *Leap Motion Controller's base frame*

Leap Motion Controller provides accurate data related to the position of the hand above the device. As a result, any small hand displacement will result to a small displacement of the robot's TCP, the robot appears to vibrate which burdens the motors of the robot. To resolve this issue, AC compares the previously received data with new. For displacements under 5 mm in all axes, the robot remains in the previous position until a higher displacement occurs.

4.5 ”*Home*” and ”*Previous*” position

The purpose of Leap Motion device is to detect the position of the hand, send the information to the AC and then move the robot manipulator. However, for the human operator to start manipulating the robot, there must be a reference point from which the process starts. This reference point is the “*Home*” position. Both robot manipulator and the position of the human’s operator’s hand are set to be in a specific initial position to start the robot manipulation. In this way, it is ensured that human operator has the control of the robot manipulator without any sudden movements. Moreover, in case that human operator could start the operation from any position, the robot’s response could lead to hazardous situations, if any obstacles were within the workspace.

“*Previous*” position defines the latest reached target of the robot manipulator. One of the defined static gestures is “*Hold Position*” gesture, with which human operator holds the current position of the robot. However, in this situation, human’s operator’s hand might be displaced. Therefore, the manipulation of the robot must not continue until human’s operator’s hand is placed again in the previous position. To overcome this issue, AC provides the freedom to human operator to approach the previous position with a deviation of 20 mm in order to continue the robot manipulation. [13]

4.6 Path improvement functions

The functions regarding the improvement of the path taught by the human operator aim to reduce the targets created from uncertain hand movements. During the implementation it was observed that the number of targets stored for generating the robot’s final code was great. Such targets are created during human operator is attempting to perform a straight movement, and thus short-hand displacements and collinear targets were added to the final path. Those functions improve the path after the human operator completes a robot’s path. So, the performing of “*Stop Recording Path*” initiate the execution of those functions

4.6.1 Collinear targets

This function is used to discard targets that lie on a straight line within the robot’s path. The function extracts three targets from the path to calculate the distance among them using. Equation (45) [73].

As an example, from Figure 29, the function calculates the distances of “*AB*”, “*BC*” and “*AC*”. Then, the sum of “*AB*” and “*BC*” targets is compared with the distance of “*AC*”.

$$\Delta = \sqrt{(\Delta_x)^2 + (\Delta_y)^2 + (\Delta_z)^2} \quad (45)$$

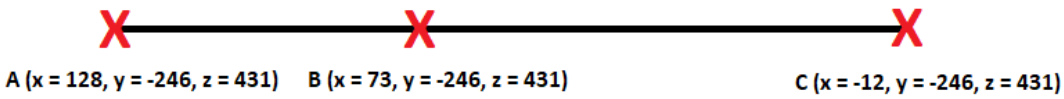

Figure 29. *Example of collinear targets*

Targets are discarded based on the two following conditions:

- In case that the targets 'A', 'B' and 'C' lie on the same line, targets "*A*" and "*C*" are added to the final path and function continues from target "*C*".
- Otherwise, if the length of 'AC' differs from the length of 'AB' and 'BC', then the function adds only target 'A' to the final path and continues searching for collinear targets from "*B*".

4.6.2 Discard targets depending the angle

While human operator attempts to perform a straight movement, such as move along Z-axis, it was observed that oscillations occur during this movement. Thus, the industrial robot manipulator had to go through an additional target to complete a task. As an example, in Figure 30, in an attempt to move from target "*B*" to target "*C*", an additional target is added to the path, target "*A*".

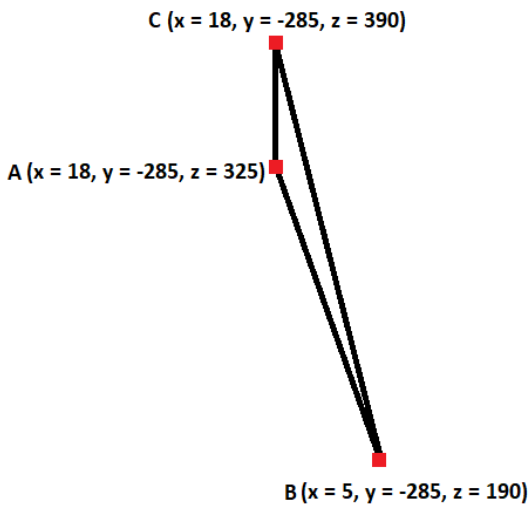

Figure 30. *Example of additional target*

The function operates by monitoring three consecutive targets from the path in order to detect if an angle between 150° and 180° degrees appear while moving from target "*B*" to target "*C*". The aforementioned range was defined after teaching the robot manipulator various tasks. It was observed that within this range it was most probable that an oscillation of human's operator hand will occur.

The calculation of this angle is achieved with the mathematical method of 'vector analysis'. The function is executing the following steps to find the angle:

- The function creates two vectors and each element of the vectors (Equation (46) and Equation (47)) contain the difference in X, Y and Z-axes between "*AB"* and "*AC"* targets.
- Calculate the magnitude for the vectors "*AB*" and "*AC"* using Equation (48) and Equation (49), [73].
- Then to calculate the angle "*θ*", function uses the Equation (50) of the dot product, [74].
- Calculate the dot product of "*AB*" and "*AC"* with the Equation (50).
- Finally, the function calculates the "*θ*" angle with Equation (51), [74].

$$AB = < B_X - A_x, \quad B_y - A_y, \quad B_z - A_z > \tag{46}$$

$$AC = < C_X - A_x, \quad C_y - A_y, \quad C_z - A_z > \tag{47}$$

$$Magnitude: \|\overrightarrow{AB}\| = \sqrt{(AB[1])^2 + (AB[2])^2 + (AB[3])^2} \tag{48}$$

$$Magnitude: \|\overrightarrow{AC}\| = \sqrt{(AC[1])^2 + (AC[2])^2 + (AC[3])^2} \tag{49}$$

$$Dot\ product: (\overrightarrow{AB} \cdot \overrightarrow{AC}) = (AB[1] * AC[1]) + (AB[2] * AC[2]) + (AB[3] * AC[3]) \tag{50}$$

$$Angle: k = \cos^{-1}\left(\frac{\overrightarrow{AB} \cdot \overrightarrow{AC}}{\|\overrightarrow{AB}\| * \|\overrightarrow{AC}\|}\right) * \left(\frac{180}{\pi}\right) \tag{51}$$

If angle in the range 150° and 180° degrees is calculated, the targets "*B*" and "*C*" are added to the final path, while target "*A*" is discarded. Otherwise the function stores also target "*B"* in the final path.

4.6.3 Short-hand displacements

Short-hand displacements by the human operators add redundant targets within the path resulting to additional cycle time for the completion of the robot's task. The function detects displacements of the hand less than 20 mm in all axes on the Cartesian space.

Three targets are examined in order to determine whether a short displacement occurs. Those targets are the current and the two following targets. The result of displacement

is stored in two Boolean flags. A third flag contains the result of the previously calculated short-hand displacement. Figure 31 depicts an example of short-hand displacements.

From Figure 31, the function calculates the distance of "*B"* and "*C"* and "*C*" to "*D*" displacements. On the distance "*BC*" the displacement does not exceed the limit of 20 mm and thus, the flag "*currentSmallChange*" will have the value of "*TRUE*". Meanwhile, the distance of "*C"* to "*D"* the displacement does not exceed the limit and the flag "*nextSmallChange"* becomes "*FALSE*". As for the flag regarding the previous displacement, the displacement is more than 20 mm and so the "*previousSmallChange"* is "*FALSE*". For this example, the function will store only the targets, "*A*", "*B*" and "*D"*.

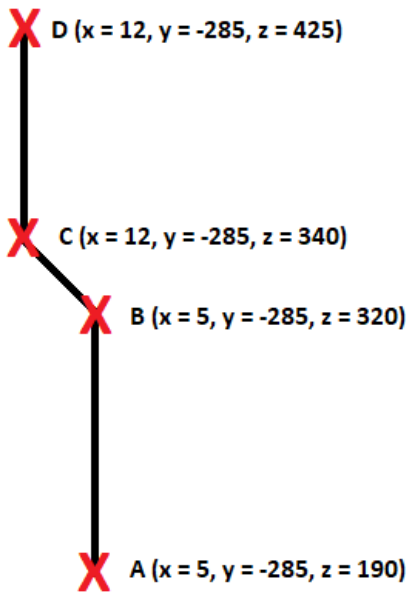

Figure 31. *Example of short-hand displacements*

Moreover, all the decision blocks are described in Table 9. The following table considers as "*previousSmallChange"* the distance "*A"*", "*currentSmallChange"* as the distance "*BC"* and finally the distance "*CD"* for the flag "*nextSmallChange"*.

Table 9. *Short-hand displacements, decision blocks*

Flags:	'previousSmallChange'	'currentSmall-Change'	'nextSmallChange'	Targets to add
	FALSE	FALSE	-	Target 'B' and 'C'
	FALSE	TRUE	FALSE	Target 'D'
	FALSE	TRUE	TRUE	Target 'B' and 'D'
	TRUE	FALSE	-	Target 'B'
	TRUE	TRUE	FALSE	Target 'B'
	TRUE	TRUE	TRUE	Target 'B' and 'D'

4.7 Creating the robot's target

The creation of the robot's target is performed twice within this Thesis Work. Initially, a robot's target in the Cartesian space is created from the received data related to the hand's position as this detected by the Leap Motion. This target is sent to the IKC for solving the inverse kinematics and hence, create the second robot's target on the Joint space.

The messages that contain the robot's target, have the "*string*" format, due to the fact that both robots' controllers accept and interpret this format. In the Cartesian space, the message four attributes, the position to be reached in the X, Y and Z-axes and the number "1" or "0" for denoting the grasping or releasing action of the end-effector. Each attribute is separated by "*;*" symbol. Similarly, the message with the joint values contains seven attributes for the IRB120 industrial robot (six robot joints values and the end-effector's action) and five for programming the eCobra 600 RPO (four robot joints values and the end-effector's action).

4.8 Generate the robot's code

Human operator must be able to program any industrial robot independently of its manufacturer. Thus, the AC contains two different templates for generating the final robot's code using joint variables for both robots. For this Thesis Work, the templates are defined for the Adept and the ABB robots.

Different paths can be recorded by the human operator using "*Start Recording Path*" and "*Sop Recording Path*" gestures. AC stores each path within a JavaScript object, the "*dictionary*". Then depending on the defined robot (4.3.4) AC executes appropriate function to generate the code.

For the Adept robot, a main function is created, through which all the different paths are executed sequentially (i.e. *CALL Path_1()*). Each path is defined within different files and contain the "*MOVE*" (MOVE #PPOINT(joint_1, joint_2, joint_3, joint_4)) for all the taught joint targets, as those calculated through the inverse kinematic calculator.

On the other hand, for the ABB robot, the joint targets' definition and execution can occur within one program file. Initially, the joint targets are defined with the data type "*jointtarget*". Similarly, with the Adept robot, a main procedure executes each path individually. As described in 4.7, the robot's target contains also the action to be taken for the robot's end-effector. For the ABB IRB120 the ABB Smart Gripper is selected as its end-effector. In order to utilize this end-effector an initialization procedure must be defined. The initialization consists of the definition of grasping speed and force, and a calibration procedure.

4.9 Human-Robot interaction area

The Human-Robot interaction area provides a site, in which human can replicate the robot's environment for the ease of the programming process. For example, in a pick and place robot application, human operator handles the corresponding work-objects that robot manipulator has to handle.

Two different structures were developed depending the position of the Leap Motion device. On the first structure, the Leap Motion device is placed on the bottom of the Human-Robot interaction area. The second structure holds the Leap Motion device on the top side, facing downwards. Within the first option, human operator is able to manipulate to industrial robot, but handling corresponding work-objects is not possible. This occurs due to the interaction area of the Leap Motion device (Figure 3) which does not detect the position of the hand is whole spectrum. On the other hand, while Leap Motion device is facing downwards, the manipulation of work-objects by the human operator is possible as those lie on the bottom side which is within the device's interaction area.

The interaction area is a structure of five sides. The structure of the interaction area was retrieved from maker case online tool, [75]. This structure has slot edge joints with slots for attaching all the sides, and slots for M3 nuts and 10 mm bolts to tight the structure. The materials used for the structure are MDF (Medium-density fiberboard) for the bottom and top sides, and acrylic glass for the right, left and back side of the interaction area

and the thickness of the materials is 3 mm. The length of the structure is 800 mm, the width 550 mm and the height 520 mm.

Figure 32 depicts the back side of the interaction area which is used for both structures. Figure 33 shows the bottom side of the interaction area with the Leap Motion Device facing upwards as this was used for the first structure. Figure 34 and Figure 35 illustrate the top and bottom sides of the Human-Robot interaction area in the case that Leap Motion device is placed on top.

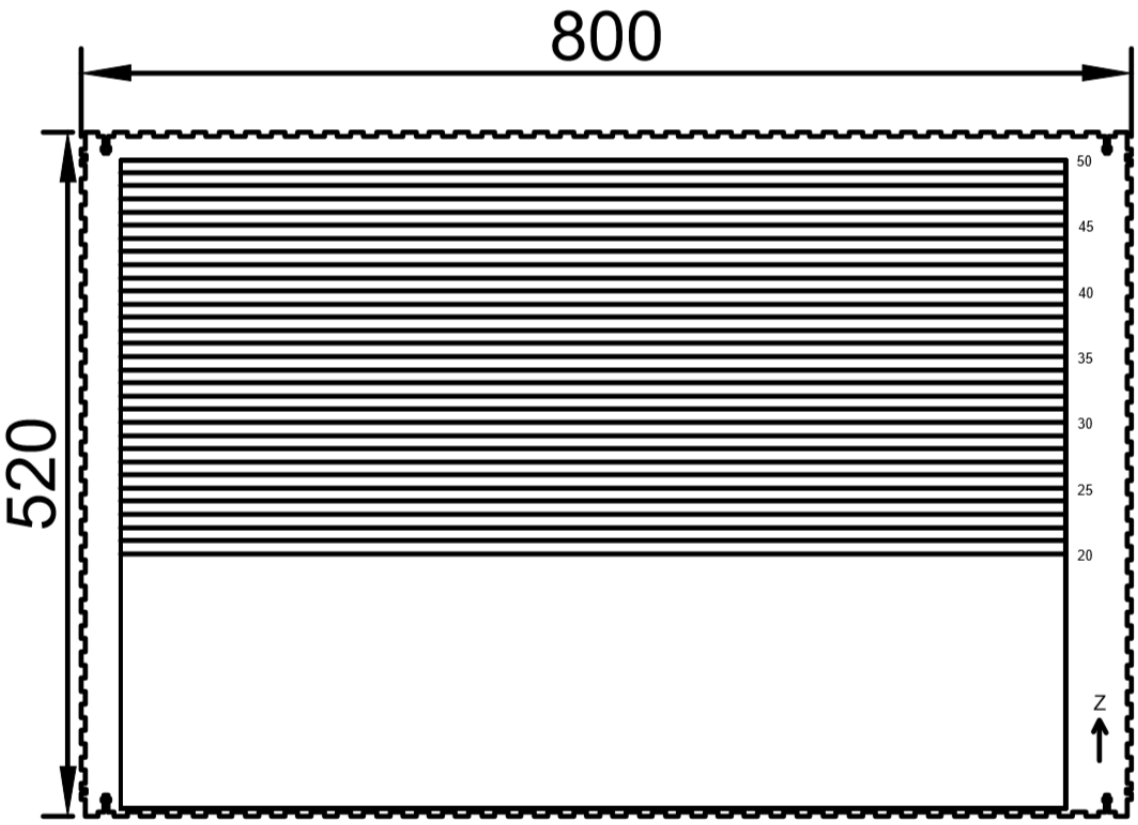

Figure 32. *Back side of Human-Robot interaction area*

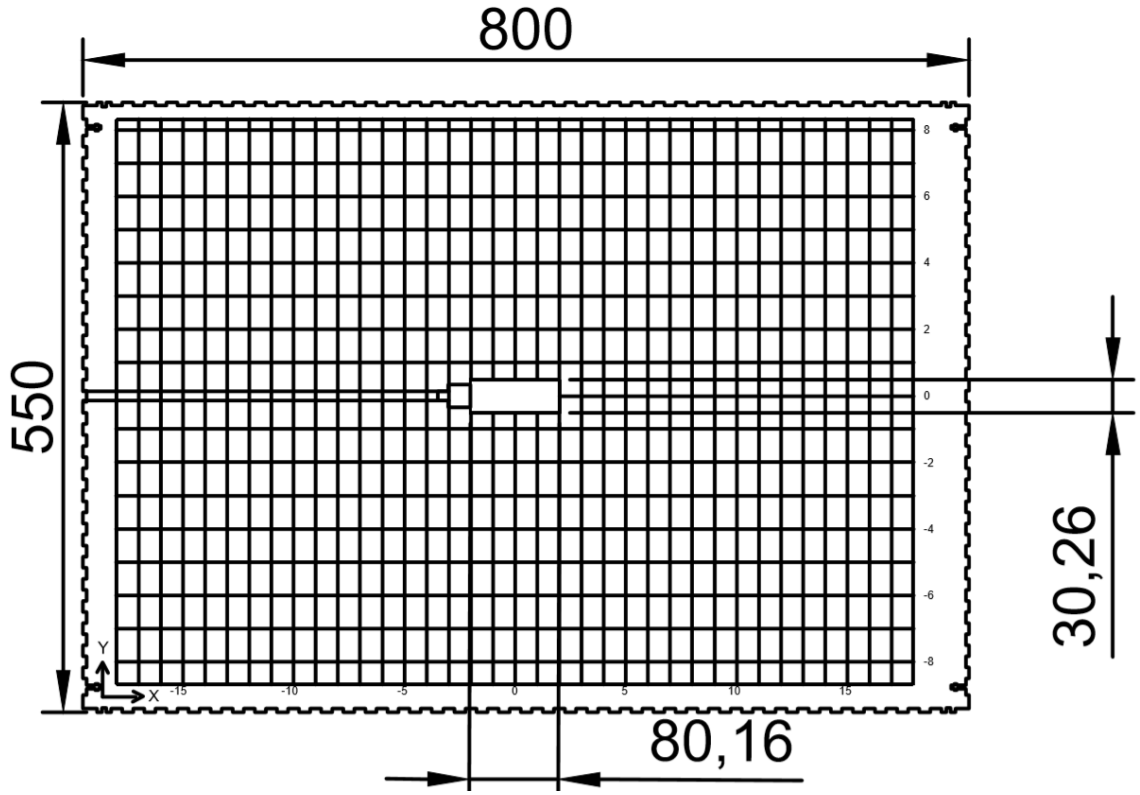

Figure 33. *Bottom side of Human-Robot interaction area, Leap Motion device faces upwards*

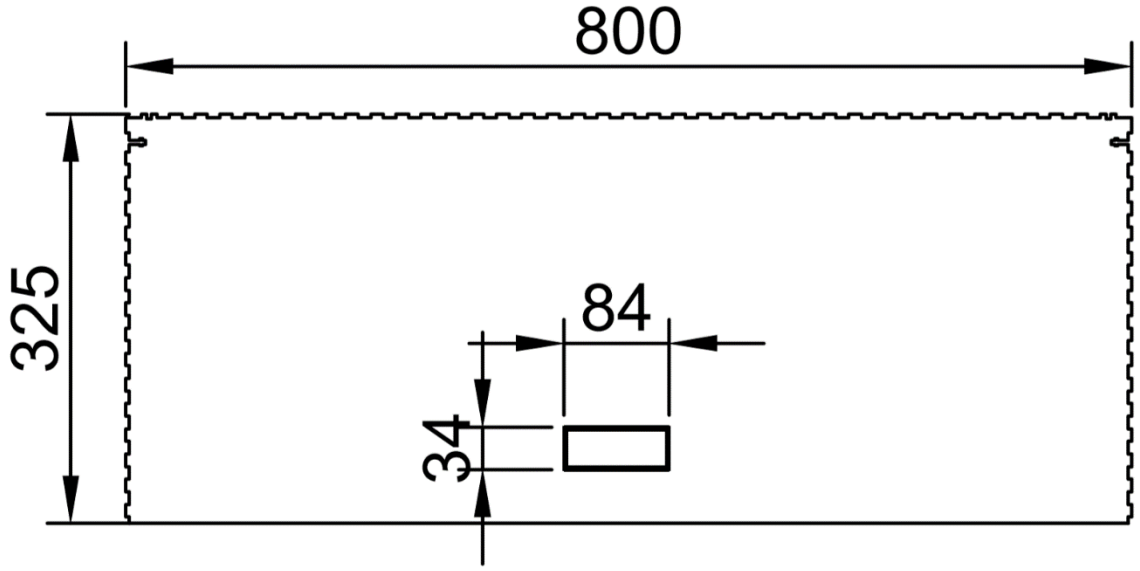

Figure 34. *Top side of Human-Robot interaction area, Leap Motion device faces downwards*

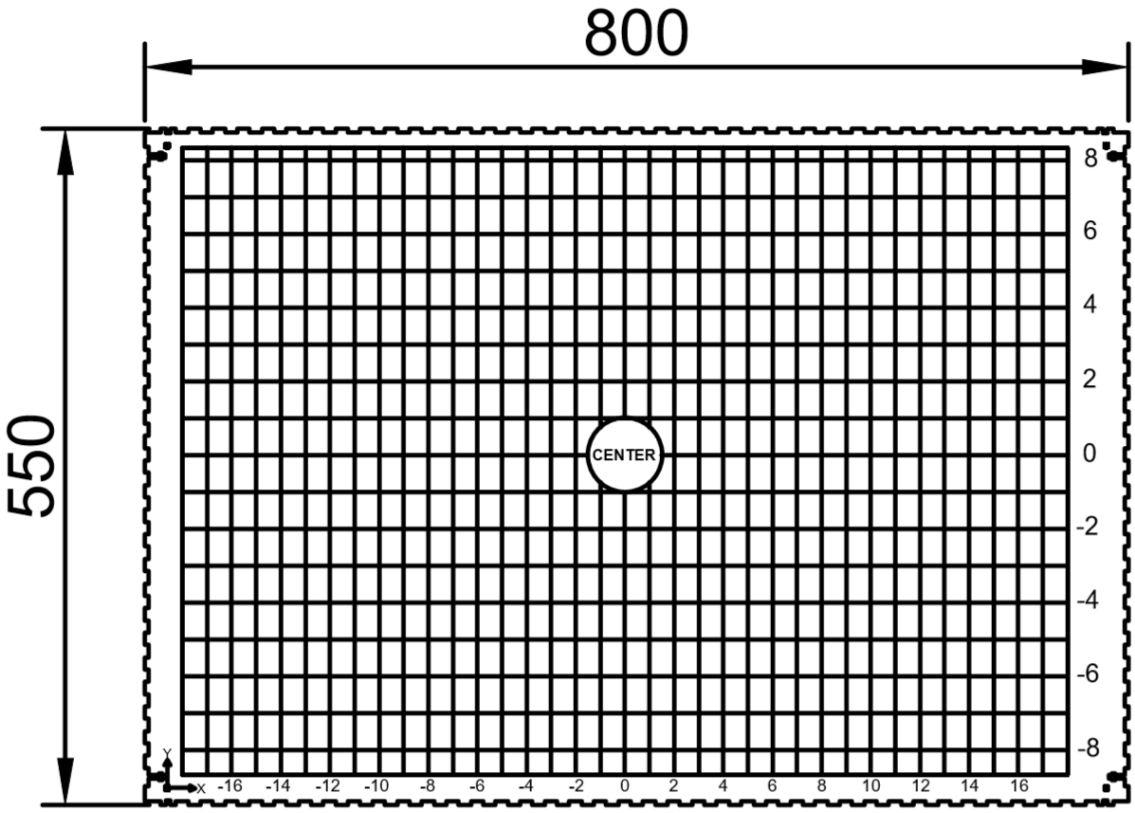

Figure 35. *Bottom side of Human-Robot interaction area, Leap Motion device faces downwards*

4.10 Limitations on AC

The industrial robot manipulators are placed within a cell or they are surrounded by fences to prevent any physical interactions with humans working close to them. Moreover, performing accidental static gestures and hand movements on the interaction area of Leap Motion Controller must be foreseen from the AC to avoid non-desirable actions to the robot manipulator.

Both industrial robot manipulators are arranged inside two different cells and as a result the available workspace is limited. Human operator must define manually through the Human-Machine Interface the lower and upper limits for the workspace that the industrial robot can be manipulated to. These limits regard the X, Y and Z-axes in the Cartesian space.

Exceeding those workspace limits, leads to the recursion of the industrial robot to the predefined initial position. Similarly, the industrial manipulator returns to "*Home*" position if the hand of the human operator is out of the Leap Motion device's interaction area. In addition, the gestures are disabled, and human operator must the programming process from the beginning.

4.11 Task optimization

The task optimization aims to improve the programmed path by the human operator, in terms of time execution. Three approaches were researched to optimize the time of the task execution, path planning, trajectory planning and genetic algorithms for reducing the time cycle.

Path planning consists of various algorithms that attempt to generate the shortest possible path when the initial and final target are given, in conjunction with any obstacles that possibly lie on the industrial robot's workspace. However, the goal of this Thesis is programming the industrial robot manipulator by imitation. Hence, adapting a path planning algorithm adds the necessity for additional procedures, which include the declaration of initial and final target along with the creation and of the industrial robot's work-envelop for generating the robot's optimized path.

The second approach is considered the introduction of the trajectory planning within the overall application. In principle, the output of the trajectory planning is a set of joint angles defined in time along with the position, velocity and acceleration to be achieved by each joint of the robot. While creating a robot's code, variables regarding position, velocity and acceleration, it is not possible to be defined as those, are controlled and defined through the robot's controller. Only the velocity is possible to be defined through the robot's code, but this is related to the velocity of the robot's TCP.

Last approach tested to optimize the task was the adaption of genetic algorithms to generate the near optimum time cycle for a robot's task by evaluating different robot configurations for each target. A genetic algorithm was developed on MATLAB in order to generate the most appropriate set of configurations.

A genetic algorithm generates the desired near optimum result through five different steps:

1. Initial population
2. Fitness function
3. Selection
4. Crossover
5. Termination. [76]

Initial population defines a defined number of random robot's joint configurations for each path. The number of the population in the developed genetic algorithm was configured from 50 to 3000 while performing the validation of the genetic algorithm.

The fitness function calculates the time cycle for each one of the joints' configuration for the path and provides a score. This score is used to discard the paths which time cycle was high. This is completed within the selection phase. The discard is achieved by selecting the percentage of the population. For example, for this Thesis Work the percentage varied from 25% to 60%.

The crossover phase reproduces a new generation from the population that was kept on the selection phase. Crossover, exchanges robot configurations for each target of the path between the population. Finally, the termination phase determines the number of repetitions of the process to generate the near optimum solution.

The genetic algorithm provided the same or similar result in a rate of 90% within the tests. However, applying the result to the industrial robot, provided almost the same time cycle with the one that human operator programmed. The differences were less than 0,5 ms and as a result this approach was either feasible for optimizing the robot's task.

5. CONCLUSION

The aim of this Thesis is the development of an application to intuitively program industrial robot manipulators by hand-gestures. Human operator performs with the one hand static gestures, recognized through a data glove, to control the next action of the robot manipulator. A hand tracking device traces the hand's position in terms of the Cartesian space. This position is converted to a target in the Joint space and the industrial robot follows the position of the human operator's hand.

The application allows human operators to program robots independently of their manufacturer. Two robots were selected to test the application, the ABB IRB120 industrial robot with 6 DOF, and the OMRON Adept eCobra 600 PRO, a SCARA type of industrial robot with 4 DOF. These types of robots (6 DOF and 4 DOF) were selected as they are commonly used within the factory floor for various manufacturing processes.

To validate the operation of the application developed within this Thesis works, various paths were tested within the scope of the industrial process pick and place. Both configurations of the Human-Robot interaction area (LEAP Motion device faces upwards or downwards) were applied. With the first configuration, LEAP Motion device faces upwards, the robot could be manipulated and programme. Although the presence of corresponding work-objects it is not possible as LEAP Motion Controller could not detect the motion of the hand. However, while placing the LEAP Motion device on the top of the workspace, human operator can manipulate the corresponding work-objects. The generated robot's code was added to the robots' controllers and robot manipulators successfully performed the taught paths.

The path improvement functions reduced the additional targets. However, the task optimization was not successfully achieved. Path planning, trajectory planning and genetic algorithms were considered for optimizing the task. The path planning algorithm requires from the human operator additional steps through the programming process and leads to an out of scope programming methodology, as the hand-gestures would not be required. Moreover, the trajectory planning provides results concerning the position, velocity and acceleration for each robot's joint. Those results are not possible to be included within a robot's code, as are internally handled by the individual robot manufacturer. Finally, the developed genetic algorithm was successfully operating, but applying the results on the industrial robot did not provide any significant reduction on the time cycle.

5.1 Future work

The development and implementation of this Thesis work suggest the following as possible future works for further extension and development:

- Expand the possible robots to program.
- Add robot code templates for additional robot manufacturers.
- Adaption of industrial robots with more degrees of freedom.
- Investigation of algorithms to optimize the task, programmed by the human operator.
- Enable the hand-gesture programming of industrial robots through teleoperation.

REFERENCES

[1] IFR, “International Federation of Robotics,” IFR International Federation of Robotics. [Online]. Available: https://ifr.org/P6. [Accessed: 08-Oct-2018].

[2] IFR, “IFR forecast: 1.7 million new robots to transform the world´s factories by 2020,” IFR International Federation of Robotics. [Online]. Available: https://ifr.org/ifr-press-releases/news/ifr-forecast-1.7-million-new-robots-to-transform-the-worlds-factories-by-20. [Accessed: 26-Sep-2018].

[3] A. Poncela and L. Gallardo-Estrella, “Command-based voice teleoperation of a mobile robot via a human-robot interface,” Robotica, vol. 33, no. 01, pp. 1–18, Jan. 2015, doi: 10.1017/S0263574714000010.

[4] A. Tellaeche, J. Kildal, and I. Maurtua, “A flexible system for gesture based human-robot interaction,” Procedia CIRP, vol. 72, pp. 57–62, Jan. 2018, doi: 10.1016/j.procir.2018.03.017.

[5] P. Tsarouchi, A. Athanasatos, S. Makris, X. Chatzigeorgiou, and G. Chryssolouris, “High Level Robot Programming Using Body and Hand Gestures,” Procedia CIRP, vol. 55, pp. 1–5, 2016, doi: 10.1016/j.procir.2016.09.020.

[6] B. Hein, M. Hensel, and H. Worn, “Intuitive and model-based on-line programming of industrial robots: A modular on-line programming environment,” in 2008 IEEE International Conference on Robotics and Automation, 2008, pp. 3952–3957, doi: 10.1109/ROBOT.2008.4543818.

[7] S. Makris, P. Tsarouchi, D. Surdilovic, and J. Krüger, “Intuitive dual arm robot programming for assembly operations,” CIRP Ann., vol. 63, no. 1, pp. 13–16, Jan. 2014, doi: 10.1016/j.cirp.2014.03.017.

[8] M. R. Pedersen and V. Krüger, “Gesture-Based Extraction of Robot Skill Parameters for Intuitive Robot Programming,” J. Intell. Robot. Syst., vol. 80, no. 1, pp. 149–163, Dec. 2015, doi: 10.1007/s10846-015-0219-x.

[9] B. Fang, D. Guo, F. Sun, H. Liu, and Y. Wu, “A robotic hand-arm teleoperation system using human arm/hand with a novel data glove,” in 2015 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2015, pp. 2483–2488, doi: 10.1109/ROBIO.2015.7419712.

[10] G. Du and P. Zhang, “Markerless human–robot interface for dual robot manipulators using Kinect sensor,” Robot. Comput.-Integr. Manuf., vol. 30, no. 2, pp. 150–159, Apr. 2014, doi: 10.1016/j.rcim.2013.09.003.

[11] “IRB 120 - Industrial Robots (Robotics) - Industrial Robots From ABB Robotics.” [Online]. Available: https://new.abb.com/products/robotics/industrial-robots/irb-120. [Accessed: 08-Oct-2019].

[12] “eCobra 600 Lite / Standard / Pro SCARA Robots/Specifications | OMRON Industrial Automation.” [Online]. Available: http://www.ia.omron.com/products/family/3516/specification.html. [Accessed: 08-Oct-2019].

[13] A. Sylari, B. Ramis Ferrer, and J. L. Martinez Lastra, “Hand Gesture-Based On-Line Programming of Industrial Robot Manipulators,” presented at the International Conference on Industrial Informatics.

[14] M. A. Goodrich and A. C. Schultz, “Human-Robot Interaction: A Survey,” Found. Trends® Hum.-Comput. Interact., vol. 1, no. 3, pp. 203–275, 2007, doi: 10.1561/1100000005.

[15] G. Michalos, S. Makris, J. Spiliotopoulos, I. Misios, P. Tsarouchi, and G. Chryssolouris, “ROBO-PARTNER: Seamless Human-Robot Cooperation for Intelligent, Flexible and Safe Operations in the Assembly Factories of the Future,” Procedia CIRP, vol. 23, pp. 71–76, 2014, doi: 10.1016/j.procir.2014.10.079.

[16] “ROBO-PARTNER Project — ROBO-PARTNER Project Portal.” [Online]. Available: http://www.robo-partner.eu/. [Accessed: 08-Oct-2019].

[17] M. Bdiwi, M. Pfeifer, and A. Sterzing, “A new strategy for ensuring human safety during various levels of interaction with industrial robots,” CIRP Ann., vol. 66, no. 1, pp. 453–456, Jan. 2017, doi: 10.1016/j.cirp.2017.04.009.

[18] J. Marescaux et al., “Transatlantic robot-assisted telesurgery,” Nature, vol. 413, no. 6854, pp. 379–380, Sep. 2001, doi: 10.1038/35096636.

[19] K. Nagatani et al., “Emergency response to the nuclear accident at the Fukushima Daiichi Nuclear Power Plants using mobile rescue robots,” J. Field Robot., vol. 30, no. 1, pp. 44–63, Jan. 2013, doi: 10.1002/rob.21439.

[20] R. R. Murphy, “Human–Robot Interaction in Rescue Robotics,” IEEE Trans. Syst. Man Cybern. Part C Appl. Rev., vol. 34, no. 2, pp. 138–153, May 2004, doi: 10.1109/TSMCC.2004.826267.

[21] S.-Y. Yang, S.-M. Jin, and S.-K. Kwon, “Remote control system of industrial field robot,” in 2008 6th IEEE International Conference on Industrial Informatics, 2008, pp. 442–447, doi: 10.1109/INDIN.2008.4618140.

[22] K. Qian, J. Niu, and H. Yang, “Developing a Gesture Based Remote Human-Robot Interaction System Using Kinect,” Int. J. Smart Home, vol. 7, no. 4, p. 6, 2013.

[23] “ABB Teach Pendant.” [Online]. Available: https://new.abb.com/products/el/3HAC028357-001/teach-pendant. [Accessed: 16-Dec-2019].

[24] L. Wang, “Collaborative robot monitoring and control for enhanced sustainability,” Int. J. Adv. Manuf. Technol., vol. 81, no. 9–12, pp. 1433–1445, Dec. 2015, doi: 10.1007/s00170-013-4864-6.

[25] P. Nilas, T. Sueset, and K. Muguruma, “A PDA-based high-level human-robot interaction,” in IEEE Conference on Robotics, Automation and Mechatronics, 2004., 2004, vol. 2, pp. 1158–1163 vol.2, doi: 10.1109/RAMECH.2004.1438084.

[26] T. Fong, C. Thorpe, and C. Baur, “Collaborative Control: A Robot-Centric Model for Vehicle Teleoperation,” p. 8.

[27] J. A. Adams and H. Kaymaz-Keskinpala, “Analysis of perceived workload when using a PDA for mobile robot teleoperation,” in IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA ’04. 2004, 2004, vol. 4, pp. 4128-4133 Vol.4, doi: 10.1109/ROBOT.2004.1308919.

[28] M. Skubic, S. Blisard, A. Carle, and P. Matsakis, “Hand-Drawn Maps for Robot Navigation,” p. 8.

[29] K. Fischer et al., “A Comparison of Types of Robot Control for Programming by Demonstration,” in The Eleventh ACM/IEEE International Conference on Human Robot Interaction, Piscataway, NJ, USA, 2016, pp. 213–220.

[30] W. Song, X. Guo, F. Jiang, S. Yang, G. Jiang, and Y. Shi, “Teleoperation Humanoid Robot Control System Based on Kinect Sensor,” in 2012 4th International Conference on Intelligent Human-Machine Systems and Cybernetics, 2012, vol. 2, pp. 264–267, doi: 10.1109/IHMSC.2012.159.

[31] “Part 1 – Introduction to Microsoft Kinect – Microsoft Pakistan DX Community Blog.” [Online]. Available: https://blogs.msdn.microsoft.com/pakistan/2013/01/26/part-1-introduction-to-microsoft-kinect/. [Accessed: 16-Dec-2019].

[32] O. Mubin, J. Henderson, and C. Bartneck, “You just do not understand me! Speech Recognition in Human Robot Interaction,” in The 23rd IEEE International Symposium on Robot and Human Interactive Communication, 2014, pp. 637–642, doi: 10.1109/ROMAN.2014.6926324.

[33] O. Mubin, C. Bartneck, L. Feijs, H. Hooft van Huysduynen, J. Hu, and J. Muelver, “Improving Speech Recognition with the Robot Interaction Language,” Disruptive Sci. Technol., vol. 1, no. 2, pp. 79–88, Jun. 2012, doi: 10.1089/dst.2012.0010.

[34] Álvaro Garcia Morcillo, “Multi-Modal Interface For Offline Robot Programming,” Tampere University of Technology, 2018.

[35] A. K. Malima, E. Özgür, E. Ozgur, M. Çetin, and M. Cetin, “A fast algorithm for vision-based hand gesture recognition for robot control,” Apr-2006. [Online]. Available: http://dx.doi.org/10.1109/SIU.2006.1659822. [Accessed: 03-Mar-2019].

[36] M. V. den Bergh et al., “Real-time 3D hand gesture interaction with a robot for understanding directions from humans,” in 2011 RO-MAN, 2011, pp. 357–362, doi: 10.1109/ROMAN.2011.6005195.

[37] D. Lu, Y. Yu, and H. Liu, “Gesture recognition using data glove: An extreme learning machine method,” in 2016 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2016, pp. 1349–1354, doi: 10.1109/ROBIO.2016.7866514.

[38] H. S. Park, E. Y. Kim, and H. J. Kim, “Robot Competition Using Gesture Based Interface,” in Innovations in Applied Artificial Intelligence, vol. 3533, M. Ali and F. Esposito, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 131–133.

[39] K. Nickel and R. Stiefelhagen, “Visual recognition of pointing gestures for human–robot interaction,” Image Vis. Comput., vol. 25, no. 12, pp. 1875–1884, Dec. 2007, doi: 10.1016/j.imavis.2005.12.020.

[40] K. Berns, T. Asfour, and R. Dillmann, “ARMAR-an anthropomorphic arm for humanoid service robot,” in Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C), 1999, vol. 1, pp. 702–707 vol.1, doi: 10.1109/ROBOT.1999.770057.

[41] Y. Pititeeraphab, P. Choitkunnan, N. Thongpance, K. Kullathum, and C. Pintavirooj, “Robot-arm control system using LEAP motion controller,” in 2016 International Conference on Biomedical Engineering (BME-HUST), 2016, pp. 109–112, doi: 10.1109/BME-HUST.2016.7782091.

[42] E. Yavşan and A. Uçar, “Gesture imitation and recognition using Kinect sensor and extreme learning machines,” Measurement, vol. 94, pp. 852–861, Dec. 2016, doi: 10.1016/j.measurement.2016.09.026.

[43] T. Grzejszczak, A. Łegowski, and M. Niezabitowski, “Application of hand detection algorithm in robot control,” in 2016 17th International Carpathian Control Conference (ICCC), 2016, pp. 222–225, doi: 10.1109/CarpathianCC.2016.7501098.

[44] K. Khoshelham and S. O. Elberink, “Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications,” Sensors, vol. 12, no. 2, pp. 1437–1454, Feb. 2012, doi: 10.3390/s120201437.

[45] P. Neto, J. N. Pires, and A. P. Moreira, “Accelerometer-based control of an industrial robotic arm,” in RO-MAN 2009 - The 18th IEEE International Symposium on Robot and Human Interactive Communication, 2009, pp. 1192–1197, doi: 10.1109/ROMAN.2009.5326285.

[46] A. Kannan, A. Ramesh, L. Srinivasan, and V. Vijayaraghavan, “Low-cost static gesture recognition system using MEMS accelerometers,” in 2017 Global Internet of Things Summit (GIoTS), 2017, pp. 1–6, doi: 10.1109/GIOTS.2017.8016217.

[47] O. Sidek and M. A. Hadi, “Wireless gesture recognition system using MEMS accelerometer,” in 2014 International Symposium on Technology Management and Emerging Technologies, 2014, pp. 444–447, doi: 10.1109/ISTMET.2014.6936550.

[48] B. Fang, F. Sun, H. Liu, and D. Guo, “A novel data glove using inertial and magnetic sensors for motion capture and robotic arm-hand teleoperation,” Ind. Robot Int. J., vol. 44, no. 2, pp. 155–165, Mar. 2017, doi: 10.1108/IR-07-2016-0179.

[49] P. Kumar, S. S. Rautaray, and A. Agrawal, “Hand data glove: A new generation real-time mouse for Human-Computer Interaction,” in 2012 1st International Conference on Recent Advances in Information Technology (RAIT), 2012, pp. 750–755, doi: 10.1109/RAIT.2012.6194548.

[50] P. Lokhande, “Data Gloves for Sign Language Recognition System,” Int. J. Comput. Appl., p. 4.

[51] C. Kohrt, R. Stamp, A. G. Pipe, J. Kiely, and G. Schiedermeier, “An online robot trajectory planning and programming support system for industrial use,” Robot. Comput.-Integr. Manuf., vol. 29, no. 1, pp. 71–79, Feb. 2013, doi: 10.1016/j.rcim.2012.07.010.

[52] C. Schou, J. S. Damgaard, S. Bøgh, and O. Madsen, “Human-robot interface for instructing industrial tasks using kinesthetic teaching,” in IEEE ISR 2013, 2013, pp. 1–6, doi: 10.1109/ISR.2013.6695599.

[53] S. Zolkiewski and D. Pioskowik, “Robot Control and Online Programming by Human Gestures Using a Kinect Motion Sensor,” in New Perspectives in Information Systems and Technologies, Volume 1, vol. 275, Á. Rocha, A. M. Correia, F. . B. Tan, and K. . A. Stroetmann, Eds. Cham: Springer International Publishing, 2014, pp. 593–604.

[54] M. R. Pedersen, D. L. Herzog, and V. Krüger, “Intuitive skill-level programming of industrial handling tasks on a mobile manipulator,” in 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2014, pp. 4523–4530, doi: 10.1109/IROS.2014.6943203.

[55] F. Weichert, D. Bachmann, B. Rudak, and D. Fisseler, “Analysis of the Accuracy and Robustness of the Leap Motion Controller,” Sensors, vol. 13, no. 5, pp. 6380–6393, May 2013, doi: 10.3390/s130506380.

[56] “Download CaptoGlove Suite, SDKs, firmware updates and more!” [Online]. Available: https://www.captoglove.com/downloads-sdk/. [Accessed: 17-Dec-2019].

[57] “CaptoGlove - Virtual reality wearable gaming motion controller,” CaptoGlove®. [Online]. Available: https://www.captoglove.com/. [Accessed: 09-Oct-2019].

[58] “leap-motion-interaction-area - Leap Motion Blog.” [Online]. Available: http://blog.leapmotion.com/how-to-build-your-own-leap-motion-art-installation/leap-motion-interaction-area/. [Accessed: 16-Dec-2019].

[59] “Leap Motion.” [Online]. Available: https://www.leapmotion.com/. [Accessed: 09-Oct-2019].

[60] “eCobra | Omron, Europe.” [Online]. Available: https://industrial.omron.eu/en/products/ecobra. [Accessed: 16-Dec-2019].

[61] “IRC5 Compact | ABB.” [Online]. Available: https://new.abb.com/ch/fr/magazine-clientele/produits/irc5-compact. [Accessed: 16-Dec-2019].

[62] “IRB 120, data sheet, PDF.” [Online]. Available: https://search-ext.abb.com/library/Download.aspx?DocumentID=ROBO149EN_D&LanguageCode=en&DocumentPartId=2&Action=Launch. [Accessed: 16-Dec-2019].

[63] Denavit Jacques and Hartenberg Richard Scheunemann, “A kinematic notation for lower-pair mechanisms based on matrices,” Journal of applied mechanics 77.2, pp. 215–221, 1955.

[64] Hartenberg Richard Scheunemann and Danavit Jacques, Kinematic Synthesis of Linkages. New York: McGraw-Hill, 1964.

[65] Richard P. Paul, Robot manipulators: mathematics, programming, and control : the computer control of robot manipulators. Cambridge, MA: MIT Press, 1981.

[66] Spong, Mark W and Vidyasagar, M, Robot Dynamics and Control. New York: John Wiley & Sons, 1989.

[67] M. W. Spong, S. Hutchinson, and M. Vidyasagar, “Robot Dynamics and Control,” p. 303, Jan. 2004.

[68] P. I. Corke, Robotics TOOLBOX for MATLAB, 7th ed. 2002.

[69] J. Fang and W. Li, “Four degrees of freedom SCARA robot kinematics modeling and simulation analysis,” vol. 2, p. 8, 2013.

[70] David Barrio Vicente, “Modeling and Balancing of Spherical Pendulum using a Parallel Kinematic Manipulator,” Lund University, Department of Automatic Control, 2007.

[71] Tyler J. Carter, “The Modeling of A Six Degree-Of-Freedom Industrial Robot For The Purpose Of Efficient Path Planning,” The Pennsylvania State University, The Harold and Inge Marcus Department of Industrial & Manufacturing Engineering, May2009.

[72] Craig John J., Introduction to robotics: mechanics and control, 4th ed. .

[73] Rorres Chris and Howard Anton, Elementary linear algebra: applications version. Wiley, 1994.

[74] MR. Spiegel, S. Lipschutz, and D. Spellman, Vector Analysis (Schaum's Outlines), 2nd ed. McGraw-Hill Education, 2009.

[75] "MakerCase - Easy Laser Cut Case Design." [Online]. Available: http://old.makercase.com/. [Accessed: 05-Jan-2020].

[76] L. E. Gonzalez Moctezuma, L. Angelica Nieto, and J. L. M. Lastra, "A genetic algorithm for optimizing vector-based paths of industrial manipulators," in 2013 11th IEEE International Conference on Industrial Informatics (INDIN), 2013, pp. 286–292, doi: 10.1109/INDIN.2013.6622897.