POLITECNICO
MILANO 1863

SCHOOL OF INDUSTRIAL AND INFORMATION ENGINEERING
Master of Science in Automation and Control Engineering

Master Degree Thesis

Teleoperation of a Dual-Arm Manipulator using Hands' Tracking

Advisor:
Prof. Andrea Maria Zanchettin

Co-Advisor:
Prof. Yasuhisa Hirata
Tohoku University, Japan

Candidate:
Andrea Massimo De Angelis

Academic Year 2017–2018

Alla mia famiglia

Contents

List of Figures

List of Tables

1 Introduction

1.1 Abstract in Italiano

Il controllo in tempo reale di un sistema robotico consente di eseguire una vasta gamma di operazioni tramite la definizione di una sola configurazione di sistema.
Nel caso specifico di un manipolatore, è possibile ricorrere all'utilizzo di diverse tecnologie per ottenere l'input in real-time. Tipicamente vengono utilizzate differenti tipologie di joystick, che consentono agli operatori umani di comunicare la posizione e l'orientamento desiderati per l'end-effector. Tali joysticks, però, presentano alcuni significativi inconvenienti in termini di versatilità e curva di apprendimento. Nel seguito verrà quindi presentata un'alternativa all'uso dei joysticks, consistente nello sfruttamento di un dispositivo ottico di nuova generazione chiamato Leap Motion.
Il Leap Motion è un sensore che si presenta sotto forma di periferica USB e che è stato progettato per essere posizionato su di un desktop fisico, sopra al quale si muovono le mani di un operatore umano. Esso utilizza due telecamere e tre LED a infrarossi per osservare, fino a una distanza di 1 metro, un'area ampia fino a 150°. I dati provenienti dalle telecamere e dai LED sono elaborati in modo tale da sintetizzare la posizione e l'orientamento 3D delle mani che si muovono sopra di essi. Con questo tipo di approccio è possibile raggiungere un livello di precisione più elevato rispetto a quanto ottenibile con altre tecnologie che si occupano di tracciare i movimenti delle mani.
L'idea è dunque quella di controllare in tempo reale i movimenti di un manipolatore, inserendo in input al sistema i dati catturati dal Leap Motion. In questo modo l'end-effector del braccio robotico eseguirà una copia speculare della traiettoria della mano umana che si muove sopra al sensore. Considerando che il Leap Motion è in grado di acquisire i dati di entrambe le mani dell'uomo, distinguendo tra destra e sinistra, sarà possibile controllare contemporaneamente entrambi gli arti di un manipolatore a doppio braccio. Il sistema proposto è rappresentato in Fig.1.1, dove è stata schematizzata l'interazione tra uomo e robot.
Il presente lavoro di tesi è quindi dedicato alla presentazione delle principali problematiche e delle possibili soluzioni relative alla realizzazione di tale tipo di struttura di controllo. In particolare, il controllo in tempo reale tramite Leap Motion è stato testato su di una coppia di bracci robotici a 6 gradi di libertà. Gli esperimenti sono stati condotti sia in un ambiente di simulazione che su di una macchina reale. Il comportamento del robot è stato infatti testato prima mediante un ambiente di simulazione personalizzato e poi su un manipolatore dual-arm Epson C3 - A600 SR, commercializzato da Seiko. Inoltre, al fine di interagire con il sistema robotico, è stata sviluppata un'interfaccia software che sfrutta il sistema operativo INTIME, che è il sistema operativo utilizzato anche dal manipolatore stesso.

1.2 Abstract in English

The control of robotic arms is typically performed according to one of two operating modes. A first option consists in programming a specific task, performed by the manipulator according to the relative computer code. In this case the robot will continuously carry out the same job without any kind of variation. Otherwise it is possible to control the robotic arm with a real-time input. In such case it will be possible to perform a wide range of tasks by defining a single operational configuration.

Such real-time control can be realised by many means, the most popular of which is by using specialised joysticks that allow human operators to communicate the desired position and orientation to the manipulator. Anyway, the use of joysticks, despite the many notable improvements during the years, still presents significant drawbacks in terms of versatility and learning curve. In the following it will be then presented an alternative to the use of joysticks, consisting in the exploitation of an optical device of new generation called Leap Motion.

The Leap Motion is a sensor which comes in the form of a USB peripheral. It is designed to be placed on a physical desktop and it uses two cameras and three infrared LEDs in order to observe, up to a distance of 1 meter, an area that is 150° wide. The Leap Motion manages to elaborate the data coming from the cameras and the LEDs in such a way to synthesize 3D position and orientation of the user's hands. With this kind of approach it is possible to achieve a significantly higher level of accuracy with respect to other technologies when it comes to tracking hand's movements.

The idea is then to perform a real-time control of the manipulator's motion by inputting the control system with the data caught by the Leap Motion. In this way the end-effector of the robotic arm will perform a specular copy of the motion of the hands of the human operator that is interfacing with the robot. Considering that the Leap Motion is capable to acquire data from both human's hands, distinguishing among left and right, it will be possible to control both arms of a dual-arm manipulator simultaneously.

The proposed system is represented in Fig.1.1, where the interaction between human and robot has been schematized.

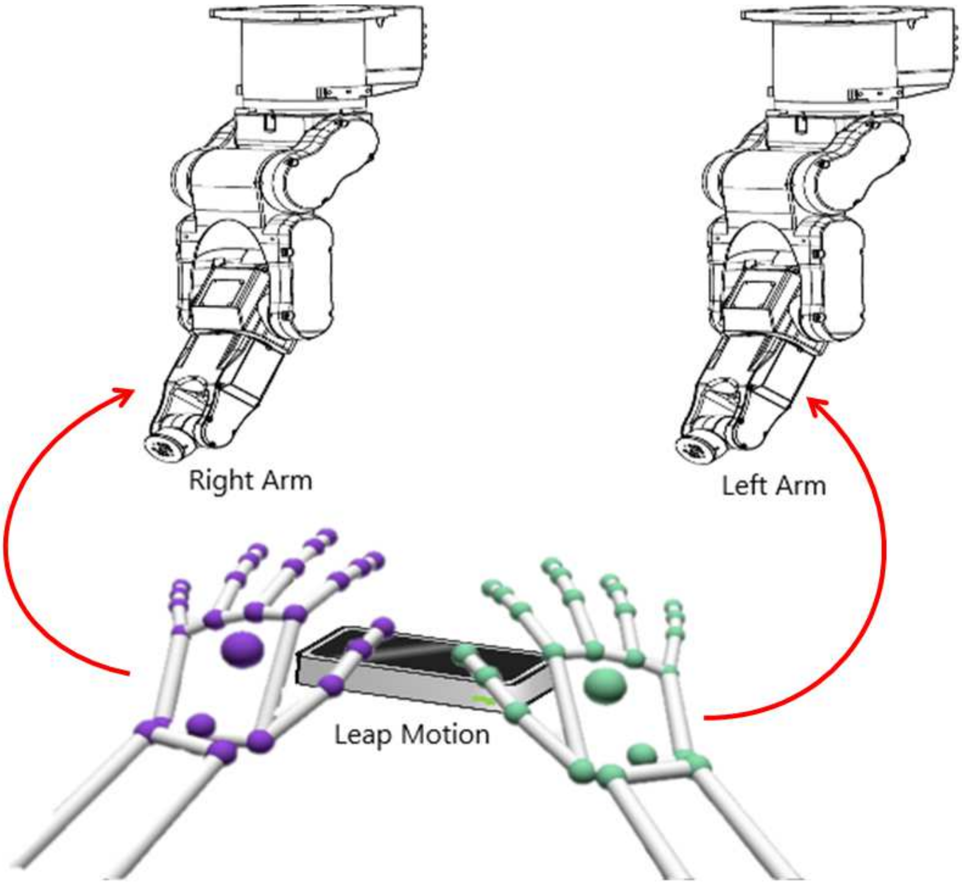


Figure 1.1: Schematization of the human-robot interaction in the proposed system of real-time control of the dual-arm manipulator.

The present thesis work is then dedicated at presenting the main problematics that arise when realizing such kind of control framework and at proposing effective solutions to them. In particular, the real-time control by Leap Motion has been tested with reference to a pair of 6 degrees of freedom manipulators. The experiments have been conducted both in a simulation environment and on the real machine. The robot's behavior has in fact been tested in a customised simulation software first and on a Epson C3 – A600 SR dual-arm manipulator, commercialised by Seiko, secondly. In addition, in order to interact with the robotic system, a software interface has been developed, which exploits the INTime operating system that is the operating system used also by the manipulator itself.

1.3 Overview of the Thesis Work

In the following, the proposed model of human-robot interaction will be explained in all of its details.
At first the general idea will be explained by its applicative point of view, by introducing the main objectives that are desired when implementing such system and by presenting the many possible applicative scenarios. The previous work related to the topic and published through academic papers, will be also introduced in chapter 2.
Subsequently, a more detailed discussion will be dedicated to the methodological aspects underlying the research project. The main matters related with the implementation of a Leap Motion based real-time control will be analysed and discussed with a focus on the adopted solutions in chapter 3.
After defining the context of interest from an high level point of view, the focus will be on the implementative aspects. In chapter 4, the experimental setup will be described, by distinguishing among the hardware and the software devices that have been utilised.
At the end of the theoretical discussion, the experimental results will be presented in chapter 5. The motion outcomes will be analysed taking in account different aspects, such as the input-output temporal delay and the motion accuracy.
In conclusion, in chapter 6, the obtained results will be summarised and the possible fields of interest for a future work will be introduced.

2 The Application

2.1 State of Art

In the context of real-time control of robotic arms many studies have been conducted with the aim of comparing the different possible technical solutions.
One of the most popular means of real-time control consists in the use of joysticks that are meant to be moved by human operators in order to input motion data to the manipulator. Zhang et al. [1] developed a scheme in which the position of the joysticks is evaluated so as to proportionally determine the velocity of the end-effector of a robotic arm. In [2] instead, a low-cost robotic system is controlled by the inputs of a force feedback joystick in order to either assist or resist the movements of a physically impaired patients during therapy.
Typically, it will be necessary to use two joysticks when trying to control a robotic arm in 6 degrees of freedom. Such solution anyway, despite of allowing the control of both position and orientation, can result in being little intuitive and in requiring a long time to train the human operator. A possible configuration of the joysticks' commands is represented in Fig.2.1, where the left controller is responsible for the robot's position and the right one determines its orientation.

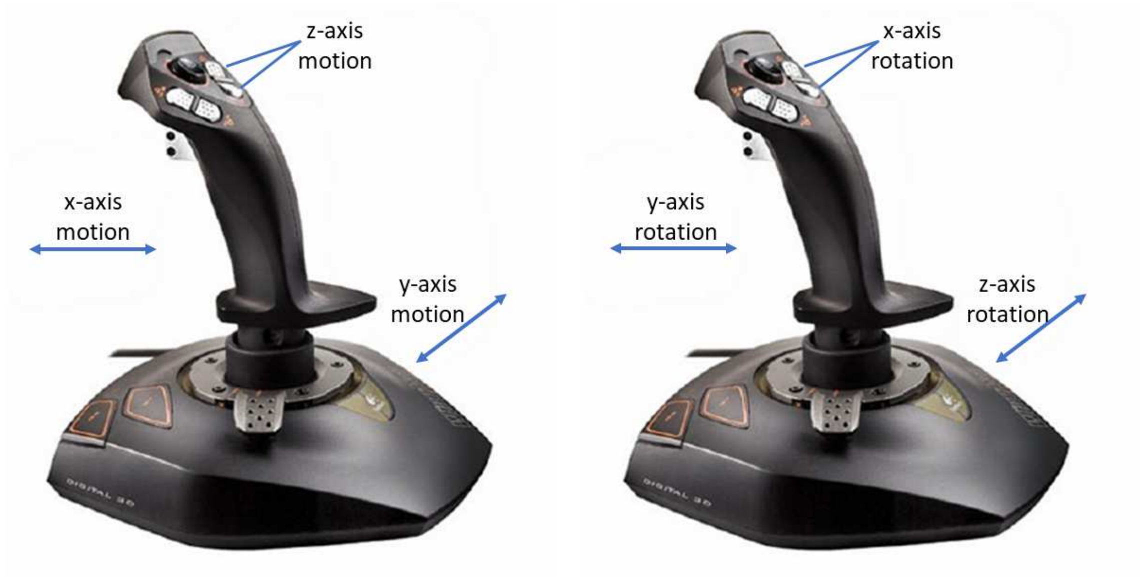


Figure 2.1: Possible configuration of the commands set-up of a couple of joysticks controlling the motion of a 6 DOF manipulator

A natural solution that comes in mind when thinking about a more intuitive way of controlling a robotic arm, consists in using hand's gestures, which can be associated with specific movements of the manipulator. There is an elevated number of research projects in which authors adopted hands' gestures so as to manipulate robots remotely, such as [3], [4],[5] and [6]. In all of these projects, beside from several specific drawbacks, it was not possible to define many different types of commands that would move the robot. In addition, during the real-time control of the manipulator, the human operator would have to focus on the specific gestures corresponding to the single motions forward or backward or up and so on. It would be better for a human operator to focus on the manipulation task instead of thinking about which gesture should be used to control the robot to move it in a certain position. Moreover, it can be easily understood how the control through gestures can result in being detrimental in terms of accuracy. In fact, while it would be easy to distinguish among gestures specifying different types of motion, it would be highly complicated to distinguish among gestures relative to the same direction of motion but with different values of desired position, orientation or velocity.

The most convenient and efficient solution would be copying the hands' motion instead of codifying it into gestures corresponding to commands. That is why in the last years an extensive use of optical tracking devices took place. Such risen in popularity is mainly due to the release of very powerful technologies such as the Microsoft Kinect [7] and the Leap Motion [8]. This kind of devices supports in fact the tracking of the human hand's motion in the 3-D space so that the operator could control the robot in a more natural way.

Du and Zhang [9] have been among the firsts to innovatively adopt the Microsoft Kinect for Xbox 360 to track the hand of the operator to control dual robot manipulators. They obtained the orientation of the hands through tracking the thumb and the index fingers. However, this method was easily affected by possible occlusions and characterized by low precision.

The Kinect is equipped with two infrared cameras for depth detection and one standard visual-spectrum camera for visual recognition. It can obtain the depth information and colour images of the operator which are used to calculate hand's position and the orientation. However, the image capture and segmentation processes of this method are of high computation complexity. Furthermore, considering also the low accuracy and stability of the device, the use of the Kinect does not represent an ideal solution in contexts where high precision and small temporal delays are required.

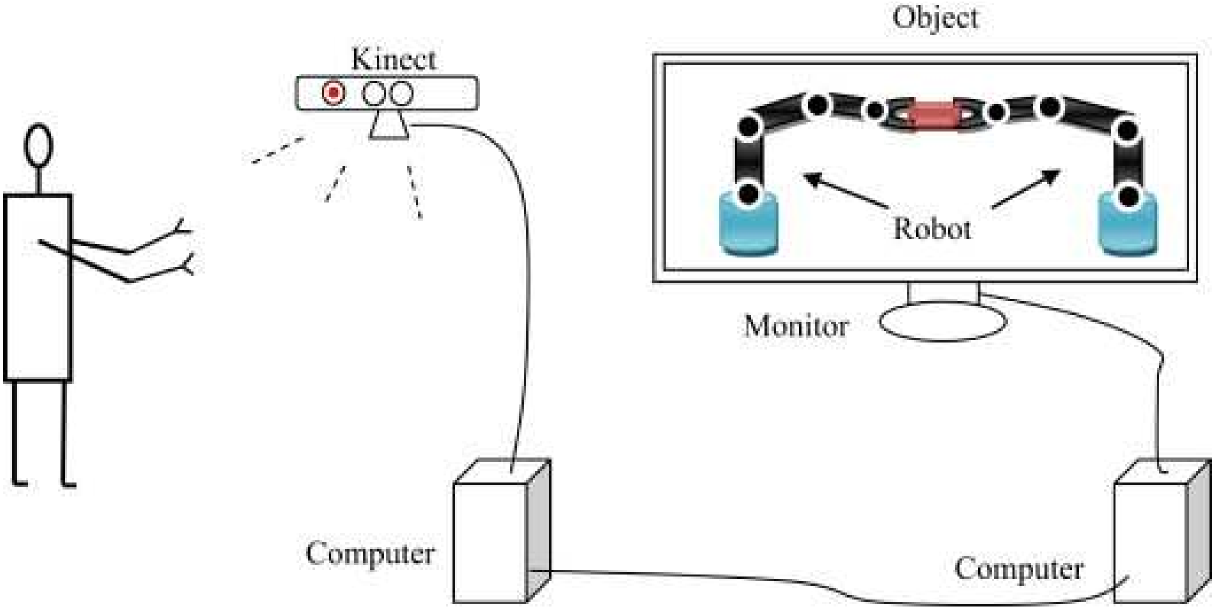


Figure 2.2: Schematization of the human-robot interface using the Microsoft Kinect by Du and Zhang.

The deficiencies shown by the Microsoft Kinect can be partially solved, and surely improved, by the usage of the Leap Motion as the optical device tracking hands' motion. Apostolellis et al. [10] used the Leap Motion in order to perform integral and separable 3D manipulation tasks in a stage lighting application. In [11] a human-machine communication interface between the Leap Motion controller and a 6-DOF Jaco robotic arm is developed in the context of ambient assisted living (AAL). The Jaco would assist the elderly and disabled people in their homes, so the system included also an algorithm to take account for hand terror and shake. Moreover, Kobayashi et al. [12] developed a human-robot handover motion where the robot measures the position and posture of human hand by the Leap Motion controller. Additional examples of human-robot interactions supported by the use of the Leap Motion device are [13], [14] and [15].
Besides from the Kinect and Leap Motion there is a wide range of possible technologies allowing for a contact-less and intuitive mean of tracking human's motion. Lee et al. [16] presented a exoskeleton robot system for human power assistance where angle sensors and inertial sensors track user's movements. Method [17], which was based on vision techniques, used some markers which were attached to the hand to assist tracking the hand of the operator. In order to remove the probability of occlusion, several camera-detected markers were attached on the body, making the system complex and inaccurate. Similar kind of problematics regarded the work of Won et al. [18], which presented a novel methodology based on one position sensor and one inertial measurement unit (IMU) to estimate position and orientation with the integration of two filter tools. Though this method could obtain relatively accurate pose estimation, the computational efficiency of orientation estimation and the operation accuracy could be further improved.
In conclusion, a concept similar to the one analysed in this thesis work, was treated by

Kofman et al. [19]. They tracked the hand-arm movements in the 3-D space so that it could allow the operator to focus on the global task naturally in the same way it goes with the Leap Motion. However, the method needed the operator to perform gestures in a dark background, with the hands being higher than the shoulder. So not only this method could not detect a precise gesture since suffering from lighting effects, but it also implied an uncomfortable position for the human operator.

2.2 Research Objectives

Among the many possible improvements that are possible with the proposed human-robot control interface, the most significant ones have been selected. According to them the development of the manipulator control system has followed specific paths. As it will be later explained, the functioning of the whole HRI system was experimentally verified in a laboratory. The main objectives, which will be now introduced, were then considered as parameters of evaluation of the achieved machine operation.

As already stated before, the most natural form of communicating even complex movements to a robot manipulator is the one where it copies the motion of human hands. So one of the main goals when tracking hands' motion is to develop a more intuitive form of control where the time that is necessary in order to learn how to control the machine is considerably reduced. The idea was then to interpret the input data coming from the Leap Motion in such a way as to have the manipulator reproducing in a specular way all the movements performed by the human hands. In particular, in order to achieve an high level of intuitiveness, two different configurations were taken in account. In the cases where the human operator was in front of the manipulator, as depicted in Fig.1.1, the movements of the right hand were reproduced by the left arm while the right arm followed the left hand. In this case we can speak of a mirror-like interaction. When instead, the human operator was standing behind or next the robotic arms, the opposite configuration was considered, letting the right arm copy the right hand and the left arm to copy the left hand. With these kinds of settings the aim is to obtain a reduced learning curve so that the manipulator can be successfully controlled even if the operator is not an expert. In fact he would only need to focus on the task to be completed rather than distracting himself by thinking about the specific motion of his/her hands.

One more advantage in preferring the Leap Motion to one of the classical means of real-time control for a manipulator, stays in the fact that the motion of one single hand is sufficient to collect all the necessary information to move one robotic arm. In fact, if we consider a classical joystick controller, it will be capable of receiving from the human operator only the data about the desired position, while one more joystick will be required in order to define the desired orientation for the end-effector. The Leap Motion device, on the other side, can track an operator's hand in six degrees of freedom capturing both the continuous 3-D position and orientation of the human hand. The main benefit coming from this is that the user will be able at controlling dual robots through both his or her hands operating two robotic arms simultaneously.

This clearly allows one to perform a wide variety of operations that require a combined action of two robotics arms in a very intuitive way. The main motivation for controlling two arms simultaneously in teleoperation setups is that the operator is used at performing bimanual tasks, and having two arms at the control site means that the operator's bimanual skills can be transferred to the robotic system.
Truth be told, there exists also some kinds of joysticks, which are widely used in many robotic applications, that are capable of tracking hand's motion in a 3D environment. They are characterized by a special design allowing for the motion of the human hand in six DOF. A popular example is the SensAble Phantom Omni [20], pictured in Fig.2.3.

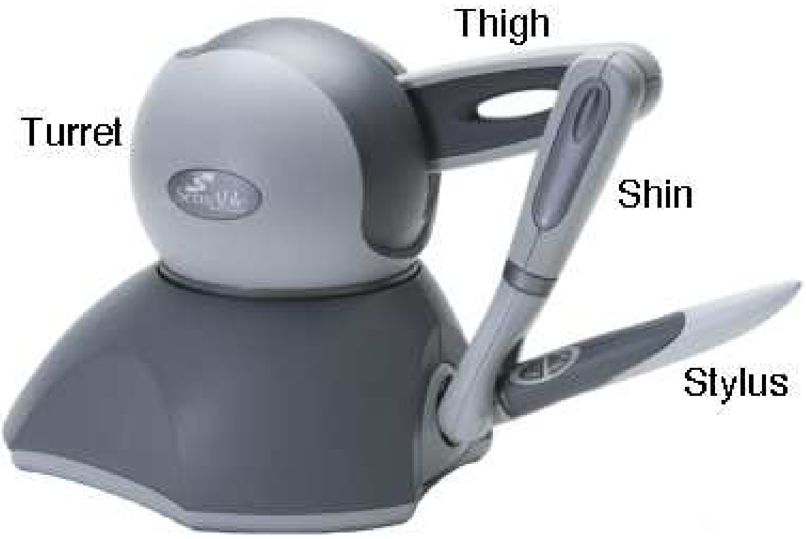


Figure 2.3: Six degrees of freedom joystick: SensAble Phantom Omni

Also using these kinds of joysticks some important drawbacks are implied. Their size is in fact quite bulky if compared to most of the traditional joysticks, while also getting used at maneuvering one of them could take some time. With the Leap Motion instead, one operator can perform the six DOF motion of his hand in a contactless environment. The devices that require contact, such as the joysticks, may prevent the human movements while the contactless ones are not. Allowing the user to be completely unimpeded in free space, it is then possible to achieve the simultaneous control of a dual-arm robot in the most effective way.

One additional objective that is pursued when implementing the control of the manipulator is the realization of an actual real-time functioning. Latency is always a critical component in real-time operations since the robotic arm needs to be able to keep up with the user, in case the user needs to interact or grasp an object. Ideally, the robotic system is supposed to move immediately as soon as the human hands start moving upon the Leap Motion controller. However, there are different components that contribute in

defining a significant, yet still reducible, delay between the input and the robotic motion. Latency is first of all due to the time required by the Leap Motion to recognize and elaborate data coming from the hands' motion. On average it takes 50ms for the Leap to calculate and transmit tracking data at each acquired frame. Additional delay is due to the time required by the central computer to further elaborate the input data when computing inverse kinematics and interpolating between consecutive joint angles. Finally, the greatest contribution to the latency is caused by the servo motors enacting the movements of the single joints composing the manipulator's arms. In fact, according to the specific model of manipulator , they might take up to 500ms before starting moving after they have been commanded to.
In the following chapters the real-time requirement will be then taken in high consideration with the aim of minimizing as much as possible the input-output delay.

A further parameter that has been considered during the design and thus also the evaluation of the HRI system is the accuracy of the performed motion. A very accurate reproduction of the motion of the hands of the human operator was required in terms of position, orientation and also velocity. In order to evaluate accuracy two contributions had to be taken into account. Firstly the Leap Motion, although promising a sub-millimeter accuracy, can influence the overall error in terms of hand's tracking. According to Weicher et al. [21] the standard deviation when moving to discrete positions on a path is less than 0.2 mm in static setups and less than 0.7 mm in dynamic cases. In addition the specific robotic manipulator influences decisively the precision at which the input movements are copied by the system. In order to account for these contributions, in the final chapters results about the obtained level of accuracy will be reported.

Finally, the real-time control of the manipulator has been designed and experimented in accordance with the high standards of safety that are always required in research, as well as in industrial, environments. In particular, when mapping the human's gestures into the robot motion, a particular attention was given in such a way as to avoid every possibility of collision, and then of damage, among the two arms of the manipulator. Also the position of the human operator was established with the intention of avoiding every possible physical contact between human and robot. Furtherly, a specific limitation was imposed to the maximum speed of motion of each end-effector and then of each joint angle motor. This has been imposed considering the likely situation in which the human operator could inadvertently lose control of his/her hands and then impose a very fast and dangerous motion to the manipulator.

2.3 Applicative Scenarios

By the applicative point of view, the technology that has been introduced up to now could be used in most of the contexts where robotic manipulations is generally required. Like all robotic systems, a manipulator controlled in real time is capable of performing a wide variety of tasks ensuring precision, safety and repeatability. Anyway, the proposed protocol of human-robot interaction, being based on the use of an optical device such as the Leap Motion and being targeted to the simultaneous control of two robotic arms, results in suiting particularly some kinds of applicative scenarios rather than others.

Let's start stating that the ideal field of work for a manipulator teleoperated in real time would not be one where it can get into physical contact with humans. Contexts such as the domestic one could hardly fit or profit from the presented technology.

The industrial and the research fields on the other hand, seem to suit better the situation of a robot system being teleoperated. Furtherly, it is necessary to consider that the real-time control is not oriented towards the continuous repetition of the same routines, but at the execution of specific and differentiated tasks. That is why repetitive works are not considered, even if in principle they could be executed through a Leap Motion controlled manipulator. Among the industrial scenarios of application, the ones that could mostly benefit from the proposed technology are the highly dynamic ones that is the ones where a different type of robotic manipulation could be required at each operation.

In addition, as already highlighted before, an especially characterizing feature is the dual-arm simultaneous control. There is in fact a wide range of so-called cooperative tasks where the combined use of two or more robotic arms is required. Among these tasks we can distinguish the un-coordinated from the coordinated ones. Considering as an example the case in which the dual-arm manipulator is working in an assembly line, an un-coordinated motion could see the left arm welding an unrelated workpiece while the right arm is palletizing parts. A coordinated task on the other hand would see both arms palletizing parts into the same box. Considered that both types of task would be a possible field of application, cooperative ones would clearly benefit more from the hand gestures control by Leap Motion. The use of both hands in a cooperative way comes in fact natural to humans that could then command the dual-arm manipulator straightforwardly.

It is then clear how a good applicative scenario is represented by all of the cases in which robots are expected to mimic human behaviours, acting and manipulating objects in ways similar to humans.

In addition, also the applications requiring an high level of accuracy could strongly benefit from the proposed Leap Motion infrastructure. In fact, as already pointed out, the Leap Motion device notably ensures elevated standards in terms of accuracy in the context of hands tracking. Moreover, in order to guarantee the basic safety rules for a working place, the manipulation via Leap Motion can exclusively be performed in environments where the position of the human operator with respect to the robot is well defined and restricted.

In order to report some concrete examples, it is possible to list some specific applicative scenarios where the use of a teleoperated dual-arm manipulator was required and that would significantly benefit from the real-time control by Leap Motion.
In 1952, American engineer Raymond Goertz teleoperated a dual-arm manipulator for handling radioactive goods [22]. In this case the use of the hand tracking via optical device would definitely improve the performance, considering that the robotic arms were used in pairs with the operator controlling one with each hand.
Same kind of improvement would have been possible when teleoperated robotic arms were used for deep-sea exploration by Fletcher et al. [23] or when they were used by NASA for space exploration [24]. In particluar, also in the NASA's Robonaut the robotic arms are teleoperated by hands' tracking, but in this case virtual-reality-based telepresence gloves are used as shown in Fig.2.4.

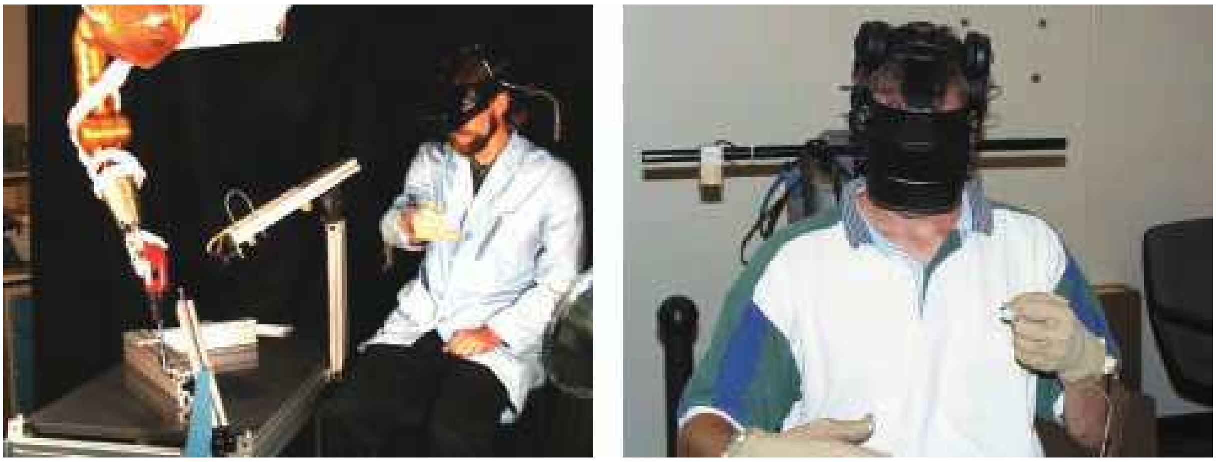


Figure 2.4: Teleoperation of the Robonaut through the virtual-reality-based telepresence gloves and helmet.

Some more examples can be reported with reference to a typical application field for dual arm systems that is parts assembly. The Leap Motion interface for controlling robot arms in real time could be in fact used in the case of gearbox assembly has described in [25], in the case of bending of metal parts [26] and in the case of folding of cartons into predetermined shapes [27]. It can also be considered, as concrete applicative cases, the ones that are typically related with dual-arm manipulation and with the ability of controlling two parts at the same time. That is the case, for example, of a typical peg-in-hole task, where one arm is positioning the peg and one arm the hole [28], or of a screw assembly, where one arm is controlling the nut and the other one the bolt [29]. This last example is pictured in Fig.2.5, where the interaction between the two arms holding the two pieces is shown.

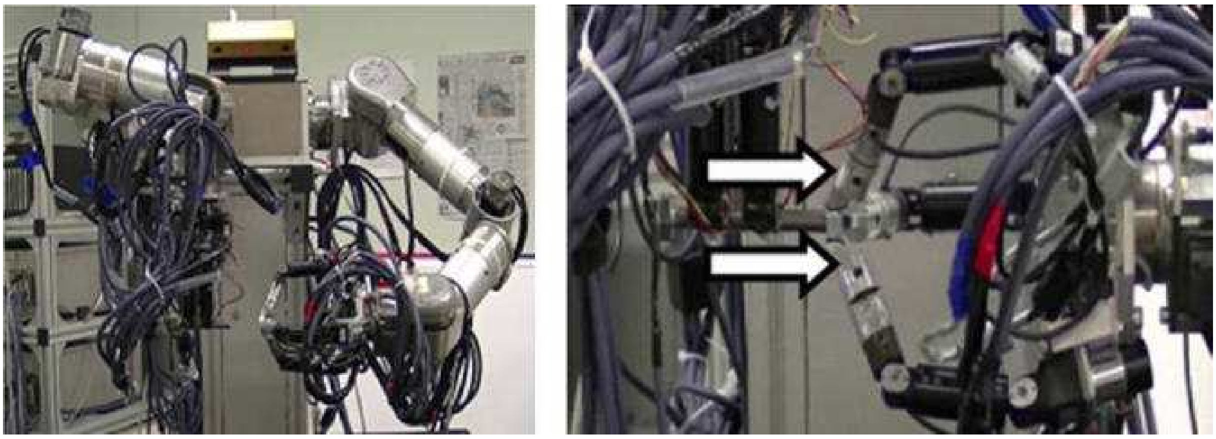


Figure 2.5: Cooperation between two teleoperated robotic arms in a typical screw assembly task.

One additional example proposing a solution that is relatively close to the one presented here is the "CellPro" dual-arm manipulator by Shibuya Hoppmann's [32]. The *CellPro* is a robot isolator designed to replace human operators in sterile-manufacturing processes that carry the risk of contaminating sensitive products or harming personnel. The system is equipped with a built-in two-armed robot whose movements are teleoperated by a couple of special joysticks allowing for a six degrees of freedom copy of the motion of the hands of a human operator.

Finally, there is also many examples of industrial dual-arm manipulators whose functioning could be considerably improved by integrating the Leap Motion real-time control system. The proposed infrastructure could be in fact implemented in relation with dual-arm robots that are typically teleoperated in real time such as the Toyota Dual Arm Robot [25], the Yaskawa Motoman SDA10D [30], or the ABB Frida [31].

3 Methodology

After introducing the general idea and its applicative scenarios, the specific implementative aspects will be presented in this chapter. When developing the real-time functioning of a manipulator, there are a series of technical issues that need to be addressed. Some of them are classical problems dealing with the kinematics and the dynamics of a six degrees of freedom manipulator (inverse kinematics, interpolation of input data), while some problematics arise from the necessity of interpreting data coming from a optical device such as the Leap Motion (coordinate systems, interpretation of input data, mapping). Some more issues are related with the requirement of having a real-time control that is interfaced with the software infrastructure sustaining the whole system (configuration of the leap motion command, shared memory interface). Each problematic will be now introduced and further discussed with the aim of defining possible solutions, among which the adopted ones will then be coherently motivated and explained.

3.1 Configuration of the Leap Motion Command

The proposed control infrastructure has been tested on a Epson C3 – A600 SR dual-arm manipulator, commercialised by Seiko. Such a manipulator comes with a dedicated software platform allowing for an efficient and intuitive human-robot interaction. Through this platform it is in fact possible to select commands imposing a specific kind of motion or task to the robot. For each of them the user will have to define also its input arguments, which in most cases correspond to the final desired pose for the robotic arms.

Some of these commands are already predefined and implemented by the software itself so that they are just ready to be tested. That is the case, for instance, of "Server ON" and "Server OFF", which are designed with the aim of activating and deactivating the arms in such a way as to put them in listen for possible motion commands coming in. In case we ask for the motion of the manipulator before turning on its servers, no kind of movement will take place and in case we forget to turn them off at the end of each cycle of work, dangerous behaviours could arise. The input argument for this kind of commands will be the number corresponding to which of the two manipulator's arms must be turned on or off.

Apart from these built-in commands, there is also the possibility for the user to define its own desired orders for the robot, which will be customized to his specific requirements. That is the case of the "Leap Motion", which is the name given to the command allowing for the real-time control of the manipulator through hands gestures.

In order to define a customized command, a new piece of coding has been written. It works the same way as a function does, taking some input arguments and producing a consequent output for the robotic system. Such function gets called by the main program, represented by the control window of the software platform, once the user has activated

the servers of the arms he wants to move.
The "Leap Motion" in particular, is different from the other commands in the fact that it does not require well defined input arguments prior to its calling. It in fact gets its input information in real-time from the optical device, so that it will be called with default arguments whose value does not actually matters. One more difference stays in the fact that it needs to be called only once by the main program, which will execute it continuously until it is interrupted by the user itself. It is possible to stop the execution of the real-time motion by two methods. Firstly, the human operator could get his or her hands out of the Leap Motion's visual range, imposing at the manipulator's arms to stop. Alternatively, in cases of dangers of some kind, it is possible to press an emergency button, causing not only the sudden stop of the arm's motion but also their servers to be turned off immediately. Such behavior is described by the finite-state machine depicted in Fig.3.1, where three possible states are identified for the considered system.

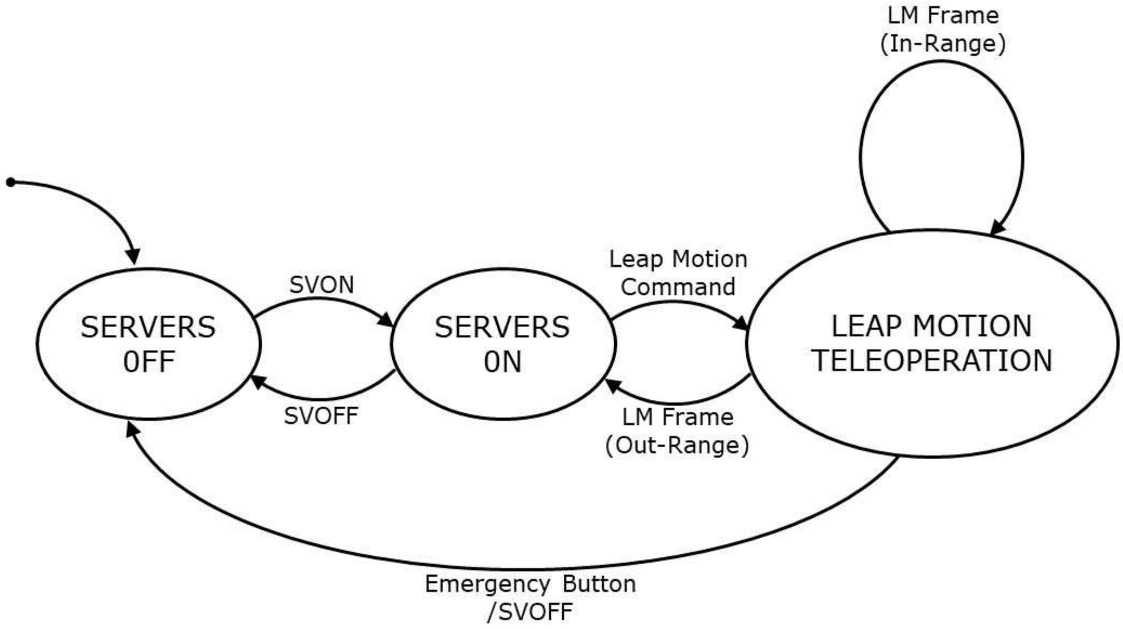


Figure 3.1: Finite-state Machine representing the relationships between the three possible functioning states of the System.

The code of the Leap Motion Command is responsible for all the necessary acquisitions and computations that are performed in order to impose to the robot a specular motion with respect to the one detected from the human operator's hands. In particular, the input data coming from the Leap Motion is accessed by the command from a shared memory source as it will be explained in next section. The Leap Motion catches a lot of information from the hands' motion so that it must be selected and then interpreted according to our needs. In particular, the desired pose will be considered in terms of displacements from the initial values caught by the sensor. The command then maps such displacements from the Leap Motion visual range to the manipulator's motion field, considering possible collision as a safety-related limitation. The obtained desired pose of the end-effectors will be then translated into desired joints angles through the inverse kinematics equations. In addition, the Leap Motion command is responsible for the interpolation among consecutive desired positions and orientations of the two robotic arms. After all of these computations and acquisitions, this personalized command manages to communicate with the robot itself in order to inform it about the motion that each one of its joints must perform. In the remaining part of this chapter, all the different features implemented through the Leap Motion command will be presented in detail.

3.2 Shared Memory Interface

It is usually relatively easy to define the motion of the manipulator in the cases when the control is not performed in real time. In such cases it is in fact possible to specify a priori the desired input arguments for the motion, using the software interface to define an offline communication of the data. On the other hand, in the cases where the control is performed in real time, it will be necessary to determine a specific communication protocol from the input source towards the manipulator. The definition of such protocol gets more complicated considering the quite common eventuality of a manipulator that is interfaced with its software supply through an operating system that is different from the one used to catch the input data. That is the case of the Epson C3 manipulator used to test the proposed infrastructure, which is commanded through the INTime operating system. The basic functioning of this manipulator consists in fact in a direct communication of the desired amplitude of each arm's joint angles from the software platform toward the INTime Kernel. The Kernel in turn provides a safe and verified access to the manipulator's hardware, commanding its motion.
Anyway, the considered case is not the basic one since the data will not be directly specified in the software platform but they must firstly be caught and also elaborated in real time before being finally sent to the manipulator. In order to achieve a well-functioning and also effective real-time control, a customized communication protocol has been designed according to the scheme represented in Fig.3.2.

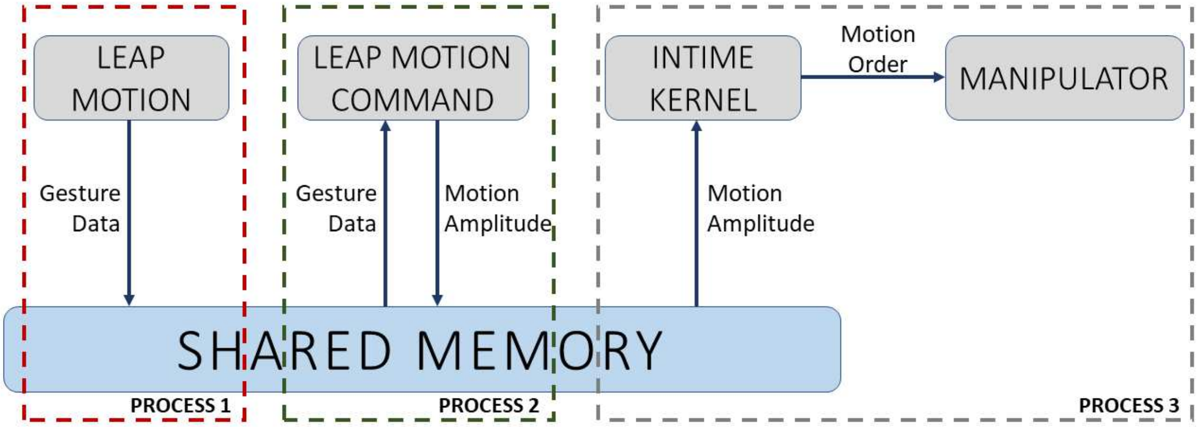


Figure 3.2: Schematization of the data exchange from the Leap Motion to the manipulator through the Shared Memory.

From Fig.3.2 it is possible to notice the presence of three independent processes. A first process (the one in red) performs the task of memorizing the data that is captured by the Leap Motion device in a dedicated space of a shared memory resource. Among the wide variety of information that is possible to retrieve from the sensor, the position of the palms of the hands, their velocities with respect to the previous frame and the orientations of the whole hands are considered. At the same time another process (the one in green) is executed. It runs the Leap Motion command that has been described in the previous section. Such command uses the gesture data that has been memorized by the first process in order to define the necessary motion of the manipulator's joint angles. The amplitude by which each joint should move is then memorized in another space of the memory that the three processes share. Such information is then directly accessed by the kernel of the INTime operating system, which is commanded by process three (the one in grey) to provide the two robotic arms with the desired motion in a safe and controlled way.

The use of a shared memory is made necessary by the fact that three different processes, with three different address spaces, are supposed to exchange information. In this way a technique of inter process communication is being performed.

3.3 Reference Systems

One more aspect that must be taken into account when dealing with the motion of a manipulator is the definition of the coordinate systems by which the input data and the motion itself are handled. In case of a real-time control by hands' tracking, it is required to interpret and understand the data in such a way as to make possible a specular copy of the hands' gestures from the human to the robot.
First of all it is necessary to establish a reference frame that is fixed with respect to the Leap Motion device. Considering that the Leap Motion is not going to move during the manipulator's functioning, but that it is going to be still in front of it, it is possible to identify the sensor's reference system as the global one with respect to which the robotic motion is going to happen. The chosen set of coordinate axes is reported in Fig.3.3.

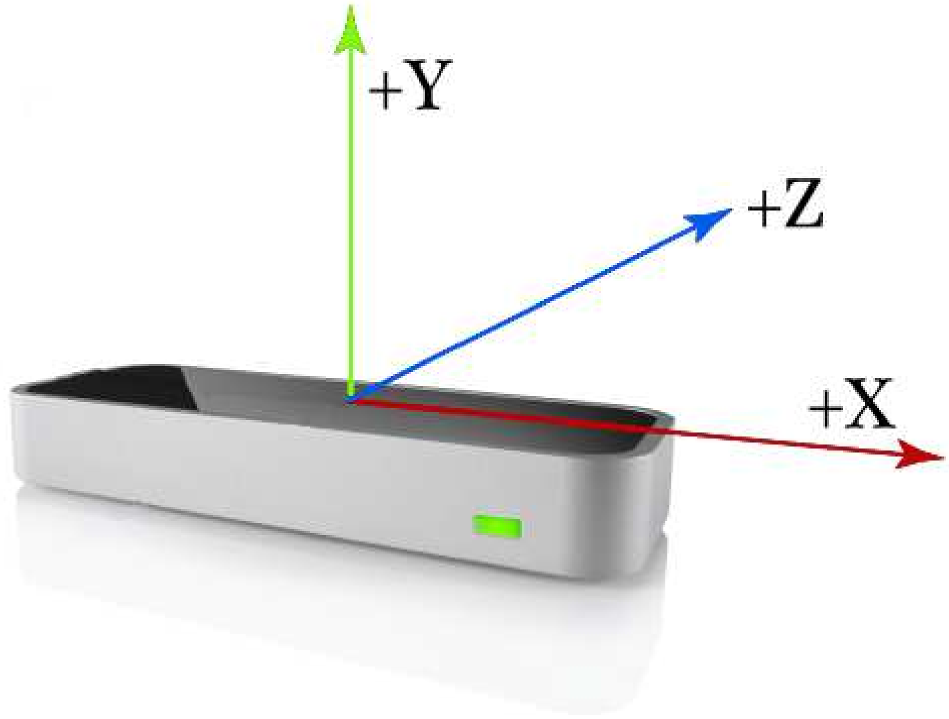


Figure 3.3: Reference system of the Leap Motion device.

In this case the choice was arbitrary and determined by the will of making the system as simple and intuitive as possible. With such purpose in mind, the same reference system used by the original system development kit of the Leap Motion has been chosen. The hands' tracking data are in fact obtained through the built-in functions that are provided by the sensor's development kit and it would be inconvenient to refer to such data with respect to a different reference frame.

After having introduced the global reference frame that is defined with respect to the Leap device, it is necessary to specify two reference systems that moves together with the end-effectors of the two robotic arms. In such way it will be possible to define the system's motion with respect to frames that gets located in the global one. In addition, in order to specify the manipulator's kinematics it is convenient to have two more reference frames that are fixed with respect to the manipulator. Such reference systems have got their origin at the base of the two arms, while the moving frames are located at the two end-effectors. The coordinate systems attached to the manipulator are depicted in Fig.3.4.

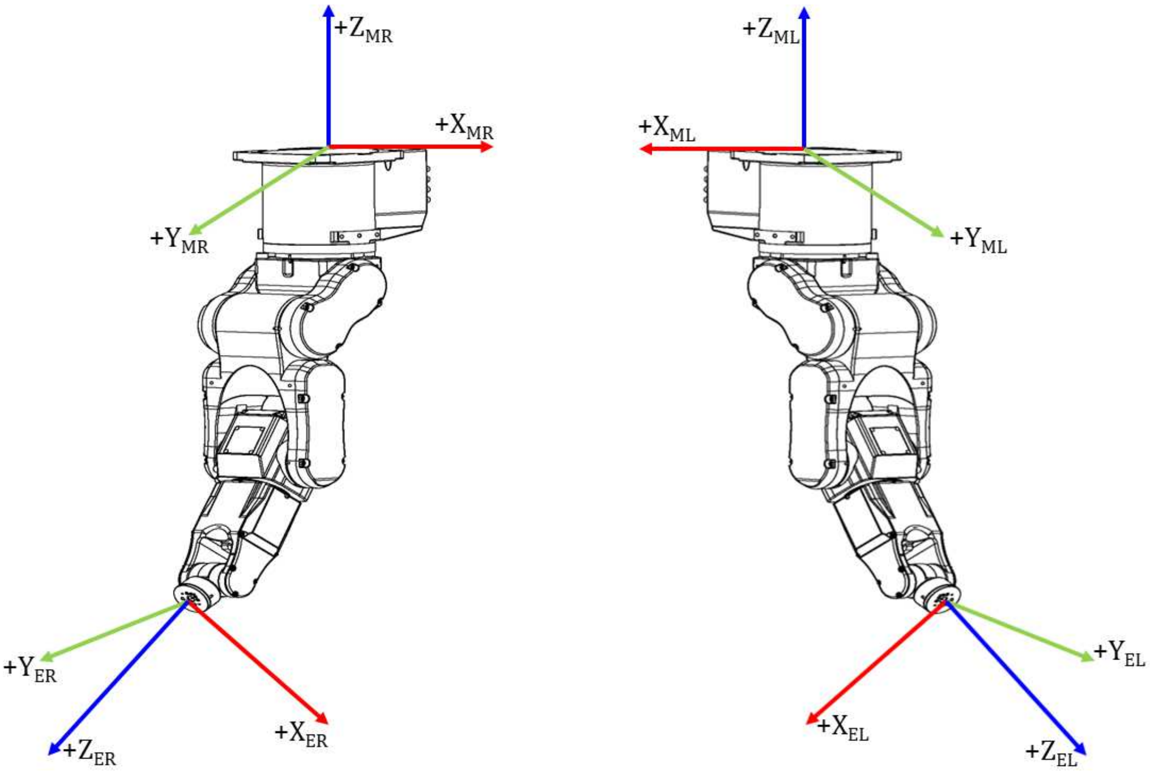


Figure 3.4: Reference systems attached to the manipulator's arms.

While the global reference system is represented by the X,Y,Z triad, the manipulator's notations present the *ER* and *EL* subscripts for the right and left endpoint frames respectively and the *MR* and *ML* subscripts for the frames fixed at the bases of the two arms.

3.4 Interpretation of Input Data

3.4.1 Displacemnts

The Leap Motion device is able to capture a wide range of data regarding the motion of the hands that are moving upon its LEDs and cameras. Among the provided information it is necessary to select the ones that are relevant in the control of the manipulator's motion. In this sense, the built-in functions, which are available through the Leap Motion developer kit, turn out to be helpful since they allow at knowing specific details regarding hands gestures. These functions allow in fact at distinguishing among left and right hand at each frame captured by the sensor. Subsequently, for each detected hand it is possible to extract different kinds of data regarding specific areas of the hand. On each finger, in fact, four focus points are provided in equidistant locations, while three focus points are located at the wrist and one more at the center of the hand's palm. Among these possible hand's areas of interest, the palm is selected as the most natural way of tracking the hands position.

The Leap Motion's functions provided by the developer kit are therefore capable of tracking the positions of the two palms of the human operator. The problem is that using this information as it is would require the manipulator to move towards the detected positions in a motion that would take a finite amount of time. Such interval of time would in turn cause a delay between the moments in which the hands and the robotic end-effectors reached the considered position. This kind of situation would not guarantee a real-time functioning, aside from a poorly intuitive control procedure.

In order to avoid that, and to have the two arms moving together with the human hands, it is necessary to feed the control system with the displacement by which the hands moved from their initial position. The positions of the two palms detected by the first sensor's frame are going to be saved and with respect to them the motion will be described. At the first frame captured by the Leap Motion the displacement will be null and at the subsequent frame it will be infinitesimal. In this way, the first type of movement to be requested to the manipulator will be small enough to be executed in an amount of time that is close to the one that elapses between two consecutive frames of the sensor. Defining by T the time that the manipulator takes to get to the desired pose and by T_L the time between two consecutive frames of the Leap Motion, the real time requirement is satisfied only in the cases in which:

$$T \leq T_L \tag{3.1}$$

The satisfaction of such requirement is not 100% guaranteed and the experimental results presented in chapter 5 are going to describe better this eventuality.

3.4.2 Orientation

In order to achieve an accurate replication of the hands movements, not only their position must be replicated by the manipulator, but their orientation must be taken into account as well. Also in the case of the data regarding the hands' orientations in space, they will be considered in terms of variation with respect to their initial value so that a real-time functioning will be possible, according to the same reasoning introduced in the case of positioning data. There are various possible manners of detecting hands' orientations in the 3D space. The Leap Motion approach consists in evaluating two different vectors:

- *Palm Normal*, which has a direction perpendicular to the plane formed by the palm of the hand and is pointing downward out of the palm.
- *Direction*, which is a vector laying on the palm's plane and pointing from the center of the palm toward the fingers.

According to the information retrieved about these two vectors, the functions defined in the developer kit are able to describe the orientation of a hand defining the relative rotation matrix. The data contained in this matrix are sufficient at defining in a unequivocal way the hands' orientation.

Anyway, the information contained in the rotation matrix cannot be sent to the manipulator controller as it is. The INTime operating system, underlying to the functioning of the manipulator, requires in fact the desired orientation of the robot's end-effectors to be expressed in terms of Euler angles instead. Euler angles are a triad of angles expressing in a more compact form the same information contained in a rotation matrix. The three angles specifically represents the three consecutive rotations that must be performed around the axes of a fixed reference frame in order to get a new coordinate system that is normal to the end-effector even after its motion. In our specific case it is possible to describe the orientation of the manipulator's end-effectors by defining the following angles with respect to the reference frames introduced in Fig.3.4:

- θ_x, angle of rotation around the X_M axis. After such rotation the $X^{'}, Y^{'}, Z^{'}$ reference system is obtained.
- θ_y, angle of rotation around the $Y^{'}$ axis. After such rotation the $X^{''}, Y^{''}, Z^{''}$ reference system is obtained.
- θ_z, angle of rotation around the $Z^{''}$ axis. After such rotation the X_E, Y_E, Z_E reference system is obtained.

Since the orientation of the hands is described by the Leap Motion in terms of rotation matrices, it is necessary to obtain the Euler angles values starting from their representation by rotation matrix. To do so it is possible to consider the three rotation matrices expressing the rotations around the frame axes by the three Euler angles just presented:

$$R_{X_M}(\vartheta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\vartheta_x & -sin\vartheta_x \\ 0 & sin\vartheta_x & cos\vartheta_x \end{bmatrix} \tag{3.2}$$

$$R_{Y'}(\vartheta_y) = \begin{bmatrix} cos\vartheta_y & 0 & sin\vartheta_y \\ 0 & 1 & 0 \\ -sin\vartheta_y & 0 & cos\vartheta_y \end{bmatrix} \tag{3.3}$$

$$R_{Z''}(\vartheta_z) = \begin{bmatrix} cos\vartheta_z & -sin\vartheta_z & 0 \\ sin\vartheta_z & cos\vartheta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

Multiplying among them these three matrices, the following manipulator's rotation matrix, function of the three Euler Angles is obtained:

$$R = \begin{bmatrix} cos\vartheta_y cos\vartheta_z & -cos\vartheta_y sin\vartheta_z & sin\vartheta_y \\ cos\vartheta_x sin\vartheta_z + sin\vartheta_x sin\vartheta_y cos\vartheta_z & cos\vartheta_x cos\vartheta_z - sin\vartheta_x sin\vartheta_y sin\vartheta_z & -sin\vartheta_x cos\vartheta_y \\ sin\vartheta_x sin\vartheta_z - cos\vartheta_x sin\vartheta_y cos\vartheta_z & sin\vartheta_x cos\vartheta_z + cos\vartheta_x sin\vartheta_y sin\vartheta_z & cos\vartheta_x cos\vartheta_y \end{bmatrix} \tag{3.5}$$

The matrix obtained from the Leap Motion's built-in functions will be known and it can expressed in the following notation

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3.6}$$

Comparing among them expressions 3.5 and 3.6 it is possible to obtain the following formulas expressing the Euler angles as functions of the elements in the rotation matrix:

$$\vartheta_x = tan^{-1}\left(-\frac{r_{23}}{r_{33}}\right) \tag{3.7}$$

$$\vartheta_y = tan^{-1}\left(\frac{r_{13}}{\sqrt{{r_{23}}^2 + {r_{33}}^2}}\right) \tag{3.8}$$

$$\vartheta_z = tan^{-1}\left(-\frac{r_{12}}{r_{11}}\right) \tag{3.9}$$

Finally, it must be considered that the X_M, Y_M, X_M manipulator's reference system is obtained from the X, Y, Z Leap Motion frame, by a counterclockwise rotation of 90 degrees around the X axis. For such reason, in the computation of the θ_x value it will be necessary to add a further contribution of 90 degrees, obtaining the following expression:

$$\vartheta_x = tan^{-1}\left(-\frac{r_{23}}{r_{33}}\right) + \frac{\pi}{2} \tag{3.10}$$

Equations 3.10, 3.8, 3.9 must then be used in order to compute the desired orientation starting from the rotation matrix provided by the Leap Motion.

3.5 Mapping

One more aspect that must be taken into account when transferring the detected data from the optical device towards the robot is the specific field of motion of the two communicating environments. Considering that the aim of the control infrastructure is to replicate the movements of the hands of the human operator, it is necessary to evaluate the size and the shape of the space in which not only the human hands, but also the manipulator's arms are free of moving.
In particular, the limitation to the motion for what concerns the operator's hands is not physically determined, but it depends on the range of view of the optical device being used. The Leap Motion provides a field of view shaped as an inverted pyramid, as shown in Fig.3.5.

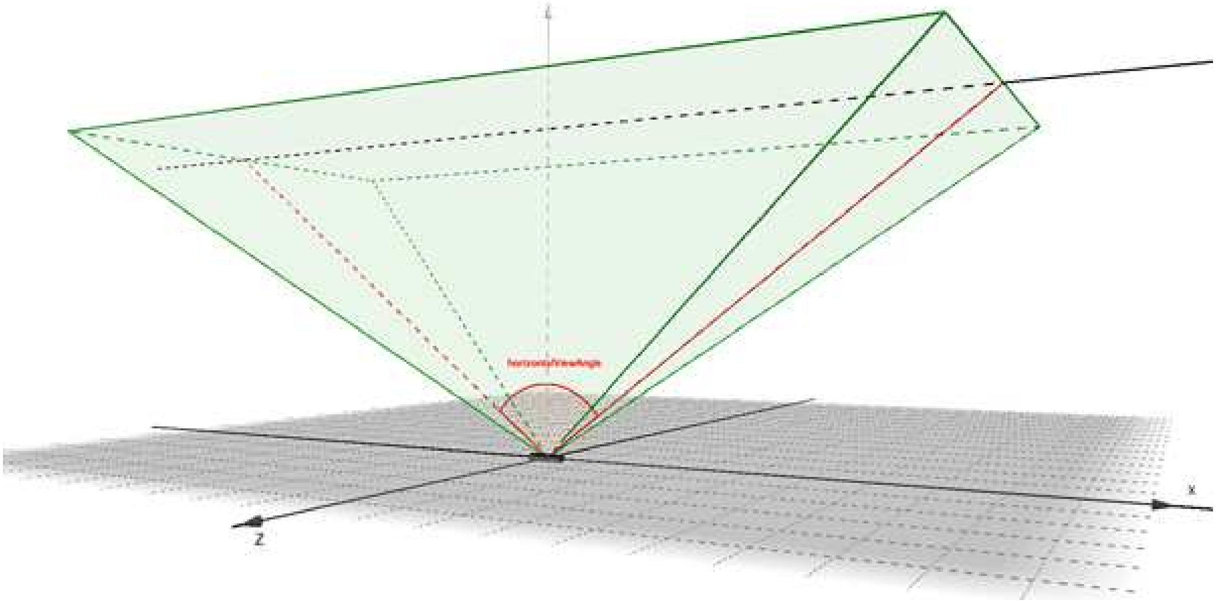


Figure 3.5: Field of view of the Leap Motion device.

According to Fig.3.5 the available range on the x and z axes is much smaller close to the device than it is near the top of the range. This would imply the impossibility of reaching points that are positioned at the bottom extremities of the manipulator's field of motion. Such problem is derived from the fact that the Leap Motion and the robotic arms are characterized by spaces of action of different shapes. In particular, the manipulator used in the experimental phase is composed of two arms fixed at the top (ceiling mounting) of a cage-like structure, defining its safe limits of motion. As shown in Fig.3.6, the field of motion of the robotic system posses the shape of a parallelepiped.

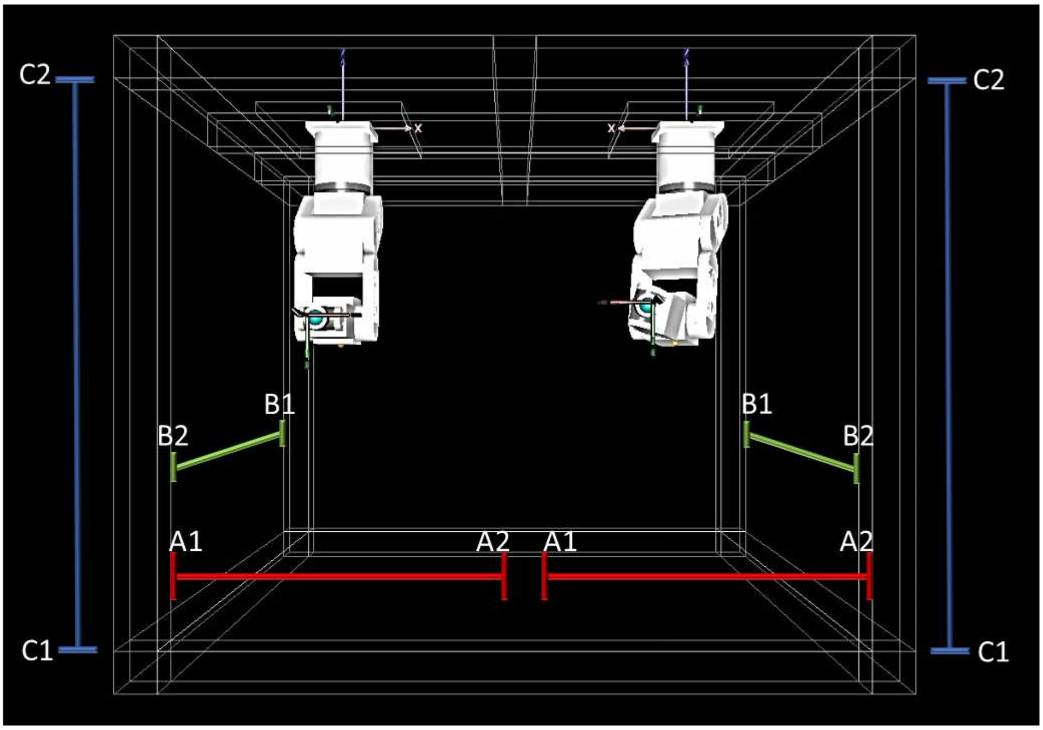


Figure 3.6: Field of motion of the Epson C3 dual-arm manipulator, determined by its mounting cage.

In order to map the two spaces of motion belonging to the human hands and to the robotic arms, an interaction box has been defined inside the Leap Motion range of view. It defines a rectilinear area as shown in Fig.3.7.

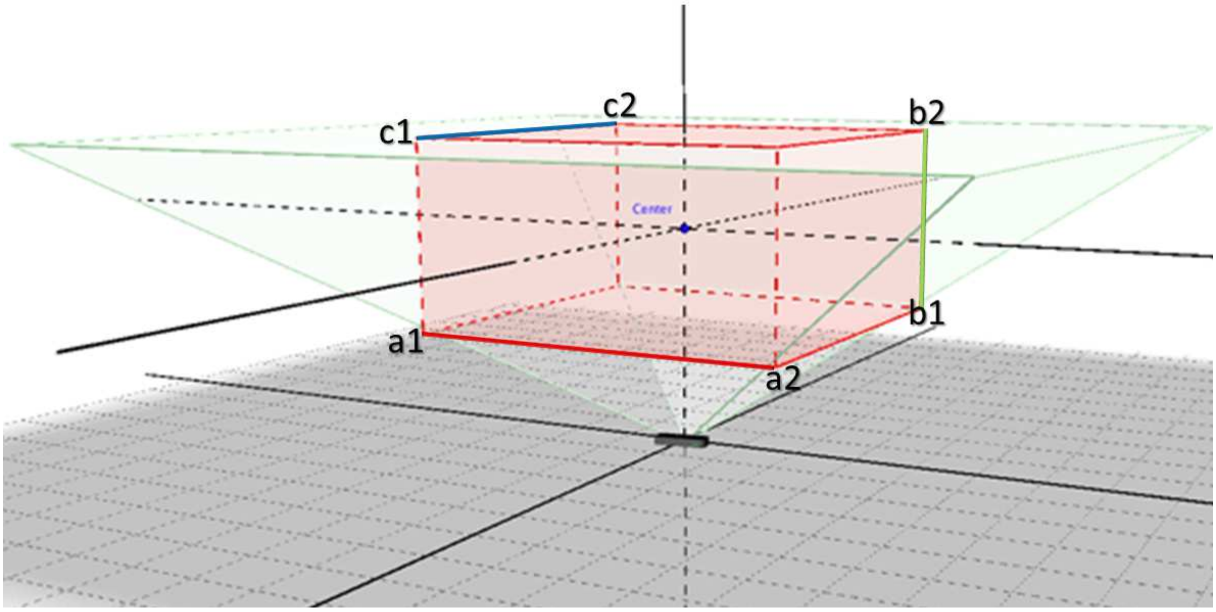


Figure 3.7: Interaction box defined inside of the Leap Motion range of view.

The shape of such area has been defined with the intent of avoiding any spatial distortion between the two considered environments, while its dimensions are evaluated with the aim of staying always inside of the Leap Motion field of view. Shaping the sensor's range in a different form with respect to its original one, will imply that the human hands could easily be detected outside of such interaction box. In such cases, the observed coordinates that are out of the admissible range will be reconducted at their specific limit. So, if for example the interaction box ranges from 100 to 200 millimetres along the z-axis and the detected palm position is of 250 mm at its z component, the coordinate considered in the further computations will be of 200 mm.

Comparing Fig.3.7 and Fig.3.6 is also possible to guess how the two motion ranges are characterized by different specific dimensions. In particular the sensor's range is way smaller than the robot one, in a way that would imply the manipulator at exploiting only a little part of its potential motion field. It is then necessary to decide how to scale the Leap Motion coordinates to suit the motion of the robotic arms. The greater the scale factor, the more affect a small physical movement on the sensor will have on the trajectory of the manipulator. This would make it easier for the user to move the end-effector from one side to another, but it would also make more difficult to obtain precise positioning. It is then necessary to find the best balance between speed and precision for the purposes of control.
The adopted solution regarding the scaling factor is due to the will of mapping the whole interaction box into the entire safe space in which the robot moves. With such objective in mind, a safe range of motion has been defined for both robotic arms with respect to their reference frames, as pictured by the A,B and C indexes in Fig.3.6. The dimensions of the interaction box, on the other hand, are chosen in such a way to have the whole box inside the inverted pyramid range of Fig.3.5. The limit values on the three coordinates of such box are reported in Fig.3.7 through the a,b and c values. In order to map the *abc* space into the *ABC* one, the following formulas for the mapping are considered, where the subscripts M and L are referred to the fields of the Manipulator and of the Leap Motion respectively:

$$x_M = \frac{A_2 - A_1}{a_2 - a_1} \Delta x_L \tag{3.11}$$

$$y_M = -\frac{B_2 - B_1}{c_2 - c_1} \Delta z_L \tag{3.12}$$

$$z_M = \frac{C_2 - C_1}{b_2 - b_1} \Delta y_L \tag{3.13}$$

In these equations, the Leap Motion coordinates are considered in terms of displacement from thei first-frame value, as explained in section 3.4. The difference between the sensor and the manipulator coordinate systems required the flipping of the robotic y-axis and z-axis with the z-axis and y-axis of the Leap Motion respectively. The minus sign in 3.12 is due to the fact that the y-axis in the *M* reference system is on the same direction as the z-axis of the sensor frame but in opposite verse, as depicted in 3.3 and 3.4.

3.6 Inverse Kinematics

Up to now we dealt with problematics related with the necessity of interpreting the information coming from an optical device such as the Leap Motion. After considering the input data in terms of displacements and after mapping them from the sensor to the manipulator fields of motion, the desired positions and orientations for the end-effectors have been obtained. It is now necessary to order the motion to the robotic arms according to the performed computations.

As always, the manipulator's movements are performed through the change in amplitude of the six joints composing each arm, which are in turn commanded by their own servo motors. It is then necessary to solve the Inverse Kinematics problem for the considered six degrees of freedom manipulator. Such problem consists in the computation of the six joint variables corresponding to the given positions and orientations of the end-effector. Solving the Inverse Kinematics is then of fundamental importance in order to transform the computed specifications, assigned to the end-effectors, into the corresponding joint space movements that allow the execution of the desired motion.

In the robot considered for the experimental purposes there is no prismatic joint since all of them are revolute, which means that the joints and thus the variables to be obtained are all angular ones. In addition, with respect to the specific infrastructure presented up to now, the resolution of the Inverse Kinematics problem is included as part of the computations performed by the Leap Motion command, which extensively elaborates the input data before commanding the desired output to the robotic system.

As already explained, this command is continuously executed once the software platform calls it. However, the frequency at which the command gets executed is constantly higher than the frequency at which the Leap Motion device catches new frame and thus new input data. Considering also that the resolution of the Inverse Kinematics is particularly demanding from the computational and temporal points of view, it will be performed only in the cycles where a new sensor's frame has been detected.

Inside of the Leap Motion command the Inverse Kinematics problem is solved through a specific function. Its input arguments are the desired positions, as expressed by the mapping equations 3.11, 3.12 and 3.13, and the desired orientations, as expressed by the 3.10, 3.8 and 3.9. Additional inputs for the Inverse Kinematics function are the geometrical data regarding the lengths of the different links composing the two robotic arms.

According to the provided information, the corresponding angular amplitudes of the six joints of each arm are obtained. The equations that have been computed in order to express the joint angles as function of the input variables are not reported here since they are long and of difficult reading. However, the procedure used to estimate them corresponds to the one introduced by Siciliano et al. in [33]. In particular, the Epson C3 robot can be included in the category of the Manipulators with Spherical Wrist, for which a simplified solution process can be identified. Such category comprehends many existing manipulators, which are composed by a three-link planar arm and by a spherical wrist. They allow to articulate the inverse kinematics problem into two subproblems, since the solution for the position is decoupled from that for the orientation. It is in fact possible to determine the amplitude of the first three joint angles, θ_1, θ_2 and θ_3, according to the desired position of the end-effector. After that, the values obtained for the first three angles will be used in order to compute the amplitudes of θ_4, θ_5 and θ_6 that guarantee the desired endpoint orientation. Therefore, on the basis of this kinematic decoupling, it is possible to solve the inverse kinematics for the arm separately from the ones for the spherical wrist.

3.6.1 Selection of Best Solution

The category of the manipulators with spherical wrist does not make any exception when it comes to the number of different correct solutions existing for the Inverse Kinematics problem. Actually for a six degrees of freedom manipulator without mechanical joint limits, there are in general up to 16 admissible solutions. The number of accepted multiple solutions for the real structure is however limited in the cases where mechanical joint limits are present. In particular, in the developed case of the teleoperated dual-arm robot system, only 8 of the 16 possible solutions are considered, since only the angular values ranging between $-\pi/2$ and $\pi/2$ are admitted.
Given the above, some criterion is required to select the best one among the admissible solutions of the Inverse Kinematics problem. It is necessary that the chosen solution guarantees continuity with the previously performed motion , avoiding sudden and dangerous movements of the robotic arms. To do so, all of the 8 solutions are compared among each other in order to evaluate which one will ensure the smoothest motion. For each solution the summation of the required motions for each joint is computed, according to:

$$\sum_{i=1}^{6} |\vartheta_i - \vartheta_{i,k}| \tag{3.14}$$

where the θ_i represents the actual amplitude of the i-th joint angle, while the $\theta_{i,k}$ represents the new desired value for the i-th joint angle according to the k-th solution of the inverse kinematics problem. Among the 8 admissible solutions, the one minimizing the quantity expressed in 3.14 is selected as the one to be sent to the manipulator in order to command its motion.

3.6.2 Singularity Points

One additional problem that must be addressed when solving the inverse kinematics, is the fact that there might be no admissible solutions, in view of the manipulator's kinematic structure. The configurations at which no reliable solution is found are called singularities and they can be detected evaluating for which configurations the manipulator's Jacobian J is not full rank.
The problem with singularities is that they constitute configurations at which the mobility of the structure is reduced in the sense that they prevent the end-effector from executing the imposed motion. Moreover, when the structure is in a singularity, a request for small velocities of the end-effector could cause large and then dangerous velocities of the arms' joints.

Singularities can be classified into:

- *boundary singularities*, taking place when the arms are either outstretched or retracted
- *internal singularities*, occurring inside the reachable workspace.

Both kinds of singularities must be avoided through specific techniques depending on the specific practical situations.
In our case the way to avoid boundary singularities is straightforward. The Leap Motion data is in fact mapped in a way that the boundaries are never requested by the controller and thus the fully outstretched or retracted configurations are never assumed by the robotic arms.
Internal singularities, on the other hand, are avoided through the process of selection of the best inverse kinematics solution. Every time one of them occurs, a specific alternative configuration, ensuring a continuous and smooth motion is selected.
To realize such methods of singularities avoidance, it is then necessary to evaluate all the existing configurations with no admissible solutions. This is possible through the computation of the values for which the determinant of the Jacobian is null. Such computation is generally of no easy solution, but in the cases of manipulators with spherical wrist it is possible to decompose the problem into two easier ones. For such category it is in fact possible to partition the Jacobian into (3x3) blocks as follows:

$$J = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \tag{3.15}$$

Through a convenient choice of the reference frames of the different links of the robotic arms, the overall Jacobian becomes a block lower-triangular matrix, obtaining a significant simplification in the computation of its determinant. In such case the determinant is in fact given by the product of the determinants of the two blocks on the main diagonal. In particular, the arm singularities, resulting from the motion of the first 3 links, are obtained by the condition $det(J_{11}) = 0$, while the wrist singularities, depending from the motion of the wrist joints, are computed through the $det(J_{22}) = 0$ condition.

3.7 Joint Space Interpolation

One more typical issue that needs to be addressed when defining the functioning of a manipulator, is the definition of its ideal trajectory. In particular, it is usually required to impose a smooth motion to the end-effector, requesting the passage through some assigned points. The robotic motion is in fact typically described as a sequence of desired poses, which can potentially be distant in space in a way that the intermediate trajectory points must be determined.
In the case under examination, the desired end-effector trajectory is fully specified trough the motion of the hands of the human operator, caught by the Leap Motion device. Apparently, there might be no need for the definition of intermediate points between the detected ones. However, it must be considered that the frequency at which new motion frames are captured by the optical device is constantly lower with respect to the frequency at which new motion commands are sent to the two robotic arms. According to the performed experiments, there is always between 15 and 25 cycles of the controller among two consecutive frames of the Leap Motion. Such condition can then be exploited in order to define a smoother motion, which is characterized by a regular interconnection of consecutives desired poses.
The path generation can be performed by the interpolation of consecutive end-effectors positions and orientations or also with respect to the motion of the six joint angles. Interpolating with respect to the endpoints poses would require to define intermediate values according to which the corresponding joint amplitudes should be computed through the inverse kinematics. Solving at each cycle of the controller the inverse kinematics would clearly be demanding by the computational point of view as well as by the temporal one. Alternatively, it will be considered a method of path generation in which the path shapes in space and time are described in terms of the joint angles. In particular, the first step is performed by specifying each path point through the conversion of the hand position in the corresponding pose of the end-effector. After that, each pose is translated in terms of angular joints amplitude through the inverse kinematics. Once the desired six joints angles are obtained, a smooth function is found for each one of them. Such function will be required to connect two consecutive angular values in an amount of time which corresponds to the interval between two consecutive frames of the Leap Motion. Moreover, it should guarantee the desired values for what concerns the initial and final velocity of the joint motions. The interpolation function is then required to satisfy the following constraints:

$$\vartheta(0) = \vartheta_0 \tag{3.16}$$

$$\vartheta(t_f) = \vartheta_f \tag{3.17}$$

$$\dot{\vartheta}(0) = \dot{\vartheta}_0 \tag{3.18}$$

$$\dot{\vartheta}(t_f) = \dot{\vartheta}_f \tag{3.19}$$

In order to satisfy four different constraints, the interpolation function must be a polynomial of at least third degree, which is characterized by four coefficients. The interpolation function is then a cubic in the following form:

$$\vartheta\left(t\right) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{3.20}$$

In order to satisfy the four constraints with respect to 3.20, the following expressions for the four coefficients are obtained:

$$a_0 = \vartheta_0 \tag{3.21}$$

$$a_1 = \dot{\vartheta}_0 \tag{3.22}$$

$$a_2 = \frac{3}{t_f^2}\left(\vartheta_f - \vartheta_0\right) \tag{3.23}$$

$$a_3 = -\frac{2}{t_f^3}\left(\vartheta_f - \vartheta_0\right) + \frac{1}{t_f^2}\left(\dot{\vartheta}_f + \dot{\vartheta}_0\right) \tag{3.24}$$

where the initial angular amplitude θ_0 has been memorized from the previous iteration and the desired final angle θ_f has been computed through the inverse kinematics. The desired value for the initial velocity $\dot{\theta}_0$ is established in order to have continuous velocity from the previous frame, so it corresponds to the final velocity reached at the previous frame. The final angular velocity $\dot{\theta}_f$ is instead computed in accordance with the instantaneous hand velocity registered by the Leap Motion. Such information is in fact used to compute the desired velocity at each joint through the inverse Jacobian matrix. Finally the t_f value corresponds to the time that elapses between two consecutive frames registered by the Leap Motion.

Finally, in order to summarise the system's functioning that has been described in this chapter, the block diagram of Fig.3.8 is here reported.

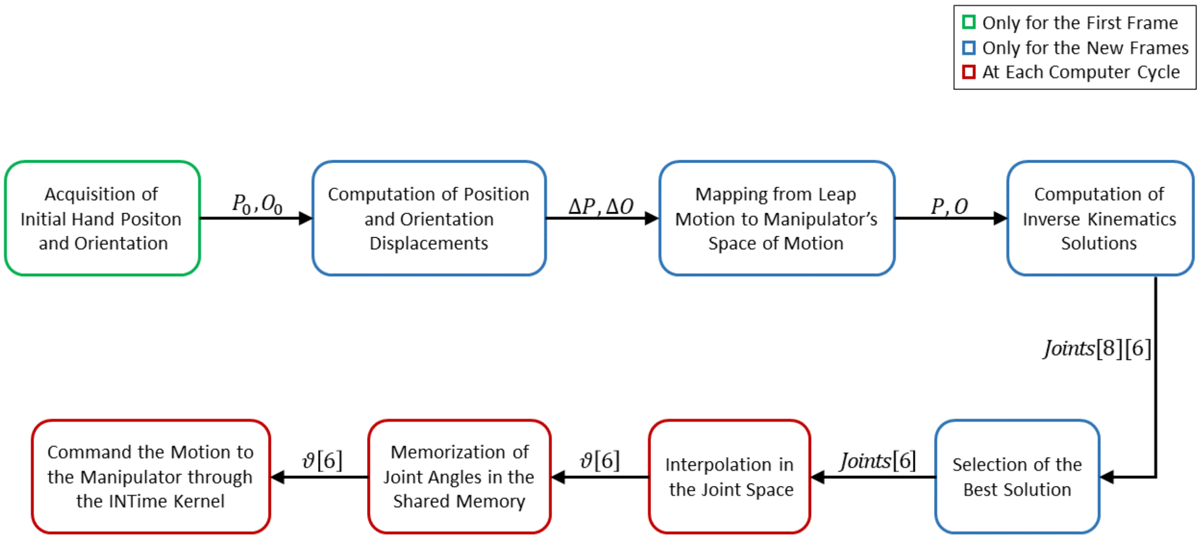

🟩 Only for the First Frame
🟦 Only for the New Frames
🟥 At Each Computer Cycle
Acquisition of Initial Hand Positon and Orientation
P_0, O_0
Computation of Position and Orientation Displacements
$\Delta P, \Delta O$
Mapping from Leap Motion to Manipulator's Space of Motion
P, O
Computation of Inverse Kinematics Solutions
$Joints[8][6]$
Selection of the Best Solution
$Joints[6]$
Interpolation in the Joint Space
$\vartheta[6]$
Memorization of Joint Angles in the Shared Memory
$\vartheta[6]$
Command the Motion to the Manipulator through the INTime Kernel

Figure 3.8: Block Diagram picturing the whole system's pipeline, from the data acquisition to the robot's motion.

4 Experimental Setup

4.1 Hardware Supply

4.1.1 The Manipulator

The teleoperation by Leap Motion has been tested in a laboratory environment on a Epson C3-A600 SR manipulator commercialised by Epson Seiko [34]. Such robot is a dual-arm manipulator, conceived for being used in applications of industrial automation, but fitting the research purposes as well. It is in fact characterised by best-in class levels of speed and accuracy, which ensure increased levels of productivity and quality. Both arms are able to exploit the surrounding space with high efficiency thanks to their small elbow areas (Joint #3) and thanks to their slim bodies. The size of the two Manipulator's arms is in fact only 1/44 of their motion range. In addition, they feature a reduced level of interference with peripherals, walls, and ceiling, allowing for a more compact installation and an improved flexibility in hand design.

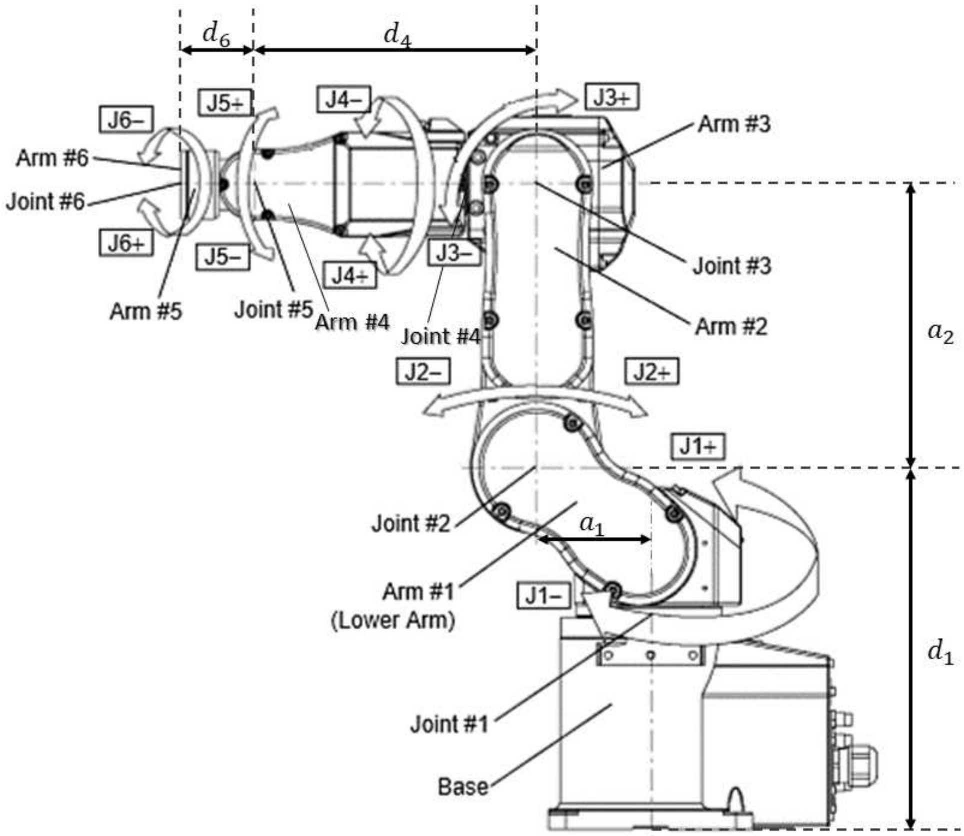


Figure 4.1: Links and joints of a C3's robotic arm.

One more fundamental property of the manipulator is in the wrists (Joint #5), which are realized with a compact design, enabling a skilful control with wide motion range, smooth movements and the ability to work from many angles.
The C3 Manipulator is composed by two identical robotic arms, which are free to move in six degrees of freedom. They are both composed of six angular joints and six links as shown in Fig4.1. From the figure is possible to identify the axes of motion of each joint, as well as a schematic organization of the physical structure, made of six connected arms.

According to Fig.4.1, each joint is intuitively responsible of the following motions:

- Joint #1 : The whole Manipulator revolves.
- Joint #2 : The lower arm swings.
- Joint #3 : The upper arm swings.
- Joint #4 : The wrist revolves.
- Joint #5 : The wrist swings.
- Joint #6 : The hand rotates.

The total length of each arm is of 600 mm in case of an outstretched configuration, while their weight is of 27 kg each. Such numbers, not only allow for a wide range of motion, but also for a potentially dangerous human-robot interaction.

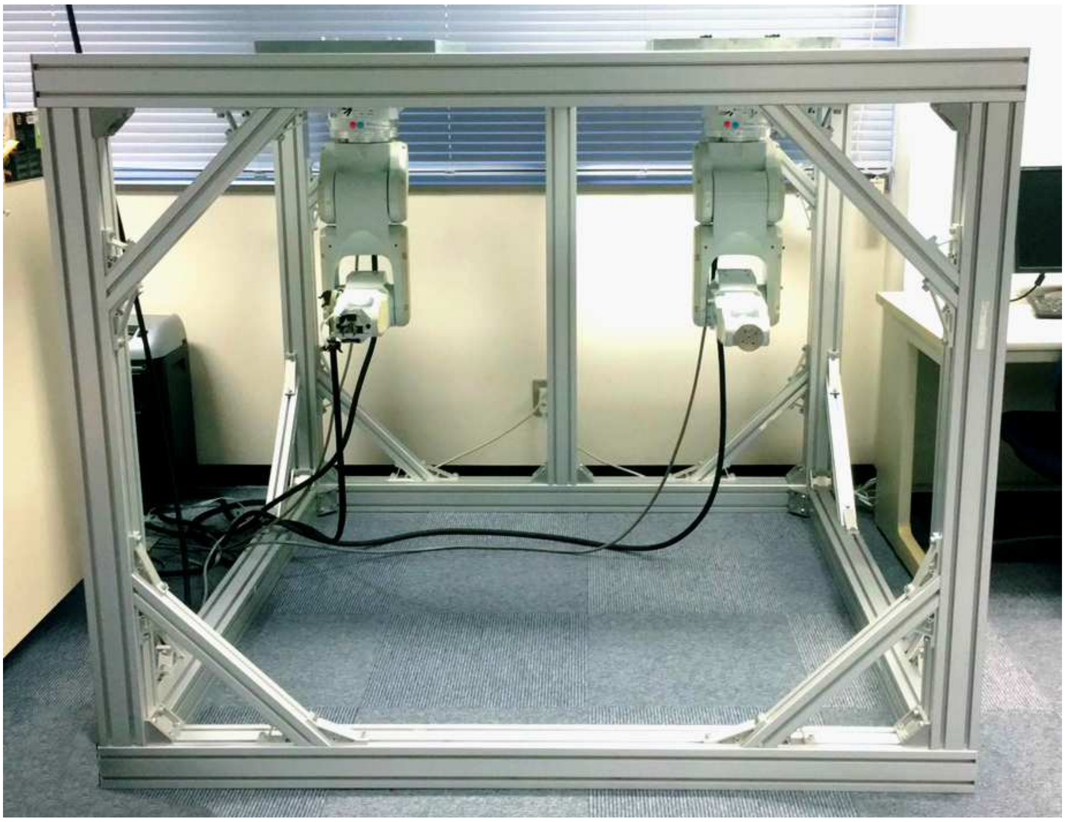


Figure 4.2: C3 Manipulator used for the laboratory texting: bearing structure with the robotic arms fixed in ”ceiling” configuration.

In order to avoid unsafe contacts or impacts with human operators and also with the surrounding environment, the two arms are mounted in a cage structure, defining their limit of interaction with the adjacent space, as shown in Fig.4.2. From the figure is possible to see how the two robotic arms have been fixed to their bearing structure through a ceiling mounting configuration. The other possible configurations for mounting would require the fixing on the table top or at the wall's surface. Among them, the ceiling mounting results in being the ideal choice in a context where the robotic arms are supposed to replicate the human ones in the most realistic and intuitive way.

In order to perform a real-time control of the considered manipulator, it is necessary to acquire catalogue information related with the motion of each joint. In particular, the movements of the six joints are limited by their maximum operating speeds, which are in turn related with the power of their AC servo motors. In addition, in order to prefer some solutions of the inverse kinematics problem among others, it is necessary to consider the maximum motion range of each joint. In the following table, the maximum operating speeds and motion ranges are reported for each joint, together with the power consumptions of their motors and the brake equipment.

	Max. Operating Speed	**Max. Motion Range**	**Motor Power Consumption**	**Brake Equipment**
Joint #1	450 deg/s	[-180°,+180°]	400 W	Optional
Joint #2	450 deg/s	[-160°,+65°]	400 W	Standard
Joint #3	514 deg/s	[-51°,+225°]	150 W	Standard
Joint #4	553 deg/s	[-200°,+200°]	50 W	Optional
Joint #5	553 deg/s	[-135°,135°]	50 W	Standard
Joint #6	720 deg/s	[-360°,+360°]	50 W	Optional

Table 4.1: Physical limitations and characteristics of the six angular joints of the two arms.

In the standard configuration of the robotic arms, which is the setup utilised during the experimental phases, the brakes were present only on the second, on the third and on the fifth joint. The brakes are electromagnetic and in case they are ON it is possible to move manually the manipulator in a safe way.

One additional limitation to the motion ranges of the six joint angles is due to the necessity of avoiding any kind of interference between the two arms. In particular, in case it is attempted the execution of a movement that would cause an impact among them, the operations of the manipulator are restricted. It will in fact stop in the eventuality where the joint angles of one arm are within the grey areas represented in the following graphs.

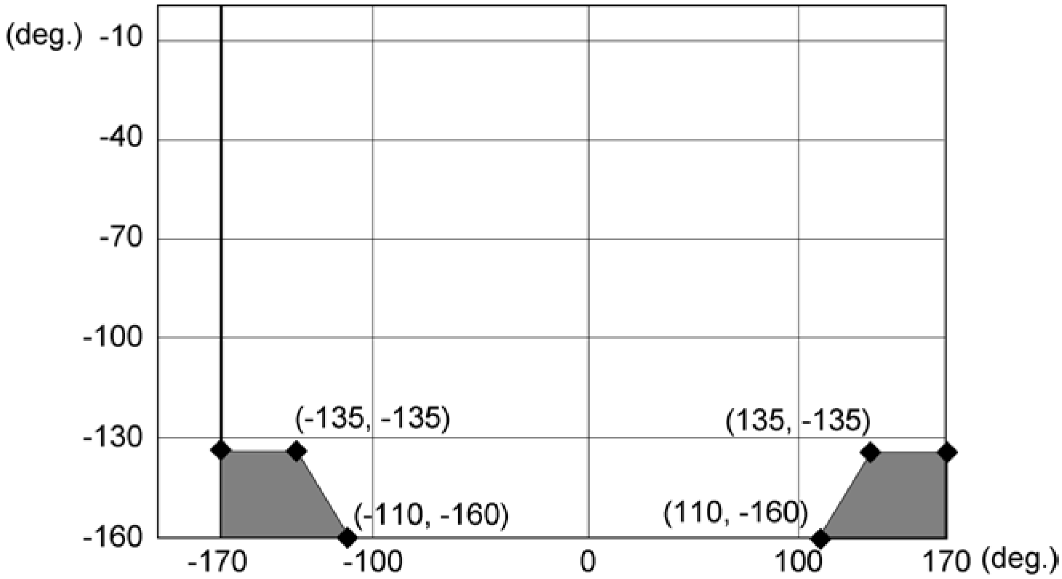


Figure 4.3: Combinations of Joint #1 and Joint #2 amplitudes, at which the manipulator stops because of possible impact between the two robotic arms.

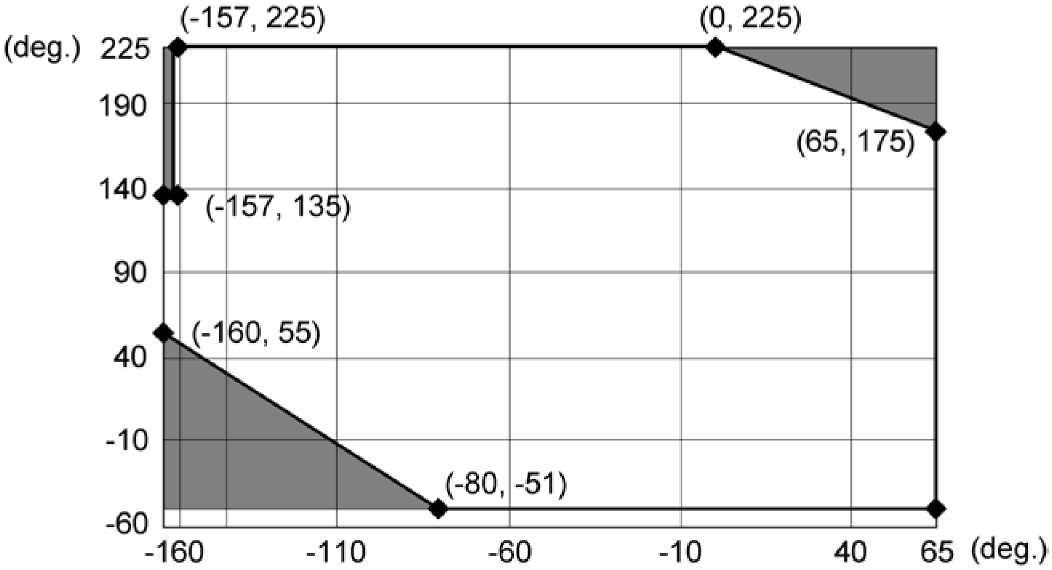


Figure 4.4: Combinations of Joint #2 and Joint #3 amplitudes, at which the manipulator stops because of possible impact between the two robotic arms.

The two arms composing the C3 manipulator, together with their bearing structure, are interfaced with the human users through a computer unit in which the controller and the software for commanding the robot are installed. The Leap Motion device is plugged in such computer and is placed at a safety distance with respect to the manipulator. From that position the human user is able to freely move his or her hands upon the sensor, commanding the motion. He is also able to visualize the control software and the graphic simulator through a monitor plugged in the main computer unit. The layout of the experimental environment is depicted in Fig.4.5.

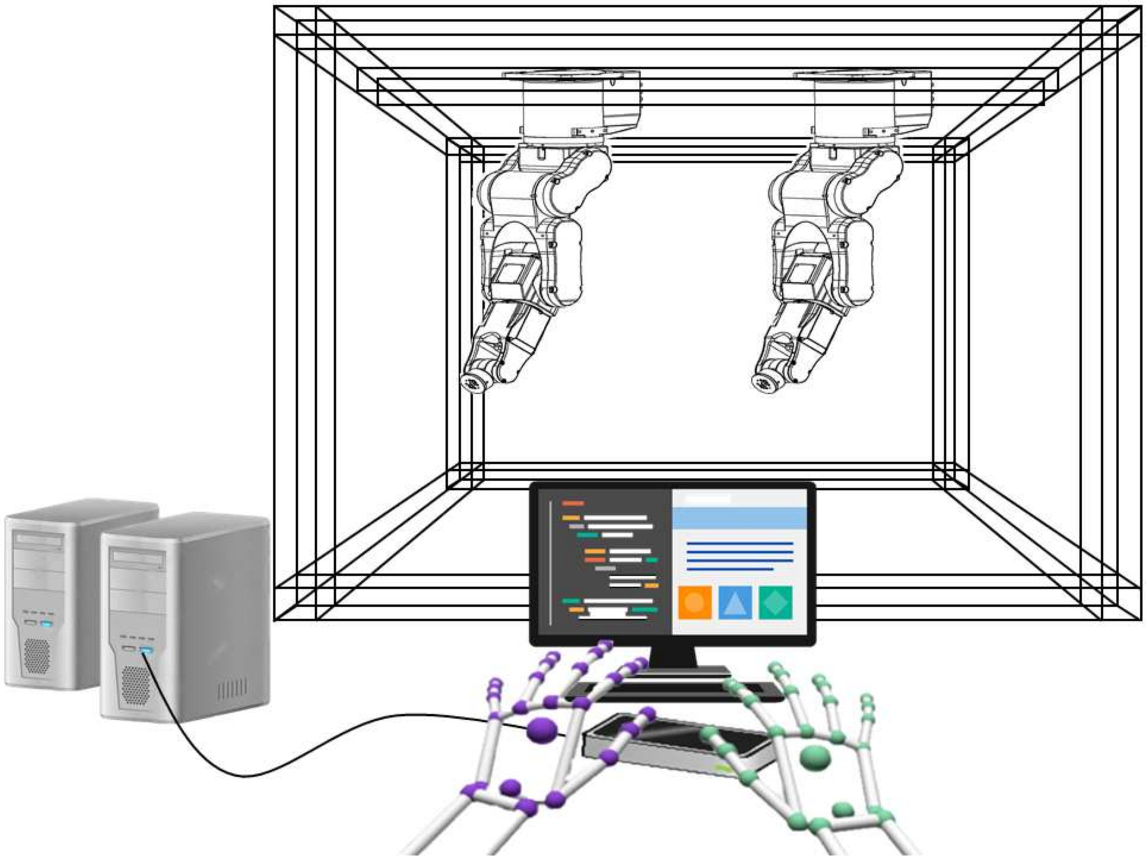


Figure 4.5: Graphic drawing of the way the hardware supply is disposed into the experimental space.

In Fig.4.5, the second computer unit, standing next to the main one, is needed to control the different tools that could possibly be mounted at the extremity of the robotic arms. It is in fact possible, depending on the specific applicative context, to equip the end-effectors with suitable tools, such as grippers or robotic hands. Also the control of such tools, can potentially be performed through hands motion, interpreting the number of the extended fingers or in general their position, as a mean to command the extremity-tools. However, such additional mode of control has not been developed here, but is let open for future works.

In order to define the kinematics of the manipulator, apart from the overall frames described in section 3.3, six additional reference systems have been attached to each link of the robotic arms, as depicted in Fig.4.6.

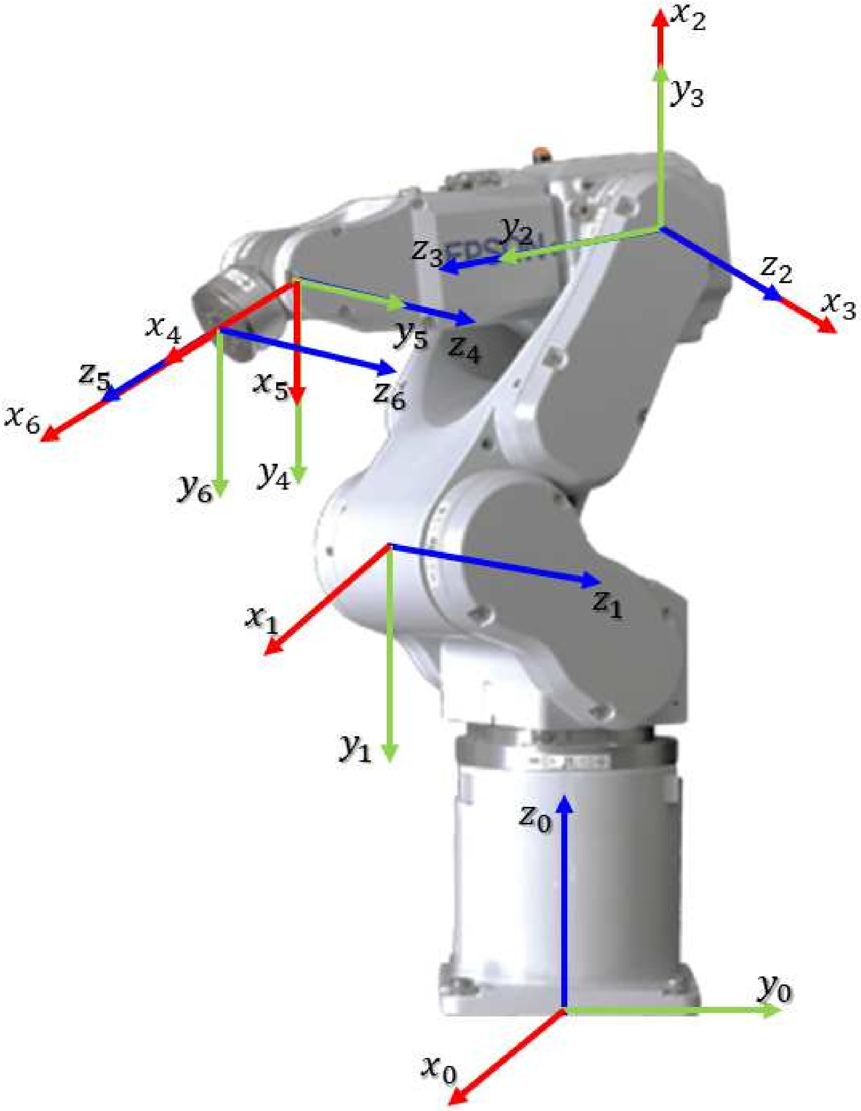


Figure 4.6: Reference systems of the six manipulator's joints.

The reference frames are defined according to the Denavit-Hartenberg convention in order to derive the position and orientation of each arm's link with respect to the preceding one. Each z_i axis is located along the axis of Joint $i+1$, putting the O_i origin at the intersection of axis z_i with the common normal to axes $z_{i-1}$and z_i. The relative x_i axis is located on such common normal with direction from Joint i to Joint $i+1$, while y_i is chosen as to complete a right-handed frame.

Once the link frames have been established, the position and orientation of Frame i with respect to Frame $i-1$ are completely specified by the Denavit-Hartenberg parameters. Such parameters, for one arm of the C3 manipulator, have been identified and reported in Table 4.2.

	$\boldsymbol{a_i}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$
Link #1	a_1	$-\frac{\pi}{2}$	d_1	θ_1
Link #2	a_2	0	0	θ_2
Link #3	0	$-\frac{\pi}{2}$	0	θ_3
Link #4	0	$+\frac{\pi}{2}$	d_4	θ_4
Link #5	0	$-\frac{\pi}{2}$	0	θ_5
Link #6	0	0	d_6	θ_6

Table 4.2: Denavit-Hartenberg parameters for a arm of the C3 manipulator

In such table the a_i and d_i values, representing the spatial distances among the reference frames, are refferred to the dimensions reported in Fig.4.1. According to the reported parameters, the forward kinematics matrix, expressing the pose of the manipulator's end-effector as function of the six joint variables, has been evaluated. Such matrix allowed subsequently at the solution of the inverse kinematics problem, as explained in section 3.6.

4.1.2 The Leap Motion

The Leap Motion controller is a small USB peripheral device that is capable at detecting the motion of human hands and fingers. It catches movements in the 3D space distinguishing among different trajectories and gestures. It is supposed to be placed on a physical desktop, facing upward, and it requires no kind of hand contact or touch.
The Leap Motion is considered to be a revolutionary technology in the field of optical sensor devices since it manages to track hands motion with a submillimetre accuracy. It allows in fact to perform a wide range of operations, such as the control of the computer's cursor or the interaction with the surrounding space in virtual reality, with a hundredth of a millimeter precision.

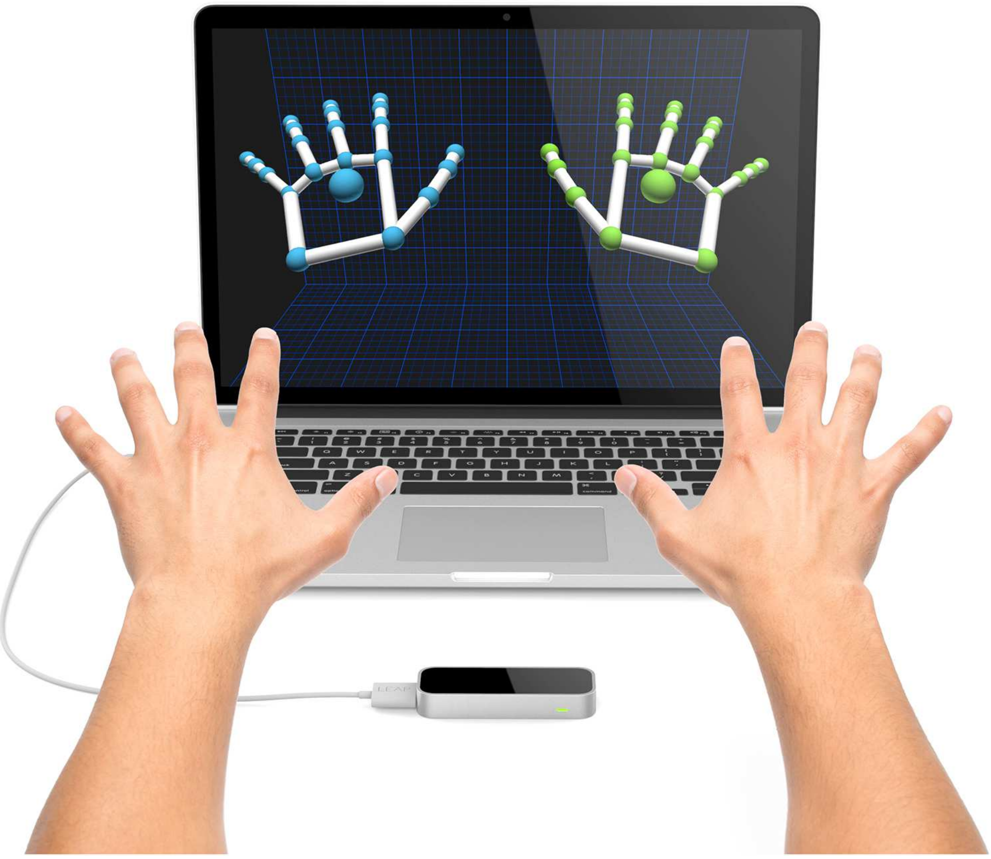


Figure 4.7: Example of use of the Leap Motion device.

There is a broad variety of practical applicative scenarios in which the Leap Motion device is currently being used. Its basic functioning consists in operations that are analogous to the ones performed by a common mouse controller. It provides in fact the possibility to comfortably navigate an operating system or to browse Web pages through finger's flick and to zoom in on maps and pictures trough finger pinching. Anyway, apart from this basic usage, the Leap Motion controller has also been employed by surgeons and researchers for medical software [35], automotive companies for concept cars [36] and musicians for composition [37]. Virtual Reality, in addition, is a field of application suiting particularly well the Leap Motion, considering that the device is being mounted on VR headsets in order to gasp commands from user's hands, as shown in Fig.4.8.

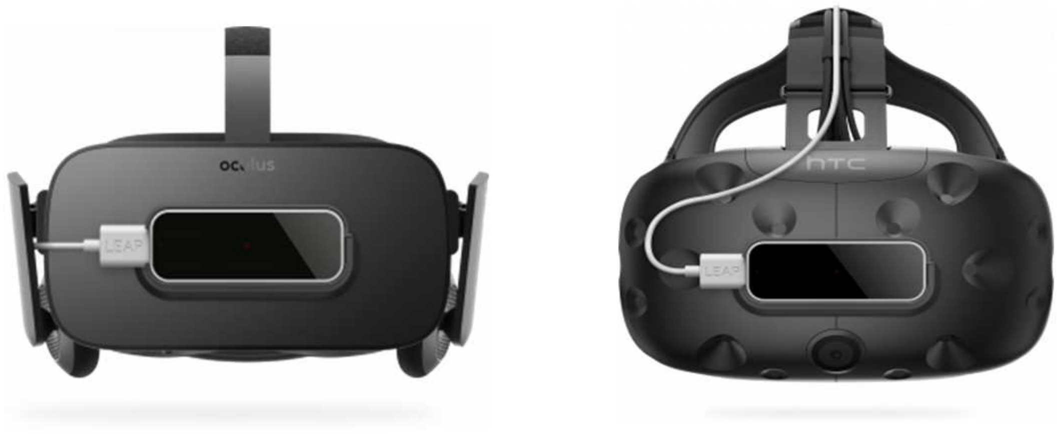
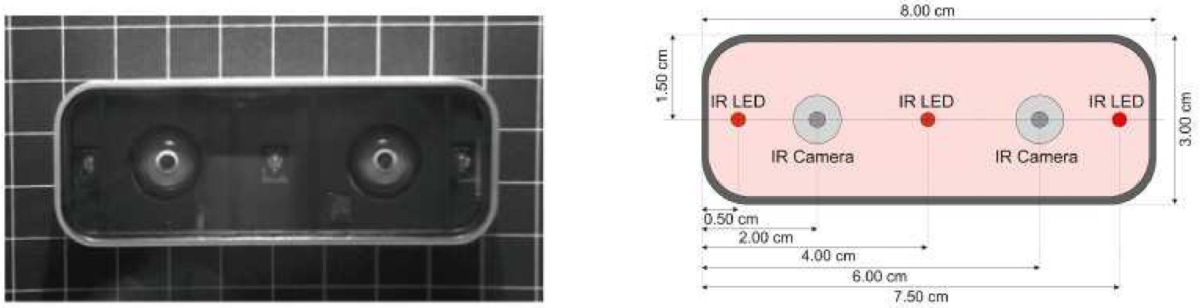


Figure 4.8: Leap Motion device mounted on a Oculus Rift VR headset (a) and on a HTC Vive headset.

Since 2013, the Leap Motion Inc. launched an app store called *Airspace*, where innovative apps, requiring the use of the optical sensor, are being sold. The apps are all thought and realised by developers and as of May 2014 the store had over 200 of them, the most popular of which is a Google Earth integration [38]. Recently published applications for 3D hand's tracking include gesture-based user interfaces, virtual clay sculpting and many types of gaming.
These applications have different requirements in terms of resolution, speed, distance and target characteristics. In particular, with regard to the teleoperation of the dual-arm manipulator, the accuracy of the optical device is crucial. The Leap Motion device has been considered in this specific applicative context taking into account its declared sub-millimeter precision at tracking hands' motion. In a 2013 study, a standard deviation between the desired 3D position and the average measured positions below 0.2 mm has been obtained for static setups [39].

Such levels of accuracy are obtained through an innovative technological setup. The Leap Motion controller is in fact composed of three infrared lights emitters (LEDs) and two monochromatic infrared cameras at high resolution, as shown in Fig.4.9.



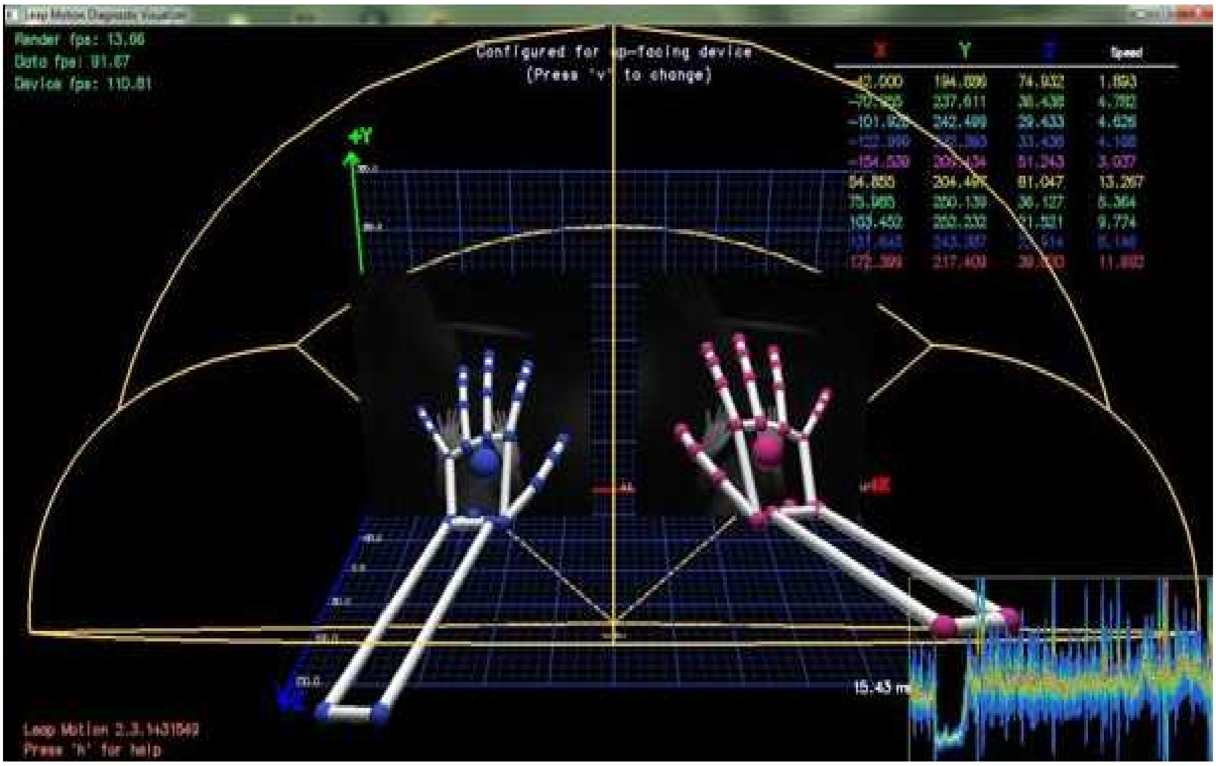
(a) (b)

Figure 4.9: Inside view of the sensors composing the Leap Motion device (a). Graphical scheme of the disposition of the sensors inside of the device (b).

It manages to observe a roughly hemispherical area, with the shape of an inverted pyramid, to a distance of about 1 meter, according to Fig.3.5. The three LEDs generate a pattern-less infrared light while the cameras can catch up to 200 frames per second of reflected data. The actual frame rate of the device ranges between 20 and 200 frames per second depending on the user's settings and the available computing power [40]. The data caught by the cameras and the LEDs is then sent through a USB cable to the host computer, where the Leap Motion software elaborates it in order to synthesize 3D pose information.
The precise mathematics underlying the data processing have not been disclosed by the company, but they are based on the stereo vision concept. Such concept uses the comparison of the 2D frames generated by the cameras in order to determine the depth in the observed scene. By searching points of correspondence among the two images, the position of predefined markers in the Cartesian space is detected.
The Leap Motion is a relatively new technology considering that it has first been realized to the public in 2012. Up to that time the most popular motion sensing device was the Microsoft Kinect, which is more suitable for whole-body tracking in a space the size of a living room. On the other side, the Leap Motion provides a smaller observation area and a higher resolution. According to a study conducted in 2012, the Kinect controller shows a standard deviation in depth accuracy of approximately of 1.5 cm [41], which is a very large number with respect to the sub-millimeter one of the Leap Motion. That explains the choice of using such optical device to track hands motion. The Leap can sense motion down to the most subtle movements of a finger, which the company says is 200 times more sensitive than anything else on the market.

What makes the Leap Motion a highly valuable device, especially in the research field, is its intrinsic open-software nature. Its company has in fact promoted the developing of many new suitable applications by providing libraries and other precious coding resources that would allow for the best exploitation of the sensor. After the purchase of the device is in fact possible to install the Leap Motion SDK (Software Developer Kit), which not only allows at using the basic applications, but provides also a simulation environment

and a continuously updated API (Application Programmer Interface).
The simulation environment is shown in Fig.4.10. It allows to perform basic diagnostics as well at evaluating the interaction volume by showing a graphic rendition of the hands motion. In particular, it displays hands' tracking information, including labelled axes, field of view, current framerate, processing latency and finger tip coordinates and speed.



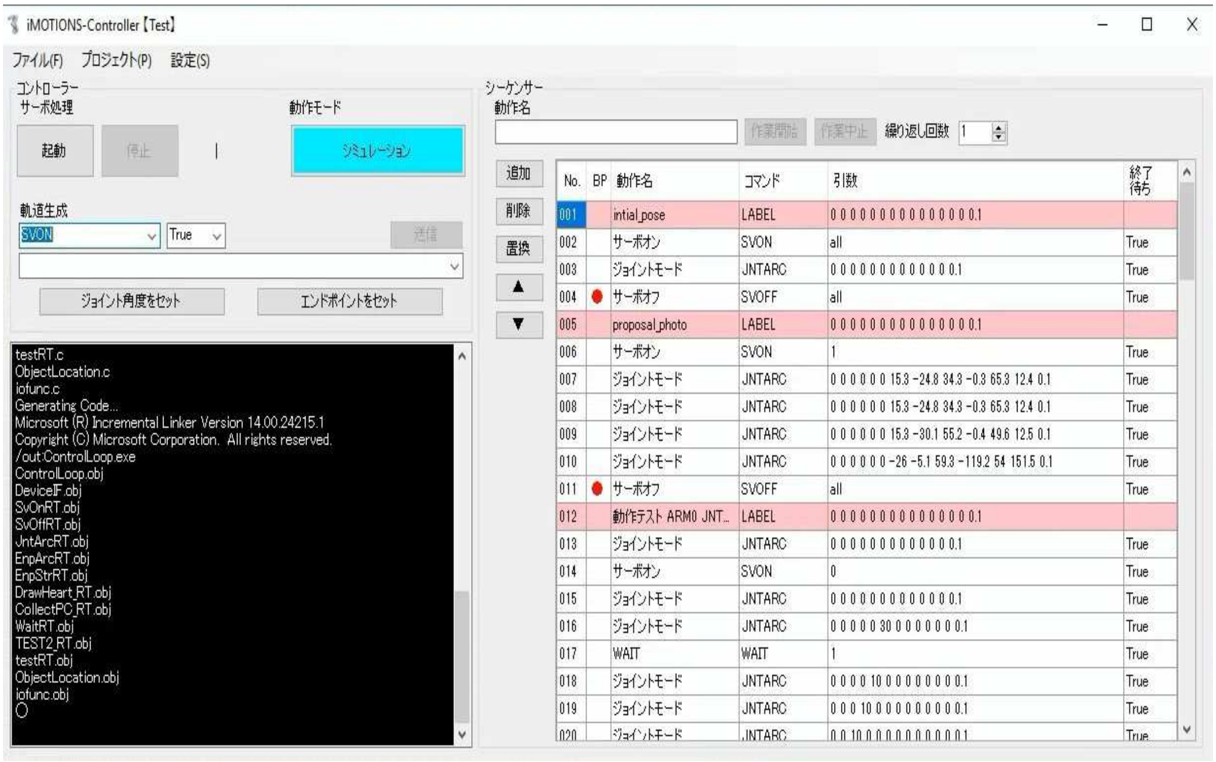
```
Render fps: 13.06
Data fps: 91.67
Device fps: 110.81
Configured for up-facing device
(Press 'v' to change)
Leap Motion 2.3.1431549
Press 'h' for help
```

Figure 4.10: Simulation Environment provided by the Leap Motion SDK in order to perform diagnostic evaluations.

The SDK comprises also a very useful API, which delivers libraries in many different programming languages, allowing for useful operations such as gesture recognition and mathematical elaborations of detected data. In the considered case, in which the captured data are used to teleoperate the robotic arms, the public developers functions were used to distinguish among left and right hands, to obtain their orientation and to interpret position and speed information in different mathematical structures. The API was furtherly used to perform advanced diagnostic and to recognize the number of extended fingers.

4.2 Software Interface

The control of the considered Epson C3 manipulator is performed trough a customized software interface, named iMOTIONS. This software platform allows for the communication between the human operator, using a Windows-based computer unit, and the robotic arms, which are commanded through the INTime mediation. INTime is a real time operating system (RTOS) conceived to run time-critical operations with cycle-times as low as $50\mu s$. It runs concurrently with a general-purpose operating system, such as Windows, on a single hardware platform ensuring for efficient use of the system. The INTime kernel is responsible for realizing a fast and safe communication between the robotic system and the shared memory where the elaborated data is saved.
The iMOTIONS software presents a main controller window as the one shown in Fig.4.11.



軌道生成: SVON True

繰り返し回数 1

```
testRT.c
ObjectLocation.c
iofunc.c
Generating Code...
Microsoft (R) Incremental Linker Version 14.00.24215.1
Copyright (C) Microsoft Corporation.  All rights reserved.
/out:ControlLoop.exe
ControlLoop.obj
DeviceIF.obj
SvOnRT.obj
SvOffRT.obj
JntArcRT.obj
EnpArcRT.obj
EnpStrRT.obj
DrawHeart_RT.obj
CollectPC_RT.obj
WaitRT.obj
TEST2_RT.obj
testRT.obj
ObjectLocation.obj
iofunc.obj
O
```

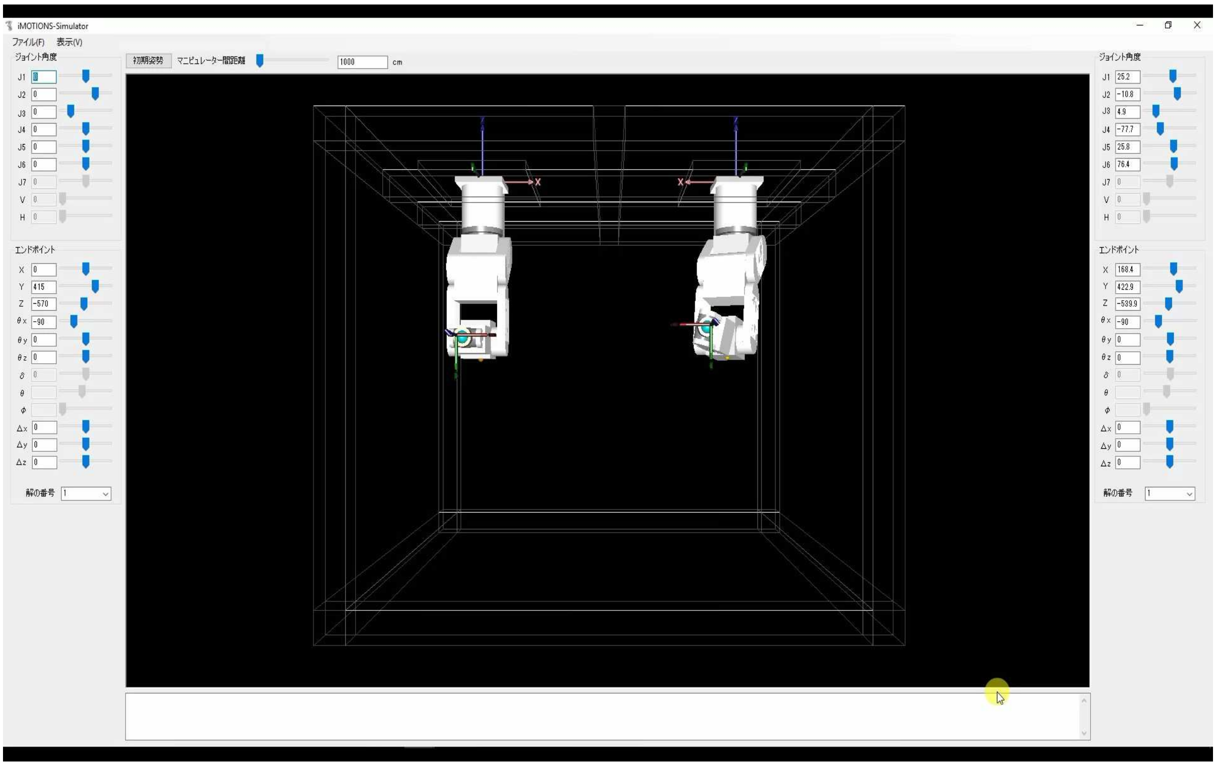
No.	BP	動作名	コマンド	引数	終了待ち
001		intial_pose	LABEL	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.1	
002		サーボオン	SVON	all	True
003		ジョイントモード	JNTARC	0 0 0 0 0 0 0 0 0 0 0 0 0.1	True
004	●	サーボオフ	SVOFF	all	True
005		proposal_photo	LABEL	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.1	
006		サーボオン	SVON	1	True
007		ジョイントモード	JNTARC	0 0 0 0 0 0 15.3 -24.8 34.3 -0.3 65.3 12.4 0.1	True
008		ジョイントモード	JNTARC	0 0 0 0 0 0 15.3 -24.8 34.3 -0.3 65.3 12.4 0.1	True
009		ジョイントモード	JNTARC	0 0 0 0 0 0 15.3 -30.1 55.2 -0.4 49.6 12.5 0.1	True
010		ジョイントモード	JNTARC	0 0 0 0 0 0 -26 -5.1 59.3 -119.2 54 151.5 0.1	True
011	●	サーボオフ	SVOFF	all	True
012		動作テスト ARM0 JNT...	LABEL	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.1	
013		ジョイントモード	JNTARC	0 0 0 0 0 0 0 0 0 0 0 0 0.1	True
014		サーボオン	SVON	0	True
015		ジョイントモード	JNTARC	0 0 0 0 0 0 0 0 0 0 0 0 0.1	True
016		ジョイントモード	JNTARC	0 0 0 0 0 30 0 0 0 0 0 0 0.1	True
017		WAIT	WAIT	1	True
018		ジョイントモード	JNTARC	0 0 0 0 10 0 0 0 0 0 0 0 0.1	True
019		ジョイントモード	JNTARC	0 0 0 10 0 0 0 0 0 0 0 0 0.1	True
020		ジョイントモード	JNTARC	0 0 10 0 0 0 0 0 0 0 0 0 0.1	True

Figure 4.11: Controller window of the iMOTIONS software.

Through this window, in the left section, is possible to control the motion of the manipulator by starting or stopping the servo processing and by selecting a specific trajectory. The trajectory generation happens by choosing a certain manipulator's command among the available ones. These commands can both be user-defined, such as the Leap Motion one, or predefined, such as the ServerON and ServerOff ones. Among the pre-installed trajectory generating commands it is possible to choose one of the followings:

- *JointArc*, which allows to directly specify the desired joint angles amplitudes, connecting consecutive points with a second-order polynomial.
- *EnpStr*, which takes the desired endpoint poses as input argument, interpolating among them through a first-order polynomial.
- *EnpArc*, which is also inputted with the requested end-effector poses but interpolates through second-order curves.

The desired input arguments can be manually specified in the dedicated space under the definition of the command name or they can be selected among some standard ones.
Furtherly, it is possible to define sequences of commands to be sent to the manipulator through the right section of the Controller window of the iMOTIONS software. Each sequence, labelled with a specific name reported in the red line, can either start with a ServerON or with a request to the reach the initial position (in such case the server was left ON from previous operations). They usually end with a ServerOFF command in order to avoid any possibly dangerous situation. For each line of a certain sequence, not only the input arguments, but also the label and the will of waiting for completion must be specified. Through this right section of the window is possible to start or stop the execution of a sequence and to choose the number of its desired repetitions.
In addition, the blue box in Fig.4.11 allows at selecting an operating mode among the "Real Machine" and the "Simulation" ones. It is in fact possible to test the functioning of a specific command sequence or of a newly defined command, in a dedicated simulation environment, that is shown in Fig.4.12.



マニピュレーター間距離 1000 cm

ジョイント角度
J1 0
J2 0
J3 0
J4 0
J5 0
J6 0
J7 0
V 0
H 0

エンドポイント
X 0
Y 415
Z -570
θx -90
θy 0
θz 0
δ 0
θ
ϕ
Δx 0
Δy 0
Δz 0
解の番号 1

ジョイント角度
J1 25.2
J2 -10.8
J3 4.9
J4 -77.7
J5 25.8
J6 76.4
J7 0
V 0
H 0

エンドポイント
X 168.4
Y 422.9
Z -539.9
θx -90
θy 0
θz 0
δ 0
θ
ϕ
Δx 0
Δy 0
Δz 0
解の番号 1

Figure 4.12: Simulation window of the iMOTIONS software.

Whenever the command inputs are sent in simulation mode they will be executed in a graphic animation where the two robotic arms are shown together with their bearing structure and their coordinate systems. The animation is as similar as possible to the effective movement that would arise from the input arguments. As a matter of fact, not only the joints of the two arms execute the same kind of movement, but also the time spent to complete the simulation is the same that would be required on the real machine. In the simulation window is also possible to read important data regarding the motion of the manipulator. For each arm the amplitude of the six joint angles is shown together with the endpoint's position and orientation, expressed in terms of Euler angles. It is also possible to observe the graphic animation from different points of view in order to extensively evaluate the manipulator's motion. Besides that, the bases of the two arms can also be moved at different mutual distances in a way to consider different configuration and safety limits. In case a specific setup, combined with a certain motion's input, can cause a collision with the surrounding space, a danger notification will be displayed and the relative motion of the real machine will not be executed.

5 Motion Outcomes

The teleoperation of the Epson C3 dual-arm manipulator has been tested in a Laboratory environment, where the experimental setup described in chapter 4 has been prepared. In particular, the Leap Motion device has been plugged in the main computer unit and located in front of the human operator standing at the control monitor.

The configuration chosen for the practical tests is then the one where the user and the robot are placed one in front of the other during the communication. As explained in chapter 2, it would be possible to perform the real-time control by hands' tracking also when the human is standing behind the manipulator. In such case it would be achieved a direct correspondence among the right human hand controlling the right robotic arm and the left human hand commanding the left robotic arm. However, in order to supervise the robot operations in a more intuitive way, the face-to-face configuration is here preferred for the testing purposes, even if in many applicative cases the opposite configuration could also be used.

The main goal of the experimentation phase has been the evaluation of the research objectives, defined prior to the implementation of the human-robot interface and already introduced in chapter 2.

5.1 Intuitiveness

One of the primary reasons why the Leap Motion device has been interfaced with the manipulator, is the will of realising a more intuitive method of control of the robotic system. The aim is to obtain a reduced learning curve so that the manipulator can be successfully controlled even if the operator is not an expert.

During the experimental sessions, the manipulator's arms have been successfully commanded by different human operators. None of them was involved with the developing of the Leap Motion control platform and none of them needed a specific training in order to understand the way the system works. All the different users managed to freely move the robotic arms along the desired trajectories and to impose also the desired end-effectors velocities and orientations. That was possible considered the relatively high order of the polynomial used to interpolate consecutive command inputs. It ensured in fact the realization of a really smooth motion, interpreting also the most impulsive hand gestures with soft and safe movements. In this way the users were led at understanding quickly and with minimum effort the way the control system works. In fact they only needed to focus on the task to be completed rather than distracting their selves by thinking about the specific motion of their hands.

5.2 Simultaneous Dual-Arm Control

One more objective that has been pursued during the developing of the Leap-based control infrastructure consistes in the possibility of teleoperating the two robotic arms simultaneously. The Leap Motion sensor, unlike most of the basic joystick controllers, allows in fact at defining the data about the desired position and orientation of the end-effector by the use of a single hand. In this way it is possible to have a one-to-one correspondence where only one hand is needed to control one arm of the manipulator. The simultaneous teleoperation is then feasible because the Leap Motion device is capable of acquiring the data from two hands at the same time, distinguishing among the left and the right one.
The simultaneous control of the two robotic arms has not been tested on the real machine but in the simulation environment exclusively. This is due to a hardware malfunction on one of the robotic arms that has prevented for a certain amount of time the use of both links at the same moment. However, this problematic didn't affect the processes of data acquisition and elaboration so that it was possible to test how the system behaves while handling the information coming from the two hands at the same time.
In particular, it was necessary to assert if the control platform was able at bearing an amount of data that is doubled in size.The simultaneous teleoperation results in fact in being much more demanding by the computational point of view, considering that challenging calculations, such has the inverse kinematics and the trajectory interpolation, must be performed twice per frame.
In addition, it is mandatory to make sure that the Leap Motion sensor does not make confusion by swapping and mistaking one hand for the other. To do so, different trajectory of the human hands were tested, considering also special paths were the two hands would be overlapped for a certain amount of time. Even the unlike, but still dangerously plausible, eventuality in which the two hands would exchange position with respect to their natural standings was tested. Such situation could not be tested with the real machine, considering that possible impacts, and especially overlaps, are forbidden and then never accepted nor executed by the manipulator. However, this special trajectories are allowed, within certain limits, by the simulator, which permits in this way to test the robustness of the developed system also in the cases of potentially ambiguous input data.
The simultaneous control in real-time was successfully tested in the simulation environment. The doubled computational effort was well endured by the computer unit responsible for the elaboration of the data detected by the Leap Motion. No major difference was in fact noticed in terms of performance with respect to the tests conducted on the single-hand control configuration. For what concerns the aim of distinguishing among the two hands, no kind of confusion was ever made between he right and the left hands of the human operator. This was possible considering the innovative technology underlying the functioning of the Leap Motion device, which considers also previous frames in order to differentiate the trajectories of the two hands. In particular, it was also possible to flip one hand on its backside, in a way that the right hand could be confused with the left arm considering the opposite mutual disposition of the five fingers. Also this kind of probable, as well as hazardous, movement was handled with no kind of error by the system.

The numerous tests showed then that it is possible to have the two robotic arms simultaneously controlled, allowing for a wide range of bimanual tasks to be successfully performed.

5.3 Real Time Functioning

The manipulator's teleoperation interface has been developed with the aim of realizing an actual real-time control. Ideally, the robotic arms are supposed to move immediately as soon as the human hands start moving upon the Leap Motion controller, following the hand's trajectories with no kind of delay. However, according to the performed simulations, there is almost always a more or less significant time interval between the moment in which the human hands are in a certain pose and the time when the end-effectors reach such pose.

The temporal delay has been observed with respect to a testing session where only one arm of the real machine was commanded to move. In order to estimate such delay, for each frame captured by the Leap Motion, the number of computer cycles required for reaching the detected pose are counted. In some cases the desired pose has never been reached by the manipulator, which, in order to follow the hands trajectories, has followed a smoother path. In such cases the delay of the closest pose with respect to the desired one has been computed. The obtained data about the temporal delay are reported on the graph of Fig.5.1.

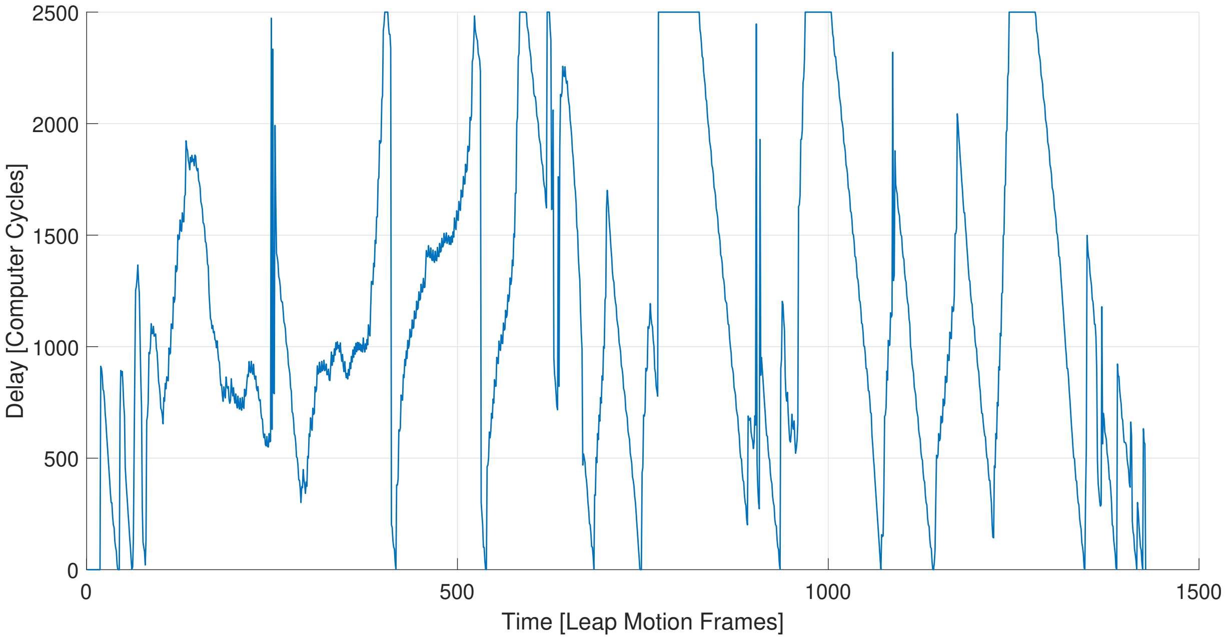


Figure 5.1: Temporal delay between the input command and the correspondant output motion. (1 Computer Cycle = $100\mu s$)

On the x-axis is reported the observation time, expressed in terms of frames of the Leap Motion sensor. Considering a Leap Motion frame-rate of 200 frames per second, the considered interval is of 7.5 seconds of the whole manipulator's motion. For each frame the relative latency has been reported on the y-axis in terms of computer cycles required to achieve the detected pose of the human hands. On average, each computer cycle corresponds to a time interval of $100\mu s$ because of the processor frequency being of 10kHz.

According to Fig.5.1, the highest peaks of delay reach a value of 2500 cycles of the computer unit, which in turn represents time intervals of a quarter of second, considering the average value of 50 cycles per frame. However, the average value of delay for the observed portion of the motion of the robotic arms is of 1184 computer cycles, as computed through the specific Matlab tool. Such value corresponds in turn to an obtained average latency of about 0.1 seconds. It is then possible to conclude that a good level of real-time approximation has been achieved for the developed system of teleoperation. It will be in fact possible to control the robotic arms even in cases where the user needs to interact with the surrounding environment with real-time requirements.

5.4 Accuracy

One of the main goals that were prefixed before working on the realization of the proposed system of human-robot interaction, was the will to achieve a highly precise method of teleoperation for the considered manipulator. In particular, it was required not only to read exactly the data relative to the hands motion, but it was also requested a faithful reproduction of the detected motion. The first objective has been satisfied immediately, by choosing as sensor unit the Leap Motion device, which ensures a submillimeter precision at tracking hands' motion. For what concerns the second goal, the data relative to the trajectory of the human hands was elaborated with the intention of obtaining an exact and specular copy of their motion.

In order to evaluate such objective, a comparison has been performed among the desired trajectory, detected by the sensor, and the obtained end-effector motion. In particular, for each frame captured by the Leap Motion, the desired end-effector position has been considered as computed through the mapping equations 3.11, 3.12 and 3.13. Such value has then been compared to the actual position obtained for the end-point, after interpolating between consecutive inputs at each cycle of execution of the Leap Motion command. The two considered kinds of data have been depicted on the graph of Fig.5.2 with reference to the z-axis position and to an experimental session on the real machine where only one arm was controlled through the motion of one hand.

From the figure is possible to see how the end-effector has been following the trajectory of the human hand in a precise way, but with evident differences. The discrepancies between the two lines of Fig.5.2 are partially due to the input-output latency that has been already discussed, but they depend mostly on the interpolation performed on the consecutive poses of the human hand. The red line, representing the trajectory of the robotic end-effector, assumes in fact a much smoother form as a result of the intention of chasing the user's hand with a polynomial of the third order.

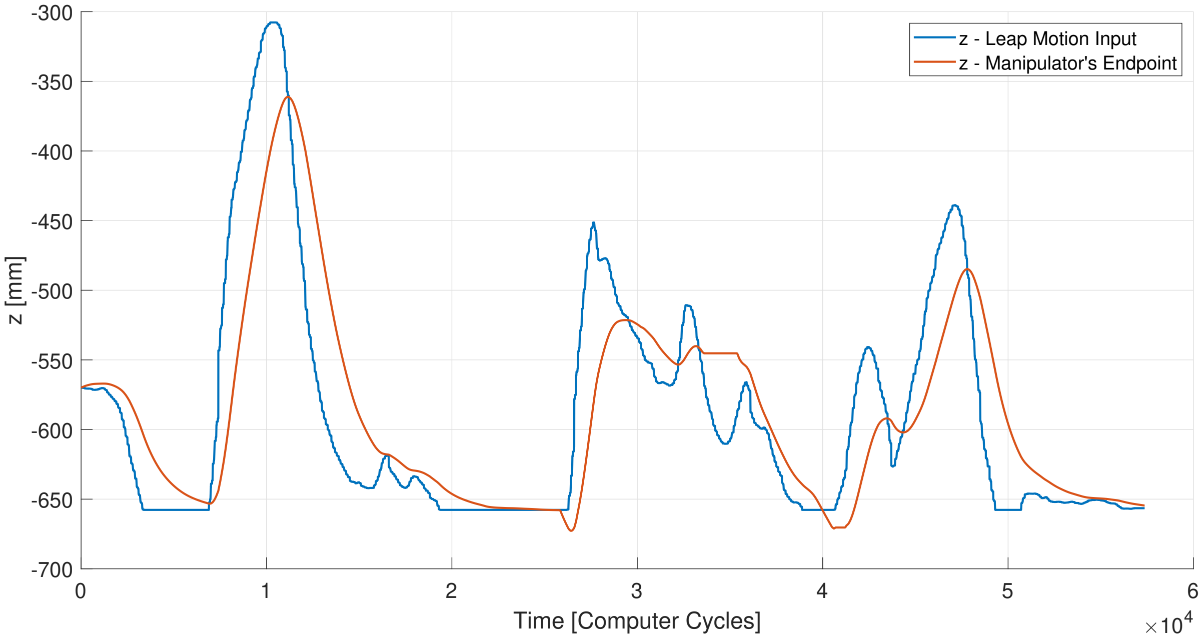


Figure 5.2: Comparison among the desired position on the z axis and the obtained trajectory of the end-effector.

A zoomed-in representation of the graph is reported in Fig.5.3, where a closer look is given to one of the peaks.

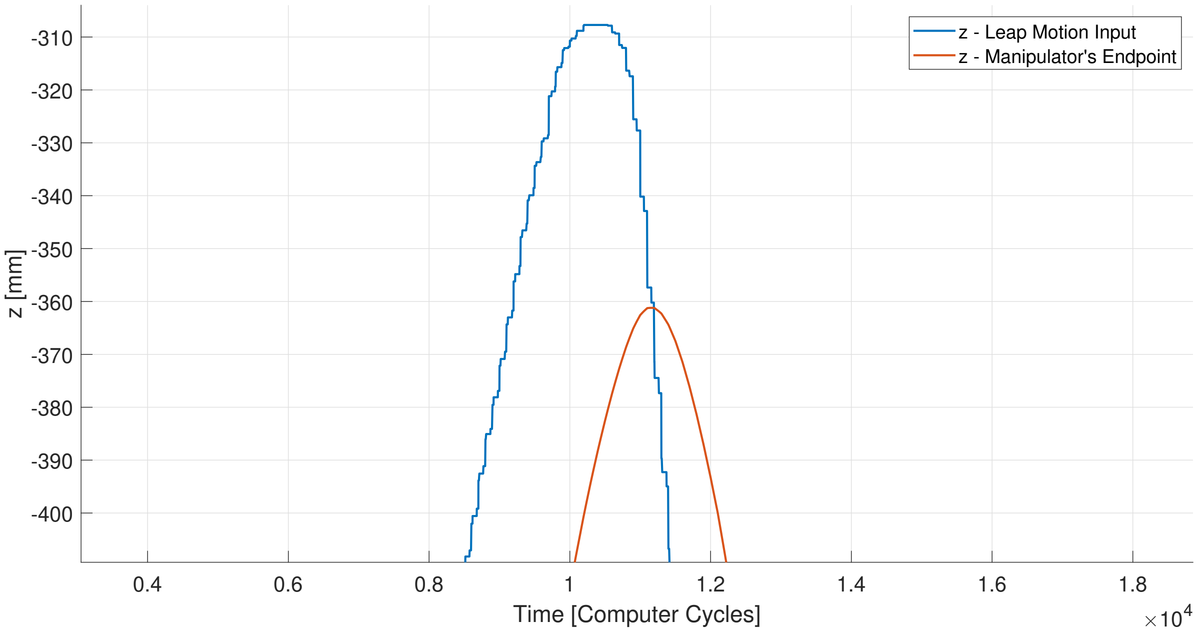


Figure 5.3: Zoomed-in comparison among the desired position on the z axis and the obtained trajectory of the end-effector.

From the figure is firstly possible to verify how the actual delay between the peaks (and also between the bases of the parabolas) is of about 0.1 seconds, in agreement with the average latency computed before. In addition, the zoomed plot allows to see how the input data is discreetly obtained with a frequency of about 200 frames per second. The obtained motion is instead represented by a continuous red line, which smoothly interpolates among the detected frames.

The zoom of Fig.5.3 is actually focused on the biggest distance noticed between the two lines of Fig.5.2. It is then possible to highlight how the highest error in the copy of the motion of the human hands corresponds to a value of about 50 mm. However the average value of such error has been computed of being not higher than 20 mm for all of the positional comparisons performed. The same confrontation of Fig.5.2 has in fact been done with reference to the x-axis and the y-axis data, as reported in Fig.5.4 and Fig.5.5.

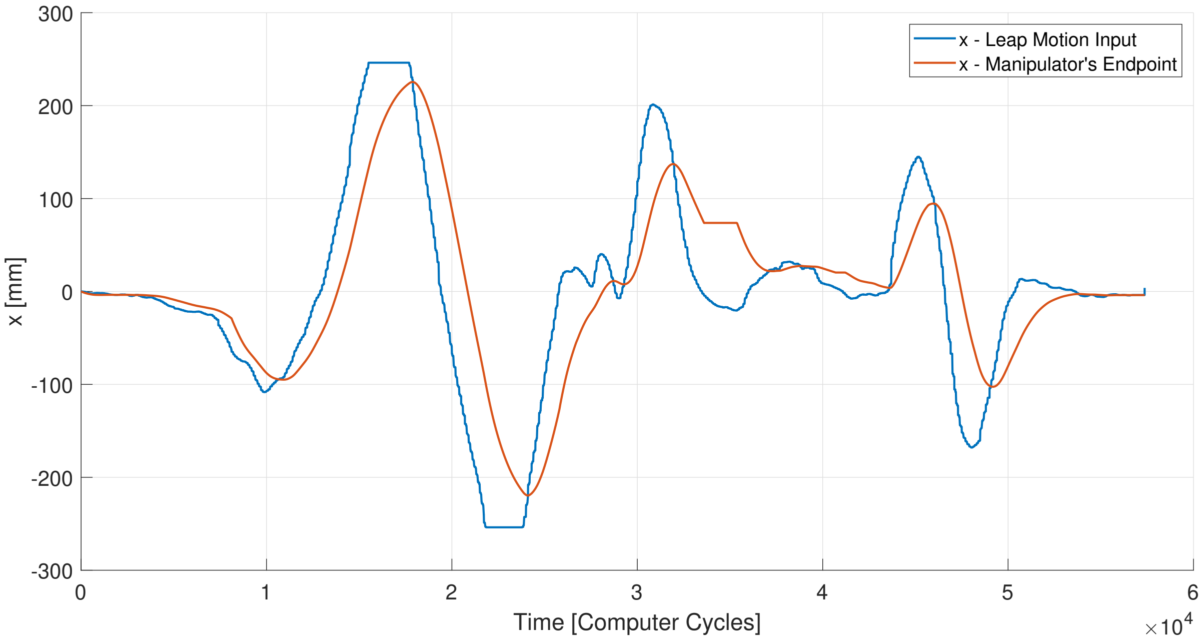


Figure 5.4: Comparison among the desired position on the x axis and the obtained trajectory of the end-effector.

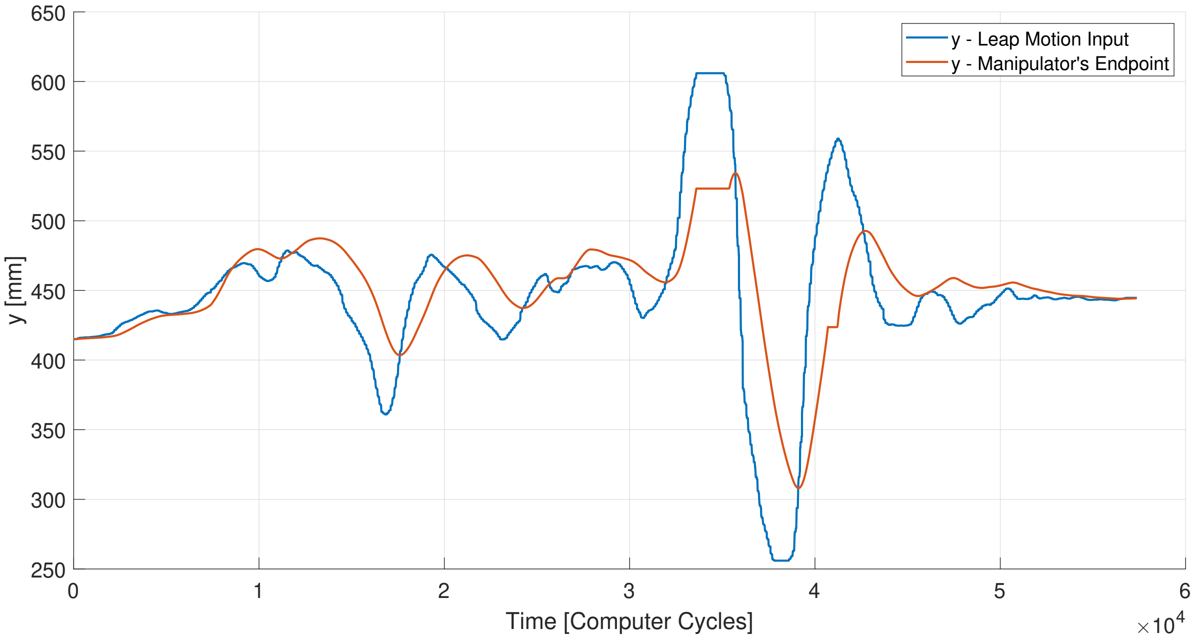


Figure 5.5: Comparison among the desired position on the y axis and the obtained trajectory of the end-effector.

A similar kind of comparison among the requested motion and the one actually obtained, has been performed for what concerns the orientation data as well. Also in this case it has been considered an experimental session where only one robotic arm of the real machine was moved. In particular, both the orientations of the human hand controlling the system and the one of the manipulator's end-effector were interpreted in terms of Roll, Pitch and Yaw angles and reported in Fig.5.6, Fig.5.7 and Fig.5.8.

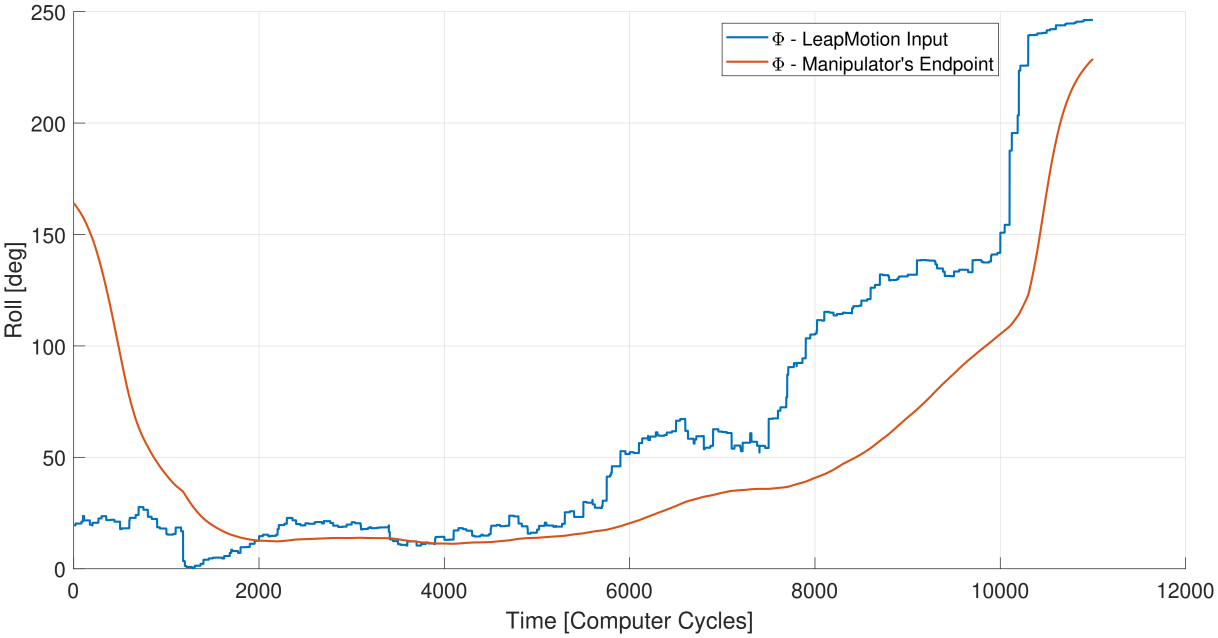


Figure 5.6: Comparison among the desired orientation and the one obtained for the end-effector, in terms of Roll angle.

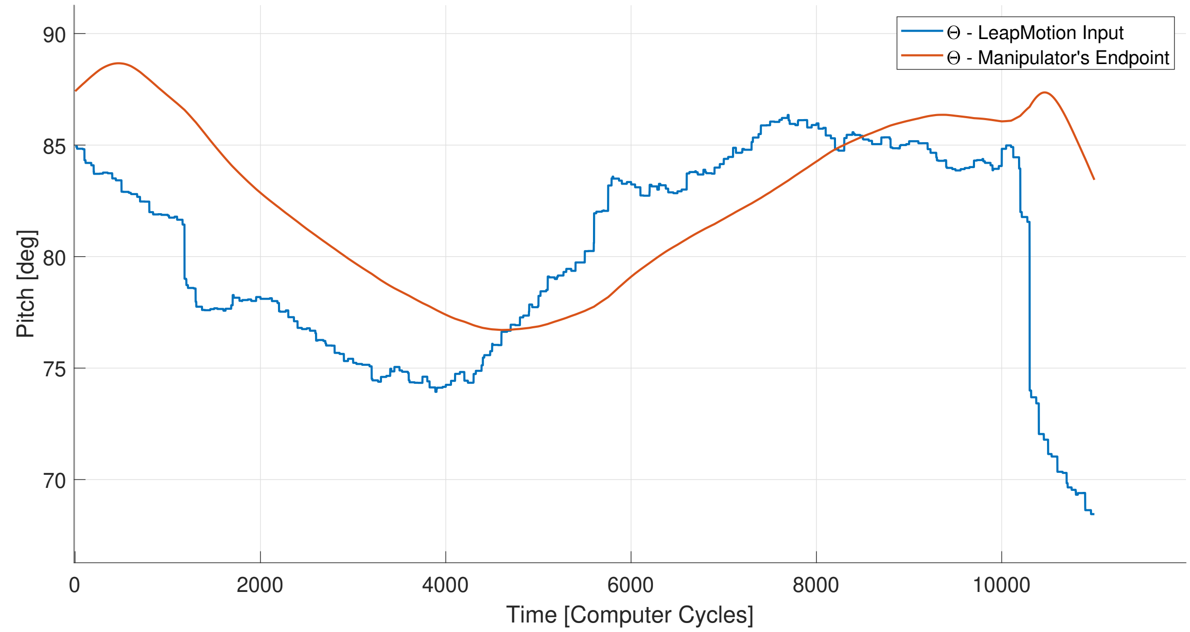


Figure 5.7: Comparison among the desired orientation and the one obtained for the end-effector, in terms of Pitch angle.

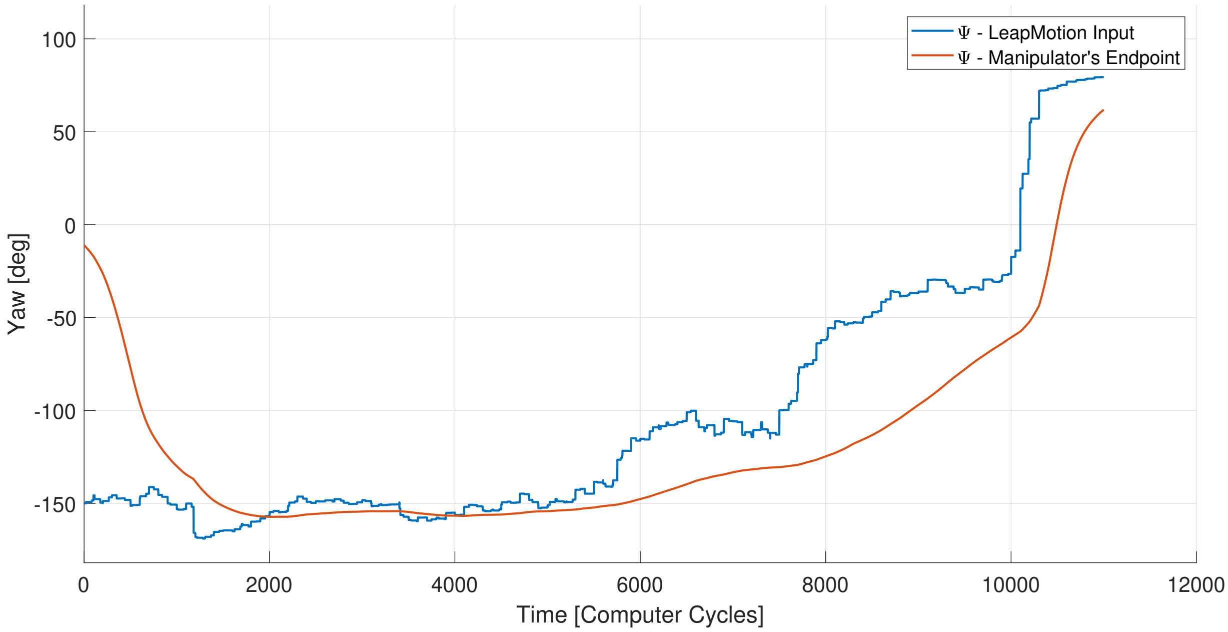


Figure 5.8: Comparison among the desired orientation and the one obtained for the end-effector, in terms of Yaw angle.

The time interval that has been observed during the considered real-machine test is relatively short, but it allows at evaluating the performance from a closer look. It is in fact possible to see how also in this case the obtained output motion smoothly interpolates among the input data. In all the three graphs, the red line, representing the resulting orientation of the end-effector, follows precisely the shape of the blue one. Nevertheless, significant differences can be spotted among the desired and the obtained motions. In particular, looking at the Roll and the Yaw angles (Fig.5.6 and Fig.5.8) it can be noticed how at the beginning of the test both angles showed their maximum error, which is standing around 150° of difference. However, it takes about 0.1 seconds for both of them to reach a level of approximation where the two lines never differ for more than 20°. The average accuracy error in terms of RPY orientation stands at about 3° for the Pitch and at about 16° for the Roll and the Yaw, as computed through the registered data.

In the following also the behaviours of the joint angles of one arm are shown in order to have one additional mean of evaluation of the obtained performance. Also in this case the time is reported on the x-axis in term of computer cycles, while the amplitudes of the joint angles are expressed in degrees on the y-axis. The blue line represents the desired values of the joints angles as computed through the inverse kinematics every time a new input frame is captured by the Leap Motion.
From all of the six graphs it can be noticed how the blue input lines present an impulsive variation in correspondence of the 12000 value on the axis of the abscissas. This is due to a configuration of the human hands that has led to the selection of the wrong solution of the inverse kinematics. In such case the optimization process, responsible for the choice of the best solution, has in fact picked a vector of joint angles that is in discontinuity with the previous values. Such vector, although being still valid in terms of admissible ranges of the joint angles, could worsen the performance or also lead to a dangerous kind of motion and must thus be ignored. Anyway, the third-order interpolation is effective also in this case, in the sense that it manages to determine a red output line, which chases the desired input without taking too much into account the unwanted input.

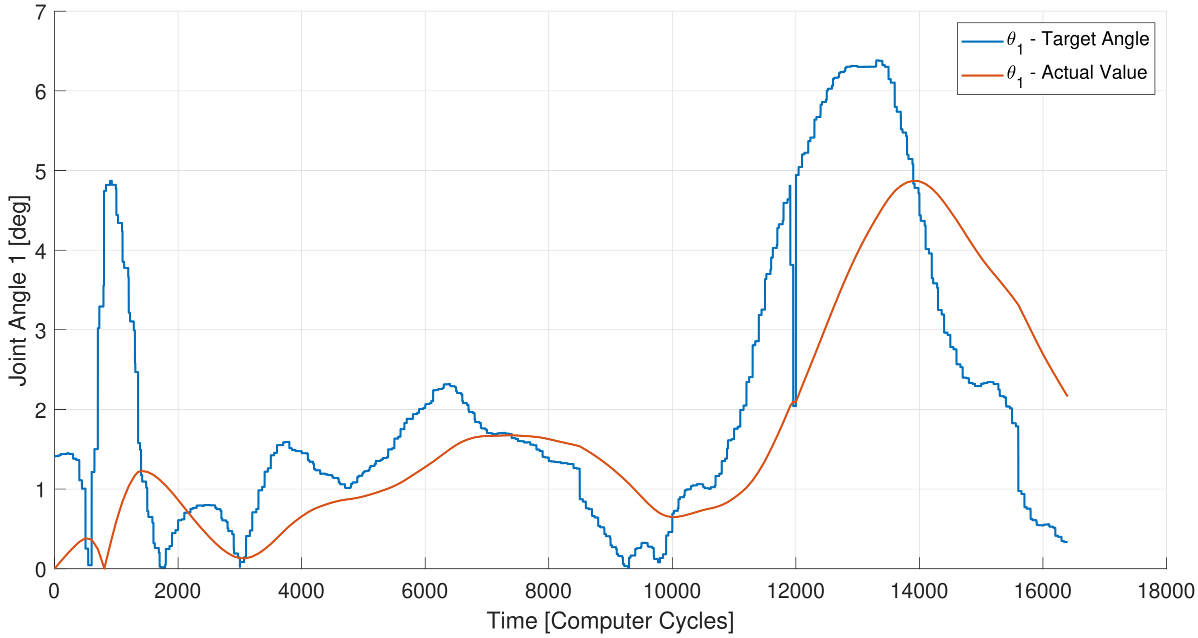


Figure 5.9: Comparison among the desired first joint amplitude and the one obtained on the robotic arm.

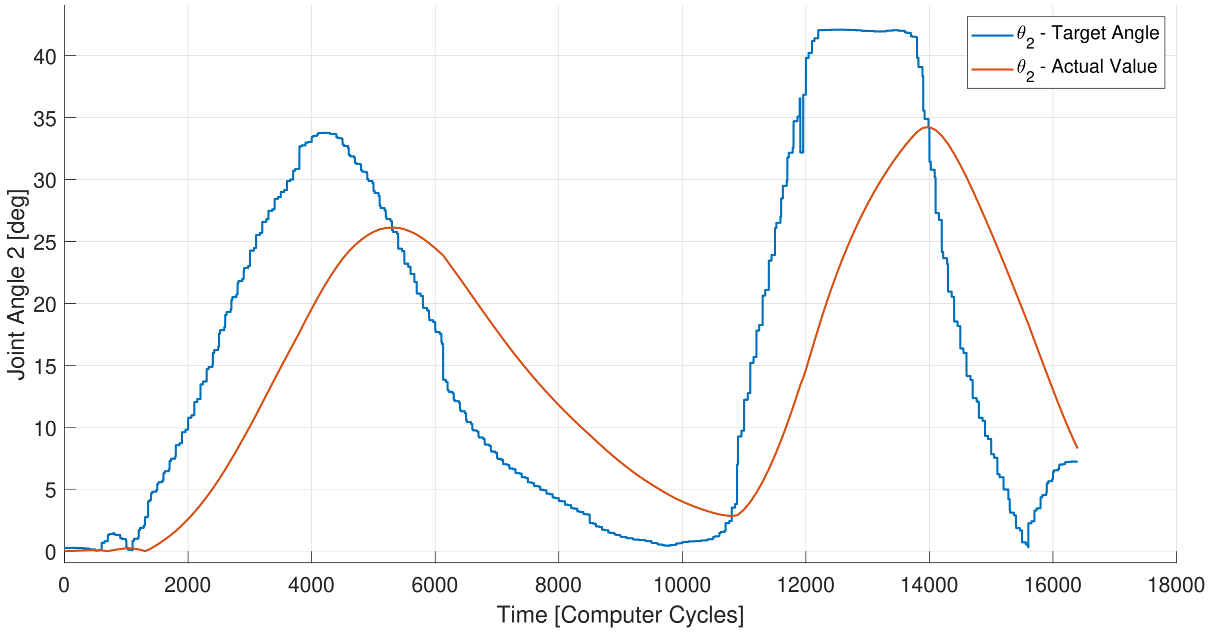


Figure 5.10: Comparison among the desired second joint amplitude and the one obtained on the robotic arm.

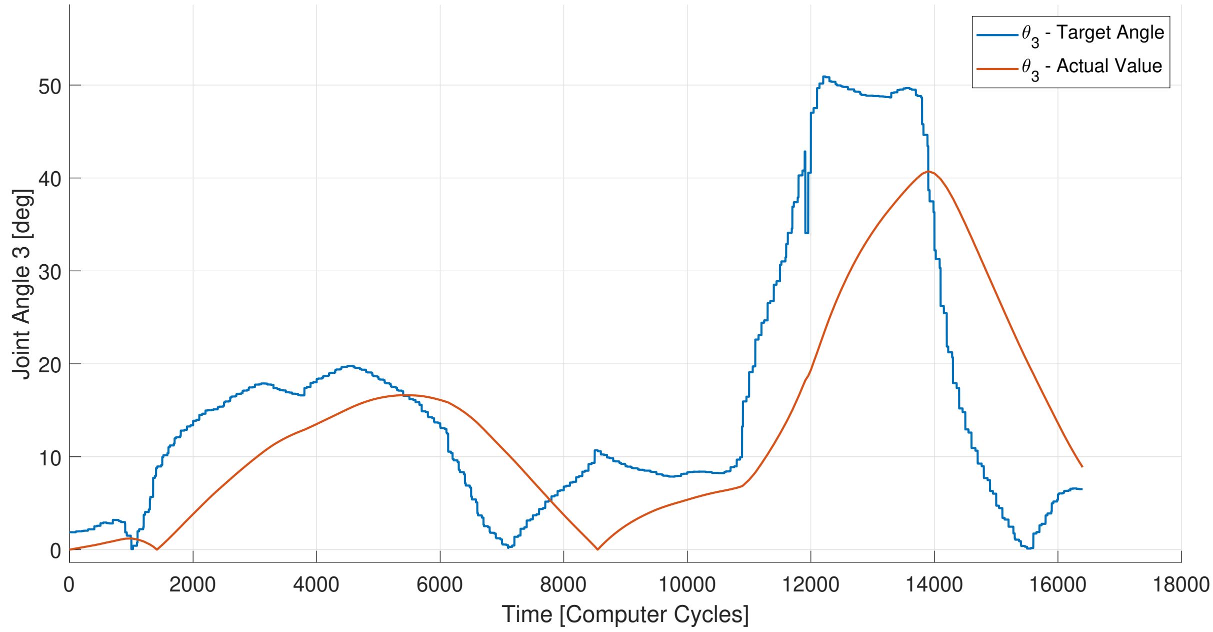


Figure 5.11: Comparison among the desired third joint amplitude and the one obtained on the robotic arm.

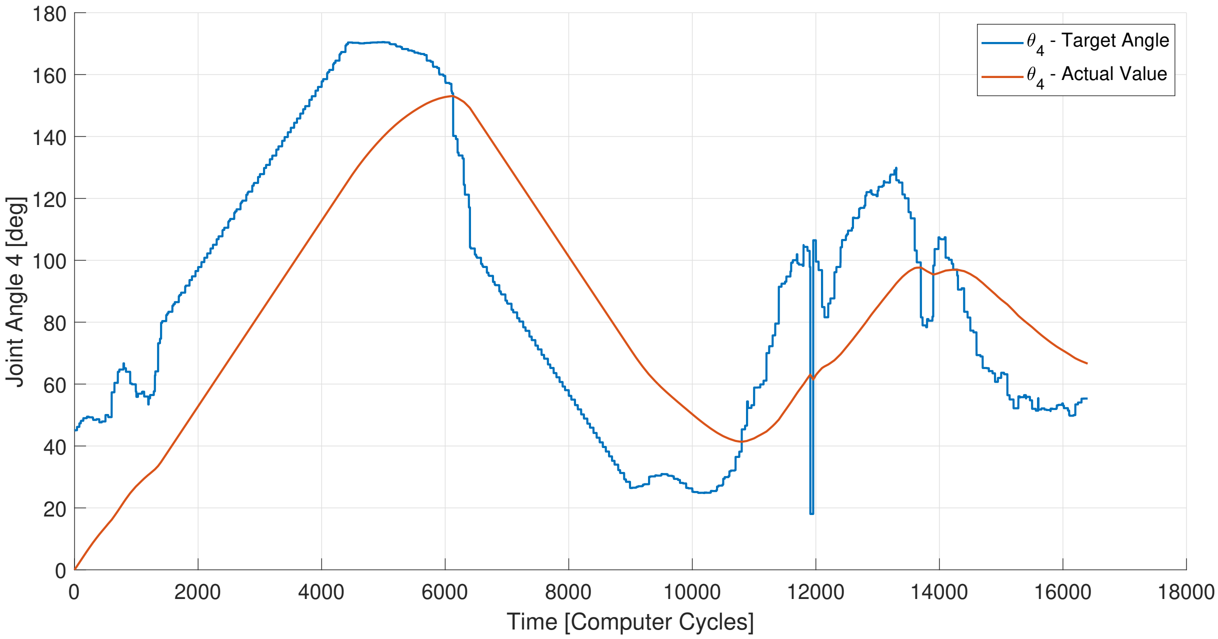


Figure 5.12: Comparison among the desired fourth joint amplitude and the one obtained on the robotic arm.

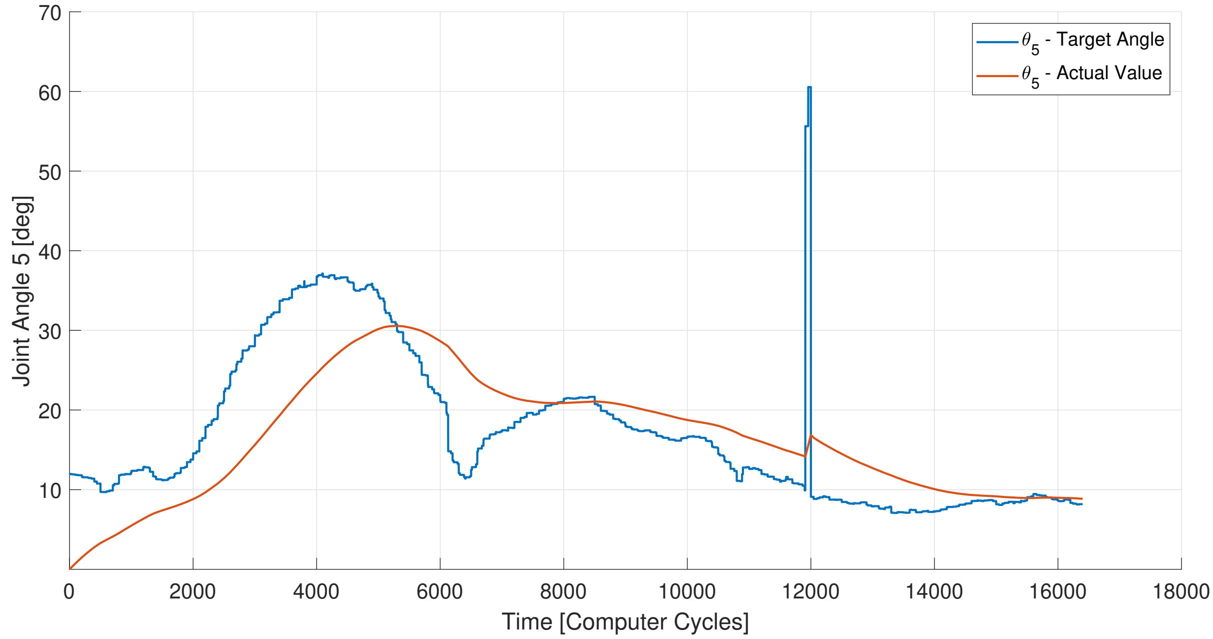


Figure 5.13: Comparison among the desired fifth joint amplitude and the one obtained on the robotic arm.

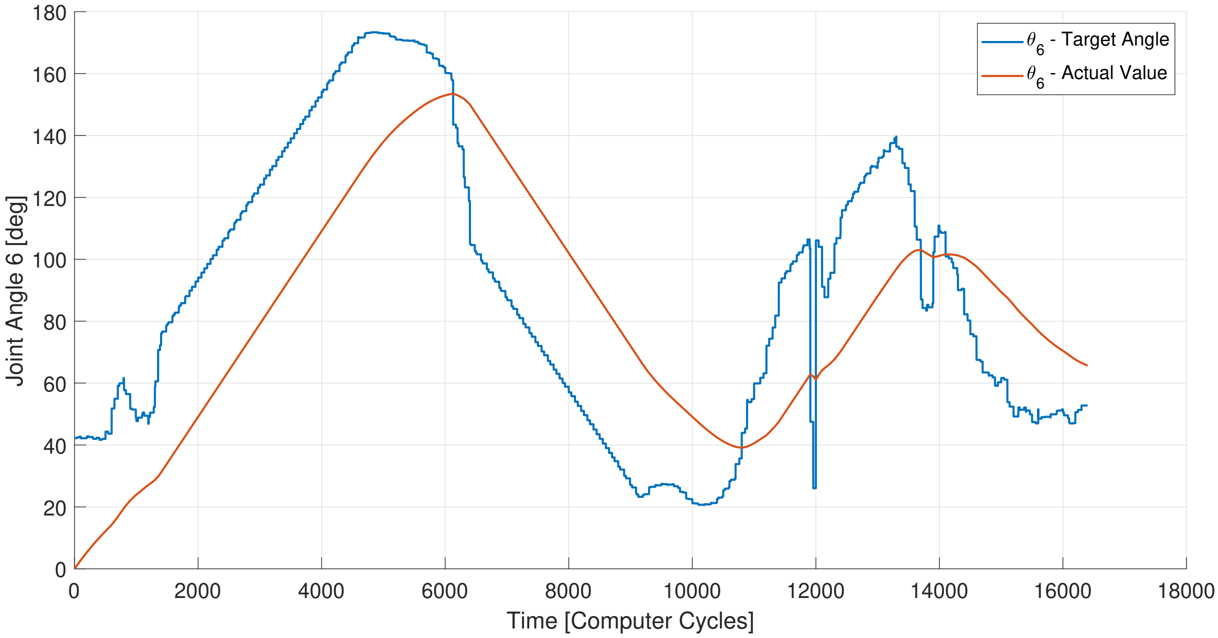


Figure 5.14: Comparison among the desired sixth joint amplitude and the one obtained on the robotic arm.

According to the collected data, which has been depicted in the previous graphs, the system controlled in real-time has been linearised. Practically, the linear model which better approximates the obtained input-output behaviour has been defined through the *System Identification* toolbox of the Matlab software. In particular the Leap Motion's inputs and the end-effector's outputs, in terms of x-axis coordinates, have been provided to the toolbox, which computed in turn the linearised system. Such system has been expressed in terms of state-space matrices first and in terms of closed-loop transfer function then. Its response in the frequency domain has been analysed through the bode diagram of Fig.5.15.

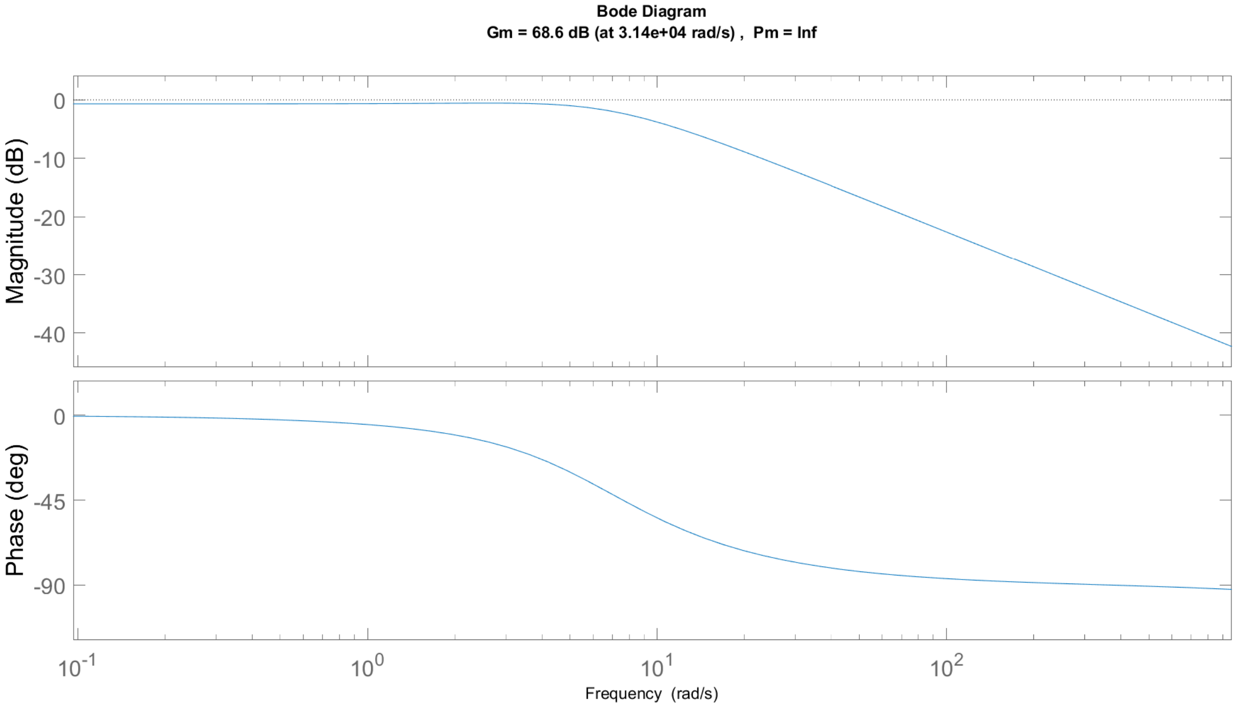


Figure 5.15: Bode diagram expressing the frequency response of the linearised system, which approximates the input-output behavior obtained for the x-axis position data.

From the diagram it is possible to evaluate the cutting frequency, and in turn the bandwidth of the system. Using the Matlab's *bandwidth* function, which returns the first frequency of the SISO dynamic system model where the gain drops below -3 dB, a value of 9.8 rad/s is obtained. Such bandwidth represents in turn the maximum ferquency of the input signal providing the x-axis coordinates that ensures a good performance of the control system.

6 Conclusions

In this chapter the main results obtained during the development of the proposed control infrastructure are presented. In particular, it will be discussed whether or not the main project objectives have been achieved, with a focus on the major encountered problematics. After that, some hints for possible future works will be proposed with the aim of achieving a more complete and exhaustive analysis of the considered topics.

6.1 Summary of the Obtained Results

The thesis work here presented aimed at introducing the possibility of teleoperating a dual-arm manipulator in real time by using the tracking of human hands as the only source of input command.

In the first place, the previous works related with the considered matter has been taken into account. They represented the primary source from which beginning the development of the desired system of human-robot interaction. After that, some project objectives have been preferred among others, in order to impose a specific direction to the work to be done, taking into account the possible applicative scenarios.

Subsequently, it has been presented the method adopted in order to have the data elaborated and communicated from the hands tracking sensor to the dual-arm manipulator. According to the available software and hardware supply, a specific protocol has been defined to interface the different parts of the system. Three different processes are in fact executed at the same time, using a shared memory resource allowing for the exchange of data among them. Furtherly, after defining all the necessary reference systems, the data input acquired through the optical sensor are interpreted in terms of displacements in such a way as to obtain an actual real-time functioning. They are also mapped from the sensor's to the manipulator's field of motion, before computing the solutions of the inverse kinematics, which are expressing the necessary joints angles. After that, the choice for the adopted interpolation function, allowing for a smooth and continuous trajectory among consecutive poses, has been explained as well.

The next step consisted in the definition of the experimental setup over which the system development was based. In particular, after introducing the main features of the available Epson C3 manipulator, it was justified the choice of the Leap Motion device as the sensor for detecting the hands motion. Contextually, also the utilised software architecture was presented.

Afterwards, a significant number of testing sessions of the developed system have been performed using both the real machine and the simulation environment. In particular, it has been asked to different users to attempt at controlling the motion of the dual-arm manipulator by using the movements of their hand.

The obtained motions have been analysed in order to evaluate whether or not the predefined research objectives were satisfied. With respect to the five project goals, the obtained results can be summarised as follows:

- *Intuitiveness*: all of the different users, who have never before controlled the manipulator through the developed infrastructure, managed to impose the desired motion since their first attempt. The obtained learning curve is then remarkably reduced with respect to the traditional hands tracking methods (joysticks).

- *Simultaneous dual-arm control*: the use of two hands at the same time didn't affect the performance of the system, which managed at bearing a computational load that is doubled in size. Also the eventuality of hazardous movements of the human hands was taken into account, obtaining a coherent motion of both the robotic arms.

- *Real time functioning*: the system showed a certain time delay between the input and output motions, as shown by the graph of Fig.5.1. However, the average delay observed was of about 0.1 seconds, with peaks of a quarter of second. It can be then claimed that the system ensures a good approximation of a real-time functioning.

- *Accuracy*: the movements of the human hands were tracked with submillimetre precision as guaranteed by the Leap Motion device. The output motion showed anyway significant differences, due in part to the temporal delay and in part to the high order of the interpolating function, which reproduces the hands motion with a much smoother trajectory. However, the end-effector average position error and orientation error were never higher then 20 mm and 16 degrees respectively. Such errors values allow for a good level of accuracy in a wide range of applicative fields.

- *Safety*: the pre-existing manipulator's precautions in terms of safety were combined with specific measures adopted during the development of the considered system. In particular, the hands trajectories have been mapped in a way that possible collisions with the surrounding environment or among the two arms would be avoided. As a result, none of the tests performed on the real machine (as well as the ones in the simulation environment) presented any event of harm or danger.

6.2 Future Works

Besides the presented research work, it is clearly possible to further elaborate about different aspects related with the teleoperation a dual-arm manipulator by hands' tracking. The reported results constitute in fact a valid framework from which is possible to process about more specific contents. Multiple focuses could hereafter be identified, in order to achieve a more complete and exhaustive analysis of the considered matter.
The presented research work has been focused on the motion itself and on the most effective way to transmit the desired trajectory from the human hands to the robotic arms. Further developments could then be related to the different plausible applicative contexts for the proposed control infrastructure. It is in fact natural, having two robotic arms simultaneously teleoperated, to think about them performing different types of bimanual tasks. However, in order to evaluate the specific jobs that could be adequately completed by the manipulator, it is necessary to furtherly test its abilities when it comes to basic duties, such as the grabbing of objects of different shapes or the execution of operations that require the cooperation between the two arms. Future works could then focus on the possibility of performing these kinds of tasks with the manipulator controlled through the Leap Motion.
Anyway, the manipulator will be able at executing these types of specific tasks only in the cases where it is supplied with the necessary items. Although this aspect has not been dealt with in this research work, it is possible, depending on the specific applicative scenario, to equip the end-effectors with suitable tools, such as grippers or robotic hands, as shown in Fig.6.1.

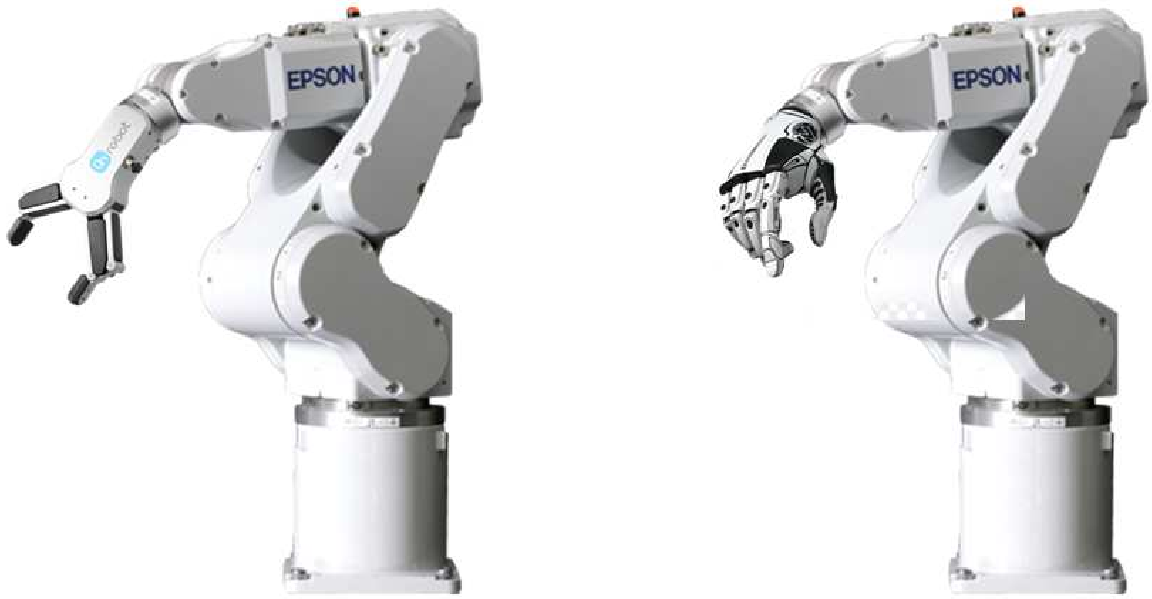


Figure 6.1: Example of a possible configuration of the manipulator, where the left arm is equipped with a gripper and the right one with a robotic hand.

Also the control of such tools can be potentially performed through the use of human gestures, interpreting the number of the extended fingers, or in general the fingers position, as a mean to command the extremity tools. Future works could then deal with the control of grippers and robotic hands through the Leap Motion (or through possibly other devices), while testing their effectiveness in different sorts of bimanual tasks.

In addition, future works could investigate on the different quality of the performance that would be obtained when preferring another kind of sensor to the Leap Motion. The tracking of the hands of the human operator could in fact be realised through the use of other devices such as the Microsoft Kinect or some type of joystick. In this way the obtained trajectories for the end-effectors would be likely characterised by different levels of accuracy or different values of input-output delay. Future developments could then establish specific performance parameters by which comparing the motions obtained with the use of different sensor devices. In this way a more coherent justification to the choice of a specific hand tracking device among the others would be provided.

Bibliography

[1] Y. Zhang, L.Xie, Z. Zhang, K.Li, L.Xiao, “Real-time joystick control and experiments of redundant manipulators using cosine-based velocity mapping”, IEEE International Conference on Automation and Logistics (ICAL), 2011.

[2] H. Sugarman, E. Dayan, A. Lauden, A. Weisel-Eichler, J. Tiran, “Investigating the use of force feedback joysticks for low-cost, robot-mediated therapy,” Int. J. Disability Human Develop., vol. 7, no. 1, pp. 95–100, 2008.

[3] G. C. Luh, H. A. Lin, Y. H. Ma, and C. J. Yen, “Intuitive muscle gesture based robot navigation control using wearable gesture armband,” in Machine Learning and Cybernetics (ICMLC), 2015 International Conference on, vol. 1, July 2015, pp. 389–395.

[4] I. J. Ding, C. W. Chang, and C. J. He, “A kinect-based gesture command control method for human action imitations of humanoid robots,” in 2014 International Conference on Fuzzy Theory and Its Applications (iFUZZY2014), Nov 2014, pp. 208–211.

[5] A.Sarkar, K.A.Patel, R.K.G.Ram and G.K.Capoor,“Gesture Control of drone using a motion controller,” in 2016 International Conference on Industrial Informatics and Computer Systems (CIICS), March 2016, pp. 1–5.

[6] A. Mishra, P. Makula, A. Kumar, K. Karan, and V. K. Mittal, “Multimodal controls of a smart robot,” in 2015 Annual IEEE India Conference (INDICON), Dec 2015, pp. 1–6.

[7] M. Inc, “Microsoft Kinect,” 2015. [Online]. Available: https://dev.windows.com/en-us/kinect

[8] L. Motion Inc, “Leap Motion Inc,” 2015. [Online]. Available: https://www.leapmotion.com/

[9] 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, 2014.

[10] P. Apostolellis, B. Bortz, M. Peng, N. Polys, and A. Hoegh, “Poster: Exploring the integrality and separability of the leap motion controller for direct manipulation 3d interaction,” in 3D User Interfaces (3DUI), 2014 IEEE Symposium on, March 2014, pp. 153–154.

[11] D. Bassily, C. Georgoulas, J. Guettler, T. Linner, and T. Bock, “Intuitive and adaptive robotic arm manipulation using the leap motion controller,” in ISR/Robotik 2014; 41st International Symposium on Robotics; Proceedings of, June 2014, pp. 1–7.

[12] F. Kobayashi, K. Okamoto, and F. Kojima, “Robot-human handover based on position and posture of human hand,” in Soft Computing and Intelligent Systems (SCIS), 2014 Joint 7th International Conference on and Advanced Intelligent Systems (ISIS), 15th International Symposium on, Dec 2014, pp. 918–921.

[13] Y. Liu and Y. Zhang, “Toward welding robot with human knowledge: A remotely-controlled approach,” Automation Science and Engineering, IEEE Transactions on, vol. 12, no. 2, pp. 769–774, April 2015.

[14] L. Peppoloni, F. Brizzi, C. A. Avizzano, and E. Ruffaldi, “Immersive ros-integrated framework for robot teleoperation,” in 3D User Interfaces (3DUI), 2015 IEEE Symposium on, March 2015, pp. 177–178.

[15] G. Du and P. Zhang, “A markerless human-manipulator interface using particle filter and Kalman filter for dual robots,” IEEE Trans. Ind. Electron., vol. 62, no. 4, pp. 2257–2264, Apr. 2015.

[16] H. Lee et al., “The technical trend of the exoskeleton robot system for human power assistance,” Int. J. Precis. Eng. Manuf., vol. 13, no. 8, pp. 1491–1497, 2012.

[17] J. Kofman et al., “Teleoperation of a robot manipulator using a vision-based human-robot interface,” IEEE Trans. Ind. Electron., vol. 52, no. 5, pp. 1206–1219, Oct. 2005.

[18] S. H. P. Won, W. W. Melek, and F. Golnaraghi, “A Kalman/particle filter-based position and orientation estimation method using a position sensor/inertial measurement unit hybrid system,” IEEE Trans. Ind. Electron., vol. 57, no. 5, pp. 1787–1798, May 2010.

[19] J. Kofman, S. Verma, and X. H. Wu, “Robot-manipulator teleoperation by markerless vision-based hand-arm tracking,” Int. J. Optomechatron., vol. 1, no. 3, pp. 331–357, 2007.

[20] SensAble Technologies Inc, “SensAble Phantom Omni “, 2011. [Online]. Available: $https://www.virtalis.com/wp-content/uploads/phantom_{o}mni_{s}pec.pdf$

[21] 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, p. 6380, 2013. [Online]. Available: http://www.mdpi.com/14248220/13/5/6380

[22] R. C. Goertz, “Fundamentals of general-purpose remote manipulators”, Nucleonics, vol. 10, no. 11, pp. 36–45, 1952.

[23] T. Fletcher, “The undersea mobot,” Nuclear Electronics Laboratory of Hughes Aircraft Company, Tech. Rep., Jan 1960.

[24] R. Ambrose, H. Aldridge, R. Askew, R. Burridge, W. Bluethmann, M. Diftler, C. Lovchik, D. Magruder, and F. Rehnmark, “Robonaut: Nasa’s space humanoid,” IEEE Intelligent Systems and their Applications, vol. 15, no. 4, pp. 57 –63, jul/aug 2000.

[25] Y. Yamada, S. Nagamatsu, and Y. Sato, “Development of multi-arm robots for automobile assembly,” in IEEE International Conference on Robotics and Automation, vol. 3, may 1995, pp. 2224 –2229 vol.3.

[26] Y. F. Zheng and M. Z. Chen, “Trajectory planning for two manipulators to deform flexible beams,” Robotics and Autonomous Systems, vol. 12, no. 1-2, pp. 55 – 67, 1994, special Issue Parallel and Multi-Arm Robotic Systems.

[27] H. Liu and J. Dai, “An approach to carton-folding trajectory planning using dual robotic fingers,” Robotics and Autonomous Systems, vol. 42, no. 1, pp. 47 – 63, 2003.

[28] A. Edsinger and C. C. Kemp, Lecture Notes in Control and Information Sciences. Heidelberg: Springer, Feb 2008, vol. 370, ch. Two Arms are Better Than One: A Behavior-Based Control System for Assistive Bimanual Manipulation, Recent Progress in Robotics: Viable Robotic Service to Human, pp. 345–355.

[29] R. L. A. Shauri and K. Nonami, “Assembly manipulation of small objects by dual-arm manipulator,” Assembly Automation, vol. 31, no. 3, pp. 263–274, 2011.

[30] R. Bloss, “Robotics innovations at the 2009 assembly technology expo”, Industrial Robot: An International Journal, vol. 37, no. 5, pp. 427–430, 2010.

[31] S. Kock, T. Vittor, B. Matthias, H. Jerregard, M. Kallman, I. Lundberg, R. Mellander, and M. Hedelind, “Robot concept for scalable, flexible assembly automation: A technology study on a harmless dual-armed robot,” in IEEE International Symposium on Assembly and Manufacturing, may 2011, pp. 1–5.

[32] Robotic Cell Culture system “CellPROi“, 2011. [Online]. Available: http://www.shibuya.co.jp/english/sby/regenerative/cellproi.html

[33] B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, “Robotics: Modelling, Planning and Control”, Ed. Springer, chapter no. 2, pp. 90–100, 2009.

[34] Epson C3 Compact 6-Axis Robot Manual (Revision 9), 2015. [Online]. Available: https://files.support.epson.com/far/docs/epson_c3_robot_manual(r9).pdf

[35] G. M. Rosa and M. L. Elizondo, “Use of a gesture user interface as a touchless image navigation system in dental surgery: Case series report”. Imaging Sci Dent. 2014 Jun;44(2):155-160.

[36] J. Fingas, “Mercedes-Benz unveils its vision of a self-driving car”, 2015. [Online]. Available: https://www.engadget.com/2015/01/05/mercedes-benz-f015-luxury-in-motion

[37] J. Vincent, “Ableton’s Connection Kit lets you make music with Lego and a banana”, 2016. [Online]. Available: https://www.theverge.com/2016/3/28/11316906/ableton-connection-kit-lego-mindstorms

[38] S. Kessler, “With New Leap Motion Support, Google Earth Lets You Touch The World”, 2013. [Online]. Available: https://www.fastcompany.com/3008654/new-leap-motion-support-google-earth-lets-you-touch-world

[39] F. Weichert, D. Bachmann, B. Rudak and D. Fisseler, “Analysis of the Accuracy and Robustness of the Leap Motion Controller”, 2013. [Online]. Available: https://www.mdpi.com/1424-8220/13/5/6380

[40] Leap Motion Developer’s Documentation. [Online]. Available: https://developer-archive.leapmotion.com/documentation/javascript/api/Leap.Controller.html

[41] K. Khoshelham and S.O. Elberink, “Accuracy and resolution of Kinect depth data for indoor mapping applications”. Sensors 2012, 12, 1437-1454.

Acknowledgements

It has been possible for me to complete this thesis work thanks to the precious and kind help of different people.

I want to thank first of all Professor Zanchettin for guiding me through the writing process of this Thesis. He has always been very helpful, providing many advices and clarifications whenever I asked for them.

I want to thank Professors Hirata and Weng from the Smart Robots Design Lab at Tohoku University (Japan), where I have had the opportunity to be a Special Research Student for 1 year. All the practical work related with the thesis project has been developed in the laboratory thanks to their precious and kind supervision. They both have been very comprehensive about my numerous requests, allowing me to work on the topics that interested me the most.

I also want to thank all the people that have been close to me during this 5 years that led me to the graduation.

Thank you to my family that has been supportive since day one, believing in me in every moment.

Thank you to all my friends in Napoli, Milano and Sendai, who have been next to me, making the hard life of a student less difficoult to bear.