Doctoral Dissertation
Doctoral Program in Mechanical Engineering (32nd Cycle)

Development of a methodology for the human-robot interaction based on vision systems for collaborative robotics

Leonardo Sabatino Scimmi

* * * * * *

Supervisor
Prof. Stefano Mauro

Politecnico di Torino
2020

Declaration

I hereby declare that, the contents and organisation of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

……………………………….....
Leonardo Sabatino Scimmi
Turin, 2020

*This dissertation is presented in partial fulfillment of the requirements for Ph.D. degree in the Graduate School of Politecnico di Torino (ScuDo).

To Ele, my family and friends

Abstract

The industrial world is facing an important change that will bring new powerful tools inside the factories. This innovation will be the result of the fourth industrial revolution, also known as “Industry 4.0”. The collaborative robotics is one of the pillars of the Industry 4.0. The goal of the collaborative robotics is to bring the robots outside from the cages in which they are now and make possible the human-robot interaction. This is a radical change in the industrial robotics because it permits to exploit the potentialities offered by the combination of the precision, velocity and repeatability of robots and the extreme adaptability of humans. In these years, several works have faced the topic of the collaborative robotics and several control algorithms have been presented.

This dissertation proposes two algorithms useful to control collaborative robots and permit safe human-robot interactions in an industrial environment. The first algorithm is a collision avoidance algorithm based on the artificial potential fields approach. The proposed algorithm controls the robot in order to avoid collisions with fixed and dynamic obstacles moving along preferred directions. The trajectory conditioning technique proposed in this work gives to the robot the possibility of avoiding collisions with highly dynamic obstacles overcoming the problem of the unknown directions of motion related to the artificial potential fields approach. In this way the operator can predict how the robot will modify the planned trajectory to avoid collisions making the human-robot interaction more natural. A hand-over algorithm is the second novelty proposed in this thesis. This algorithm permits to obtain fluent hand-over between a robot and a human operator. The proposed algorithm makes possible bidirectional, reactive and fast hand-over actions and the pose of the operator’s hand is considered as target that the robot has to reach. Both the algorithms need as input the information related to the position of the human operator working with the robot. The position and the movements of the human worker are acquired by a markerless vision system. In particular, Microsoft Kinect v2 sensors have been used in this work.

A simulation environment has been developed to permit to a human operator to start interacting with a simulated collaborative robot. A virtual collaborative robotics environment has been developed and presents graphical representations of a human body and of a collaborative robot. The simulated worker and robot reproduce the movements of the human operator and of the collaborative robot helping the worker to interact with the robotic manipulator. The movements of the

human body and of the robot are calculated by models developed in MathWorks environment. A kinematic model of the human body permits to properly move the simulated human body having as inputs the position vectors of the Kinect joints. A kinematic/dynamic model of collaborative robot controlled by the algorithms here proposed permits to understand the behaviour of the robot. A first phase of tests has been conducted using this simulation environment. The aim of these tests was to verify the effectiveness of the proposed algorithms.

A second phase of tests has been conducted in an experimental set-up built up to obtain a collaborative robotics workspace where an operator can interact with a real collaborative robot. Two markerless sensors are used to acquire the position of the human worker and three PCs handle the data from the sensors and from/to the controller of the collaborative robot. The results of experimental tests show the performances of the proposed control algorithms. In fact, the collision avoidance algorithm with trajectory conditioning technique permits a human worker to safely share the workspace with a robot. The evasive movements of the collaborative robot occur along directions that had been previously defined by the operator, making the robot movements predictable and acceptable by the worker. The hand-over algorithm drives the robot to adapt the pose of its tool centre point to the pose of the hand of the human operator. In this way the worker and the robot can fluently exchange objects between them. The developed algorithms make possible a natural and seamless human robot interaction.

Contents

List of Tables

List of Figures

Chapter 1

Collaborative robotics: state of the art

1.1 “Industry 4.0” and collaborative robotics: an overview

The second decade of this century has seen strong evolution of automation and information technology, which is deeply impacting on the production technologies and on the organisation and management of the production plants and of the complete supply chain. The impact is so strong that it is commonly considered the beginning of the fourth industrial revolution, that will radically modify how the manufacturing industry will be structured. This industrial revolution is generally known as “Industry 4.0” and this term was used for the first time by the German Federal Ministry of Education and Research to indicate a set of actions aimed to improve the productivity of the German industry sector in 2011. The name “Industry 4.0” was then adopted in the rest of the world to refer to the new incoming phase of the manufacturing industry.

The idea behind the “Industry 4.0” is to introduce in the industrial world several different technologies (e.g.: Internet of Things, Big Data Analytics etc…) that will improve the production processes. In figure 1.1 a scheme shows the characteristic elements of the four industrial revolutions.

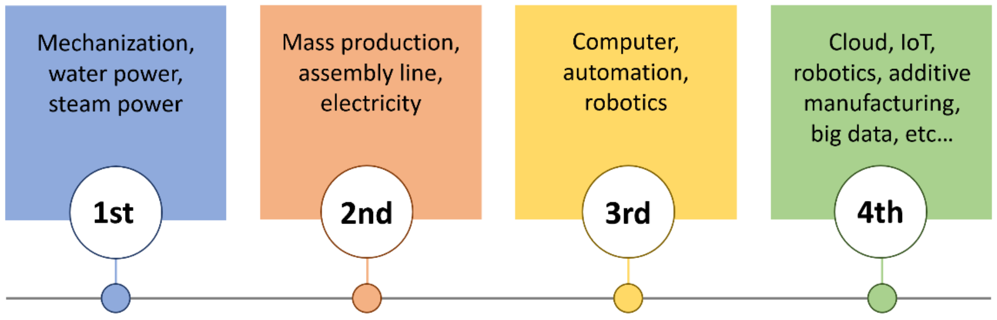

Mechanization, water power, steam power
1st
Mass production, assembly line, electricity
2nd
Computer, automation, robotics
3rd
Cloud, IoT, robotics, additive manufacturing, big data, etc…
4th

Figure 1.1: Characteristics elements of the four industrial revolutions.

Robotics has an important role in both the third and fourth revolutions. Robots were introduced inside the factories in the third industrial revolution and they had the characteristics that are commonly associated to industrial robots: expensive, precise, big, fast and highly productive, but as most of the production machines they are potentially dangerous for the human operators. For this reason, robots are always segregated inside spaces surrounded by physical fences that do not allow the entrance of the operators, unless robots are stopped and deenergized. This kind of robots are generally expensive and programming is a task for dedicated, skilled operators only. The necessity of developing areas dedicated to robots, their cost and the effort needed to program them, entail that robotic production lines must work continuously and for several years on the same kind of products, to amortise the cost associated to their set up. Industrial robots are widely used in the automotive and electronics fields and the number is increasing in others (e.g. food sector) [1]. In the last five years there has been a growth of the annual installations of industrial robots and it will be probably continuing to grow in the next years [1]. The annual report of the International Federation of Robotics (IFR) [1] indicates also that there will be four million industrial robots in the world's factories in 2022.

The growth of the market of industrial robots, led by the running transformation, sees the introduction of two new types of robots: mobile and collaborative robots (also known as *cobots*). In this thesis attention will be focused on collaborative robots only. Collaborative robotics is the new frontier of industrial robotics and the idea behind it is to eliminate the physical fences that segregate the robots and develop factories where humans and robots can share the workspace and collaborate to perform some tasks. The concept of collaborative robotics was firstly introduced by Colgate and al. in [2] in 1996 and the first collaborative robotic manipulator was the DLR Light Weight Robot LWR III developed by DLR Institute of Robotics and Mechatronics in collaboration with KUKA and presented in 2003. In the last fifteen years several companies producing collaborative robots were founded (e.g.: Universal Robots (UR), Rethink Robotics etc…) and the historical manufacturers of industrial robots started to sell their own cobots. In table 1.1 some collaborative robots and their characteristics are reported. It is clear that cobots present a significant variety in terms of payload (from 0.5kg to 170kg) and kinematic layout (e.g.: anthropomorphic manipulators with 6 or 7 degrees of freedom (*dofs*), double scara robots etc…).

A most relevant property of cobots is the relatively easiness of programming, that makes possible to train an operator in few days.

Table 1.1: List of collaborative robots.

Company	*Robot*	*Kinematic*	*Payload [kg]*	*Reach [m]*
ABB	YuMi	Dual 7dofs serial arms	0.5 each arm	0.5 each arm
Comau	AURA	6 dofs serial	170	2.21
Fanuc	• CR 4iA • CR 7iA • CR 35iA	6 dofs serial arms	• 4 • 7 • 35	• 0.55 • 0.717 • 1.813
Kawasaki	DUARO	Double SCARA	2	0.76
KUKA	• LBR iiwa 7 R800 • LBR iiwa 7 R800	7dofs serial arms	• 7 • 14	• 0.8 • 0.82
Rethink Robotics	• Sawyer • Baxter	• 7dofs serial arm • Double 7dofs serial arms	• 4 • 4 each arm	• 1.26 • 1.26 each arm
Universal Robots	• UR3 – UR3e • UR5 – UR5e • UR10 – UR10e • UR16e	6 dofs serial arms	• 3 • 5 • 10 • 16	• 0.5 • 0.85 • 1.3 • 0.9

In [1] it is clear how the collaborative robotics is still a niche. In fact, the collaborative robots sold in the 2018 were approximately 14000, 3% of all industrial robots sold in that year. However, it is important to highlight that collaborative robots sold in the 2017 were about 11000. This means that the yearly growth has been 25%. Furthermore, a report of Interact Analysis [3] reveals that the collaborative robotics market is strongly growing in the next years and cobots will be 30% of the total robot market in the 2027. The same document reports that cobots will be widely used in the electronics field and in logistics.

The future growth of the collaborative robotics is strictly related to the possibilities that the cobots offer to small and medium-sized enterprises (SMEs). In fact, cobots are cheaper than other industrial robots and the possibility to use them without the need to install physical safety barriers reduces the cost associated to the adoption of robots in factories. Moreover, the easiness in programming is going to allow SME to use this system with internal resources, without needing expensive external support for simple reconfiguration activities. This is going to permit the SMEs to enter in the world of robotics and start to innovate their productive processes introducing robots. Collaborative robots will be an important tool not only for the SMEs, but also for the big companies that are already using industrial robots. In fact, they can use cobots to partially automate processes that are fully manual now. Cobots and human workers can work on the same process exploiting their own peculiar features, such as repeatability and precision for the robots and adaptability for the humans. This permits to answer to the consumers' demand of more customized products, as reported in [4]. As shown in figure 1.2, the industrial robots are useful with manufacturing processes characterized by big production volumes and reduced product variety. Instead, cobots can be used in those cases with reduced production volumes and quite wide production variety.

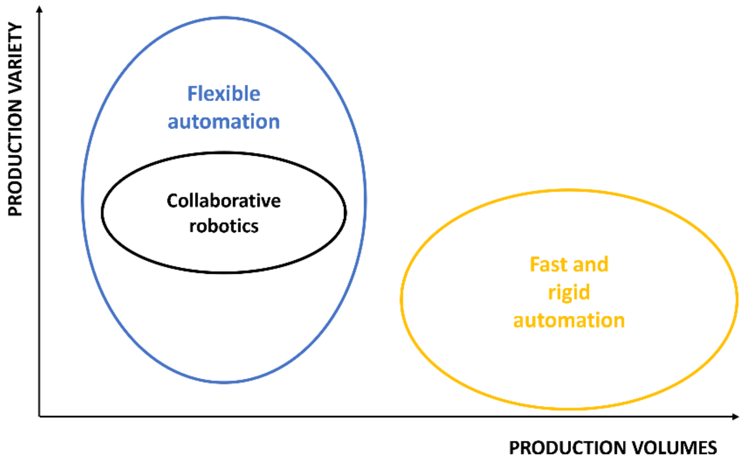


Figure 1.2: Different characteristics of flexible automation and fast and rigid automation.

These production processes require the possibility of modifying easily the production lines to adapt them to the different productions. Collaborative robots have suitable characteristics: they do not require physical fences and their programming is simple, therefore modifying the lines is not expensive or time consuming. Furthermore, the presence of human operators can play an important role in those cases where it is required an adaptability that robots do not have. Only the cobots permit to have workspace where humans and robots can work side by side. Cobots can also represent a possible solution to reduce problems related to musculoskeletal disorders (MSDs), that are the most common health problems in the workplace in the European Union [5]. In fact, collaborative robots can relieve human workers whose activities can cause them MSDs, such as lifting payloads or actions that produce unhealthy vibrations.

Collaborative robots can be used in different scenarios and the IFR, based on [6], defined four types of human-robot interaction (*HRI*) (cf. figure 1.3), here described:

1) *coexistence*: no fences but the workspace is not shared;

2) *sequential collaboration*: the workspace is shared but the movements are sequential;

3) *cooperation*: worker and robot work at the same time in the same part of the workspace, but on different product/component;

4) *responsive collaboration*: robot responds in real-time to movements of the human operator and they work simultaneously on the same product/component.

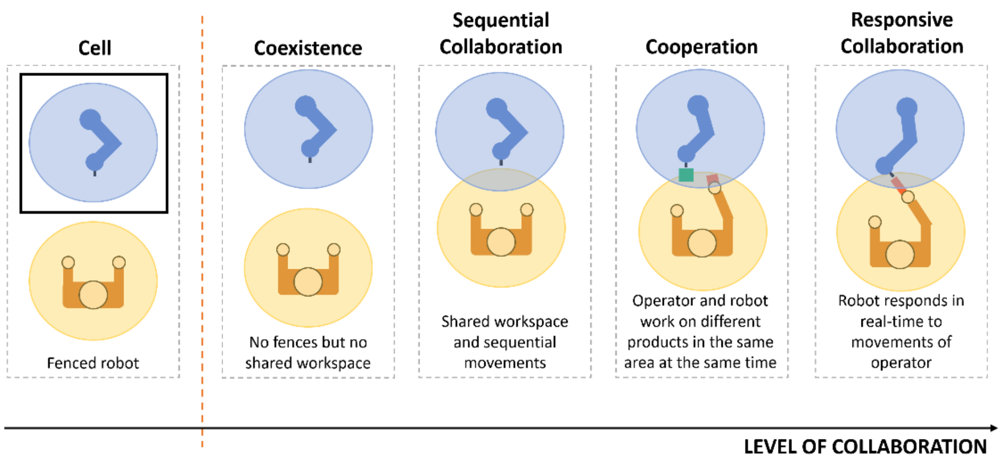

Cell
Fenced robot
Coexistence
No fences but no shared workspace
Sequential Collaboration
Shared workspace and sequential movements
Cooperation
Operator and robot work on different products in the same area at the same time
Responsive Collaboration
Robot responds in real-time to movements of operator
LEVEL OF COLLABORATION

Figure 1.3: Four different types of human-robot interaction. On the left, the case of fenced robot with no human-robot interaction.

For what concerns safety regulations, collaborative robotics was introduced in the international standards UNI EN ISO 10218-1 [7] and UNI EN ISO 10218-2 [8]. Here four requirements for collaborative operations were presented. These specifications were then extended in ISO/TS 15066:2016 [9], providing safety requirements for collaborative robot systems. These operating methods are (cf. figure 1.4):

1) *safety-rated monitoring stop*: the robot must cease its motion when the human operator enters the collaborative workspace, otherwise it can operate non collaboratively;

2) *hand guiding*: the operator can move the robot using a hand-operated device located near the end-effector to transmit commands to the robotic system. A safety-rated monitoring stop must be achieved before the operator enters in the collaborative workspace;

3) *speed and separation monitoring*: differently from the previous methods, in this case robot and human operator can move both inside the collaborative workspace, but a minimum safety distance (protective separation distance) between them must be guaranteed. If the human-robot distance is less than the protective separation distance, the robot must be stopped. The value of the protective separation distance can change during the collaborative operation, depending on the operative conditions (e.g.: velocity of the robot, velocity of the operator etc…);

4) *power and force limiting*: in this method, physical contacts (intentional or unintentional) between robot and operator are permitted, unlike the previous method where contacts must be avoided. Hazard associated to the contacts must be kept below threshold limit values and a guidance to establish these values is reported in the Annex A of the standard.

In [9] it is written that the operating methods “may be used singularly or in combination when designing a collaborative application”.

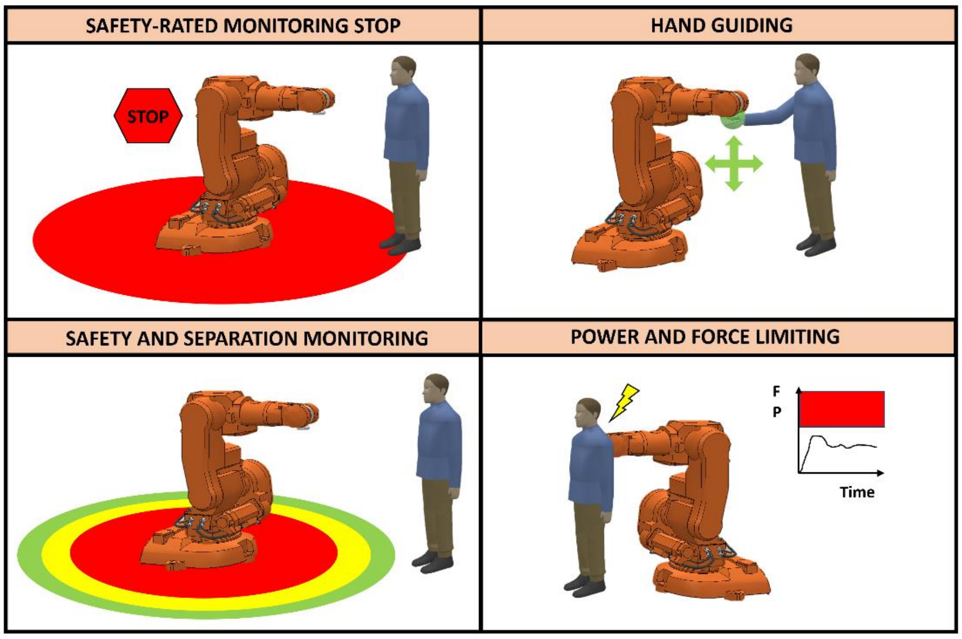

SAFETY-RATED MONITORING STOP
STOP
HAND GUIDING
SAFETY AND SEPARATION MONITORING
POWER AND FORCE LIMITING
F
P
Time

Figure 1.4: The four safety requirements reported in ISO/TS 15066:2016.

In these years, different enterprises had introduced and tested collaborative robots in their factories and in [10] the IFR lists some of these applications. In most of these applications collaborative robots are used to relieve human operators of doing repetitive, monotonous or ergonomically unsuitable tasks. But there are not industrial applications that imply human-robot cooperation or responsive collaboration.

Whereas collaborative robotics is moving its first steps inside the factories, it has been for years now a trending research topic in academia. Figure 1.5 shows the result of an estimation of research papers on the topic of collaborative robotics, published between 1996 and 2019.

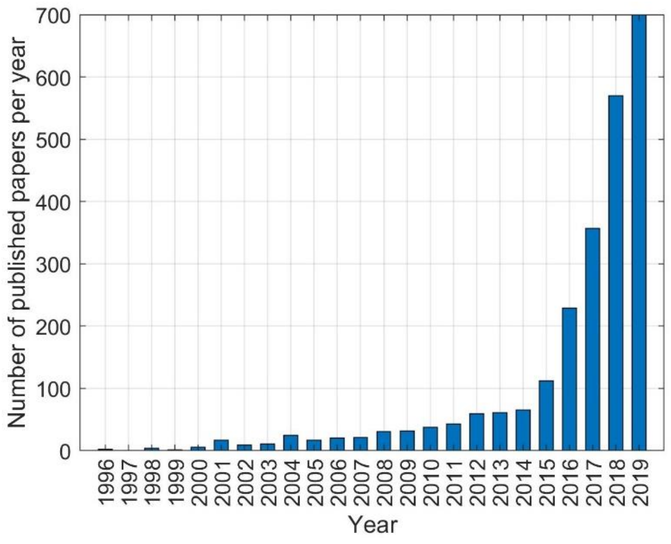


Figure 1.5: Numbers of published papers on the topic "collaborative robotics" in each year from 1996 to 2019.

These research papers address different aspects of collaborative robotics, from safety to programming robot through augmented reality. Table 1.2 reports a possible classification of the papers published on this topic. This result is a possible classification of these works and it is not intended to be exhaustive, but it could be helpful to understand the problems and the complexity related to the collaborative robotics.

Table 1.2: Classification of the research papers on the topic of "collaborative robotics".

Category	***Sub-categories***
Safety	• Safety by design • Safety by pre-collision strategies • Safety by post-collision strategies
Human-robot interactions modalities	• Vision-based systems • Vocal-based systems • Force-based systems
Robot programming methods	• Off-line techniques • On-line techniques
Virtual and Augmented reality	-
Fault tolerance	-
Applications	• Handling objects • Hand-over • Welding • Assembly

In the following paragraphs, the most relevant contributions of each category presented in table 1.2 are briefly described. A more detailed analysis of the works related to the topic "Safety by pre-collision strategies" and "Hand-over" is presented in Chapter 2.

1.1.1 Safety

Safety is a crucial aspect in the collaborative robotics scenario. In fact, the first characteristic associated to the cobots is the possibility to use them without the unavoidable obligation to install physical fences between humans and robots. This implies that the safety of human workers must be ensured with different solutions. As can be seen in table 1.2, safety presents three subcategories: 1) "Safety by design"; 2) "Safety by pre-collision strategies"; 3) "Safety by post-collision strategies".

"Safety by design" was the first approach adopted to guarantee the operators' safety. The idea is to design robots that are intrinsically safe. Various solutions have been proposed, from the reduction of the weights of the links of the robot

([11]) to the adoption of soft coverings, airbags or energy adsorbing protective layers to cover the robots ([12-13]). In other works, robots can handle collisions with other elements in the workspace thanks to the integration of mechanisms inside their structure (e.g. Series Elastic Actuators (SEA) [14] and Variable Impedance Actuators (VIA) [15]). The SEA permits to decouple the inertia of the link involved in a collision from the inertia of the previous links, thanks to an elastic element between the motor and the link. In this way it is possible to reduce the levels of injury risk. The VIA is an actuator designed to vary the value of stiffness, damping, or gear-ratio while the task is executing. VIA is an evolution of the SEA, where the compliance properties are fixed.

"Safety by post-collision strategies" embodies those approaches that actively reduce the impact of undesired collisions between robot and human. The basic solution is stopping the robot after the collision occurred. More advanced solutions permit to control the torque of the robot and adapt its behavior in order to reduce the impact energy moving the robot away and not just stopping it. For example, in [16] a damping controller was developed to modify online the tool velocity, power and contact force. In [17] a method to estimate the contact point and the contact force and using this information to control the relative motion and the exchanged forces is presented.

1.1.2 Human-robot interactions modalities

To obtain a more efficient and fluent human-robot interaction, different works have faced the human-robot communication topic and explored several modalities of interaction. It is possible to divide the systems presented in these works in three main groups: vision-based systems, vocal-based systems and force-based systems.

For what concern the first group, the human worker's movements are acquired by vision sensors and information extracted from them are given to the robot as input. Several papers of this group presented methods to understand the intentions of the human operator working with the cobot. In [18], the authors presented a multi-labeled framework useful to recognize human actions. The inputs of the system are 3D skeleton joints positions data obtained from a RGBD camera. Recognizing the actions permits to improve the human-robot collaboration. A simple and intuitive way to control robots is by gestures. Different works presented methods to recognize gestures and control the robot giving commands associated to the recognized gestures. In [19-20], a Leap Motion sensor is used to track the movements of the hands of a worker and he/she can control a KUKA Youbot using gestures recognition tools. Start and stop commands obtained from gesture recognition features are exploited to control a robot in [21]. A face recognition feature can be an additional security measure adopted to ensure the safety of the operator. In fact, using face recognition tools it could be possible to identify if the operator is allowed to work with the cobot or not [22]. In this way, only trained workers can interact with the robots reducing the probabilities of incidents.

Vocal communication is the preferred one in human-human interaction and it is very interesting also for human-robot interaction. In fact, the vocal communication channel is fast and it is useful when the hands of the operator are not free and other interaction modalities are not exploitable (e.g. commands based on gestures). In [23] a method to generate from language instructions a code for both virtual and real robot to execute assembly tasks is presented. An important problem for the applicability of vocal-based systems in factories is the environmental noise. The noise can generate misrecognition of commands, possibly leading to dangerous effects. A method to improve performances of commercial vocal recognizers in an industrial environment is described in [24].

In the previous cases, there is not the necessity of contacts between humans and robots. The operator can control and command the robot remotely. Instead, the force approach implies the analysis of the interaction forces to determinate the operator's intentions and command the robot properly. This HR interaction modality has been used in different cases, from collaborative object transportation [25] to posture assistance [26] and cooperation in assembly lines [27].

1.1.3 Robot programming methods

Different modalities have been developed to facilitate and speed-up the programming phase of the robotic systems. They can be divided in off-line and on-line techniques.

The off-line techniques [28] are characterized by using software that presents a 3D model of the robot to have an immediate feedback. Several modules are offered to simulate operations like, for instance, polishing or welding. The 3D model of the robot permits to check possible collisions and modify the program to avoid them. After the testing phase, the program can be uploaded to the robot control unit. The main advantage of this kind of robot programming is the possibility to write the program and testing it without stopping the real robot, that can continue to work. The main throwback is that every robot manufacturer has its own off-line software and they are usually expensive.

The on-line techniques permit the operator to program the robot directly interacting with it. The "Walk-through programming" and the "Programming by demonstration" are two on-line programming techniques. In the first case, the operator moves the end-effector of the robot in different positions and the controller records the desired trajectories in order to reproduce them later [29-30]. In this case, the robot reproduces exactly the movements performed by the operator. "Programming with demonstration" is an evolution of the "Walk-through programming". In fact, the acquired movements are not simply reproduced, but they are generalized so that the robot can adapt and used them in different scenarios [31-32].

1.1.4 Virtual and augmented reality

Virtual (VR) and augmented reality (AR) are tools used in different aspects of collaborative robotics, especially in the training phases of the human workers and as additional tools to guarantee the safety of the operators.

In [33-34] AR techniques are implemented to give to the operator useful information regarding the production processes. In [35-37] human operators are trained to operate with a collaborative robot in different scenarios thanks to virtual scenes that reproduce the collaborative workspace. AR and VR tools are also used to program robots.

Projection of the workspace of the robot and safe areas are presented in [38]. A Kinect v2 sensor is used to acquire the movements of a human operator and reproduce them in a virtual environment with a virtual robot to test collision avoidance algorithms, [39].

1.1.5 Fault tolerance

The fault tolerance of collaborative robots is an important problem in human-robot interaction. The fault tolerance permits to reconfigure the robotic system in order to continue working properly even in presence of faults [40]. In this way it is possible to avoid an anomalous and dangerous behavior of the robot while it is working near or with human operators [41].

1.1.6 Applications

Several papers described case studies in which cobots have been used. In this paragraph, a brief overview of the most relevant collaborative robotics applications is presented.

Handling objects, together with hand-over tasks, is the most common application in collaborative robotics. It can be found in different branches of the manufacturing sector. Using cobots to handle materials permits to reduce the physical effort of the operator when lifting or moving objects and to let the worker focus on the tasks that require the human adaptability and sensibility. For example, in [42] a cobot holds workpiece in a precise position and orientation and the operator is responsible for the final steps of a polishing operation. Another work presented a robot assistant that helps tyre dealer lifting and positioning tyres in the proper location, [43].

Welding is another interesting application in which collaborative robots can be used. In [44-45], walk-through programming techniques have been implemented to program robots in order to perform welding operations.

Assembly operations can benefit from the introduction of collaborative robots, [46]. In [47], a collaborative robot that can be used in automotive assembly lines is presented.

1.2 Conclusions

Collaborative robotics is a radical change in the industrial robotics world, opening to the possibilities to have shared workspace between robots and human workers, without the need of physical fences to ensure the safety of the operators. This is an important change in the manufacturing sector, because it permits to adopt solutions that involve human-robot collaboration to replace production processes that are now fully manual. In this way the human operator can engage in high value-added activities, leaving monotonous, repetitive or ergonomically unsuitable tasks to the robots. Furthermore, the possibility to not install physical and expensive fences, the easy programming of this kind of robots and their relatively low cost are interesting characteristics for the SMEs, that in many cases can not consider the introduction of traditional robots in their plants. The data reported in [1] show that the collaborative robotics market is still small, but a significant growth is expected in the next years [3].

The international standards UNI EN ISO 10218-1 and UNI EN ISO 10218-2 have introduced the collaborative robotics in the existing regulatory framework, defining four classes of safety to adopt when designing collaborative application. These safety requirements are extended and explained in the technical specification ISO/TS 15066:2016. The first collaborative applications that respect the safety standards are now emerging in the factories around the world, [10].

Collaborative robotics is moving its first steps inside the factories, but it is an interesting and trending research topic since years. More than 2400 research papers on the topic of collaborative robotics were published from 1996 to 2019, facing different aspects of this content.

In this chapter, a review of the research literature on this topic was given and a possible classification of the papers in categories was presented.

Chapter 2

Innovative vision-based control algorithms for human robot interaction

2.1 Introduction

In the previous chapter, the four different types of human-robot interaction have been described. For the “coexistence” and the “sequential collaboration” cases, one of the main aspects of the human-robot interaction to consider is the safety of the human operator. In fact, in these cases the cobot and the human worker do not collaborate to achieve a joint task, but they perform distinct operations. The robot must recognize the presence of the operator to avoid hurting him/her. In the case of the “responsive collaboration”, the safety is still a critical issue, but the cobot also has to be able to adapt its motion to the movements of the human operator in order to perform common tasks. Control algorithms to command properly the robot in order to collaborate with the operator must be implemented in the collaborative robotics system.

The topics of the safety of the human-operator and of responsive collaborative tasks have been studied in this work and algorithms useful to control the robot have been developed. Collision avoidance and human-robot hand-over control algorithms will be presented in this chapter. The state of the art related to these two topics and the novelties of the developed algorithms will be explained.

2.2 Collision avoidance algorithm

Collision avoidance algorithms must ensure the safety of human operators working with collaborative robots. A robot controlled by these algorithms must follow a trajectory avoiding any undesired collisions with the workers while the cobot is moving. The topic of “collision avoidance algorithms” is one of the most studied problems in academia. The problem of the collision avoidance algorithms was faced by the robotics community before the arrival of collaborative robotics. At the beginning, the idea was to plan trajectories that avoid collisions between the robots (mobile or manipulators) and other objects present in the workspace, in order to not damage the robots or other tools. The human operators were not considered because robots and humans were separated by physical fences and any interactions were not possible. The development of collaborative robotics introduced human workers inside the robots' workspace and the goal of the collision avoidance algorithms became to control the robots so to avoid hurting human operators.

Collision avoidance algorithms can be divided in two main groups: *offline* algorithms, that plan offline the trajectories that the robot has to follow so to avoid collisions with fixed and well known obstacles, and *online* algorithms, that plan or replan online the trajectories of the robot in order to avoid collisions with static and dynamic obstacles (e.g. human operators).

The offline collision avoidance techniques were the most studied before the introduction of the collaborative robotics. These algorithms were interesting because they permitted to obtain trajectories that were globally optimal ones. The offline algorithms can be divided in sub-categories:

- configuration space or C-space, in which the C-space is the space of all the possible robot configurations (also the initial and final configurations that the robot has to assume to perform the task) and $C_{obstacle}$ is the space with the obstacles' configurations. The path of the robot has to be defined in the free space $C_{free} = C - C_{obstacle}$. The complexity of the problem is related to the number of the degrees of freedom of the robot, [48, 49];
- Probabilistic planners. These algorithms start looking for collision free robotic configurations in the proximity of the initial configuration. After they found a configuration, they search other configurations near this one. The research of collision free configurations ends when the robot reaches the final configuration. The path that the robot has to follow is the connection of these collision free configurations, [50-52].

The main limitation of these techniques is that they need a previous complete knowledge of the workspace in which the robots must move and operate. So, the obstacles that the robot should avoid must be fixed. Furthermore, they are time and memory resource consuming, because they need time and computational resources to plan a proper trajectory. So, it is not possible to use them in a dynamic unconstructed environment where the trajectory of the robot must be rapidly modified to avoid collisions with moving obstacles. In [52], a Rapidly-exploring random tree (RRT) algorithm is used with a moving obstacle (the hand of a human operator) and it permits to modify the path of the robot. This is a preliminary result and the hand is only moving initially. For these reasons, only the online collision avoidance algorithms can be used in an unconstructed collaborative workspace.

The online collision avoidance techniques will be presented and discussed in the next paragraph.

2.1.2 State of the art

The online collision avoidance algorithms presented in the papers here reported can be divided in the following groups: reducing velocity/stopping approach, optimization problems and artificial potential fields.

The works related to the first group ([53-55]) present methods to avoid collisions between humans and robots based on the “safety-rated monitoring stop” and the “speed and separation monitoring” (taking in consideration the speed reduction approach) operating methods reported in ISO/TS 15066:2016. Here the velocity of the robot is reduced if operators enter in zones near the manipulator and the cobot must be stopped if the worker enters in a stopping zone. The robot can move again only when the worker leaves the stopping zone. This is a very precautionary approach, because it can lead to a stop of the robot even with a short-time entry of the operator in the stopping zone. This is an intermediate step from the industrial robotics to the collaborative robotics. The physical fences are replaced by virtual ones, but in both cases the robot cannot work if the operator is nearby.

For what concern the algorithms of the second group, here the trajectory that the robot has to follow is the online result of an optimization problem, where the collision avoidance is associated to one of the constraints of the problem ([56-61]). In [56] a safety index was presented and an optimization problem was used to plan collision free trajectories for robots. The safety index is constrained to be under a certain threshold. Other constrains are defined to limit the joint velocities and accelerations. This control algorithm was tested only in a simulation environment, no experimental results are presented. In [57] an optimization-based control algorithm that permits to maximize the productivity respecting safety constraints is presented. Here the human operator’s movements are acquired by two RGBD cameras and thanks to a kinematic model of the human body, the volume occupied by the human operator is predicted. This information is one of the inputs of the online optimization problem, that has as goal to avoid collisions between the cobot and the worker without moving the robot too far from the planned path. In this way it is possible to ensure the safety of the human operator and maximize the productivity. In [58], a method to modify the planned path of a collaborative robot to avoid a human operator is presented. The constraints of the optimization problem are defined in order to avoid that the robot collides against the human operator, walls and other objects in the workspace. Other constraints permit to avoid that the robot reaches the limits of its workspace while modifying the path. Results of test conducted only with a fixed obstacle are presented and discussed (the operator puts is hand near the path of the robot, but he/she does not move the hand). In [59-61], machine learning techniques are used to obtain data that become inputs of the optimization problems. Machine learning techniques are becoming quite popular also in the field of the collision avoidance algorithms. In fact, they are used to obtain algorithms that permit to predict the movements of the human operators and anticipate the collision avoidance motions of the robot. However, these techniques are useful when the actions that the human operators have to perform are well known and repetitive, as for example in the case of assembly tasks. In [59] the actions that the operator performs are reported and a video dataset is used to train the system. The necessity to know the actions of the operator limits the applicability of the approach based on machine learning techniques. The proposed optimization-based algorithms are fast enough to modify online the path of the robot, so they can be used with fixed and moving obstacles. But there are situations in which the problem has not a solution and the robot must be stopped. The optimization problem is a good solution if the obstacle is moving slow and the robot does not need fast evasive motion to avoid the collision. [59-61] shows improvements of the evasive ability of the robot in case of predicted movements of the operator, but these methods can be applied when the movements of the operator are repetitive. They are not suitable in highly dynamic environment. Moreover, they consider only some parts of the human body, like hands or arms. Therefore, collisions between the robot and other parts of the human body (e.g. the head) are possible.

The artificial potential fields technique is a famous approach to plan trajectories for both mobile and manipulator robots. This technique was firstly presented by Khatib in [62]. The idea is to associate attractive potential fields to the destination that the robot has to reach and repulsive potential fields to the obstacles. Combining these two types of fields, the robot can avoid collisions with the obstacles and reach its destination. The attractive and repulsive actions can be expressed as forces acting on the robot or velocities. Several works used this technique to develop collision avoidance algorithms. In [63], a fast method to evaluate the distances between a robot and a worker is presented. The position of the operator is acquired by an RGBD sensor and points on the structure of the robot are considered for the distance calculation. The calculated distances are the inputs of the collision avoidance algorithm. If the operator is near the tool centre point (hereafter *TCP*) of the robot, the algorithm calculates the repulsive velocity useful to modify the planned path. If the obstacle is near one of the other points on the structure of the robot, the algorithm reduces the velocities of the joints related to the movements of that point. If the collision is unavoidable, the robot is stopped. In [64], an industrial robot and a human operator share the same workspace. The worker can move freely inside the space while the robot performs its task. Simulation tests are conducted to verify the effectiveness of the collision avoidance algorithm. No experimental tests have been performed. Attractive and repulsive potential fields are used to plan the trajectory of a cobot in the presence of a human operator in [65]. The pose of the worker was acquired by Polhemus Liberty magnetic tracking sensors or IMU sensors and a laser scanner. The magnetic tracking sensors and the IMUs are attached to the human upper body, that could limit the movements of the operator. In other works, the artificial potential fields are used in combination with other path planning approaches. In [66] repulsive fields are used to locally modified planned path in presence of fixed and moving obstacles. Here the planned path was defined using learning from demonstration (LfD) techniques. An RRT technique and artificial potential fields are used together in [67]. The RRT algorithm is used to calculate virtual target points that permits to move the robot if it is stack in local minima. The approach was tested only in a simulation environment and with fixed obstacles. The artificial potential fields method is computationally fast and permits to obtain fast evasive motions in highly dynamic environments. A problem related to the artificial potential fields is that the resultant movements of the robot are not predictable. These movements depend on the magnitude of the attractive and repulsive actions and on their directions. The resultant movements depend on the local conditions. In table 2.1, a summary of the papers previously presented is reported.

Table 2.1: "Collision avoidance" summary.

Paper	***Collision avoidance technique***	***Characteristics***	***Limitations***
[53]	Reducing velocity/stopping approach	The robot speed is adapted depending on the distance human-robot. The human position is acquired by Kinect sensor	The robot is simulated
[54]	Reducing velocity/stopping approach	A safety system to control an industrial robot where the movements of the operator are acquired by Kinect and laser scanners	-
[55]	Reducing velocity/stopping approach	Here the authors used only laser scanner to evaluate the position of the operator	-
[56]	Optimization problem	A safety index that considers the human-robot distances and the momentum of the links of the robot is presented. The index is used as input in an optimization problem	Only simulation results
[57]	Optimization problem	A duplex RGBD system acquires motions of an operator. The volume occupied by the worker is predicted and used as input in an optimization problem	The human performs slow movements

[58]	Optimization problem	A vision system acquired the movements of the operator and used these data as inputs of an optimization problem with several constraints on the motion of the robot	Results with only fixed obstacle are reported
[59]	Optimization problem	Machine learning techniques predict actions of the operator and modify the path of an UR robot to avoid collisions	The robot can avoid collision only with the hand
[60]	Optimization problem	The motions of the human operator are modelled with a time series model	The robot can avoid collision only with the hand
[61]	Optimization problem	Prediction of the volume occupied by the human when collaborating with a robot in an assembly task is input of an optimization problem	The robot can avoid collision only with the forearms and the hands
[63]	Artificial potential fields	A collision avoidance algorithm permits to avoid collisions between all the links of the robot and the operator. A method to rapidly calculate the distances is presented	Only the TCP of the robot performs evasive motions
[64]	Artificial potential fields	Artificial potential fields are used to modify the path of an industrial robot	Only simulation results
[65]	Artificial potential fields	Classic artificial potential fields are used to avoid collisions with the operator. The operator's movements are acquired by magnetic tracking sensors or IMUs and laser scanner	The magnetic tracking sensors and the IMUs could limit the movements of the worker in a real industrial scenario
[66]	Artificial potential fields	Artificial potential fields algorithm modifies locally the path learned by the robot thanks to learning from demonstration (LfD) techniques	The human operator is not considered

[67]	Artificial potential fields	RRT and artificial potential fields algorithms are used together to overcome the local minima problem	Only simulation results

Some interesting considerations can be obtained from the analysis of the collision avoidance methods presented in this paragraph: the optimization method permits to consider several constraints and obtain a resultant motion that satisfy all the requested conditions. But it could not be useful in case of fast moving obstacles. Artificial potential fields approach permits to obtain reactive evasive motion even with highly dynamic obstacles, as human operators. Unfortunately, the direction of the resultant motion is not known a priori. This can be an important issue in a human-robot interaction case. In fact, the impossibility to predict the movements of the robot can reduce the level of safety perceived by the human operator. In the following paragraph, a novel technique that permits to define preferred directions of motion in case of a human-robot interaction will be proposed. It is called "collision avoidance algorithm with the trajectory conditioning technique" and permits to impose directions of motion to the cobot while it is sharing the workspace with human workers. This approach is based on the artificial potential fields and gives the opportunity to have predictable directions of motion even with a highly dynamic obstacles as human operators.

A standard collision avoidance algorithm called "basic collision avoidance algorithm" and the collision avoidance algorithm with the trajectory conditioning technique will be described in the next paragraph.

2.2.2 Developed algorithms

In this paragraph, the developed collision avoidance algorithms will be presented. Two different types of algorithms will be described: the first one is a basic collision avoidance algorithm, that is a reformulation and extension of the algorithm reported in [63]. The algorithm here reported permits to obtain evasive motions not only for the TCP, but also for other parts of the structure of the robot. It does not simply reduce the joint velocities as in [63] but generates evasive movements for all the structure of the cobot. The second algorithm is the collision avoidance algorithm with the trajectory conditioning technique, that permits to avoid collisions while forcing the TCP of the robot to move along desired directions. These directions are defined using attractive elements that can be fixed in the workspace or associated to the human worker. The basic collision avoidance algorithm is firstly presented and then the collision avoidance algorithm with the trajectory conditioning technique. In both cases the repulsive and attractive actions are expressed as velocities in the operative space.

In the following the theoretical analysis of the algorithm is described, while an experimental setup is described in the next chapters.

2.2.2.1 Basic collision avoidance algorithm

The structure of the robot is simplified considering points on the body of the robot, as can be seen in figure 2.1.

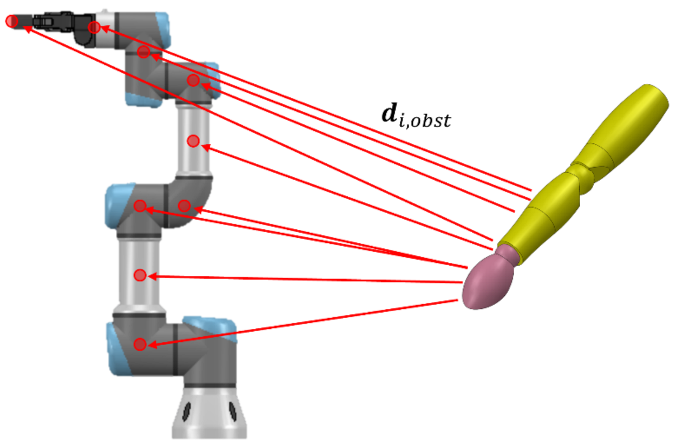


Figure 2.1: Distances $\boldsymbol{d}_{i,obst}$ between the robot and the human arm.

In the figure above, the distances $\boldsymbol{d}_{i,obst}$ between the robot and human body obtained from the distance calculation algorithm are reported (i is referred to the i^{th} point on the manipulator, $obst$ is the operator or a generic obstacle). The obstacles can be modelled as a series of points (like the structure of the robot) or as solids. Therefore, the distances here considered can be point-to-point distances or point-to-surface distances. The algorithm can work with both types of distances and the following theoretical explanation is valid in both cases.

The distances are used to calculate the magnitude of the repulsive velocities,

$$v_{i,obst} = \frac{V_{MAX}}{\left(1 + e^{\left(\|\boldsymbol{d}_{i,obst}\|\frac{2}{\rho}-1\right)\alpha}\right)}$$

$$\boldsymbol{V}_{i,obst} = v_{i,obst}\left(\frac{\boldsymbol{d}_{i,obst}}{\|\boldsymbol{d}_{i,obst}\|}\right) \tag{2.1}$$

where V_{MAX} is the maximum value of the magnitude of the repulsive velocity. The repulsive velocity has null magnitude if the distance is larger than a reference distance ρ. α is a shape factor that determinates how the value of the magnitude of the velocity changes with the distance. The repulsive velocities have the same directions of the distances $\boldsymbol{d}_{i,obst}$. An example of the curve of the magnitude of the repulsive velocity versus the norm of the distance between a point on the robot and the obstacle is reported in figure 2.2. The values of the parameters used to draw the example curve in the figure are V_{MAX} = 0.25 m/s, ρ = 0.5 m and α = 6.

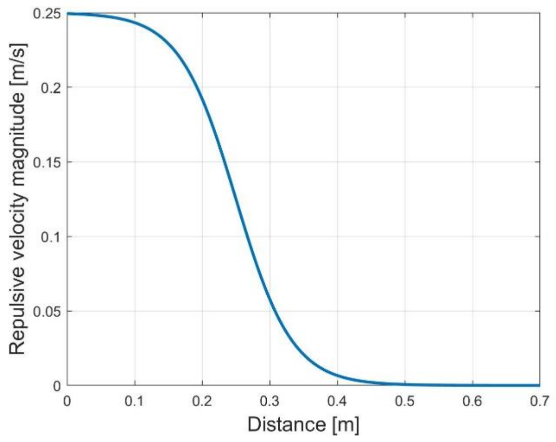


Figure 2.2: Magnitude of the repulsive velocity $\boldsymbol{V}_{i,obst}$ versus the norm of the distance $\boldsymbol{d}_{i,obst}$.

To control properly the robot, the collision avoidance velocities defined in the operative space must be translated in the joint space. The joint velocity vector $\dot{\boldsymbol{q}}$ is calculated by the following equation

$$\dot{\boldsymbol{q}} = \sum_{i=1}^{n_{points}} \boldsymbol{J}_i^{-1} \boldsymbol{V}_{i,obst} \tag{2.2}$$

where $\boldsymbol{J}_i$ is the partial Jacobian associated to the i^{th} point on the robot.

To avoid collisions and achieve the planned task, the trajectory of the TCP must be the result of a combination of the collision avoidance actions and the task action. Given the velocity of the TCP associated to task, $\boldsymbol{V}_{task}$, the resultant action in terms of joint velocity vector is

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_{TCP}^{-1} \boldsymbol{V}_{task} + \sum_{i=1}^{n_{points}} \boldsymbol{J}_i^{-1} \boldsymbol{V}_{i,obst} \tag{2.3}$$

The joint velocity vector $\dot{\boldsymbol{q}}$ permits the robot to avoid collisions and achieved the task when it is possible.

The basic collision avoidance algorithm can be used also with redundant robots. This can be obtained using the well-known null space of the Jacobian,

$$\begin{gathered} \boldsymbol{V}_{rep,TCP} = v_{TCP,obst} \left(\frac{\boldsymbol{d}_{TCP,obst}}{\left\| \boldsymbol{d}_{TCP,obst} \right\|} \right) \\ \dot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger} \left(\boldsymbol{V}_{task} + \boldsymbol{V}_{rep,TCP} \right) + \left(\boldsymbol{I} - \boldsymbol{J}^{\dagger} \boldsymbol{I} \right) \dot{\boldsymbol{q}}_0 \\ \dot{\boldsymbol{q}}_0 = \sum_{i=1}^{n_{points}-1} \boldsymbol{J}_i^{-1} \boldsymbol{V}_{i,obst} \end{gathered} \tag{2.4}$$

where $\boldsymbol{J}^{\dagger} = \boldsymbol{J}_{TCP}^T (\boldsymbol{J}_{TCP} \boldsymbol{J}_{TCP}^T)^{-1}$ is the Moore-Penrose right pseudo-inverse of the Jacobian. The vector $\dot{\boldsymbol{q}}_0$ considers all the points on the robot except for the TCP, for this reason the summation goes from 1 to $(n_{points} - 1)$.

Projecting the vector $\dot{\boldsymbol{q}}_0$ into the null space of the Jacobian, the evasive movements of the $(n_{points} - 1)$ points move the structure of the cobot but the TCP continues performing the planned task. The evasive movements of the TCP are obtained by $\boldsymbol{V}_{rep,TCP}$. In this way it is possible to exploit the redundancy of the robot. In fact, if the obstacle is near the body of the cobot but far from its TCP, the robot can avoid collisions while continuing to follow the planned path.

2.2.2.2 Collision avoidance algorithm with trajectory conditioning technique

In the basic collision avoidance algorithm, the combination of the repulsive velocities and of the task velocity defined the trajectory that the robot must follow. This trajectory is not known a priori, because it depends on the local values and directions of the velocities. The robot can modify the planned path with planar or vertical movements, depending on the relative pose of the robot and of the obstacles. This could be a problem, because the cobot can move along trajectories that are safe because they don't produce human-robot collisions, but they could be considered unsafe by the worker (e.g. a vertical motion that moves the robot near the head of the operator). The collision avoidance algorithm with the trajectory conditioning technique permits to defined desired trajectories of the TCP. The idea behind this technique is to use elements with a well-defined pose producing attractive actions that force the TCP to move along pre-set directions. These attractive objects can be fixed in the workspace or moving. To obtain motions that can be considered safe by the operator, these objects can be associated directly to parts of the human body. For example, associating one of these elements to a hand of the operator, it is possible to force the robot to move in front of the hand instead of above or below it. In this way the robot cannot move near the head of the operator and it is always visible by the worker. Three different attractive objects have been defined:

1) *spherical surface*: given a sphere with centre O and radius r, the attractive velocity is here defined

$$\boldsymbol{a}_{act} = (\|\boldsymbol{p}_{TCP} - \boldsymbol{p}_O\| - r)(\boldsymbol{p}_{TCP} - \boldsymbol{p}_O)$$
$$v_{act} = \frac{V_{MAX,act}}{\left(1+e^{\left(\|\boldsymbol{a}_{act}\|\frac{2}{\rho_{act}}-1\right)\alpha_{act}}\right)} \tag{2.5}$$
$$\boldsymbol{V}_{act} = v_{act}\left(\frac{-\boldsymbol{a}_{act}}{\|\boldsymbol{a}_{act}\|}\right)$$

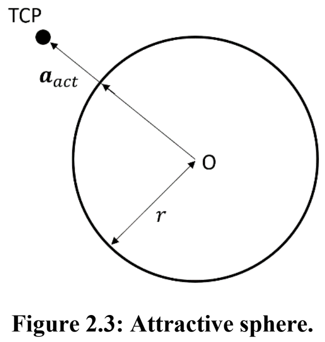


Figure 2.3: Attractive sphere.

$\boldsymbol{a}_{act}$ is the distance vector between the attractive element (in this case the spherical surface) and the TCP. The velocity $\boldsymbol{V}_{act}$ attracts the TCP to the surface of the sphere. $V_{MAX,act}$ is the maximum value of the magnitude of the attractive velocity and ρ_{act} is the distance of influence of the attractive action. α_{act} is a shape factor.

2) *planar surface*: a reference frame is associated to the planar surface and ${}^{world}\hat{A}_{plane}$ is the homogenous matrix of the plane referred to the world reference frame. The projection on the plane of the TCP is used to define the attractive velocity $\boldsymbol{V}_{act}$:

$$
\begin{gathered}
{}^{plane}\boldsymbol{p}_{TCP} = {}^{plane}\hat{A}_{world}\boldsymbol{p}_{TCP} \\
\boldsymbol{v} \text{ is the versor of the axis } z_{plane} \\
\boldsymbol{p}_{TCP,plane} = {}^{world}\hat{A}_{plane}\left({}^{plane}\boldsymbol{p}_{TCP} - \left({}^{plane}\boldsymbol{p}_{TCP}\cdot\boldsymbol{v}\right)\boldsymbol{v}\right) \\
\boldsymbol{a}_{act} = \left(\boldsymbol{p}_{TCP,plane} - \boldsymbol{p}_{TCP}\right) \qquad (2.6) \\
v_{act} = \frac{V_{MAX,act}}{\left(1 + e^{\left(\|\boldsymbol{a}_{act}\|\frac{2}{\rho_{act}}-1\right)\alpha_{act}}\right)} \\
\boldsymbol{V}_{act} = v_{act}\left(\frac{\boldsymbol{a}_{act}}{\|\boldsymbol{a}_{act}\|}\right)
\end{gathered}
$$

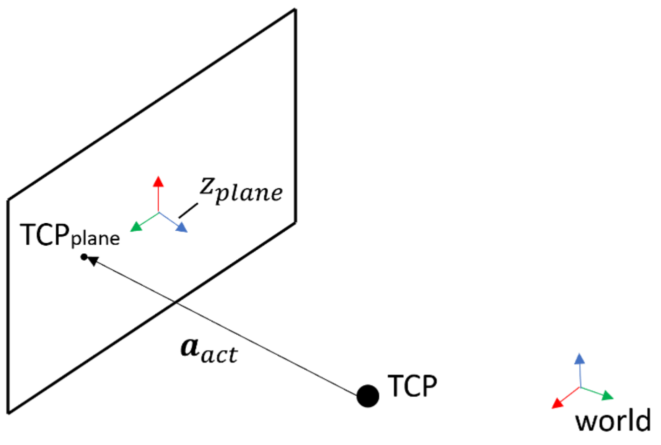


Figure 2.4: Attractive plane.

$\boldsymbol{p}_{TCP}$ is the position vector of the TCP with respect to the world reference frame and ${}^{plane}\boldsymbol{p}_{TCP}$ is the TCP position vector with respect to the plane reference frame. $\boldsymbol{p}_{TCP,plane}$ is the position vector of the projection on the plane of the TCP.

3) *cylindrical surface*: the cylinder has a radius r. A reference frame is associated to the cylinder and ${}^{world}\hat{A}_{cylinder}$ is the homogenous matrix of the cylinder referred to the world reference frame.

The attractive action is calculated defying a point on the axis of the cylinder:

$$ {}^{cylinder}\boldsymbol{p}_{TCP} = {}^{cylinder}\hat{A}_{world}\boldsymbol{p}_{TCP} $$

$\boldsymbol{v}$ is the versor of the axis $z_{cylinder}$

$$ \boldsymbol{p}_{TCP,cylinder} = {}^{world}\hat{A}_{cylinder}\left(\left({}^{cylinder}\boldsymbol{p}_{TCP}\cdot\boldsymbol{v}\right)\boldsymbol{v}\right) $$

$$ \boldsymbol{a}_{act} = \left(\left\|\boldsymbol{p}_{TCP} - \boldsymbol{p}_{TCP,cylinder}\right\| - r\right)\left(\boldsymbol{p}_{TCP} - \boldsymbol{p}_{TCP,cylinder}\right) \quad (2.7) $$

$$ v_{act} = \frac{V_{MAX,act}}{\left(1 + e^{\left(\|\boldsymbol{a}_{act}\|\frac{2}{\rho_{act}}-1\right)\alpha_{act}}\right)} $$

$$ \boldsymbol{V}_{act} = v_{act}\left(\frac{\boldsymbol{a}_{act}}{\|\boldsymbol{a}_{act}\|}\right) $$

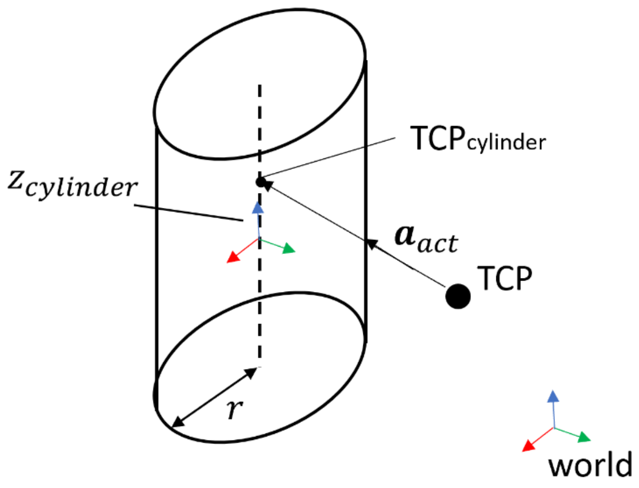


Figure 2.5: Attractive cylinder.

${}^{cylinder}\boldsymbol{p}_{TCP}$ is the TCP position vector with respect to the plane reference frame and $\boldsymbol{p}_{TCP,cylinder}$ is the position vector of the projection on the axis of the cylinder of the TCP. The velocity $\boldsymbol{V}_{act}$ attracts the TCP to the surface of the cylinder.

Associating one of these elements to parts of the human body means to define where the centre of the reference frame of the attractive element is positioned with respect to the related body part. It is also possible to define the orientation of the attractive element, choosing if it is fixed or depends on the one of the related body part.

Given the attractive velocity, the joint velocity vector for no-redundant and redundant robots is respectively

$$ \dot{\boldsymbol{q}} = \boldsymbol{J}_{TCP}^{-1}(\boldsymbol{V}_{task} + \boldsymbol{V}_{act}) + \sum_{i=1}^{n_{points}} \boldsymbol{J}_i^{-1}\boldsymbol{V}_{i,obst} $$

$$ \dot{\boldsymbol{q}} = \boldsymbol{J}^{\dagger}\left(\boldsymbol{V}_{task} + \boldsymbol{V}_{act} + \boldsymbol{V}_{rep,TCP}\right) + \left(\boldsymbol{I} - \boldsymbol{J}^{\dagger}\boldsymbol{I}\right)\dot{\boldsymbol{q}}_0 \quad (2.8) $$

To better understand how the collision avoidance algorithm with the trajectory conditioning technique works, the case of an attractive cylindrical surface associated to the hand of the operator is shown in figure 2.6. The attractive cylindrical surface (in yellow in the figure 2.6) is adopted to obtain planar directions of motion. The red circumference in figure 2.6 represents the volume in which the repulsive action obtained with the basic collision avoidance algorithm is present.

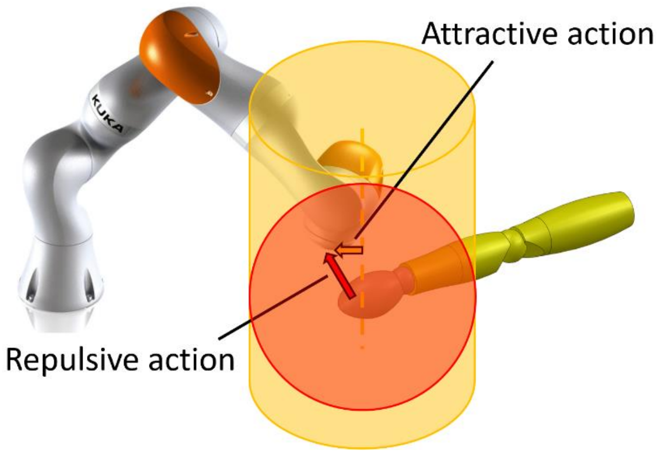


Figure 2.6: The repulsive action pushes the robot vertically. The cylindrical surface generates a planar attractive action.

Applying the basic collision avoidance algorithm and considering the values of the parameters $V_{MAX} = 0.25\ {}^{m}/_{s}$, $\rho = 0.3\ m$ and $\alpha = 6$, the repulsive action related to the human hand generates the streamlines reported in figure 2.7. It is possible to see that on the $X - Y$ plane, the streamlines point outwards from the centre of the circumference, that is the position of the hand, pushing the robot away from the obstacle. The repulsive action has a vertical component, as can be seen by the vertical streamlines on the Z plane. The repulsive action is null when the robot – obstacle distance is larger than ρ.

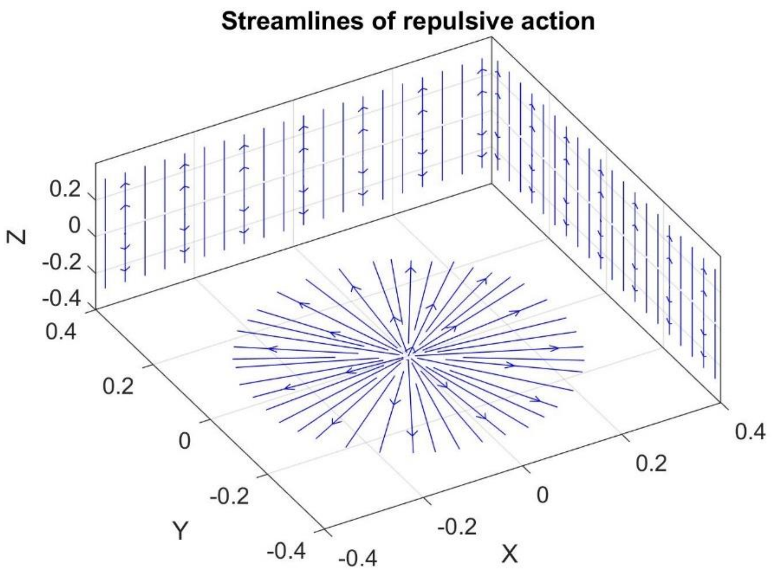


Figure 2.7: Streamlines obtained by the repulsive field.

Instead, the streamlines associated to an attractive cylindrical surface are shown in figure 2.8. These streamlines are obtained with the following values of the parameters of the cylindrical surface: $V_{MAX,act}$ = 0.35 $^{m}/_{s}$, r = 0.3 m, ρ_{act} = 0.4 m and α_{act} = 6.

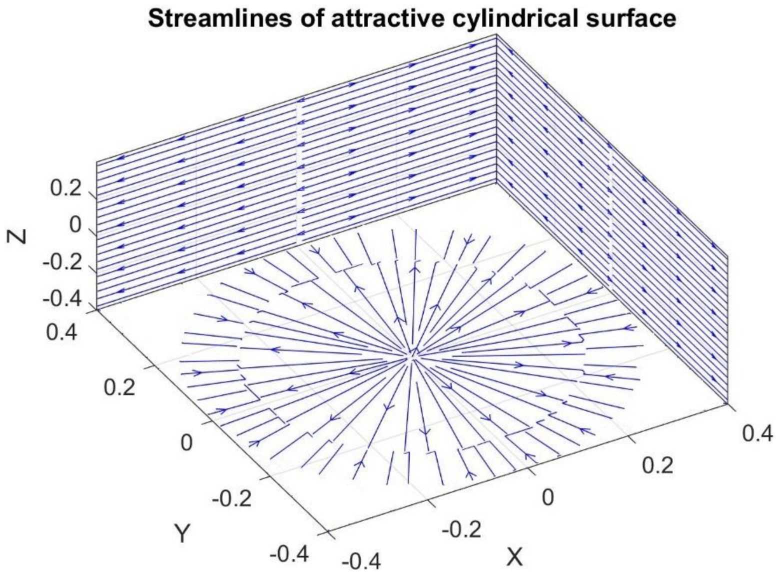


Figure 2.8: Streamlines obtained by the attractive cylindrical surface.

On the $X - Y$ plane, the streamlines associated to the attractive field point to the surface of cylinder, that is the inner circumference. The outer circumference is the area of influence of the cylinder outside its surface. The attractive field has no vertical component, as can be seen on the Z plane where the streamlines are horizontal. Combining these two fields, the resultant streamlines are shown in figure 2.9.

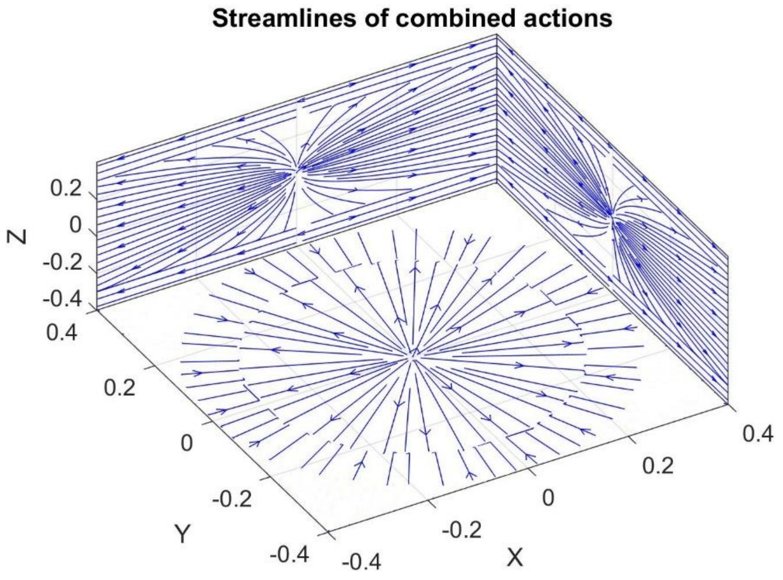


Figure 2.9: Streamlines obtained from the combination of the repulsive and attractive fields.

It can be seen in the Z plane that the combination of the two fields produces streamlines that are almost horizontal. This means that the attractive cylindrical surface is able to force the robot to follow planar evasive movement directions, that were previously chosen by the operator as preferred directions of motion in case of collision avoidance actions.

Applying the basic collision avoidance algorithm, the hand of the operator generates a mainly vertical repulsive action. Implementing the collision avoidance algorithm with the trajectory conditioning technique, the TCP of the robot is attracted by the surface of the cylinder that produces planar movements. The robot moves in front of the hand of the worker. The trajectory conditioning technique permits to defined preferred directions of motion for the robot even in presence of highly dynamic obstacles as human operators. Tests to evaluate the effectiveness and the potentialities of the proposed algorithms are presented and discussed in Chapter 3 (simulation tests) and Chapter 4 (experimental tests).

2.3 Hand-over

The human-robot hand-over consists in human and robot handing objects to one another. This is a "responsive collaboration" operation because it requires that the robot modifies and adapts online its motion to the one of the human operator. Human-robot hand-over is a useful element to study in human-robot interaction. It can be useful both in service robotics (for example to hand food or medicines to patients in a hospital) and in the industrial environment. In the industrial sector the hand-over between human and robot can be implemented in assembly and boxing operations. This type of interaction permits to increase the fluency and efficiency of the human-robot collaboration.

Human-robot hand-over is a trending research topic and different aspects of this kind of human-robot interaction have been studied. The main characteristics elements of a human-robot hand-over task are motion planning, that involves the path planning and the determination of the object transfer point (*OTP*), and the grasping [68-71].

The hand-over motion planning is also closely related to the ergonomic and psychological aspects of the human-robot interaction. In this work, motion planning algorithms have been studied and developed, based on previous works and assumptions about the ergonomics and psychological effort of the operator.

The state of the art related to the motion planning and the developed algorithm will be presented and discussed in the next paragraphs.

2.3.1 State of the art

There are three different types of human-robot hand-over: robot-to-human hand-over, in which the robot gives an object to the operator, human-to-robot hand-over, in which the human worker is the giver and the robot the receiver and bidirectional hand-over, in which the robot and the operator can be both giver and receiver.

The robot-to-human hand-over is the most studied type ([72-83]) and several works defined strategies to accomplish the hand-over. Some of these works developed these strategies starting from the study of human-human hand-over. This permits to obtain useful information and data that are adapted and transferred to the robot-to-human hand-over. In [72, 73], human-human and human-robot hand-over tests show that the operators prefer fast and less precise hand-over tasks to precise and slow ones. The timing of the hand-over is a fundamental element for the perceived fluency of the hand-over. Slow hand-over operations are perceived less natural than fast ones. However, the operators feel safe when the robot slows down slowly when it approaches the OTP. For this reason, it is important to avoid motion of the robot that requires high decelerations. In [74], human-robot hand-over tests have been conducted to verify that the fluency of the hand-over is influenced by the fact that the human operator can infer the robot's intention. In this work, the intention of the robot is communicated to the operator defining an approach position, called carrying pose, different from the hand-over pose. When the robot reaches the carrying position, the operator should understand that the robot is ready for the hand-over task. In the paper it is shown that there are different solutions to communicate to the worker the robot's intent, e.g. using gaze, speech, body movements etc. The experimental results show that there is an improvement of the hand-over fluency when the robot's intent is communicated. In [75], a seamless human-robot hand-over is achieved introducing a delay in the hand-over that increases the awareness of the operator of the robot's gaze and the fluency of the hand-over. Human-human hand-over had been studied in [76] in order to obtain the preferences of the human operator during a hand-over task. These data are then used as inputs to a motion planner that permits to obtain trajectories that are considered natural and appropriate by the operator. [77] is focused on the pose of the object that the robot has to hand to the human. A method to find the appropriate part of the object to be given to the worker is used. The position and orientation of the person with respect to the robot are evaluated from data obtained by a RGBD camera. The motion planner permits to give the object with the right pose, but the operator has to wait to take the object until the robot stops its motion. This reduces the fluency of the hand-over. Furthermore, the planned motion cannot be modified if the operator changes position after the robot starts to move. In [78], HRI constraints have been defined and used as inputs for a motion planner for mobile robot with robotic arm. The constraints are the comfort of the human operator (this is a combination of different aspects such as robot visibility, robot proximity, musculoskeletal comfort for a given motion etc…), space constraints (presence of obstacles) and fluency of the hand-over (related to the time to perform of the hand-over). Given the position of the operator, the motion planner chooses the best trajectory from a trajectories database. This approach permits to accomplish the hand-over task, but it is not reactive to a change of the operator's position after the robot starts moving. A motion planner like the one presented in [77] is described in [79, 80]. Here the inputs of the motion planner are safety, visibility and arm comfort. The motion planner first chooses the OTP, then calculates and controls the robot trajectory. The robot starts moving only after the motion is completely calculated and validated, and this implies slow hand-over operations. Moreover, if the operator moves away from his previous position, the motion planner has to calculate and validate a new trajectory, increasing the hand-over timing and reducing its fluency. In [81], human-human hand-over is studied to analyse movements and trajectories of the givers. The results show that the giver has bell-shaped profile velocities. These results permit to develop a hand-over model, that selects the OTP for the robot and plans a trajectory. Machine learning techniques have been used in [82] to recognize human gestures. Actions are associated to each gesture. In this way a fluent and natural HRI is achieved, but the motion planner is not reactive to human position changes. Also in [83], machine learning techniques and data from human-human hand-over tests are used to train a dynamic system. Here the motion planner is able to plan trajectory in both cases of static and variable positions of the hand of the human operator. In this work, also the orientation of the hand is considered. Unfortunately, this method requires to use ARtags on the hand of the human receiver that can limit its applicability in a real case scenario.

Few works focused on the human-to-robot and the bidirectional hand-over cases. In [84, 85], a bidirectional motion planner based on Dynamic Movement Primitives (DMP) formalism is presented. The DMP approach produces a trajectory composed of two parts, a planned part and a reactive one. The planned part implies to make assumptions about the position of the OTP and a trajectory is planned to reach this point in a certain period. The planned trajectory is initially followed to permit the robot to immediately starts to move. At the beginning, the contribution of the reactive part of the DMP is null. After few seconds, the importance of the planned trajectory decreases, and the contribution of the reactive trajectory becomes more important. In this way, the robot can follow moving target. In these works, only the position of the hand is considered. Experimental tests show, as in [72, 73], that the operator prefers fast hand-over tasks to slow ones and the DMP technique can be a good solution to implement. In [86], a reactive motion planner is presented. The technique used in this paper is similar to the one of DMP. In fact, the motion planner has a planned trajectory part and a reactive part, as in [84, 85]. The reactive part permits the robot to follow the object handled by the operator even if he/she suddenly changes its position. The position of the object is tracked using passive markers. The position of the human worker is not tracked. This means that the planner is reactive when the operator is the giver and the robot the receiver, but the planner used only the planned trajectory part when the robot is the giver. In fact, if the position of the person is not tracked, the robot presents the object and waits in the same position until the operator takes the object. So, the motion planner is reactive in the human-to-robot hand-over, but not in the robot-to-human case. Furthermore, only the position of the object is considered.

In [87], a reactive motion planner for bidirectional human-robot hand-over is described. In this work human-human hand-over tests have been performed and interesting data about the *reaction time* and the *response time* (the time between the giver starts moving and when the hand-over is accomplished) are obtained. The human-human reaction time is 0.425 ± 0.035 *s* and the response time is 1.212 ± 0.051 *s*. These values can be used as reference data in the human-robot hand-over tasks. The planner proposed in this paper is based on the same logic of the DMP, so a planned part of the trajectory helpful to reduce the reaction time and a reactive part to follow dynamic target. Here the target is the position of the hand of the human operator. A method based on machine learning techniques useful to predict the position of the hand is presented in order to reduce the response time. Experimental tests show that the proposed method reduces the response time, but it is still 2 seconds longer than the response time of the human-human hand-over. In table 2.2, a summary of the papers previously presented is reported.

Table 2.2: "Hand-over" summary.

Paper	***Hand-over type***	***Characteristics***	***Limitations***
[72]	Robot-to-human	Human-robot hand-over tests shows that long waiting times reduce the fluency of the hand-over	-
[74]	Robot-to-human	Human-robot hand-over tests conducted to show that communicating the robot's intent to human increases the fluency of the hand-over	-
[75]	Robot-to-human	Results of tests show that forcing the operator to pay attention to the gaze of the robot permits to communicate the OTP and increase the fluency of the hand-over	-
[76]	Robot-to-human	Human preferences during human-human hand-over tasks are collected and used as input in a motion planner for human-robot hand-over	Not reactive planner
[77]	Robot-to-human	Evaluation of the appropriate orientation of the object to hand	Slow and not reactive motion planner
[78]	Robot-to-human	Motion planner that use as input the comfort of the human operator, space constraints and time of the hand-over	Not reactive motion planner

[79, 80]	Robot-to-human	Motion planner that evaluate the OTP and calculates a proper trajectory	Slow motion planner
[81]	Robot-to-human	Human-human tests performed to extrapolate features useful to develop a model that calculates the OTP and plans robot trajectory.	Only the position is considered
[82]	Robot-to-human	Machine learning techniques that permit to identify human gestures	Not reactive motion planner
[83]	Robot-to-human	Motion planner based on machine learning techniques that permit to calculate trajectories that follow the dynamic pose of the hand	ARtags limit the applicability of the system
[73]	Bidirectional	Results of tests conducted to study human-human and human-robot hand-over tasks	-
[84, 85]	Bidirectional	Motion planner based on Dynamic Movement Primitives (DMP) formalism to follow dynamic target	Only the position of the hand is considered
[86]	Bidirectional	Fast reactive motion planner based on an approach like DMP	Only the human-to-robot hand-over is reactive, the robot-to-human is not. Only the position is considered
[87]	Bidirectional	Reactive motion planner provided with a method to predict the OTP	Only the position of the target is considered. Slow hand-over

Looking at the table 2.2, it is clear that there are several motion planners developed for robot-to-human hand-over. These are based on ad-hoc assumptions made on human-human hand-over tests, but they are useful only in the case of robot-to-human hand-over. Few motion planners are useful in the other human-robot hand-over cases and only some of them are able to follow dynamic targets, but they consider only the position of the target. In this thesis, a novel hand-over motion planner is presented. It permits to have fast reactive bidirectional hand-over considering not only the hand position, but its pose. In the next paragraph the developed algorithm is presented and discussed in detail.

2.3.2 Developed algorithms

As highlighted by the analysis of the state of the art, human-robot hand-over algorithms must have some features to obtain a fluent and natural hand-over:

1) the operator should be able to understand when the robot is ready to perform the hand-over task;

2) the hand-over must be performed in a way that is ergonomically suitable for the operator;

3) the robot must be fast in order to reduce waiting times of the operator;

4) the hand-over must be directional, it is important to have robot-to-human and human-to-robot hand-over operations;

5) the robot should be able to follow dynamic target, both in position and in orientation.

The hand-over algorithm presented in this work has these features and these characteristics will be described in detail. It is important to highlight that the following algorithm has as inputs the positions of the human operator and of parts of his/her body (hands, wrists etc…). So, it is necessary to have hardware and software that permit to have this information.

The selected hardware will be presented in the next chapters. Here the assumption that these positions data are the inputs of the algorithms is made.

The motion planner here developed is a reactive one, so it gives the robot the ability to follow dynamic target. The target of this motion planner is a point connected to the hand of the worker, called “virtual hand”. This virtual hand is a point aligned with the forearm of the human body and positioned in front of the real hand of the operator. In figure 2.10 the virtual hand, the forearm and the hand of the operator are shown.

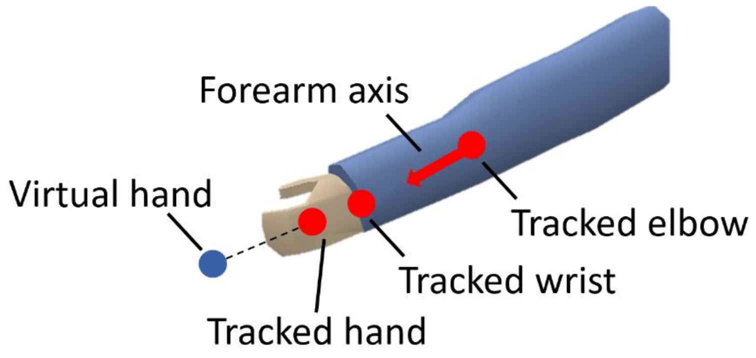


Figure 2.10: Virtual hand, forearm axis and tracked hand of the operator.

There are two reasons behind the choice of the virtual hand: first, most vision systems usually define as “hand” a point which actually is inside the hand of the operator. This implies that the robot cannot reach exactly the position of the hand of the operator without hurting him. So, it is necessary to define a volume around the hand inside which the robot cannot enter and the hand-over is considered accomplished. Using this safety volume, there are cases in which the robot stops its motion when it is above (cf. figure 2.11a) or below (cf. figure 2.11b) the hand.

This limits the fluency of the hand-over. Using the virtual hand, it is not mandatory to define a safety volume.

As a second point, choosing properly the distance of the virtual hand from the real one, it is possible to consider the dimensions of the object to handle (cf. figure 2.11c). Schemes that can give a visual explanation of the reasons related to the adoption of the virtual hand are presented in figure 2.8.

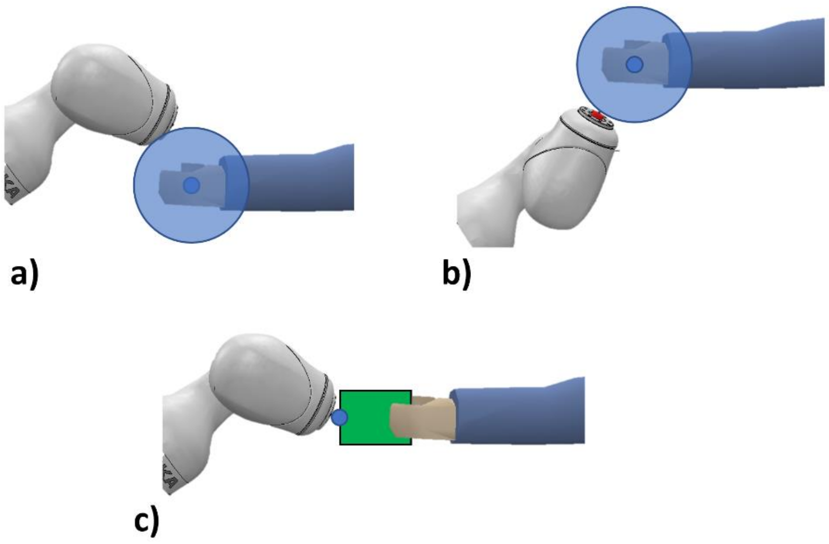


Figure 2.11: a – b) The safety volume limits the positioning precision of the hand-over; c) objects dimensions are took in consideration.

The position of the virtual hand is calculated from the position vectors of the wrist and the elbow of the operator, respectively $\boldsymbol{p}_{wrist}$ and $\boldsymbol{p}_{elbow}$.

From these two position vectors it is possible to define a versor called *forearm axis*

$$\boldsymbol{a}_{FA} = (\boldsymbol{p}_{wrist} - \boldsymbol{p}_{elbow})/\|\boldsymbol{p}_{wrist} - \boldsymbol{p}_{elbow}\| \tag{2.9}$$

The virtual hand is defined at a certain distance from the real hand along the forearm axis (cf. figure 2.10). This distance depends on the dimensions of the object that has to be handled.

Given the position of the virtual hand and of the TCP of the robot (obtained from direct kinematics), it is possible to calculate their relative distance,

$$\boldsymbol{d}_{VH-TCP} = \boldsymbol{p}_{VH} - \boldsymbol{p}_{TCP} \tag{2.10}$$

To control properly the robot and accomplish the hand-over task, an approach based on the artificial potential fields is adopted. An attractive velocity vector $\boldsymbol{v}_{pos}$ permits to push the TCP of the robot towards the target. The velocity $\boldsymbol{v}_{pos}$ is directed from the TCP to the virtual hand. In this way the robot can reach the target and perform the hand-over. It is also able to modify its path in order to follow the virtual hand if it changes its position even after the robot starts to move.

The magnitude of the attractive velocity is a function of the distance $\boldsymbol{d}_{VH-TCP}$,

$$\begin{cases} \dfrac{V_{max,ho}}{\left(1+e^{\left((\rho_{ho,1}-\|\boldsymbol{d}_{VH-TCP}\|)\frac{2}{\rho_{ho,1}}-1\right)\alpha_{ho,1}}\right)}, & if\ \|\boldsymbol{d}_{VH-TCP}\| \leq \rho_{ho,1} \\ \dfrac{V_{max,ho}}{\left(1+e^{\left((\|\boldsymbol{d}_{VH-TCP}\|-\rho_{ho,1})\frac{2}{\rho_{ho,2}}-1\right)\alpha_{ho,2}}\right)}, & if\ \|\boldsymbol{d}_{VH-TCP}\| > \rho_{ho,1} \end{cases} \tag{2.11}$$

where $V_{max,ho}$ is the maximum value of the magnitude of the velocity $\boldsymbol{v}_{pos}$, $\rho_{ho,1}$ is the value of the distance at which the magnitude of the velocity has the maximum value and $\rho_{ho,2}$ defines the length of the right branch of the curve. $\alpha_{ho,1}$ and $\alpha_{ho,2}$ are the shape factors of the two branches of the curve.

An example of curve of the magnitude of the velocity $\boldsymbol{v}_{pos}$ versus the distance $\boldsymbol{d}_{VH-TCP}$ is shown in figure 2.12. The values of parameters used to draw the example curve are $V_{max,ho}$= 0.4 $^{m}/_{s}$, $\rho_{ho,1}$= 0.4 m, $\rho_{ho,2}$= 0.3 m and $\alpha_{ho,1}$ = $\alpha_{ho,2}$ = 7

This hand-over algorithm permits to slow down slowly the robot when it is near the virtual hand. This is in accordance with the statements reported in [73, 74].

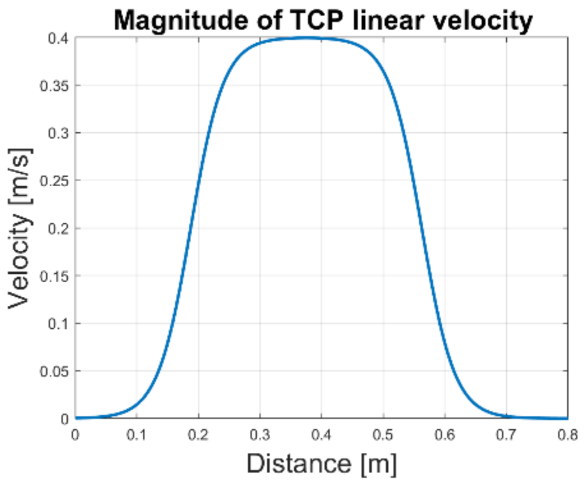


Figure 2.12: Magnitude of the linear velocity versus the TCP-tracked hand distance.

This attractive velocity $\boldsymbol{v}_{pos}$ is the linear component of the TCP velocity. This permits to reach the virtual hand position but it does not control the orientation of the TCP. Controlling the orientation of the TCP of the robot means modifying the orientation of the reference frame associated to the TCP. In figure 2.13a, the reference frame associated to TCP is given. This is a generic reference frame that could be obtained for example using the Denavit-Hartenberg convention. The versors of the axes u, v and w are respectively $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ and $\boldsymbol{\upsilon}$.

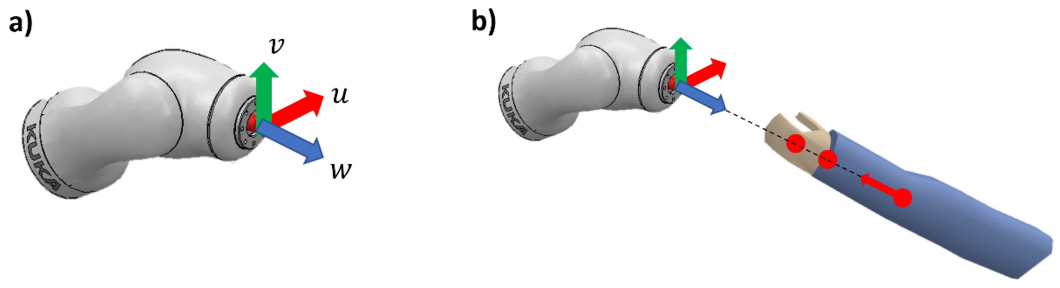


Figure 2.13: a) Reference frame associated to the TCP of the robot; b) the axis w is aligned to the forearm axis for an optimal hand-over.

In this work, the orientation that the TCP has to reach is related to the forearm axis. In fact, the motion planner permits to align the axis w of the TCP reference frame (usually this is the z axis) with the forearm axis (cf. figure 2.13b). The new axis w, called w_R, points in the opposite direction of the forearm axis. The versor $\boldsymbol{v}_R$ and the vertical versor $\boldsymbol{o} = [0 \quad 0 \quad 1]^T$ are opportunely combined to obtain a right-handed reference system,

$$\begin{aligned} \boldsymbol{v}_R &= -\,\boldsymbol{a}_{FA} \\ \boldsymbol{\lambda}_R &= (\boldsymbol{o} \times \boldsymbol{v}_R)/\|\boldsymbol{o} \times \boldsymbol{v}_R\| \\ \boldsymbol{\mu}_R &= (\boldsymbol{v}_R \times \boldsymbol{\lambda}_R)/\|\boldsymbol{v}_R \times \boldsymbol{\lambda}_R\| \end{aligned} \tag{2.12}$$

This reference system is the one that the TCP must follow. Unit quaternions can be calculated from the actual reference frame associated to the TCP and the set one, [88]. The difference between the desired unit quaternion Q_d and the actual one Q_a is calculated as reported in [88],

$$\begin{aligned} Q_d &= \{\eta_d, \boldsymbol{\epsilon}_d\} \\ Q_a &= \{\eta_a, \boldsymbol{\epsilon}_a\} \\ \boldsymbol{e}_0 &= \eta_a \boldsymbol{\epsilon}_d - \eta_a \boldsymbol{\epsilon}_d - \boldsymbol{S}(\boldsymbol{\epsilon}_d)\boldsymbol{\epsilon}_a \end{aligned} \tag{2.13}$$

Where η and $\boldsymbol{\epsilon}$ are the scalar and vector part of the quaternion and $\boldsymbol{S}$ is the skew symmetric operator. An attractive velocity $\boldsymbol{v}_{rot}$ is proportional to the quaternions difference $\boldsymbol{e}_0$ calculated in equation 2.13. This is the angular velocity of the TCP. Given the Jacobian $\boldsymbol{J}$, it is possible to calculate the joint velocities that are the inputs of the controller of the robot,

$$\begin{aligned} \boldsymbol{v}_{rot} &= \boldsymbol{K}\boldsymbol{e}_0 \\ \boldsymbol{v}_{TCP} &= [\boldsymbol{v}_{pos}; \boldsymbol{v}_{rot}] \\ \dot{\boldsymbol{q}} &= \boldsymbol{J}^{-1}\boldsymbol{v}_{TCP} \end{aligned} \tag{2.14}$$

The joint velocity vector $\dot{\boldsymbol{q}}$ permits the robot to follow a target that is dynamic both in position and in orientation.

An important element of this motion planner is that the human-operator is always in charge of the hand-over task. In fact, the robot follows the pose of the virtual hand. The operator can choose any position inside the robot workspace with the orientation that he/she prefers to perform the hand-over. In this way the operator can give/receive the object with his/her arm in the most comfortable and ergonomically suitable configuration.

The last feature presented is related to the possibility to communicate to the operator the intent of the robot to perform the hand-over. As described in [74], the robot can give information about its intent through different communication channels, e.g. gaze, speech etc…. Here a different approach is used. In fact, the idea is to modify the orientation of the TCP in order to follow the operator giving the impression that the robot is aware of the presence of the worker and it is ready to achieve the hand-over task. Given the position of the wrist $\boldsymbol{p}_{wrist} = [p_{wrist,x} \quad p_{wrist,y} \quad p_{wrist,z}]^T$ and of the TCP of the robot $\boldsymbol{p}_{TCP} = [p_{TCP,x} \quad p_{TCP,y} \quad p_{TCP,z}]^T$, the position of a point with coordinates $[p_{wrist,x} \quad p_{wrist,y} \quad p_{TCP,z}]^T$ is considered and the axis w has to point at it. A set reference system for the TCP is then obtained with the equation 2.12. Figure 2.14 shows the robot following the operator in two different positions.

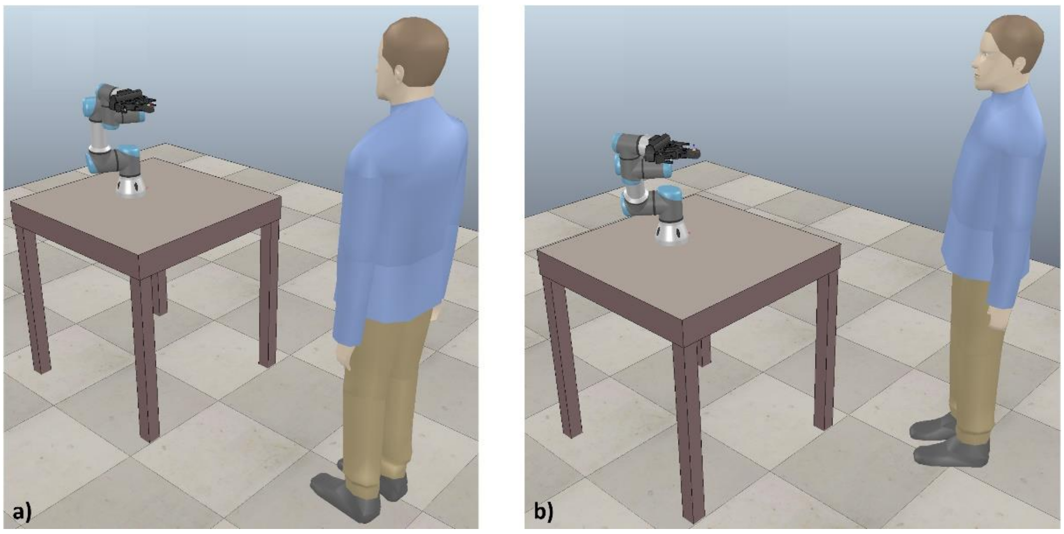


Figure 2.14: The robot follows the human operator in two different positions.

In this way the robot starts to move the TCP when the operator is approaching the workspace of the robot. When he/she is far from the robot workspace, the cobot is still in its Home configuration.

This motion planner has all the features written at the beginning of this paragraph. Preliminary simulation results will be given in the next chapter, whereas the results of experimental tests conducted to show clearly all the features of the motion planner will be presented and discussed in Chapter 4.

2.3 Conclusions

In this chapter, the topics of the collision avoidance and hand-over have been studied and the results of the study of the state-of-the-art has been presented.

The collision avoidance algorithms can be divided in offline (configuration space, probabilistic planners) and online (reducing velocity/stopping approach, optimization problems, artificial potential fields) path planning algorithms. The first ones are used when the environment in which the robot has to work is perfectly known. This kind of algorithms permits to obtain optimal paths avoiding the fixed obstacles present in the workspace. These algorithms are not fast enough to be implemented online. The online algorithms can plan and modify online the path of the robot. The obstacles can be fixed or moving and a limited a priori knowledge of the workspace is needed. The optimization-based algorithms are useful to plan trajectory that respects several constraints. But they permit to avoid slow obstacles without stopping the robot. The artificial potential fields algorithms generate fast evasive motions even in presence of fast obstacles. An important issue is that the direction of the resultant motion is not known a priori. In case of a human-robot interaction, the human operator can perceive as unsafe robot movements that are impossible to predict. A collision avoidance algorithm based on the artificial potential fields approach that permits to define a priori the collision avoidance directions of motion of the cobot is presented in this chapter.

For what concern the hand-over topic, several works related to the robot-to-human hand-over are published. Only few works face the problem of human-to-robot or bidirectional hand-over and some of the algorithms presented in these works are not reactive. They are not able to properly control the robot if the human suddenly moves his/her hand in a different position. It is important to highlight that almost all reactive algorithms consider the object transfer position and not the pose (the only exception is [83]). In this chapter, a bidirectional fast reactive hand-over algorithm able to track the pose of the hand is presented. In this way the operator can choose any position inside the robot workspace to perform the hand-over. In fact, the robot is able to follow the hand of the operator and reach it in any point of the workspace.

In the next chapters, the results of simulation and experimental tests conducted to show the effectiveness and the performances of the proposed algorithms of collision avoidance and hand-over will be presented and discussed.

Chapter 3

Analysis in simulation environment

3.1 Introduction

In the previous chapter, the algorithms of hand-over and collision avoidance have been presented. To verify the effectiveness of these algorithms, a simulation environment has been developed. This is a preliminary phase to study the algorithms before implementing them in an experimental setup. The developed simulation environment permits to a human operator to interact with a simulated collaborative robot. In fact, the movements of the operator are acquired by a vision sensor and a kinematic model of the human body replicates them in the simulation environment. A scene created in the software CoppeliaSim [89] shows the movements of the worker and the cobot and helps the operator to interact with the robot. In figure 3.1 a scheme that illustrates the logic of the simulation environment is shown.

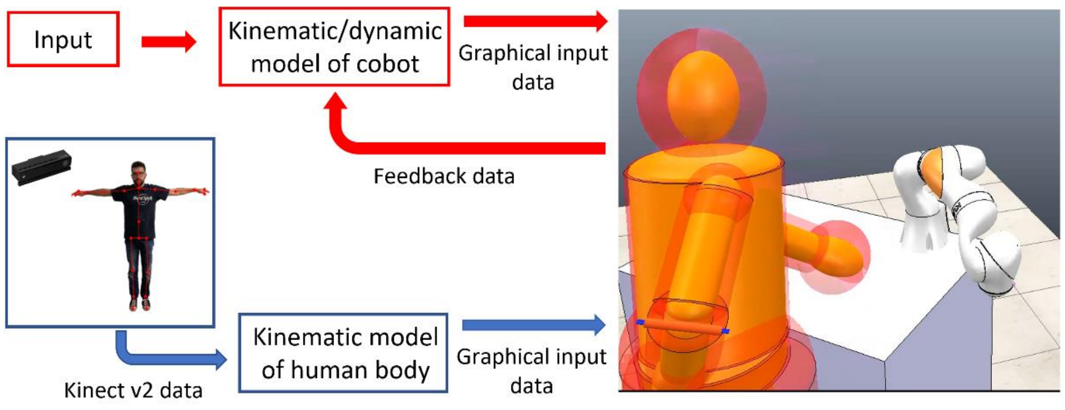


Figure 3.1: Scheme of the logic of the simulation environment.

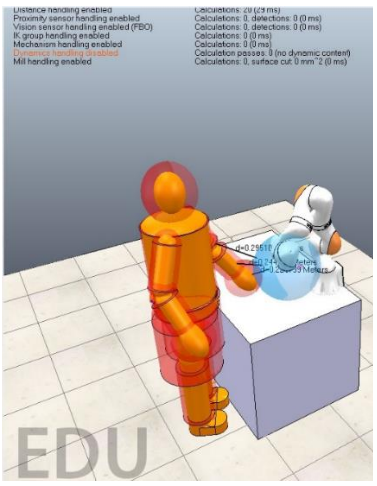
As can be seen in the above figure, the main elements of the simulation environment are a kinematic/dynamic model of collaborative robot and the kinematic model of the human body. The movements of the operator are acquired by the Microsoft Kinect v2 vision sensor. To evaluate the performances of this kind of sensor and to verify if it is suitable for human-robot interaction tasks, the data obtained from the Kinect v2 have been compared to the ones of the Optitrack V120:Trio sensor [90]. In the section dedicated to the vision sensor, the results of the comparison will be presented.

In the following paragraphs, the software/hardware tools used in the simulation phase and all the elements of the simulation environment will be described in detail. The results of tests conducted to verify the effectiveness of the algorithms are reported and discussed in the last paragraph.

3.2 Software/Hardware

3.2.1 Software

Two kinds of software have been used to design the simulation environment: Matlab/Simulink/SimScape from MathWorks and CoppeliaSim. The software from MathWorks were employed to develop the models of cobot and human body, as it will be described in more detail in the following paragraphs. The MathWorks software are used to study kinematic and dynamic models of physical systems. Unfortunately, an important drawback of the software from MathWorks is their limited 3D graphical tools that makes it difficult to reproduce complex scenes that have different elements in them. For this reason, it was decided to use CoppeliaSim for the graphical representation of the movements of the cobot and of the human body. In fact, CoppeliaSim is a powerful software that permits to realize easily complex scene with several elements; furthermore, it has several interesting tools that permit to obtain important information. One of them is the "Minimum distance calculation" function, that calculates the relative distance between two elements in the scene. This tool was used inside the simulation environment to obtain the distances between the human body parts and the robot links. The scene designed in CoppeliaSim and the distances between the end-effector and three parts of the left arm of the dummy calculated with the distance calculation function are shown in figure 3.2.



```
Distance handling enabled               Calculations: 20 (29 ms)
Proximity sensor handling enabled       Calculations: 0, detections: 0 (0 ms)
Vision sensor handling enabled (FBO)    Calculations: 0, detections: 0 (0 ms)
IK group handling enabled               Calculations: 0 (0 ms)
Mechanism handling enabled              Calculations: 0 (0 ms)
Dynamics handling disabled              Calculation passes: 0 (no dynamic content)
Mill handling enabled                   Calculations: 0, surface cut 0 mm^2 (0 ms)
```

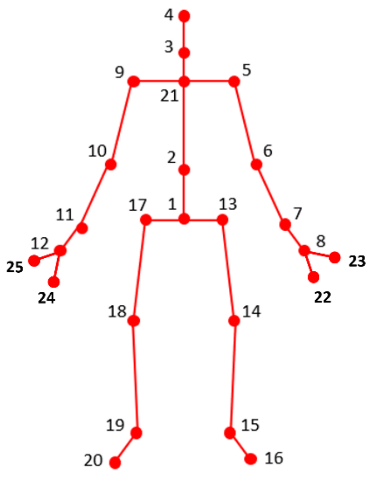
Figure 3.2: Graphical models of the human body and the cobot.

These distances are one of the inputs of the control unit of the robot and they are indispensable data to accomplish the collision avoidance or the hand-over tasks. The distances are calculated every simulation step, as the cobot and the human body change their configuration. The graphical models of the robot and of the body are moved thanks to the position data calculated by the models inside the MathWorks environment. The data are exchanged between the software by Remote API functions [91], that permit to establish a connection between MathWorks software and the CoppeliaSim scene and to send data from one software to another.

3.2.2 Motion tracking

A kinematic model of the human body was developed to reproduce the movements of the human operator who has to interact with the collaborative robot to perform human-robot interaction tasks. The model is described in the next paragraph, and it was developed in order to consider movements acquired by the sensor Kinect v2 by Microsoft. This is a 30Hz RGBD camera that can estimate with infrared rays and time-of-flight technology the distances of objects inside the framed scene from the camera. In this way it is possible to obtain a 3D reconstruction of the framed environment.

Algorithms developed by Microsoft can examine the 3D point cloud obtained from the sensor and identify people present in the scene. The data associated to a person are expressed as position vectors of 25 joints on the human body refer to the reference frame associate to the camera. This information is called *skeleton* and the Kinect v2 is able to track up to six skeletons at the same time. In figure 3.3 the positions of the skeleton joints on the human body are represented and their names are given.



Index	Body part	*11*	Wrist right	*22*	Hand tip left
1	Spine base	*12*	Hand right	*23*	Thumb left
2	Spine mid	*13*	Hip left	*24*	Hand tip right
3	Neck	*14*	Knee left	*25*	Thumb right
4	Head	*15*	Ankle left		
5	Shoulder left	*16*	Foot left		
6	Elbow left	*17*	Hip right		
7	Wrist left	*18*	Knee right		
8	Hand left	*19*	Ankle right		
9	Shoulder right	*20*	Foot right		
10	Elbow right	*21*	Spine shoulder		

Figure 3.3: Indexes of human joints and their name.

The data obtained from the Kinect sensor are imported in the Matlab/Simulink environment thanks to a toolbox designed by MathWorks. The Kinect v2 is a markeless technology and this is an interesting feature for its use in industrial applications. In fact, technologies that need markers involve placing them on the human operators' skin or coverall, possibly limiting their movements. Markerless technologies do not have this drawback but they are less precise than the technologies with markers. So, it is important to verify that the Kinect v2 is precise enough to be used in human-robot interaction applications, where the data obtained from vision sensors are very important inputs for the algorithms that have to ensure the safety of the human-operator. In [107] an analysis of the depth accuracy of the sensor was reported. The results showed that the Kinect v2 has an average accuracy below 2 *mm* at 3 meters from the sensor. But this study was conducted acquiring scenes that contain only static elements. This is not the case of workspaces where humans and robots have to work side by side. In fact, the human workers and the robots are highly dynamic objects, so the accuracy of the sensor can be lower than the one reported in [107]. So, it is important to evaluate the accuracy of the Kinect sensor in dynamic situations, acquiring moving operators. For this reason, a comparison with a vision sensor that use markers was conducted. The chosen sensor is an Optitrack V120:Trio. This is a 120Hz multiple infrared cameras sensor with sub-millimetre accuracy. It uses both passive and active markers. To make the comparison, the movements performed by the operator were acquired by the two sensors. A schematic representation of the experimental setup prepared for the comparison tests is reported in figure 3.4.

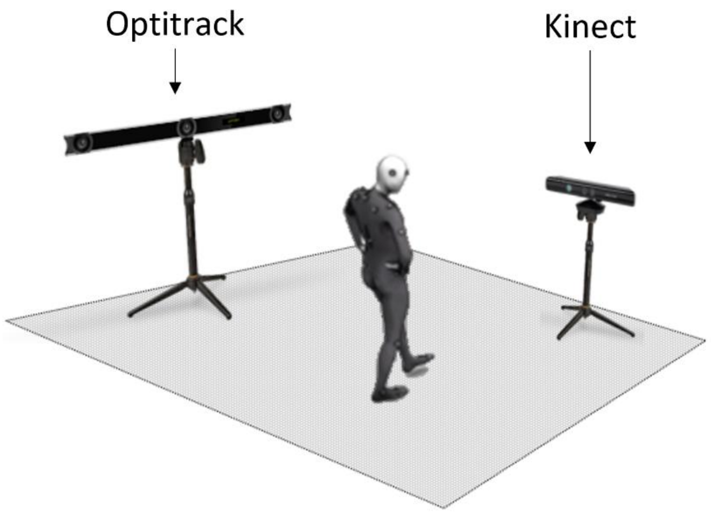


Figure 3.4: Schematic representation of the experimental setup used to compare the Kinect v2 sensor and the Optitrack V120:Trio.

It was decided to compare the results of the sensors related to the positions of the skeleton joints on the arm of the operator. In fact, the upper limbs are the parts most involved in human-robot interaction tasks in the industrial environment.

As can be seen in figure 3.5, two markers are positioned in correspondence of each Kinects joint. From each couple of markers, the position of a virtual intermediate marker was calculated and compared to the positions of the joints obtained from the Kinect v2 sensor.

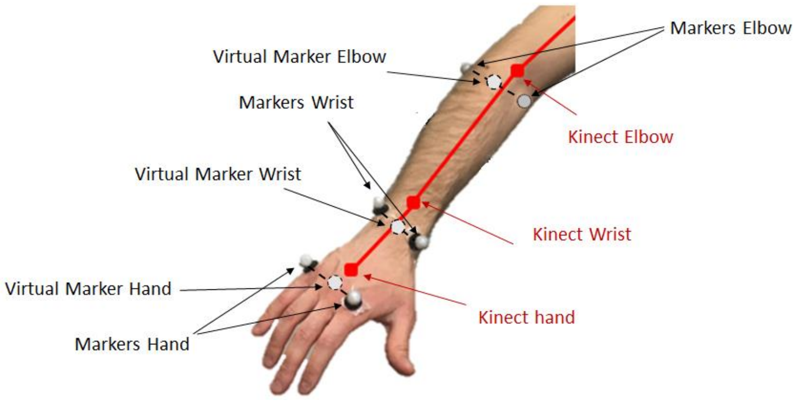


Figure 3.5: Real and virtual markers and the joints of the Kinect v2.

Frames of the movements performed by an operator are shown in figure 3.6. These kinds of movements have been chosen to test the performances of the Kinect v2 in different operative conditions. The first movement (cf. figure 3.6a, 3.6b and 3.6c) are related to the optimal operative conditions, in which the operator is in front of the sensor and the movements are on the image acquisition plane. Figure 3.6a', 3.6b' and 3.6c' show the case in which parts of the body of the operator cover the rest of the human figure. This kind of situation is common when the operator is acquired by the sensor from one side and not frontally. The third movement (cf. figure 3.6a'', 3.6b'' and 3.6c'') permits to test the sensor in case of movements performed along the axis perpendicular to the image acquisition plane. In the following, the results of the third kind of movement will be shown and discussed. The distances between the virtual markers and the Kinect joints for the elbow and the wrist can be seen in figure 3.7.

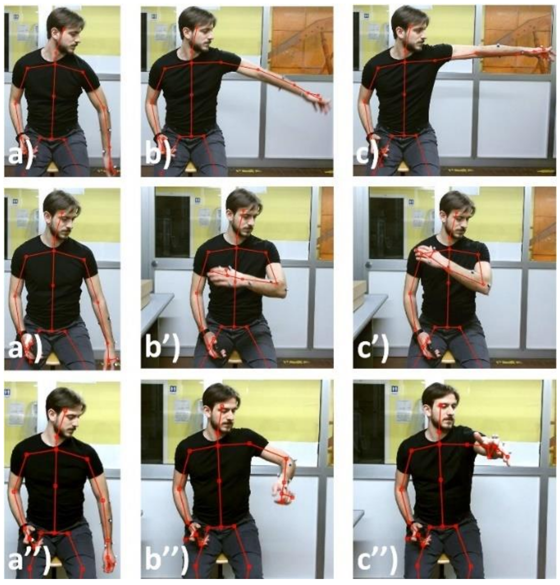


Figure 3.6: Frames of the movements performed to test the Kinect v2.

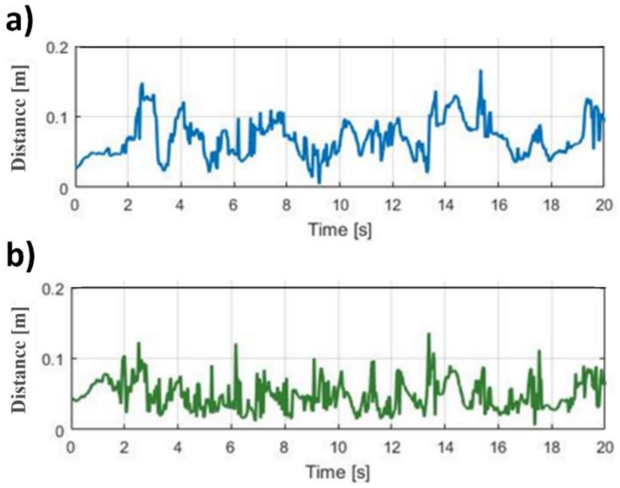


Figure 3.7: a) Distance between the virtual marker elbow and the Kinect joint elbow; b) distance between the virtual marker wrist and the Kinect joint wrist.

In both cases the mean value of the distance is about 0.050 m. This is an offset related to the fact that Kinect places the joints inside the human body, whereas the virtual markers are outside the body and they are above the skin of the operator. So, it is better to use the standard deviation to establish if the Kinect is precise enough to be used in HRI tasks. The standard deviation is 0.02 m and the data obtained from the other movements gave similar results. This value can be considered good enough for HRI tasks. In fact, this is the value obtained from the raw data given by Kinect and it can be improved, for example implementing algorithms that exploit data from multiple sensors and set useful constraints (e.g. fixing the distances between two consecutive joints). The next paragraph will deal with the models of the cobot and of the human body.

3.3 Model

3.3.1 Collaborative robot KUKA LBR iiwa 14 R820 and Universal Robots UR3

The kinematic/dynamic models of two collaborative robots have been developed. The robots are the KUKA LBR iiwa 14 R820 [92] and the Universal Robots UR3 [93]. The first one is a 7 dofs robot and the other one is a 6 dofs robot (cf. figure 3.8).

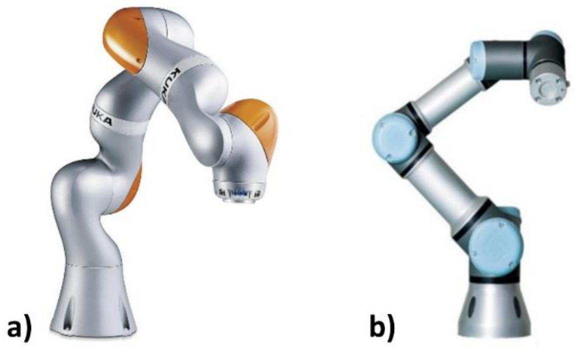


Figure 3.8: a) KUKA LBR iiwa 14 R820; b) Universal Robots UR3.

The Denavit-Hartenberg (hereafter written as DH) convention [94] was used to develop the kinematic model of the robots. Both the DH standard convention and the modified John Craig's one [95] were adopted. In the table 3.1 the values of the standard DH parameters are reported, while in figure 3.9 the reference frames for each link of the KUKA robot obtained with the standard DH convention are shown.

Table 3.1: Standard DH parameters.

UR3				
Link	**α [rad]**	**a [m]**	**d [m]**	**θ [rad]**
1	$\pi/2$	0	0.1519	0
2	0	-0.24365	0	0
3	0	-0.21325	0	0
4	$\pi/2$	0	0.11235	0
5	$-\pi/2$	0	0.08535	0
6	0	0	0.0819	0
KUKA LBR iiwa 14 R820				
Link	**α [rad]**	**a [m]**	**d [m]**	**θ [rad]**
1	$-\pi/2$	0	0.36	0
2	$\pi/2$	0	0	0
3	$\pi/2$	0	0.42	0
4	$-\pi/2$	0	0	0
5	$-\pi/2$	0	0.4	0
6	$\pi/2$	0	0	0
7	0	0	0.163	0

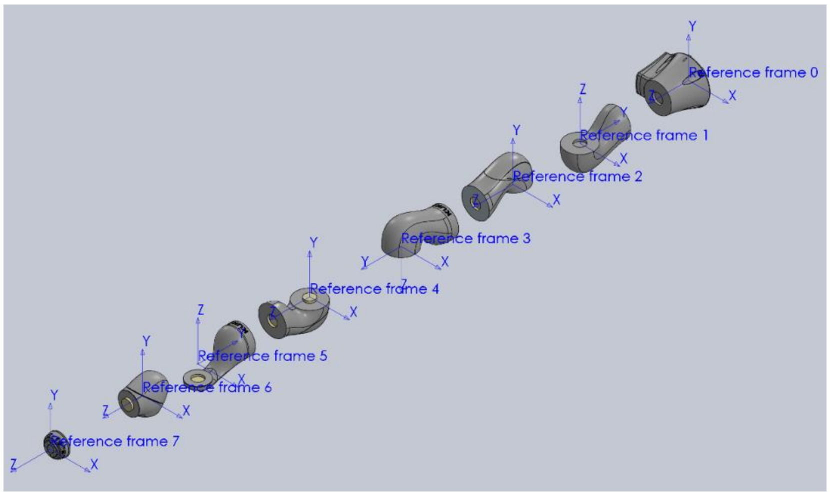


Figure 3.9: Standard DH reference frames of KUKA LBR iiwa 14 R820.

The dynamic models were developed following two different approaches. For the UR3 robot, data related to the mass and the position of the centre of mass of each link were given by Universal Robots company [96]. Given these data, it is possible to obtain the tensor at the centre of mass of each link using any CAD software from official CAD files of the robot. The dynamic data of the UR3 cobot are reported in table 3.2. The data are referred to the reference frames of each link defined with standard DH convention. The dynamic data are the mass m of the link, the position of the centre of mass $\boldsymbol{p}_{COM}$, the central inertia tensor $\boldsymbol{I}_{COM}$ and the orientation matrix of the reference frame of the central axes of inertia with respect to the standard DH reference frame ${}^{DH}A_{COM}$.

Otherwise, for what concern the KUKA robot, no dynamic data were provided by KUKA, and only the mass of the entire robot is known. The dynamic data were calculated from the CAD files of the robot assuming that the mass of the robot is evenly distributed. In this way the average density is calculated from the ratio of the mass and the volume, obtained from the CAD files. The value of the density permits to estimate the mass, the centre of mass and the tensor at the centre of mass of each link using any CAD software.

In table 3.2, the values of the dynamic data for KUKA cobot are reported.

Table 3.2: Dynamic data of the Universal Robots UR3 and KUKA LBR iiwa 14 R820.

UR3				
Link	m **[kg]**	p_{COM} **[m]**	I_{COM} **[kg·m^2]**	${}^{DH}A_{COM}$
Base	2.37	$\begin{matrix}0\\0\\0.06\end{matrix}$	$\begin{matrix}0.003 & 0 & 0\\0 & 0.0031 & 0\\0 & 0 & 0.004\end{matrix}$	$\begin{matrix}0.9927 & 0.0905 & -0.0799\\-0.0902 & 0.9959 & 0.0072\\0.0802 & 0 & 0.9968\end{matrix}$
1	2	$\begin{matrix}0\\-0.02\\0\end{matrix}$	$\begin{matrix}0.0023 & 0 & 0\\0 & 0.0031 & 0\\0 & 0 & 0.0032\end{matrix}$	$\begin{matrix}0 & -0.3782 & 0.9257\\1 & 0 & 0\\0 & 0.9257 & 0.3782\end{matrix}$
2	3.42	$\begin{matrix}0.13\\0\\0.1157\end{matrix}$	$\begin{matrix}0.0038 & 0 & 0\\0 & 0.0031 & 0\\0 & 0 & 0.0032\end{matrix}$	$\begin{matrix}0.9968 & 0 & -0.0797\\-0.0797 & 0 & -0.9968\\0 & 1 & 0\end{matrix}$
3	1.26	$\begin{matrix}0.05\\0\\0.0238\end{matrix}$	$\begin{matrix}0.0008 & 0 & 0\\0 & 0.0094 & 0\\0 & 0 & 0.0096\end{matrix}$	$\begin{matrix}0.996 & 0 & 0.0896\\0.0896 & 0 & -0.996\\0 & 1 & 0\end{matrix}$
4	0.8	$\begin{matrix}0\\0\\0.01\end{matrix}$	$\begin{matrix}0.0005 & 0 & 0\\0 & 0.0006 & 0\\0 & 0 & 0.0007\end{matrix}$	$\begin{matrix}-1 & 0 & 0\\0 & 0.9996 & -0.03\\0 & -0.03 & -0.9996\end{matrix}$
5	0.8	$\begin{matrix}0\\0\\0.01\end{matrix}$	$\begin{matrix}0.0005 & 0 & 0\\0 & 0.0006 & 0\\0 & 0 & 0.0007\end{matrix}$	$\begin{matrix}0 & 0.9996 & 0.0300\\-1 & 0 & 0\\0 & -0.0300 & 0.9996\end{matrix}$
6	0.35	$\begin{matrix}0\\0\\-0.02\end{matrix}$	$\begin{matrix}0.0001 & 0 & 0\\0 & 0.0001 & 0\\0 & 0 & 0.0002\end{matrix}$	$\begin{matrix}1 & 0 & 0\\0 & 1 & 0\\0 & 0 & 1\end{matrix}$

KUKA LBR iiwa 14 R820				
Link	m [kg]	p_{COM} [m]	I_{COM} [kg·m^2]	${}^{DH}A_{COM}$
Base	6.98	$\begin{matrix}-0.0128\\0\\0.0715\end{matrix}$	$\begin{matrix}0.0242&0&0\\0&0.0308&0\\0&0&0.0335\end{matrix}$	$\begin{matrix}0.9976&0&0.0698\\0&1&0\\-0.0698&0&0.9976\end{matrix}$
1	5.1	$\begin{matrix}0\\0.0801\\-0.0338\end{matrix}$	$\begin{matrix}0.0112&0&0\\0&0.0330&0\\0&0&0.0344\end{matrix}$	$\begin{matrix}0&1&0\\-0.9757&0&-0.2190\\-0.2190&0&0.9757\end{matrix}$
2	4.95	$\begin{matrix}-0.0003\\0.042\\0.0588\end{matrix}$	$\begin{matrix}0.0099&0&0\\0&0.0306&0\\0&0&0.0316\end{matrix}$	$\begin{matrix}0&1&-0.0120\\-0.2307&0&0.9730\\0.9730&0&0.2307\end{matrix}$
3	4.18	$\begin{matrix}0\\-0.089\\-0.0291\end{matrix}$	$\begin{matrix}0.0063&0&0\\0&0.0251&0\\0&0&0.0252\end{matrix}$	$\begin{matrix}0&1&0\\0.9627&0&-0.2707\\-0.2707&0&-0.9627\end{matrix}$
4	3.49	$\begin{matrix}0\\-0.0348\\0.0674\end{matrix}$	$\begin{matrix}0.0054&0&0\\0&0.0171&0\\0&0&0.0172\end{matrix}$	$\begin{matrix}0&0&1\\-0.2496&0.9684&0\\-0.9684&-0.2496&0\end{matrix}$
5	2.19	$\begin{matrix}0\\0.14\\-0.0219\end{matrix}$	$\begin{matrix}0.0028&0&0\\0&0.0101&0\\0&0&0.0103\end{matrix}$	$\begin{matrix}0&1&0\\-0.8821&0&-0.4711\\-0.4711&0&0.8821\end{matrix}$
6	2.11	$\begin{matrix}0\\-0.00015\\0.00057\end{matrix}$	$\begin{matrix}0.0035&0&0\\0&0.0047&0\\0&0&0.0049\end{matrix}$	$\begin{matrix}0&0&1\\-0.2&-0.9798&0\\0.9798&-0.2&0\end{matrix}$
7	0.5	$\begin{matrix}0\\0\\-0.0277\end{matrix}$	$\begin{matrix}0.00022&0&0\\0&0.00022&0\\0&0&0.00033\end{matrix}$	$\begin{matrix}0&-1&0\\1&0&0\\0&0&1\end{matrix}$

The kinematic and dynamic data have been used to develop the model of the cobot inside the SimScape environment. The SimScape block model of the KUKA LBR can be seen in figure 3.10.

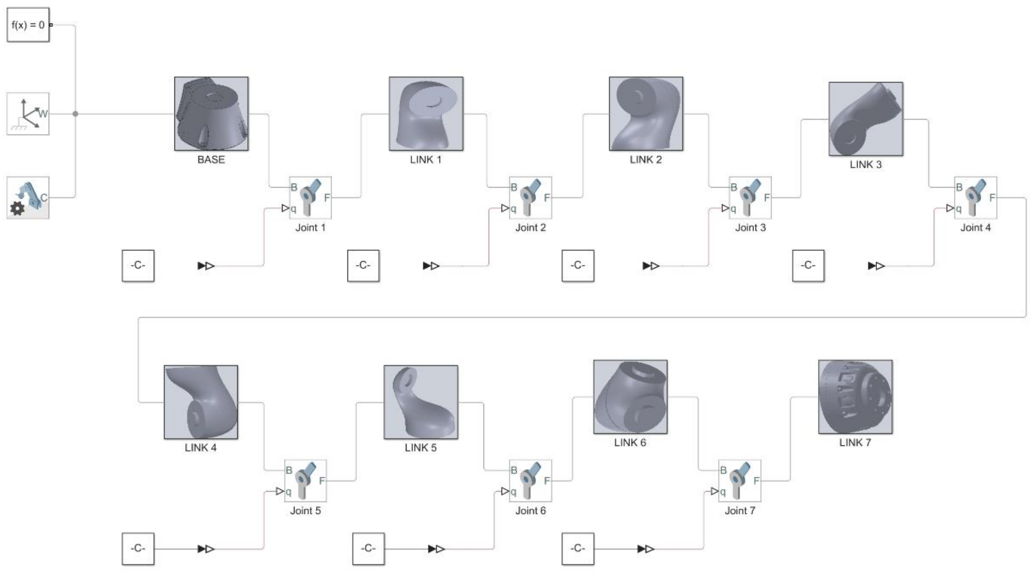


Figure 3.10: SimScape block model of KUKA LBR iiwa 14 R820.

3.3.2 Human body

A simplified kinematic model of the human body was developed using Matlab/SimScape software and its graphical representation (hereafter also called *dummy*) was designed in CoppeliaSim.

The kinematic model was built considering the human body as a group of different kinematic chains. In fact, the upper and lower limbs can be seen as four kinematic chains connected to the torso. The upper limbs joints are the shoulder, the elbow and the wrist, while the rigid bodies are the arms, the forearms and the hands. The shoulder and the wrist are spherical joints, while the elbow is a revolute one. For the lower limbs, the joints are the hips, the knee and the ankles (hips and ankles are spherical joints, knee a revolute one). The rigid bodies are the upper legs, the lower legs and the feet. So, there are 7 dofs for each limb. Six more dofs are related to the torso (three for the translation and three for the rotation). The three translational dofs of the torso permit to place the dummy inside the workspace. Instead, the three rotational dofs make it possible to properly reproduce the movements of the limbs of the human operator, as it will be explained in the following. The SimScape block model in figure 3.11 permits to understand the dofs of each part of the human body and the kinematic chains considered to make the model of the body.

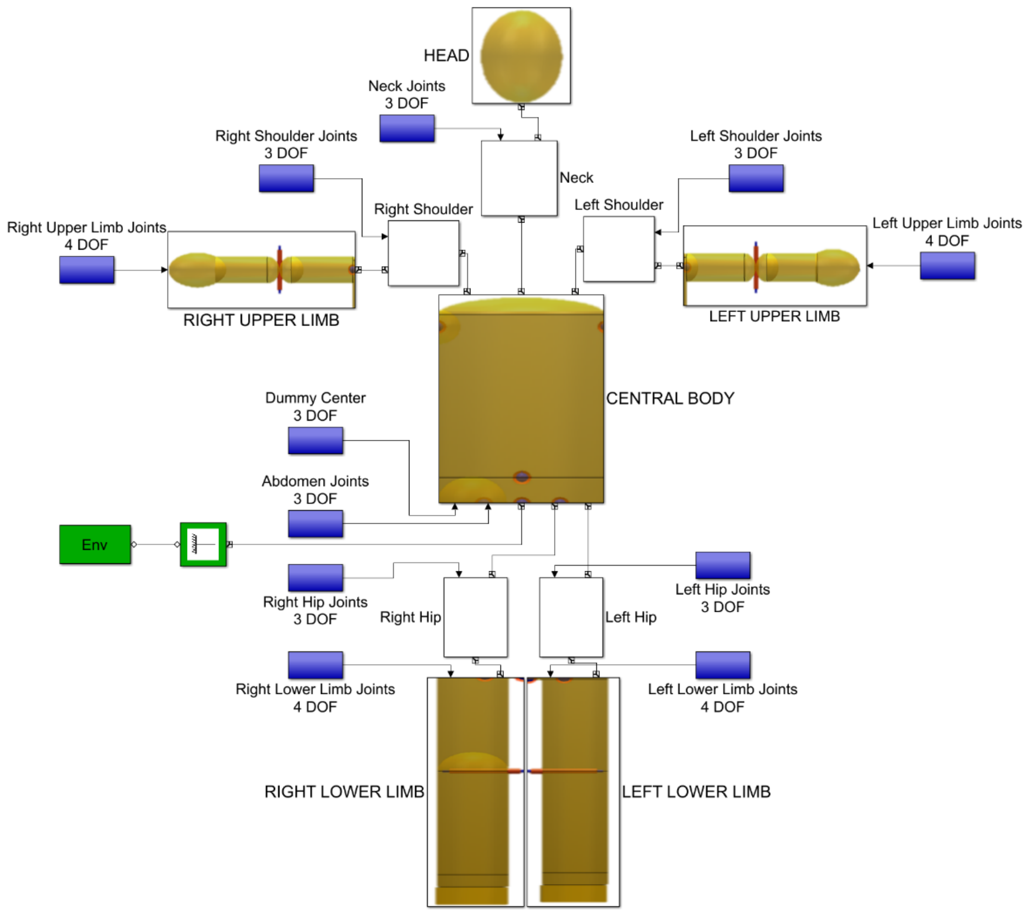


Figure 3.11: SimScape block model of the human body.

It is clear that to properly reproduce the movements of the human operator, the parts of the model must be moved following a rigid kinematic order. In fact, the first to move is the torso, because the kinematic chains of the limbs are connected to it. After the torso, the limbs can be moved independently, each of them following its own kinematic chain. For example, in the case of an arm the first dofs to consider are the ones of the shoulder, then the elbow and the last the dofs of the wrist. In this simplified kinematic model, the dofs of the wrists and the ankles were not considered, because they were not useful for the tests performed to study the collision avoidance/hand-over algorithms.

An important aspect to face is that the movements considered in this paragraph are rotations, but the Kinect data associated to the human body are the positions of the human joints. So, it is important to establish a relation between the position vectors given by Kinect and the rotations considered in the model. To do that, it is important to define the reference frames associated to each part of the human body and an initial configuration. In figure 3.12 the initial configuration of the human body and the reference frames are shown using the graphical model in a CoppeliaSim scene.

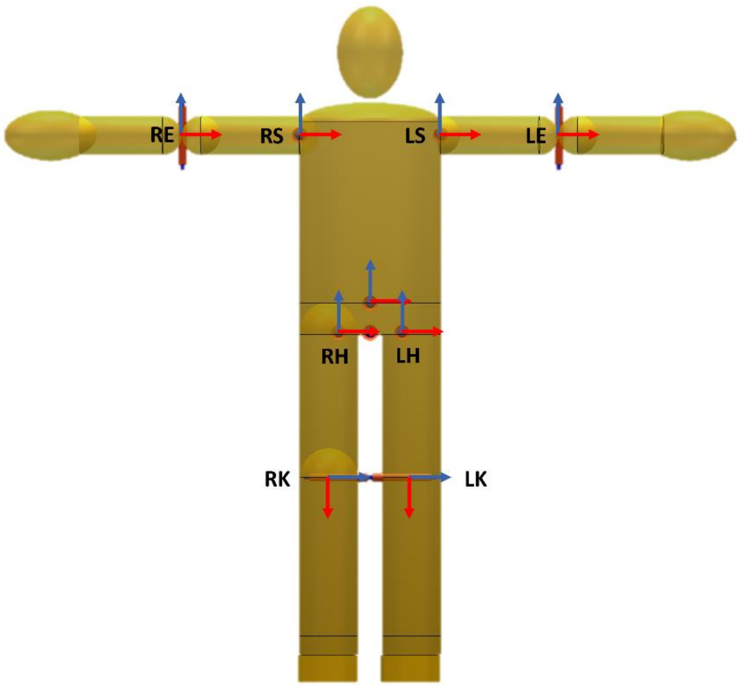


Figure 3.12: Initial configuration of the model and the reference frames.

The strategy adopted to calculate the rotations of the joints of the kinematic model from the Kinect positions data implies to define the orientation matrices associated to the parts of the body. From the matrices of two consecutive links (${}^{FR}A_{i-1}$ and ${}^{FR}A_i$), the matrix ${}^{i-1}A_i$ is calculated and then the related RPY (Roll-Pitch-Yaw) Euler angles. The Euler angles are the inputs of the dummy in CoppeliaSim. In the rest of the paragraph, the strategy to calculate the Euler angles for the torso and the left arm is described. Firstly, it is important to highlight two elements: 1) the joints position vectors are referred to a common reference frame F_R, that will be introduced in the paragraph 3.4; 2) the torso was divided into two parts, a lower one related to the hips and an upper one (the trunk).

To evaluate the rotations of the lower part of the torso (LT), the orientation matrix ${}^{F_R}A_{LT}$ related to this part of the body is calculated in this way (cf. figure 3.13a, where the numbers of the joints are the same of figure 3.3):

$$\begin{gathered}\boldsymbol{x}_0 \text{ is the projection of } (\boldsymbol{p}_{LH}-\boldsymbol{p}_{RH})/\|\boldsymbol{p}_{LH}-\boldsymbol{p}_{RH}\| \text{ on the } x_{F_R} \text{ - } y_{F_R} \text{ plane}\\ \boldsymbol{z}_0 \parallel \boldsymbol{z}_{F_R} \\ \boldsymbol{y}_0=\boldsymbol{z}_0\times\boldsymbol{x}_0 \\ {}^{F_R}A_{LT}=[\boldsymbol{x}_0 \quad \boldsymbol{y}_0 \quad \boldsymbol{z}_0]\end{gathered} \tag{3.1}$$

where $\boldsymbol{p}_{LH}$ and $\boldsymbol{p}_{RH}$ are the position vectors of the left and right hips.

The next orientation matrix to consider is the one of the trunk (T) and it is obtained from the versors here calculated (cf. figure 3.13b):

$$\begin{gathered}\boldsymbol{v}_1=(\boldsymbol{p}_{SS}-\boldsymbol{p}_{SB})/\|\boldsymbol{p}_{SS}-\boldsymbol{p}_{SB}\| \\ \boldsymbol{v}_2=(\boldsymbol{p}_{LS}-\boldsymbol{p}_{RS})/\|\boldsymbol{p}_{LS}-\boldsymbol{p}_{RS}\| \\ \boldsymbol{z}_1 \parallel \boldsymbol{v}_1 \\ \boldsymbol{y}_1=\boldsymbol{v}_1\times\boldsymbol{v}_2 \\ \boldsymbol{x}_1=\boldsymbol{y}_1\times\boldsymbol{z}_1 \\ {}^{F_R}A_{T}=[\boldsymbol{x}_1 \quad \boldsymbol{y}_1 \quad \boldsymbol{z}_1]\end{gathered} \tag{3.2}$$

where $\boldsymbol{p}_{SB}$ and $\boldsymbol{p}_{SS}$ are the position vectors of the spine base and the spine shoulder, and $\boldsymbol{p}_{LS}$ and $\boldsymbol{p}_{RS}$ are the position vectors of the left and right shoulders.

The RPY Euler angles associated to the trunk are calculated from ${}^{LT}A_T$, obtained from ${}^{F_R}A_{LT}$ and ${}^{F_R}A_T$.

For what concerns the left shoulder, the versors used to calculate the orientation matrix are the following:

$$\begin{gathered}\boldsymbol{v}_3=(\boldsymbol{p}_{LS}-\boldsymbol{p}_{LE})/\|\boldsymbol{p}_{LS}-\boldsymbol{p}_{LE}\| \\ \boldsymbol{v}_4=(\boldsymbol{p}_{LW}-\boldsymbol{p}_{LE})/\|\boldsymbol{p}_{LW}-\boldsymbol{p}_{LE}\| \\ \boldsymbol{x}_2=-\boldsymbol{v}_3 \\ \boldsymbol{z}_2=\boldsymbol{v}_3\times\boldsymbol{v}_4 \\ \boldsymbol{y}_2=\boldsymbol{z}_2\times\boldsymbol{x}_2 \\ {}^{F_R}A_{LS}=[\boldsymbol{x}_2 \quad \boldsymbol{y}_2 \quad \boldsymbol{z}_2]\end{gathered} \tag{3.3}$$

where $\boldsymbol{p}_{LE}$ and $\boldsymbol{p}_{LW}$ are the position vectors of the left elbow and the left wrist. From the matrix ${}^{T}A_{LS}={}^{T}A_{F_R}\cdot{}^{F_R}A_{LS}$, the Euler angles are then calculated.

For the elbow, a similar procedure is followed to calculate the related Euler angles:

$$\begin{gathered}\boldsymbol{x}_3 = \boldsymbol{v}_4 \\ \boldsymbol{z}_3 \parallel \boldsymbol{z}_2 \\ \boldsymbol{y}_3 = \boldsymbol{z}_3 \times \boldsymbol{x}_3 \\ {}^{FR}A_{LE} = [\boldsymbol{x}_3 \quad \boldsymbol{y}_3 \quad \boldsymbol{z}_3]\end{gathered} \tag{3.4}$$

The following figure summarizes the strategy adopted to calculate the Euler angles of the upper part of the body.

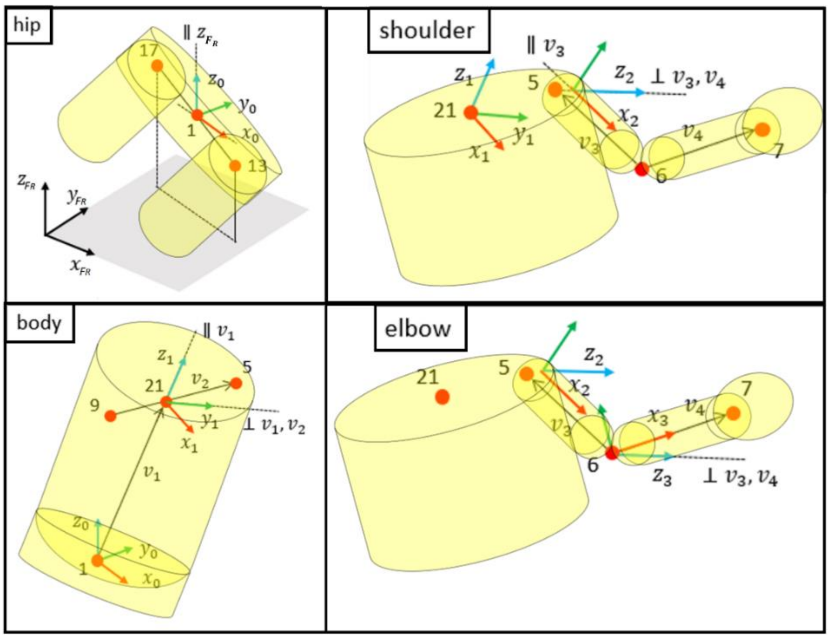


Figure 3.13: Definition of reference frames for the elements of the upper part of the human body.

In this way the dummy can reproduce the movements of the operator, given the joints position vectors estimated by Kinect sensor. The dummy designed using CoppeliaSim is composed of basic elements such as cylinders, sphere and ellipsoids, as can be seen in figure 3.12. This permits to modify the sizes of the different parts of the model in accordance with the anthropometric characteristics of the operator. In this way it is possible to have a dummy for each worker who has to interact with the cobot.

3.4 Spatial matching

In order to perform the tests properly, the data related to the human operator and the robot must be referred to a common frame F_R. The operation conducted to define a common reference frame is called "spatial matching" and the set-up prepared for this operation is shown in the following figure.

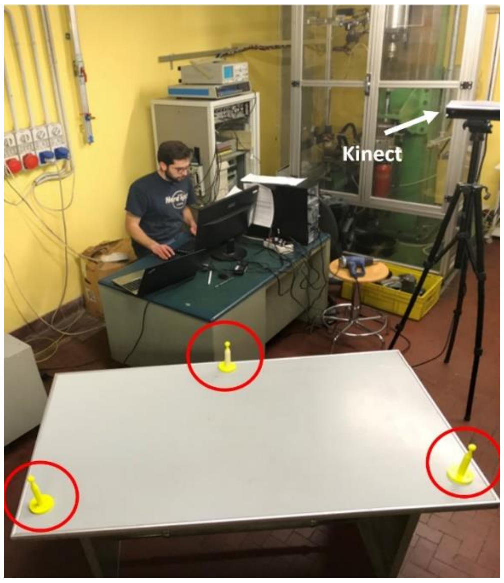


Figure 3.14: Setup used to define the common reference frame F_R.

Three 3D printed narrow-conical supports have been designed to define the common reference frame. The dimensions of the supports are reported in figure 3.15.

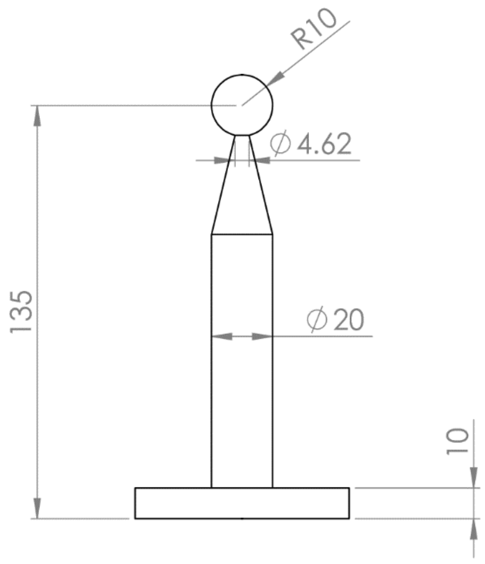


Figure 3.15: Dimensions of the narrow-conical support.

The 3D point cloud representation of the framed field permits to measure the spatial coordinates of objects within the environment. In this way it is possible to have the coordinates of the tips of the supports referred to the Kinect frame and from them define the common reference frame F_R.

In figure 3.16, the 3D point cloud view of the three supports is shown.

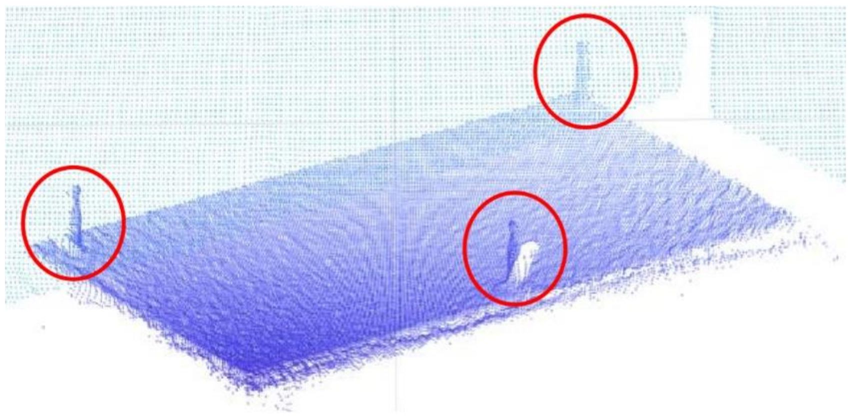


Figure 3.16: 3D point cloud representation of the three supports.

For what concern the simulated robotic manipulator, the reference frame of the base was assumed coincided with the reference frame F_R. The reference frame F_R and the virtual cobot are reported in figure 3.17.

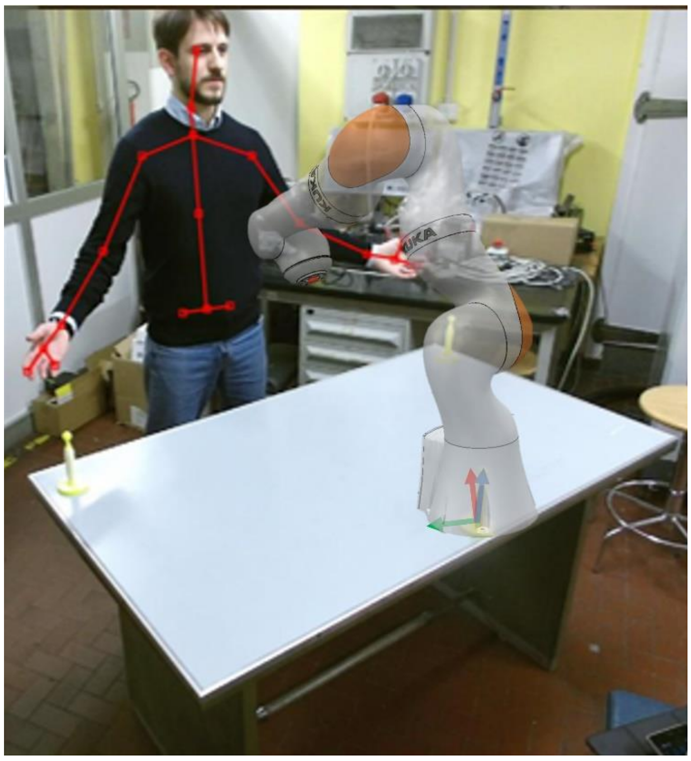


Figure 3.17: Human operator and the virtual cobot in the same environment.

Before discussing the results of the tests, it is important to summarize how the simulation environments works. In figure 3.18 a scheme of the simulation environment is shown.

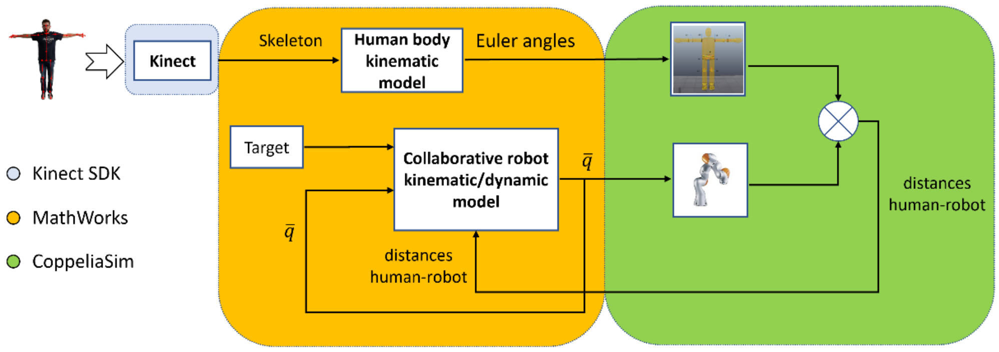


Figure 3.18: Scheme of the simulation environment with the hardware, software and the models.

The different software and the hardware used to simulate hand-over or collision avoidance tasks are shown. The movements of the human operator are acquired by Kinect v2 and the skeleton data associated to the worker are the inputs of the kinematic model of the human body, developed in MathWorks environment. The output of the kinematic model are the Euler angles, that permit to move the dummy in the CoppeliaSim scene. In this scene, the graphic models of the human body and of the robot are presented and the distances between the body parts and the links of the cobot are calculated using a tool of the software CoppeliaSim. This information becomes one of the inputs of the Simulink/SimScape model of the robot. The output of the model of the cobot are the joints positions, that are the inputs of the robot in the CoppeliaSim scene. In this way the graphic model can reproduce the movements of the simulated robot and the "minimum distance calculation" tool can properly estimate the human-robot distances.

3.5 Tests in simulation environment

3.5.1 Collision avoidance

Two kinds of tests of been performed to verify the effectiveness of the collision avoidance algorithms. In both cases, the cobot considered is the KUKA LBR iiwa 14 R820.

The first kind of tests permits to study the basic collision avoidance algorithm. Here the robot must follow a planned trajectory and avoids the worker if the distance between them is less than a certain threshold. In figure 3.19 the planned path that the robot must follow can be seen. The time to follow the path is 8 s.

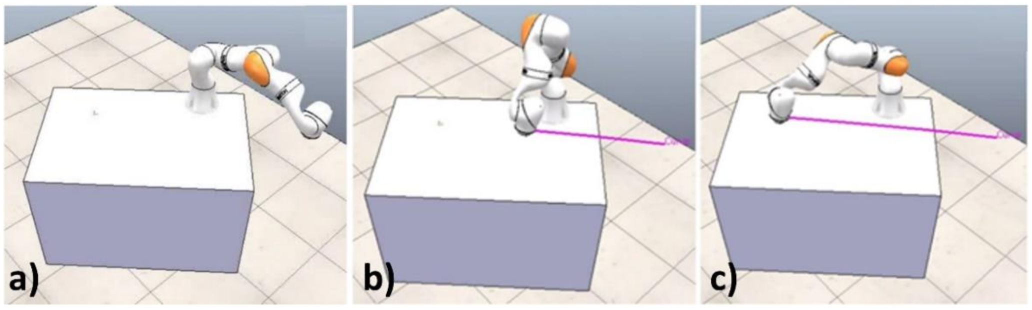


Figure 3.19: The planned path.

The collision avoidance algorithm is active for all the links of the cobot and the distances are calculated between them and the parts of the upper body of the dummy. The collision avoidance threshold ρ is 0.2 m and V_{MAX} is equal to 1 $^{m}/_{s}$. In figure 3.20 it is shown how the robot modified its path in order to avoid collisions with the dummy. The distances between the left hand and the left forearm and the end-effector of the cobot are reported in figure 3.20. The robot initially follows the planned path (figure 3.20a). When the human-robot distance is less than the threshold ρ, the robot modifies the path in order to avoid collisions with the dummy (figure 3.20b). In figure 3.20c, the human-robot distance is larger than ρ and the robot reaches the final destination.

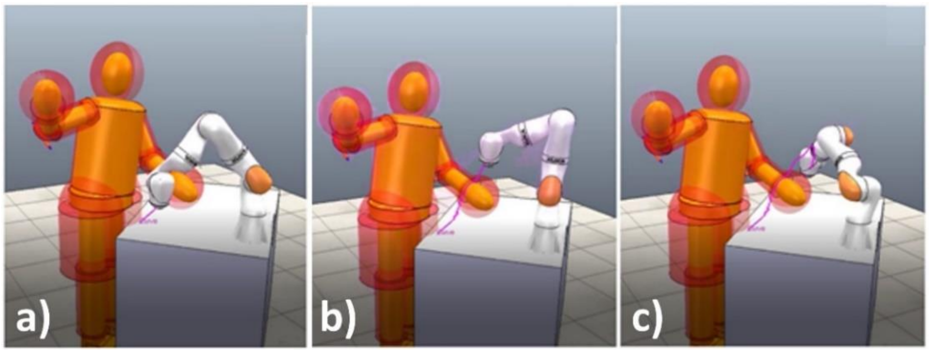


Figure 3.20: Three phases of the test.

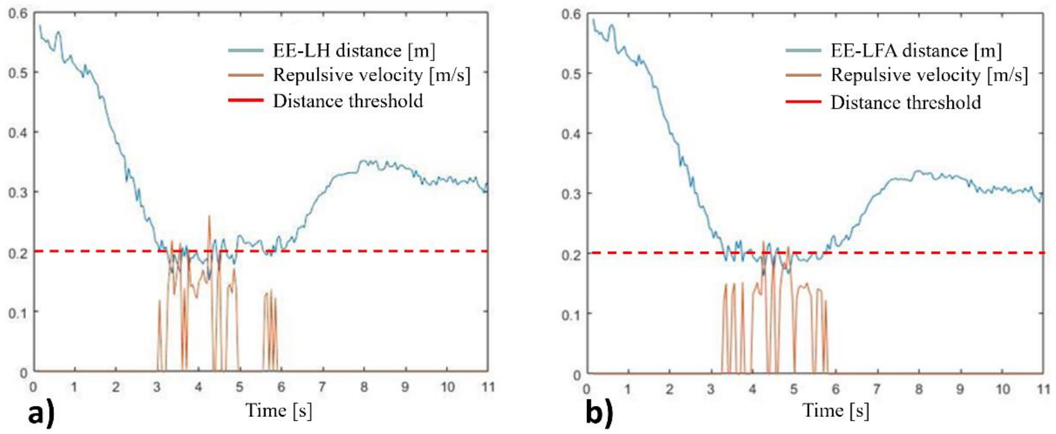


Figure 3.21: a) Distance between the left hand and the end-effector and the repulsive velocity; b) distance between the left forearm and the end-effector and the repulsive velocity.

It can be seen in figure 3.21 that the repulsive velocity is larger than zero when the human-robot distance is less than the threshold. It is also possible to notice that the distance initially decreases and then remains constant, after the repulsive velocity grows. This means that the basic collision avoidance algorithm works effectively and it is able to avoid collisions between the dummy and the robot.

The second test permitted to verify the collision avoidance algorithm with trajectory conditioning technique. In this test, an attractive cylinder is associated to the right hand of the human worker. The test was conducted to show that using the attractive cylindrical surface it is possible to properly modify the path of the end-effector of the robot in order to move it in certain areas or in certain directions that can be considered more comfortable by the human operator. In this case it was decided to adopt a cylindrical surface in order to obtain planar collision avoiding movements instead of vertical ones. The cylinder is placed in correspondence of the hand of the operator and its orientation is associated to the orientation of the right forearm of the human worker. The radius r of the cylinder is 0.4 m and the peak magnitude $V_{MAX,act}$ of its attractive velocity is 1.5 $^{m}/_{s}$. The values related to the repulsive velocity are the same of the previous test. In order to have a clear visualization of the results, a simplified representation of the dummy, the robot and the lines of the paths in Matlab environment was given. The Peter Corke's Robotics Toolbox [97] was used to obtain the graphical model of the collaborative robot In figure 3.22 a schematic representation of the human operator, the Corke graphical model of the robot and the attractive cylindrical surface associated to the worker's hand are shown. The human model is on the upper left side of the figure and it is divided in three parts. The yellow part is the right arm, the green one is the left arm and the red one is related to the torso and the head. The black line is the path of the robot obtained by the presence of the attractive cylindrical surface. The orange line is the path of the hand of the operator.

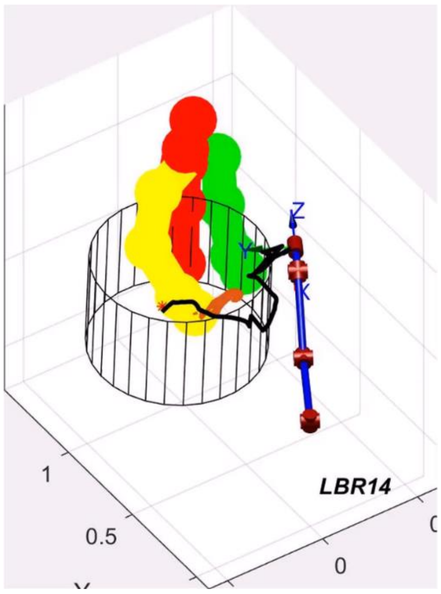


Figure 3.22: Attractive cylindrical surface associated to the right hand of the human operator.

The different paths obtained with and without the trajectory conditioning algorithm are shown, by three different points of view, in figure 3.23. The red line is the path obtained with the basic collision avoidance algorithm. As can be clearly seen in the image in the top right corner of figure 3.23, the basic collision avoidance algorithm generates vertical evasive movements and the TCP of the robot is always above the right hand of the operator. The algorithm worked correctly avoiding any collisions between the operator and the robot, but the resultant path of the TCP is not predictable a priori and could be perceived as unsafe by the worker. In fact, if the human operator wanted to move the hand away from the cobot, he/she could only retract the hand, because the robot is in front and above the worker. Using the trajectory conditioning technique, it is possible to define a priori the directions of the evasive movements. The directions can be chosen in order to give the operator a greater freedom of movement if he/she wanted to move the hand away. Choosing an attractive cylinder, the evasive movements will be planar. The path obtained with the trajectory conditioning technique is the black line in figure 3.23. In this figure, the path obtained without the attractive cylindrical surface is the red line and the one obtained with the trajectory conditioning technique is the black line.

It can be seen in the images on the right that the TCP moved in front of the right hand of the operator and no vertical displacements above the hand have been obtained. The cobot retracted itself and the operator could move the hand vertically, because this direction is not any longer barred by the presence of the robot.

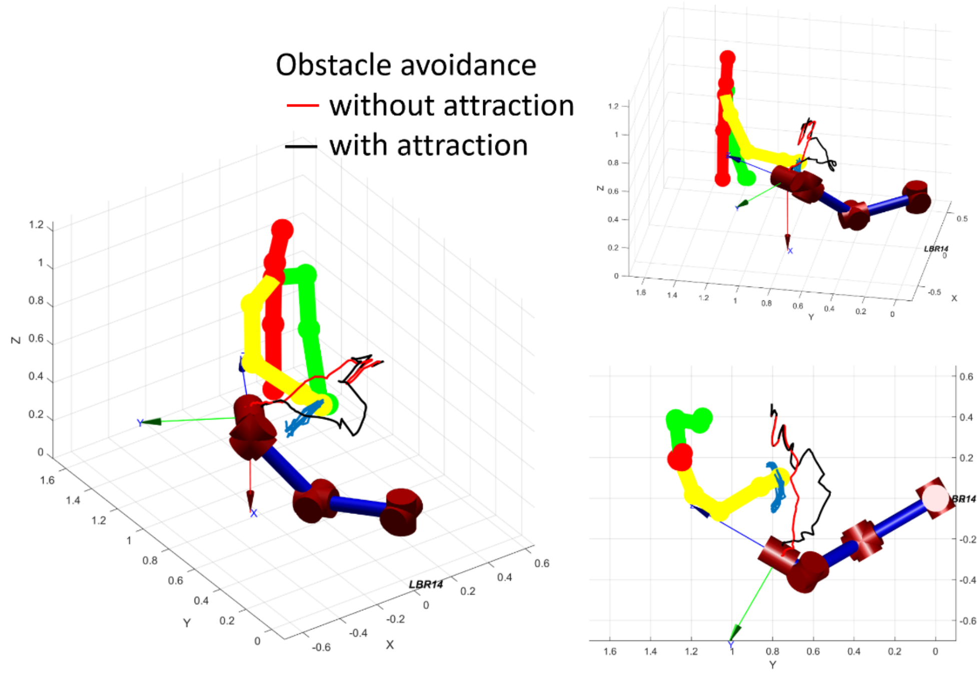


Figure 3.23: Different views of the two paths.

These results show that the trajectory conditioning approach works properly and permits to obtain collision avoidance movements in the desired directions.

3.5.2 Hand-over

To show the effectiveness of the hand-over algorithm, the results of four tests will be presented in this paragraph.

In these tests the operator moved his hand in four different positions inside the workspace of the robot. The robot is the KUKA LBR iiwa 14 R820. The Corke graphical model of the robot and the skeleton representation of the upper part of the human operator are shown in figure 3.24. Here the grey sphere represents the workspace of the robot and the other four coloured spheres are positioned in correspondence of the four distinct hand-over positions considered in the rest of this paragraph.

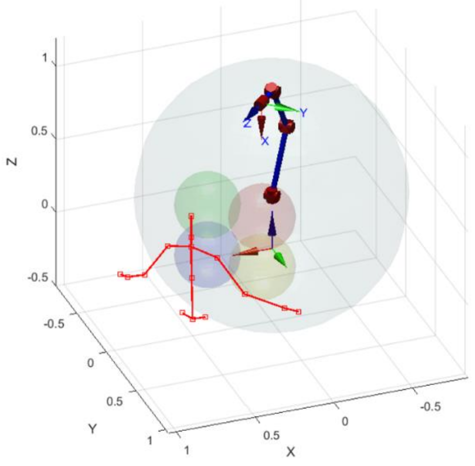


Figure 3.24: Skeleton of the upper part of the human body, the model of the robot and the four positions for the hand-over.

The tests were conducted using the following values for the hand-over parameters:

$$V_{max,ho} = 0.25 \ {}^{m}/_{s}$$

$$\boldsymbol{K} = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 10 \end{bmatrix} 1/s$$

The paths of the robot and of the hand in the four hand-over actions are shown in figure 3.25. In the images on the right, the four hand-over tasks are reported. The KUKA cobot was able to reach the hand of the operator in all cases. This means that the hand-over algorithm controlled properly the robot with a moving target. The ability of following moving target is clearly visible in the hand-over labelled with 4. In the left image in figure 3.25 (where the paths of the end-effector of the cobot and of the hand are reported), the yellow line shows the path of the hand during the hand-over 4. It is clear how the path of the hand is initially linear and then suddenly moved to the right. The robot was able to reach the hand of the operator even in this case, proving the effectiveness of the hand-over algorithm.

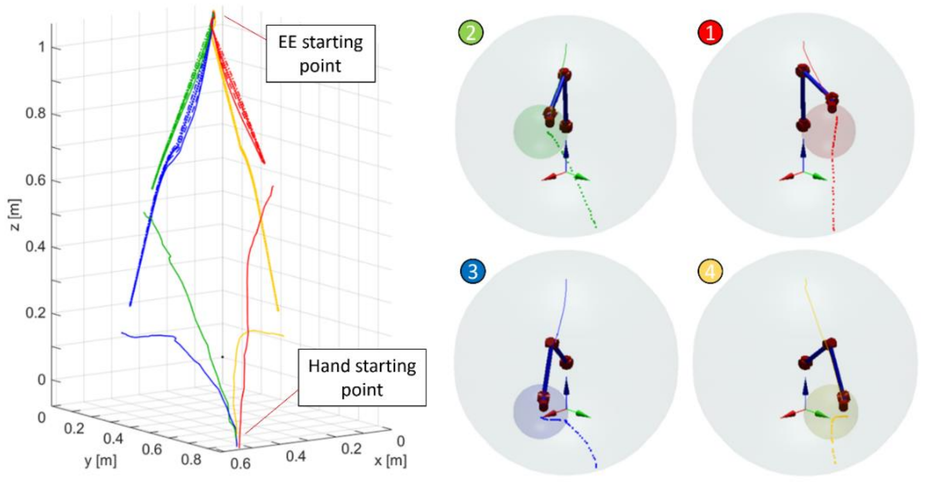


Figure 3.25: On the left, a representation of the paths of the hand and the end-effector. On the right, frames of the movements in the four different cases.

The graph on the left side of the figure 3.26 gives information about the time spent by the robot to reach the hand-over position in the two different cases shown in the images on the right. The cobot starts to move after the hand enters the robotic workspace (in both cases the hand enters the workspace between 0.5 s and 1 s). After one second, the operator stops the hand in the exchange positions (between 1.5 s and 2 s). The hand-over is achieved when the robot reaches the exchange positions.

In the hand-over labelled with 1, the robot stops at time 2.85 s. So, the response time (the time between the hand of the operator enters in the workspace and the robot stops) is 2.2 s. Instead in the hand-over 3, the response time is 4 s. In both cases the hand-over actions are not as fast as the human-human hand-over because of the low value of $V_{max,ho}$ adopted in these tests.

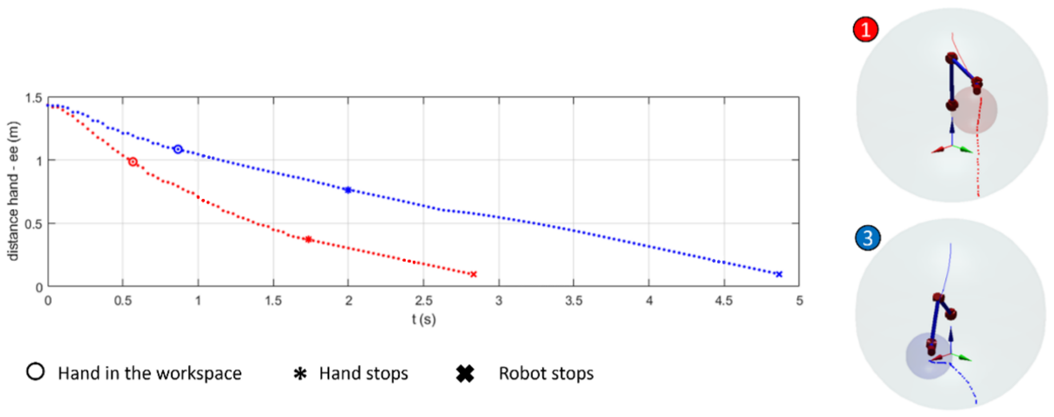


Figure 3.26: On the left, the time spent by the robot to reach two different hand-over positions. On the right, the two hand-over cases considered.

It is important to highlight that the tests conducted using the simulation environment had the goal to show the effectiveness of the algorithms. A deeper analysis of the algorithms will be presented in the chapter related to the experimental phase of the work.

3.6 Conclusions

In this chapter, the simulation phase of the work was presented. This was a fundamental step to test the algorithms of collision avoidance and hand-over presented in Chapter 2 before using them in a real experimental setup.

The simulation environment was designed using software from MathWorks and Coppelia companies. The applications can exchange data thanks to Remote API functions by Coppelia. Models of two collaborative robots and of the human body were developed in order to simulate tasks that imply human-robot interaction. The kinematic model of the human body permits to replicate the movements of a human operator given as inputs the positions of human joints obtained from Kinect v2 sensor. This is a RGBD sensor that gives a 3D point cloud representation of the framed environment. A dedicated software can estimate the positions of 25 joints on the bodies of people present in the framed scene. The kinematic/dynamic model of the robots are used to understand the behaviour of the cobots in case of applications that require collision avoidance or hand-over tasks.

Graphical models of the robot and the human body present in a CoppeliaSim scene permit to visualize the movements of the robot helping the human operator to interact with the cobot.

Tests were conducted to verify the effectiveness of the algorithms. The results of the tests showed that the collision avoidance (both with and without trajectory conditioning technique) and the hand-over algorithms work properly and can control effectively collaborative robots.

In the next chapter, the experimental setup created to develop collaborative robotics applications is presented and the results of tests conducted to study and evaluate the performances of the algorithms presented in the previous chapter will be discussed.

Chapter 4

Experimental phase

4.1 Experimental set-up

A collaborative robotics workspace was developed to verify the applicability and the performances of the algorithms presented in Chapter 2. In fact, the simulation tests described in Chapter 3 showed the effectiveness of the algorithms, but the problems related to the implementation of the algorithms in an experimental scenario were not considered. These problems can be occlusions of the vision sensor, communication delays, variable light conditions, errors in motion tracking etc…. All these problematics could undermine the effectiveness and the performances of the algorithms. In the next sub-paragraphs, the hardware and the software tools used in the collaborative robotics workspace will be described. A photo of the collaborative robotics workspace is shown in figure 4.1.

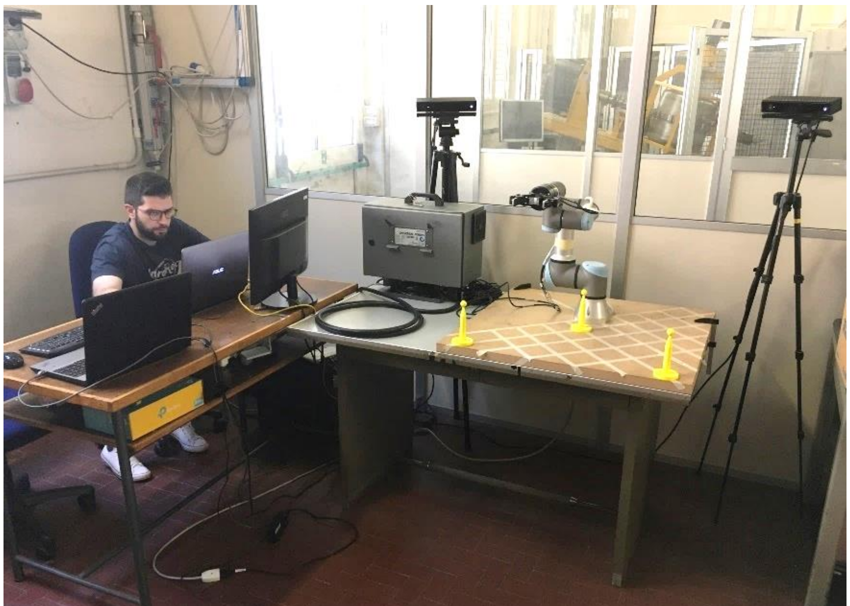


Figure 4.1: Photo of the collaborative robotics workspace.

4.1.1 Hardware

A scheme of the hardware used in the experimental set-up is reported in figure 4.2.

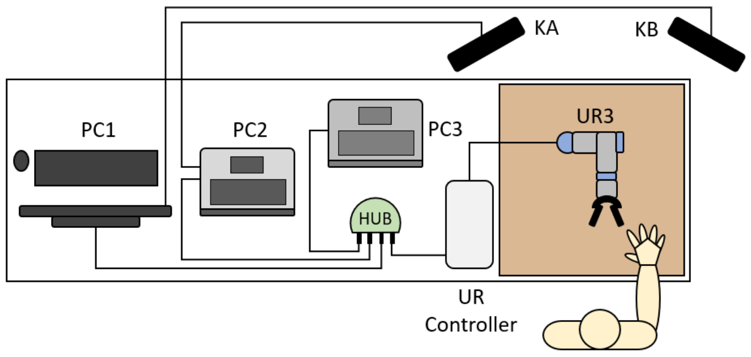


Figure 4.2: Scheme of the hardware used in the experimental set-up.

The collaborative robot used in this experimental set-up is the Universal Robots UR3, CB3.1 series. To perform hand-over task and exchange objects between the robot and the human operator, the robot was equipped with a Robotiq Gripper 2F-85 [98].

To track the movements of the human operator, a vision system based on two Microsoft Kinect v2 sensors is used. The choice of using two Kinect v2 sensors was made to overcome problems related to the occlusions of the sensors. In fact, using only one vision sensor, information related to the movements of the operator could be absent or unreliable in case of occlusions of the sensor. Therefore, the control algorithms of the robot could not control properly the cobot and an incorrect and dangerous behaviour of the robotic manipulator could be obtained. A possible source of occlusions is the robot itself, because it can position itself between the human worker and the vision sensor. This situation was not considered in the case of the simulation environment, where only one Kinect sensor was used. In fact, the robot was only simulated. To overcome the problem of occlusions, a standard solution is to use multiple vision sensors. In this way, if one sensor is occluded, the others can be used to obtain reliable motion data of the operator. The information of each sensor must be then combined in order to give correct data to the control algorithms of the robot. The procedure implemented to combine the data from the two Microsoft Kinect is described in the next sub-paragraph.

A problem related to the use of multiple Kinect v2 sensors is the large amount of data generated by Kinect v2 sensor. In fact, one single PC is not able to handle the information of two or more Kinect v2. So, it is necessary to use several PCs, one for each Kinect v2. This hardware limitation related to the use multiple Kinects leaded to the choice of using two sensors. In this way, only two Pcs are needed to handle the data from the Kinects.

The characteristics of these two PCs are:

1) PC1: i7-6700, 32 GB RAM, NVIDIA GTX 970. Hereafter, PC1 is referred as “PC-Master” in the following;

2) PC2: i7-7500U, 16 GB RAM, NVIDIA GTX 950M. PC2 is referred as “PC-Slave” in the following paragraphs.

A third PC is used to communicate with the UR Controller and control the cobot. The specifications of the PC3 are: i7-6700 HQ, 16 GB RAM, NVIDIA GTX 960M. In the rest of the chapter, this PC is referred as “PC-UR”. All the three PCs are Windows based.

To permit the communication between the three PCs and the UR Controller, a Wireless Router (HUB) 300 Mbps with 4 Ethernet ports is used. The communication is based on TCP/IP network.

In the next sub-paragraph, the software and the control architecture implemented in the robotic system will be presented and described.

The Kinects are placed behind the robot and their principal axis intersect approximately at the centre of the human workspace, that is the space measured considering the arms and the hands as the radius of two sphere centred on the shoulders (cf. figure 4.3). In this way the sensors can acquire the collaborative workspace from different angles, making the vision system architecture more robust to sensor occlusions.

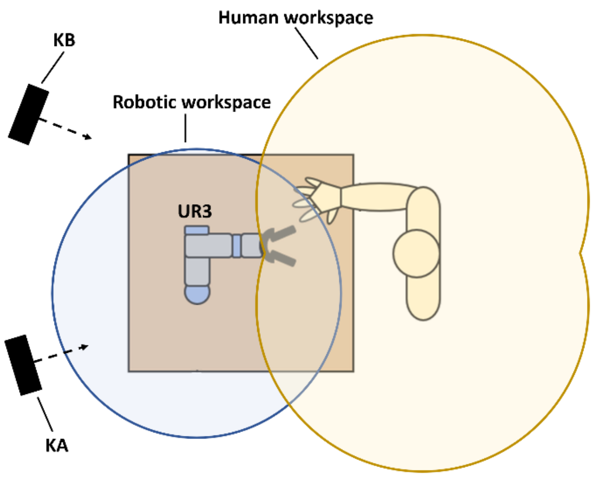


Figure 4.3: Schematics of positioning of Kinect sensors and collaborative workspace.

Concerning the light conditions, the Kinect v2 has optimal performances in terms of accuracy in any ambient light situations, as stated in [104-106]. However, the accuracy of the sensor decreases under direct sunlight. For these reasons, the collaborative robotics workspace has been set-up avoiding that sun lights directly hit the sensors, undermining the performances of the vision system.

4.1.2 Software and control architecture

A scheme of the control architecture is reported in figure 4.4. It is indicated where each operation is performed and the related calculation time.

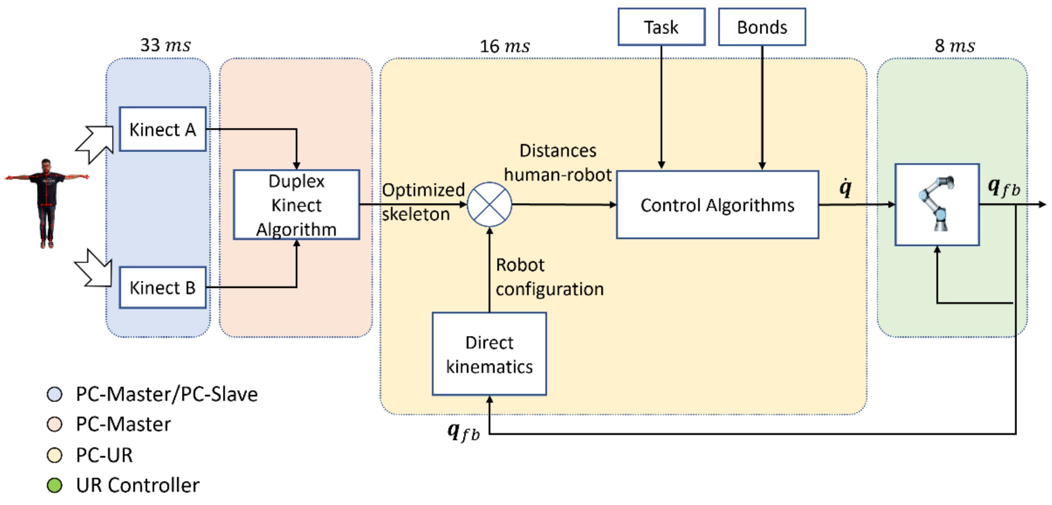


Figure 4.4: Scheme of the control architecture.

The algorithms that run in the PC-Master and the PC-Slave are written in Matlab 2019b. Here, a toolbox that permits to acquire the data from a Kinect v2 sensor is installed. The PCs connected to the Kinect sensors have to work synchronously, in order to properly combined the Kinect data. For this reason, the PC-Master sends a trigger to the PC-Salve through the HUB and they capture the collaborative robotics workspace at the same time. The PC-Slave then sends the Kinect data to the PC-Master. Here a duplex Kinect algorithm can generate a skeleton that is the optimal combination of the skeletons given by the two Kinects. The algorithm is taken from [99] and it was optimised to run at a sampling rate of 30 Hz. The algorithm is reported in the following equation

$$\begin{aligned} \min \quad & \sum_{i=1,2,\dots,n} w_i^A \|\boldsymbol{p}_i^* - \boldsymbol{p}_i\|^2 + w_i^B \|\boldsymbol{p}_i^* - \boldsymbol{q}_i\|^2 \\ \text{s.t.} \quad & \sum_{i,j=1,2,\dots,n} \left(\|\boldsymbol{p}_i^* - \boldsymbol{p}_j^*\| - l_{i,j}\right)^2 = 0 \end{aligned} \tag{4.1}$$

where n is the number of the joints, $\boldsymbol{p}_i$ is the position of i^{th} joint, $l_{i,j}$ is the distance between two consecutive joints and $\boldsymbol{p}_i^*$ is the resultant position of the optimised joint. w_i^A and w_i^B are coefficients obtained from Kinect software that indicate the tracking state of the i^{th} joint. In fact, a joint can be "tracked" (that means that the joint is correctly detected by the sensor), "inferred" (that means that the joint is not detected and its position is estimated) or "not tracked" (that means that the joint is not detected or estimated). The duplex Kinect algorithm permits to obtain a skeleton that overcomes the problems related to possible occlusions. As can be seen in figure 4.5, the right arm of the operator is not visible by the Kinect B and this leads to an unreliable skeleton. The duplex Kinect algorithm outputs a skeleton that is the correct combination of the skeletons from the two Kinects. Furthermore, the algorithm permits to keep constant the distances between consecutive joints, that would otherwise change.

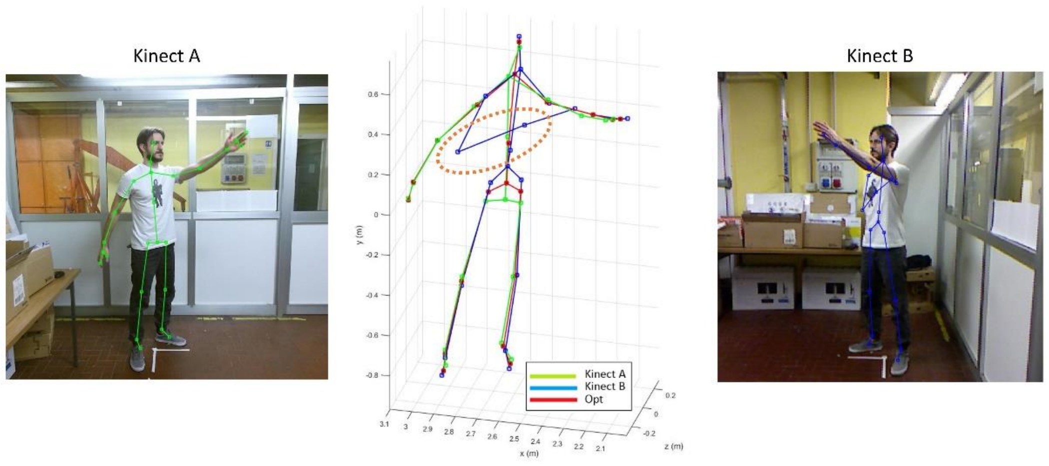


Figure 4.5: The skeletons from Kinect A (in green) and Kinect B (in blue) and the optimised one (in red) obtained from the duplex Kinect algorithm.

The optimised skeleton data are then sent to the PC-UR. These data are inputs of the control algorithms present in the PC-UR. Other inputs of these algorithms are the feedback data from the UR3. These data are sent by the UR Controller with a frequency of 125 Hz (that is the control frequency of the UR Controller). Further information related to the feedback data can be found in [100].

All the algorithms that runs in the PC-UR are written in Matlab 2019b. A kinematic model of the UR3 present in the PC-UR uses the feedback data from the UR Controller to obtain the actual configuration of the cobot. The distance vectors between the cobot and the optimized skeleton from the PC-Master are calculated by the MathWorks *knnsearch* algorithm and are inputs of the control algorithms. Other inputs are data related to the task that the robot has to achieve (usually is a planned trajectory and the data are the TCP position and velocity vectors) and to some bonds (e.g. attractive elements of the trajectory conditioning technique). The control algorithms are the collision avoidance and hand-over algorithms presented in Chapter 2.

In both cases, the output is the joint velocity vector $\dot{\boldsymbol{q}}$ and it is calculated with a frequency of 62.5 Hz. The vector $\dot{\boldsymbol{q}}$ is sent to the UR Controller as input of *URScript* commands *speedj* and *stopj*. These commands are sent as strings to the UR Controller. Further information about the *URScript* commands can be found in [101].

The *URScript* command is a high-level control of the UR3 and permits to benefit of the default safety functions of the robot. In this way, another safety tool beyond the collision avoidance algorithm is present in this collaborative robotics system. The UR safety functions permit for example to set the maximum values of the TCP velocity and of the joint velocities. Furthermore, a safety function permit to define planes that the TCP of the cobot has to not cross. Two planes have been defined in this robotics system. One plane is parallel to the table and positioned 50 mm above it; the other plane is vertical and positioned in front of the Kinect sensors. In this way, the TCP does not collide with the table and the Kinect v2 sensors. Violations of safety functions lead to a stop of the UR3 cobot.

4.2 Tests

To properly perform the experimental tests, the data from the two Kinects and the UR3 must be referred to the same reference frame. The 3D printed narrow-conical supports used for the spatial matching phase are visible on the table in figure 4.1. The spatial matching procedure for the Kinects is the same reported in paragraph 3.4. For what concerns the robot, UR3 is moved in order to place its TCP in correspondence of the tips of the supports. In this way, the positions of the tips are calculated by direct kinematics of the robot and the reference frame is built as described in paragraph 3.4.

4.2.1 Collision avoidance

The results of experimental tests conducted to show the effectiveness and the performances of the collision avoidance algorithms are presented and discussed in this subparagraph. The results of three types of tests will be shown. The first test has been conducted to study the basic collision avoidance algorithm. The last two tests permit to understand the differences between the basic collision avoidance algorithm and the one with the trajectory conditioning technique, both in case of fixed and moving obstacles. Before to report and discuss the results of the tests, it is important to highlight the operating conditions: the task of the UR3 is to follow a linear path for three times in 37.5 s. The robot goes from the starting position to the final one, then returns to the starting position. The trajectory ends with the TCP that goes again to the final position. The path is reported in figure 4.6.

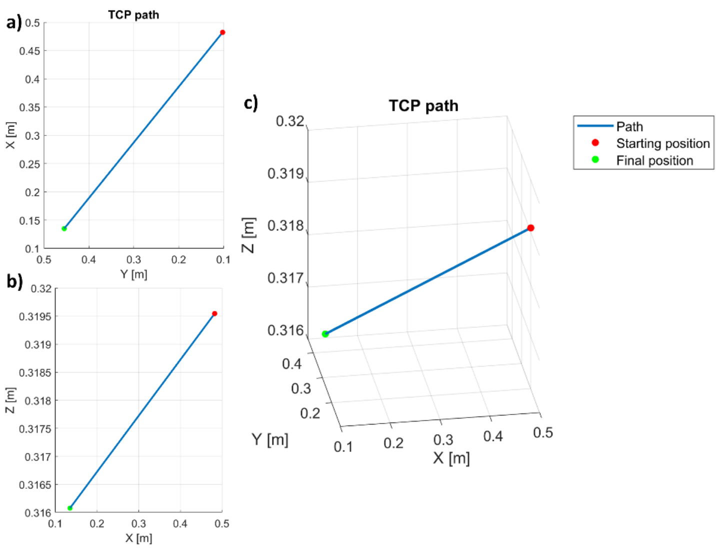


Figure 4.6: The planned path in three different views.

The poses of the TCP of the robot associated to the extreme points of the path are

$$^{FR}\hat{A}_{TCP,starting} = \begin{bmatrix} -0.6368 & 0.0260 & 0.7706 & 0.4822 \\ 0.7710 & 0.0355 & 0.6359 & 0.1023 \\ -0.0108 & 0.9990 & -0.0427 & 0.3195 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$^{FR}\hat{A}_{TCP,final} = \begin{bmatrix} -0.6368 & 0.0260 & 0.7706 & 0.1350 \\ 0.7710 & 0.0355 & 0.6359 & 0.4550 \\ -0.0108 & 0.9990 & -0.0427 & 0.3161 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

So, the velocity associated to the task of the TCP is $\boldsymbol{V}_{task} = \boldsymbol{v}_{TCP} + \boldsymbol{C} \cdot \boldsymbol{e}$, where $\boldsymbol{v}_{TCP}$ is the planned velocity vector of the TCP, $\boldsymbol{e}$ is the error between the desired planned pose of the TCP and the actual one and $\boldsymbol{C}$ is a matrix with the error gains. The orientation error is calculated as quaternion error.

The planned position and velocity vectors versus the time are shown in figure 4.7.

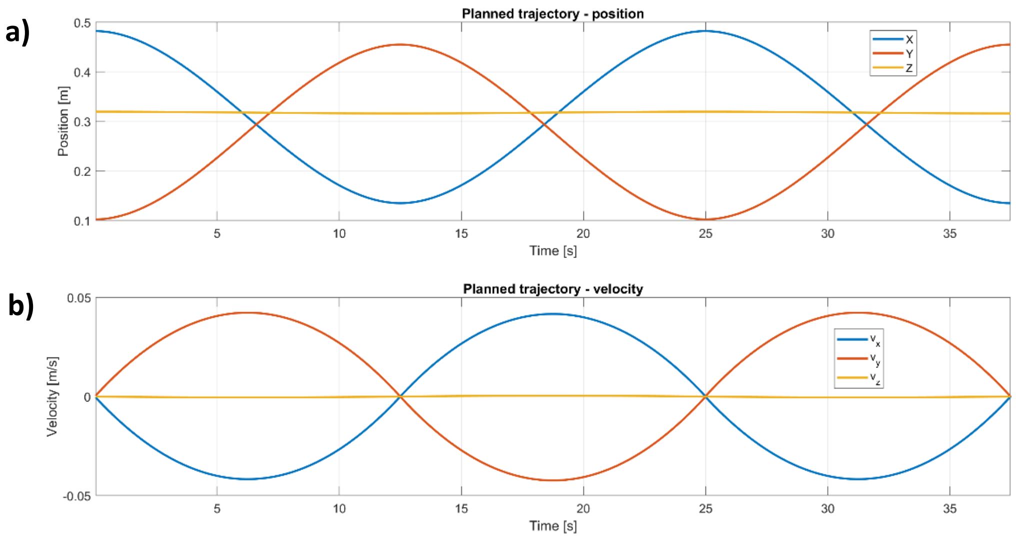


Figure 4.7: a) Coordinates of the position of the TCP; b) Components of the planned velocity.

The collision avoidance points considered on the structure of the UR3 are the ones visible in figure 2.1. The values of the parameters of the collision avoidance algorithm used in the tests are

$$V_{MAX} = 0.5\ m/s$$

$$\rho = 0.3\ m$$

$$\alpha = 0.3\ m$$

$$\boldsymbol{C} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix} 1/s$$

For what concern the human operator, the 15 joints of the upper part of the human body were considered. So, the *knnsearch* algorithm calculates the distances between these 15 points and the 9 of the cobot. In case of unavoidable collisions, the algorithms stop the robot if the minimum human-robot distance is less than 0.1 m.

Test 1 – Basic collision avoidance algorithm: in this test the operator moves the right hand near the TCP of the robot. The basic collision avoidance algorithm has to move the TCP away from the human operator modifying its path. When the minimum distance between the robot and the worker is larger than the reference distance ρ, the TCP returns to follow the planned path. In figure 4.8, frames from the test are reported.

In the figure, the planned path is shown with a red line and the TCP is highlighted with a blue circular marker.

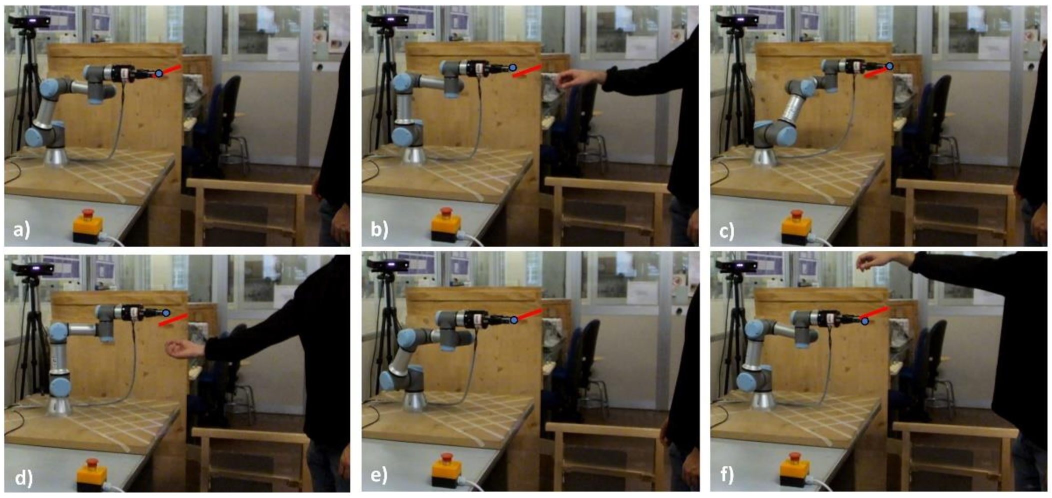


Figure 4.8: Different phases of Test 1.

The phases of the test are: a) the robot follows the planned path; b) the operator moves the hand towards the TCP; c) robot reaches the final pose; d) the hand is below the TCP; e) the UR3 returns to the starting pose; f) the operator moves the hand above the robot.

The operator approaches the TCP frontally (figure 4.8b), from below (figure 4.8d) and finally from above (figure 4.8f). In all the three cases, the UR3 leaves the planned path so to increase the distance with the human worker and avoid any collision. In fact, it is possible to see that in figures 4.8b, 4.8d and 4.8f the blue marker associated to the TCP is not on the planned path (red line), showing the evasive movements generated by the basic collision avoidance algorithm.

Experimental data are presented in figure 4.9.

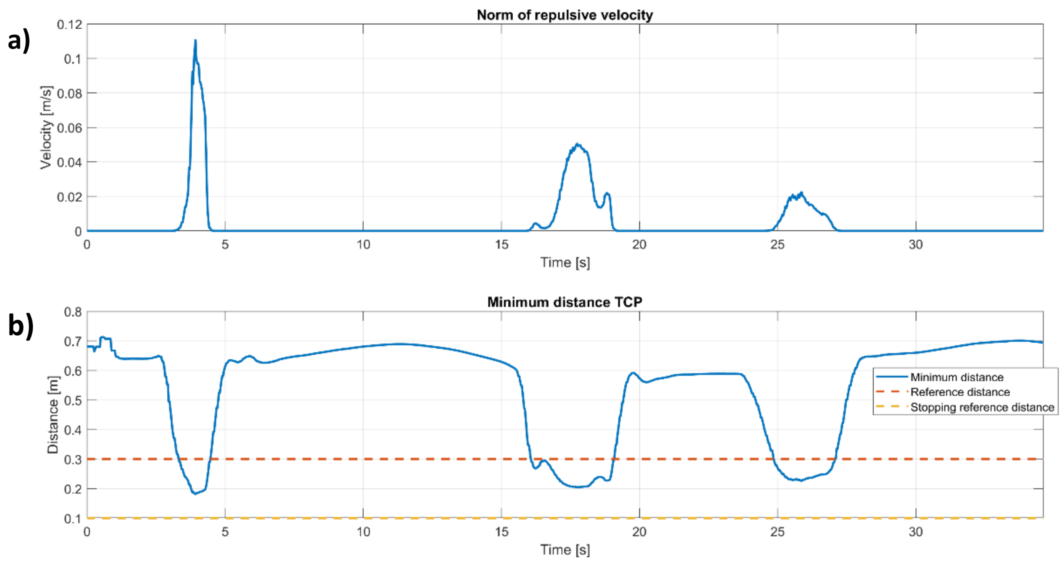


Figure 4.9: a) Norm of the repulsive velocity; b) minimum distance between the robot and the human worker.

The graph in figure 4.9b shows the minimum distance between the human operator and the robot versus the time. It is possible to see that when the distance is less than the reference distance ρ, the collision avoidance algorithm generates a repulsive velocity with norm larger than zero. This repulsive velocity modifies the path of the TCP in order to avoid collisions. The modified path and the planned one are shown in figure 4.10. Figures 4.10b and 4.10c show that the basic collision avoidance algorithm produced purely vertical movements.

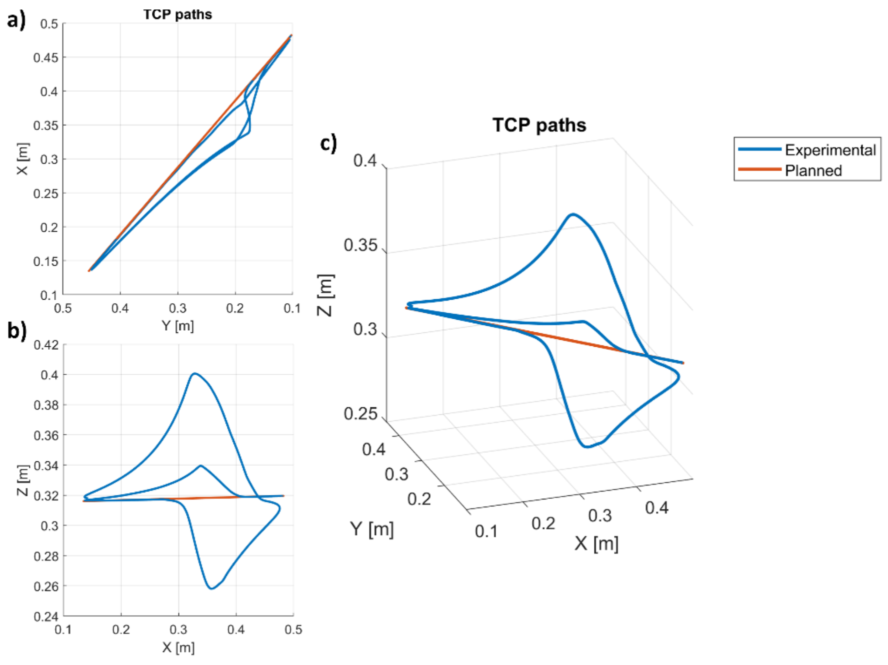


Figure 4.10: The planned path and the modified one in three different views.

Test 2 – Comparison with fixed obstacle: the operator moves the hand towards the path of the TCP and puts it in a fixed position. The standard collision avoidance algorithm and the one with the trajectory conditioning technique were used to control the UR3. Attractive cylinders associated to the hands of the human operator are considered in the trajectory conditioning technique. The origins of the reference frames of the cylinders coincide with the Kinect joints of the hands. The orientation is fixed and it is the same of the common reference frame F_R. The values of the parameters associated to the cylinders are

$$V_{MAX,act} = 0.2\, m/s$$
$$r = \rho_{act} = 0.35\, m$$
$$\alpha_{act} = 5$$

Using these types of cylinders, the robot should avoid the hand of the operator moving in front of the hand, without large vertical movements. The results obtained with the basic collision avoidance algorithm and the ones with the trajectory conditioning technique are reported in this section. In figure 4.11, frames of the movements of the robot in both cases are shown. The movements related to the basic collision avoidance algorithm are presented in the first row, while the motions obtained with the trajectory conditioning technique are in the second row. The red line is the path obtained with the basic collision avoidance algorithm (figure 4.11a-b-c). Instead, the yellow line is the path obtained with the trajectory conditioning technique (figure 4.11a'-b'-c').

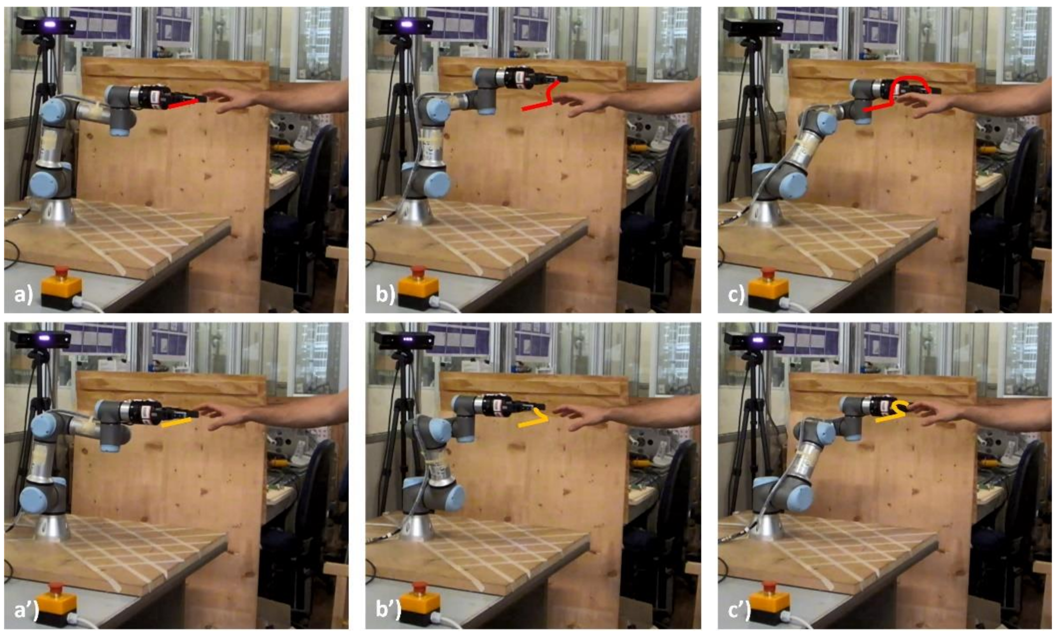


Figure 4.11: Different collision avoidance movements with fixed obstacle.

The frames in the first row in figure 4.11 (a-b-c) are related to the basic collision avoidance algorithm and show that the robot moved vertically to avoid the hand of the operator. Instead, the robot controlled by the collision avoidance algorithm with the trajectory conditioning technique moved in front of the hand to avoid the obstacle (cf. figure 4.11a'-b'-c').

The paths with and without the trajectory conditioning technique can be seen in figure 4.12. It is clear that with the trajectory conditioning technique the collision avoidance movements of the TCP are planar (yellow line), while with the basic algorithm the evasive movements are purely vertical (red line). So, the collision avoidance algorithm with trajectory conditioning technique works properly with fixed obstacle.

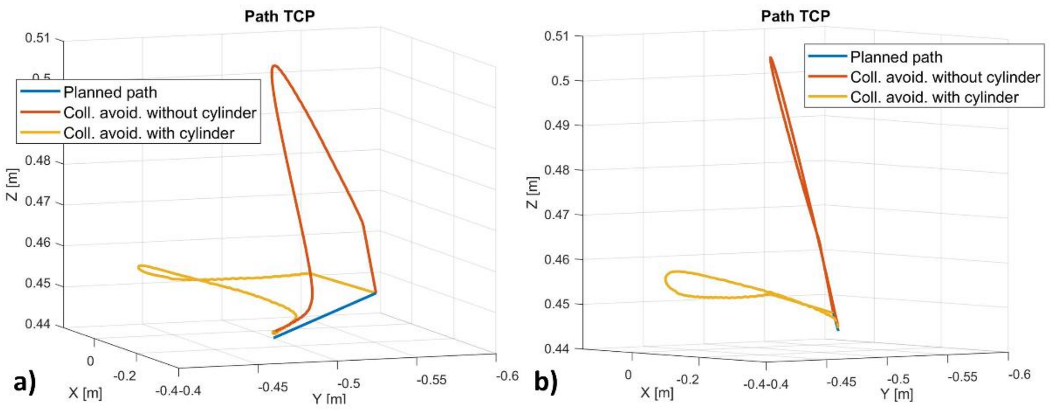


Figure 4.12: Collision avoidance paths with fixed obstacle.

Test 3 – Comparison with moving obstacle: To properly compare the algorithms with moving obstacles, the same movements of the hand of the Test 1 are considered in this test. The paths obtained with and without the trajectory conditioning technique can be seen in figure 4.13.

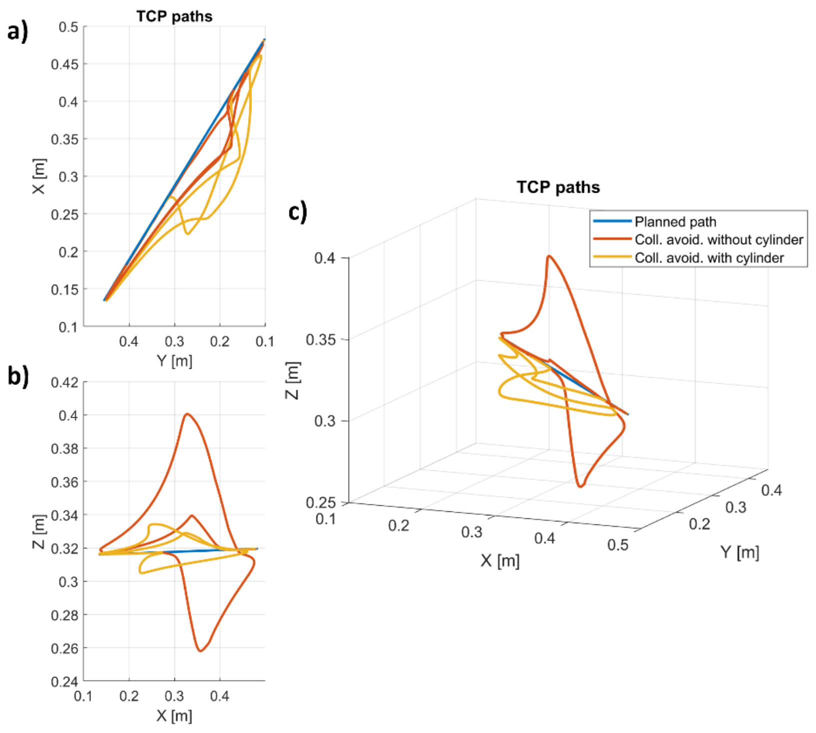


Figure 4.13: Collision avoidance with moving obstacle.

The red line in figure 4.13 shows that large vertical movements are obtained with the basic collision avoidance algorithm. Instead, with the trajectory conditioning technique planar collision avoidance movements have been obtained. The robot avoided the hand of the operator moving in front of it with limited vertical displacements, as can be clearly seen in figure 4.13b where the yellow line is the modified path obtained with the trajectory conditioning technique.

The results of the tests reported in this paragraph showed that the collision avoidance algorithm with the trajectory conditioning technique works properly with fixed and moving obstacles and permits to define a priori preferred collision avoidance trajectories.

4.2.2 Hand-over

Experimental tests have been conducted to verify and evaluate the characteristics of the hand-over algorithm. The algorithm should control the cobot in order to perform hand-over operations that are:

- bidirectional, the algorithm must work in case of human-to-robot and robot-to-human hand-over;
- reactive, it must permit to perform the hand-over in every point inside the workspace of the robot and to follow a dynamic target;
- it has to consider the pose of the hand of the human operator, not only the position. In this way it is possible to obtain more ergonomic hand-over and the operator can choose the hand pose that he/she prefers;
- it must produce fast hand-over operations, so to not undermine the fluency of the tasks.

The tests were carried out under the following conditions: the object to handle was an aluminium plate and to properly exchange it, the chosen distance between the virtual hand and the track hand was 0.15 m. This value of the distance was chosen in order to place the OTP in the middle of the aluminium plate. After the plate was exchanged, the cobot came back to its "Home" configuration only after the hand was outside the robot workspace. The joint position vector related to the Home configuration is

$$\boldsymbol{q} = \begin{bmatrix} -\pi/4 & -\pi/2 & -\pi/2 & 0 & \pi/2 & 0 \end{bmatrix}^T.$$

In these tests, the values of the parameters are $V_{max,ho} = 0.35\ m/s$, $\rho_{ho,1} = 0.4\ m$, $\rho_{ho,2} = 0.2\ m$ and $\alpha_{ho,1} = \alpha_{ho,2} = 6$. The proportional gain of the angular velocity is

$$\boldsymbol{K} = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{bmatrix} 1/s.$$

For what concerns the Robotiq gripper, this can be controlled from the PC-UR by *URScript* commands sent as strings. Further information about the *URScript* commands of the Robotiq gripper can be found in [102].

Test 1 – Bidirectionality: a human-to-robot and robot-to-human hand-over actions were performed to verify the effectiveness of the algorithm in terms of bidirectionality of the hand-over.

In figure 4.14, frames from the human-to-robot hand-over task are reported.

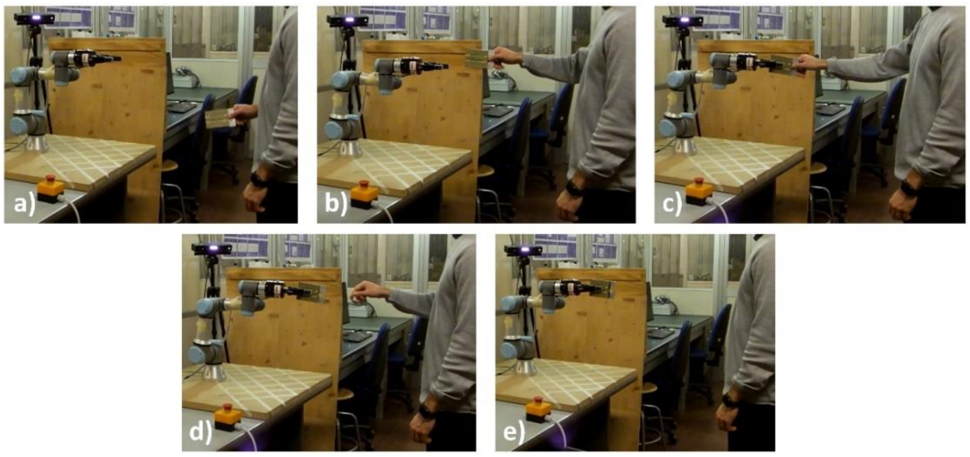


Figure 4.14: Five steps of human-to-robot hand-over.

Human-to-robot hand-over steps are:

a) the operator moves the hand towards the robot;

b) the hand enters in the workspace and the robot starts moving;

c) the robot reaches the OTP, stops and closes the gripper;

d) the human retrieves the hand;

e) after the hand is outside the workspace, the robot goes back to the Home configuration.

The algorithm controlled properly the cobot and the human-to-robot is achieved. A robot-to-human hand-over example is reported in figure 4.15. Here the robot moved towards the human operator after the hand entered in the robotic workspace (cf. figure 4.15b) and gave the worker the object (cf. figure 4.15c). The robot returned to its Home configuration when the operator retrieved the hand outside the workspace (cf. figure 4.15e). The hand-over task is achieved also in this case.

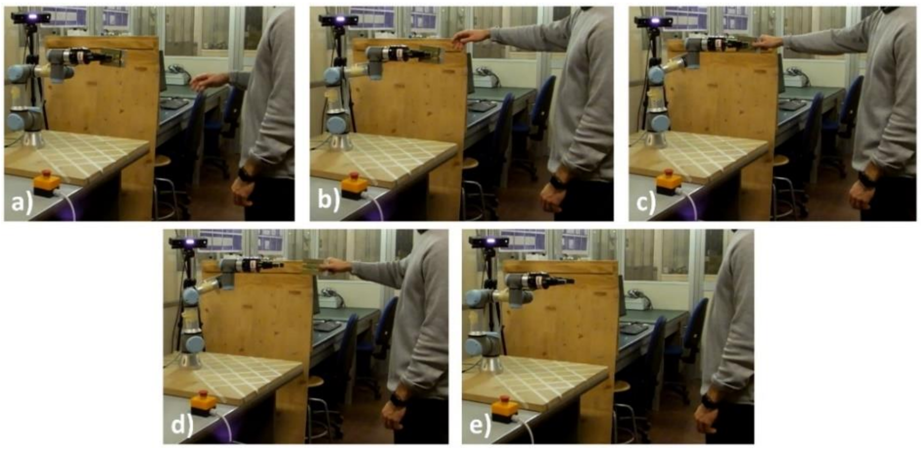


Figure 4.15: Five steps of robot-to-human hand-over.

In both cases, the operator starts the hand-over moving the hand towards the robot. In this way the worker is free to choose the time and the place in which perform the object exchange and the cobot adapts online its behaviour to the movements of the operator.

Test 2 – Reactivity: to verify the reactivity of the hand-over algorithm, three different hand-over actions have been considered. There are both human-to-robot and robot-to-human hand-over. In figure 4.16, each row shows three frames of the same hand-over. These hand-over actions will be studied again in the section *Test 4 – Response time*. In figure 4.16, it can be seen that the robot is able to follow the hand of the operator in different configurations.

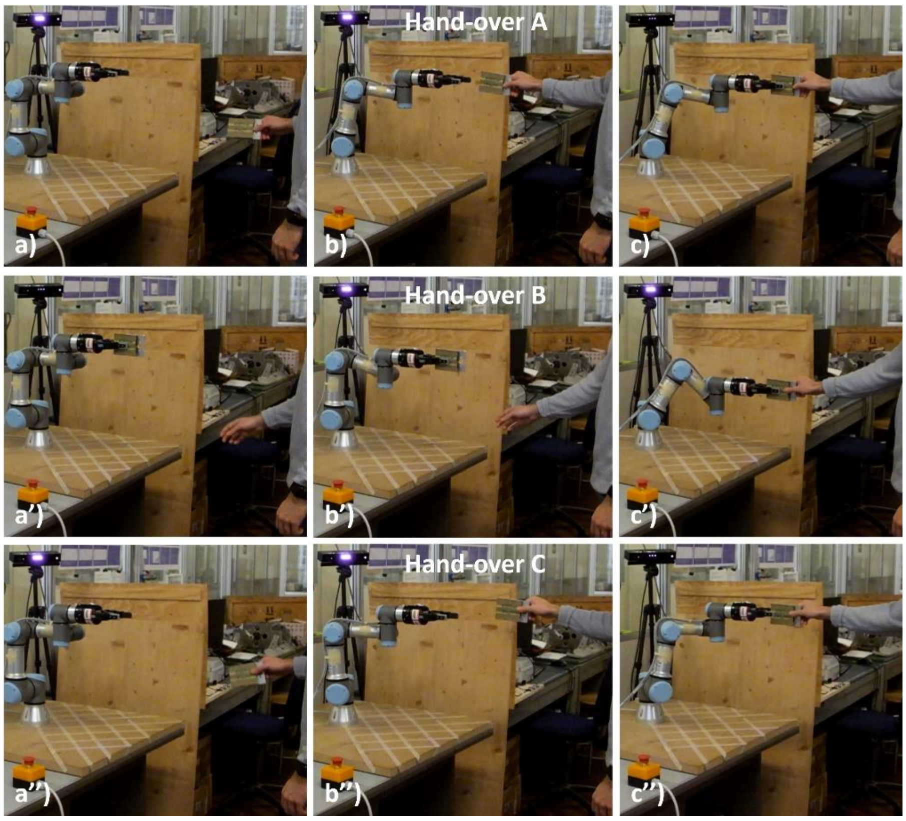


Figure 4.16: Three different hand-over actions performed in the different points of the workspace of the UR3.

Test 3 – Pose dynamic tracking: the results of this kind of tests show the performances of the algorithm in terms of following dynamically the pose of the hand. To show clearly this characteristic of the algorithm, the TCP of the cobot is fixed in position and the operator changes the orientation of the forearm. In figure 4.17 some frames are reported.

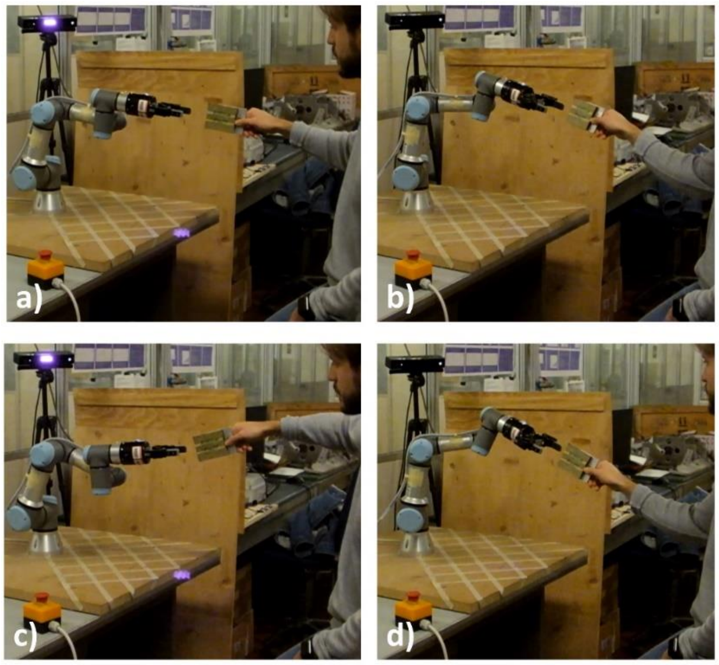


Figure 4.17: The robot is able to modify its orientation in order to properly perform the hand-over.

It is clear how the cobot can follow the orientation of the virtual hand and adapts the orientation of the TCP in order to properly perform the hand-over.

Following the pose of the hand and not only its position is an interesting feature. In fact, it could be useful to perform hand-over that are ergonomically suitable. In figure 4.18, the results of a test conducted with a sitting operator and a hand-over algorithm that considers only the position of the hand are shown.

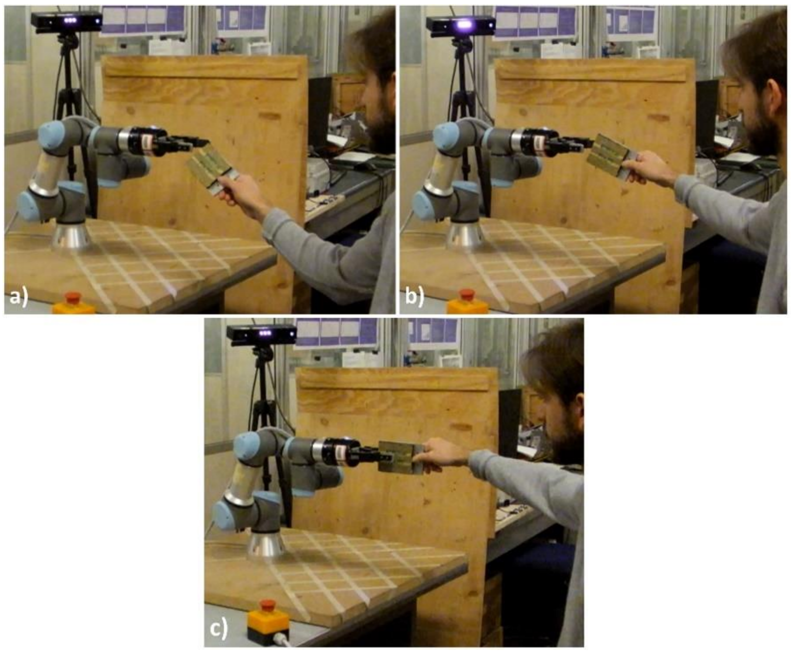


Figure 4.18: Hand-over movements obtained with a hand-over algorithm that considers only the hand position.

In figure 4.18a, the robot reaches the OTP. The operator has to move the arm to align the plate to the gripper (cf. figure 4.18b). In figure 4.18c, the plate is in the correct pose and the hand-over can be performed. Figure 4.18b and 4.18c show clearly that operator has to move the arm in order to properly give the object to the cobot after it reached the OTP. Repeating these adaptation movements several times could be annoying and tiring for the human operator.

The adaption movements are not necessary if the hand-over algorithm considers also the orientation of the hand (cf. figure 4.19). Figure 4.19b and 4.19c show that the robot adapts its TCP pose to the one of the hand of the worker to properly perform the hand-over. The operator can choose the hand pose that he/she considers most ergonomically suitable and the robot is able to adapt its motions to perform the hand-over with the right pose.

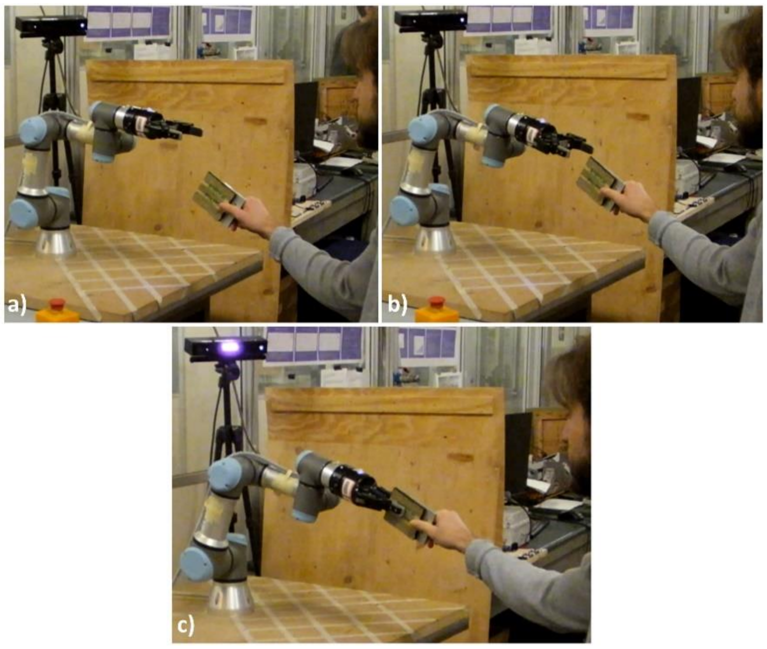


Figure 4.19: Hand-over movements obtained with the proposed hand-over algorithm considering the pose of the hand.

Test 4 – Response time: To show the time performances of the algorithm, the cases considered in Test 2 are analysed. In Chapter 2, it was reported that the mean *response time* in human-human hand-over is about 1.2 s. This is an important parameter that can be used to define if a hand-over operation is fast or not. In this section, the reaction times of the movements previously seen will be calculated. It is important to highlight that the robot starts to move after the virtual hand enters in the workspace of the robot. The reason is to avoid the singular configurations that are at the limits of the workspace. These configurations can be reached if the target that the robot tries to follow is outside the workspace. Therefore, the response time is calculated from the time instant in which the virtual hand enters in the workspace to the time the robot reaches the hand pose.

In figure 4.20, the numerical data related to the hand-over A of Test 2 are reported. In figure 4.20a the norm of the TCP linear velocity is reported. Figure 4.20b and 4.20c show the velocities of the joints of the arm and the ones of the wrist joints. Figure 4.20d shows the distances between the virtual hand and the shoulder centre of the robot. This distance permits to understand if the virtual hand is inside the workspace or not. It is inside if the distance is below the red dashed line, that is the radius of the spherical workspace of the robot. It is possible to see that the robot starts to move when the target is inside the workspace. The joint velocities are zero when the OTP was reached by the TCP of the robot. In this case, the response time is about 1.8 s.

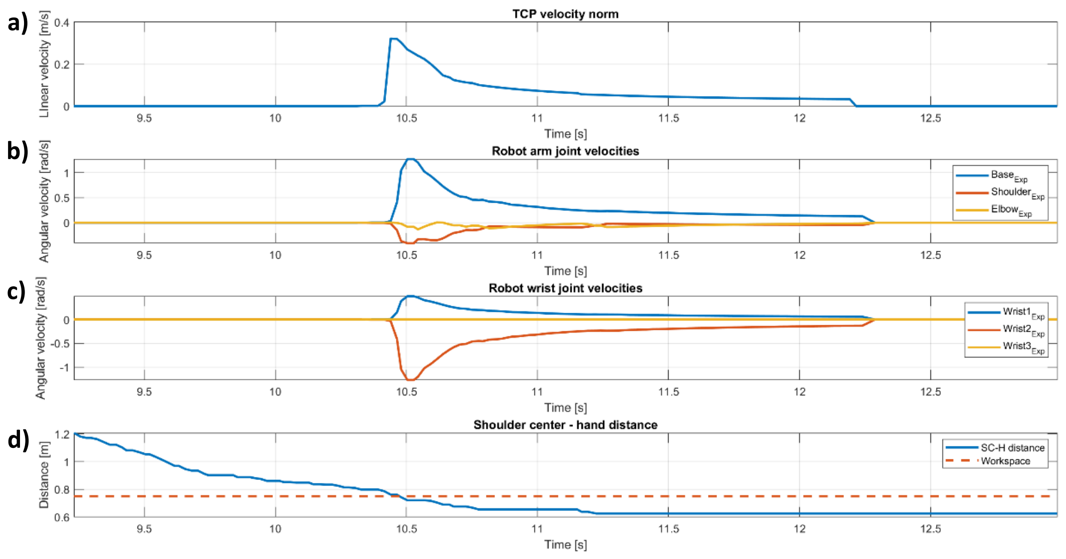


Figure 4.20: Experimental data of Hand-over A visible in the first row of figure 4.16.

The results obtained for the hand-over B are shown in figure 4.21. Figure 4.21a shows the norm of the TCP linear velocity. The joints velocities are reported in figure 4.21b and 4.21c. The distance between the virtual hand and the shoulder centre is shown in figure 4.21d. The response time is about 2.3 s.

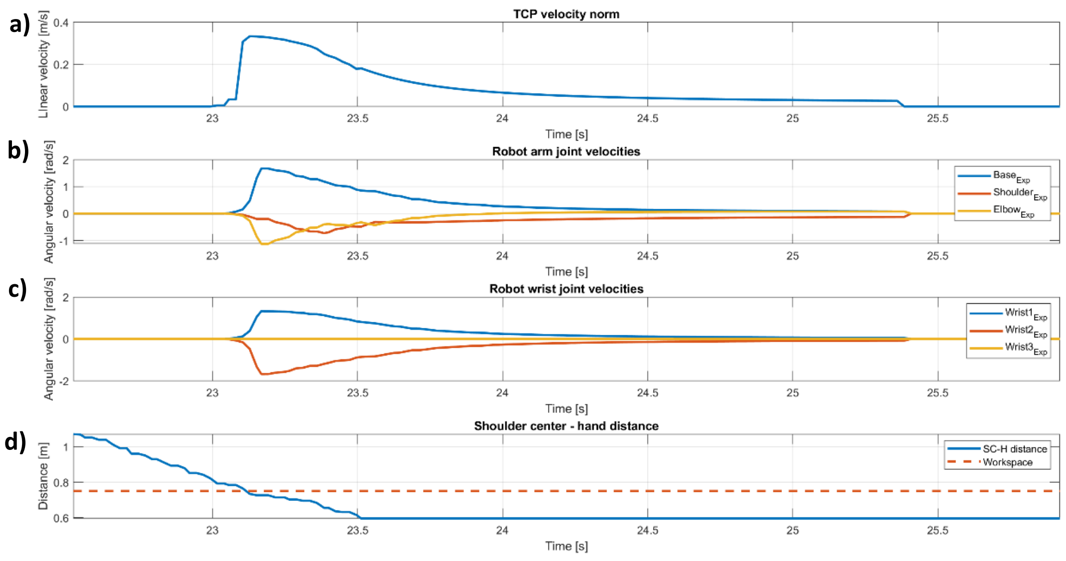


Figure 4.21: Experimental data of Hand-over B visible in the second row of figure 4.16.

The data related to the hand-over C of figure 4.16 are reported in figure 4.22. When the distance between the hand and the shoulder centre is less than the radius of the robotic workspace (cf. figure 4.22d), the TCP linear velocity norm and the joints velocities became larger than zero and the robot starts to move (cf. figure 4.22a, 4.22b and 4.22c). The response time is about 2 s.

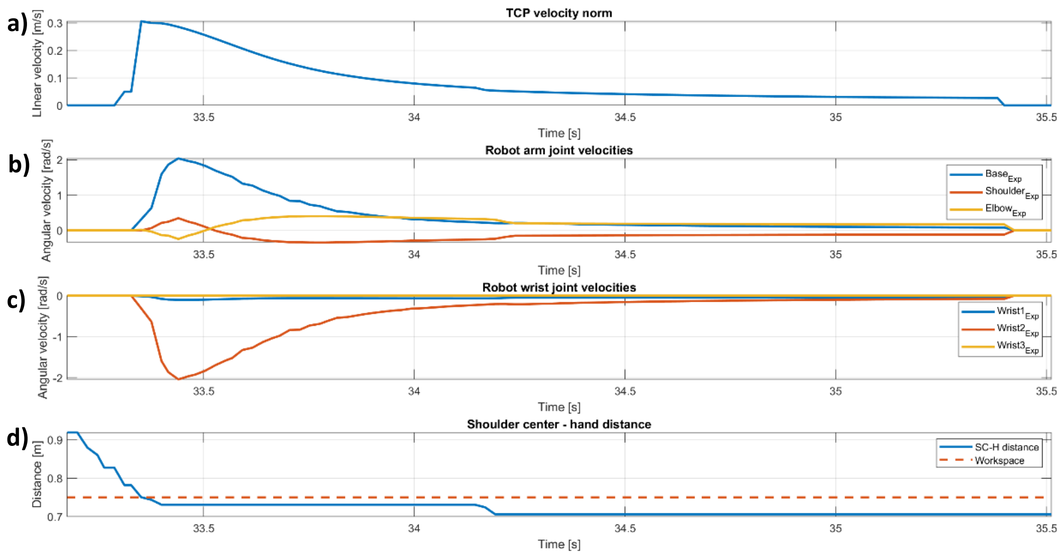


Figure 4.22: Experimental data of Hand-over C visible in the third row of figure 4.16.

The mean response time is about 2 s. This time is larger than the mean human-human response time, but the difference is less than 1 s (it is about 0.8 s). These results demonstrate that the human-robot hand-over operations performed with the proposed algorithm are not as fast as the human-human hand-over actions, but they are fast enough to obtain a fluent hand-over.

The results reported in this paragraph show clearly that the hand-over algorithm proposed in this work has all the desired features.

4.3 Conclusions

In this chapter, the experimental phase of the work is described. A collaborative robotics workspace has been set up to test the control algorithms proposed in Chapter 2. In the collaborative robotics workspace, a human operator can safely share the workspace with a cobot thanks to the collision avoidance algorithms or perform hand-over tasks. The collaborative robot presented in the experimental set-up is a UR3 from Universal Robots, equipped with a Robotiq Gripper 2F-85 in order to exchange objects with the worker. The movements of the operator are acquired by two Microsoft Kinect v2 sensors. Using two vision sensors permits to avoid the problems related to the occlusions of a sensor that could make impossible to acquire the motions of the human worker. An algorithm permits to combine the data of the two Kinects obtaining an optimized skeleton. The output of the duplex Kinect algorithm is one of the inputs of the control algorithms. The other inputs are the robot feedback data sent from the UR controller and the data associated to the task that the robot must perform. All the algorithms are written in Matlab 2019b.

Three Windows based PCs are used in the experimental set-up: two to handle the data from each Kinect and the third to control the UR3 sending commands to the controller of the cobot. A Wireless Router handles the TCP/IP communications between the PCs and the controller of the UR3.

Tests have been conducted to verify the effectiveness of the control algorithms and evaluate their performances in a real experimental scenario. The collision avoidance algorithms have been tested with fixed and moving obstacles. The results showed that the algorithms work properly and the trajectory conditioning technique permits to define preferred collision avoidance trajectories with the human operator fixed or moving in the robotic workspace. In fact, associating attractive cylindrical surfaces to the hands of the worker, the robot avoids the collision moving in front of the hand without large vertical displacement.

For what concern the hand-over algorithm, the tests showed that the algorithm proposed in this work has all the desired features and permits to perform fast, bidirectional, reactive, dynamic-hand-pose-tracking hand-over tasks.

In the next chapter, the conclusions and possible future developments of the work are discussed.

Chapter 5

Conclusions

In this thesis, two novel algorithms designed for controlling collaborative robots have been presented. These algorithms are tools that permit to human operators to safely share the workspace with collaborative robots or exchange objects with a cobot. The movements of the human worker are inputs of these control algorithms and they are acquired by markerless vision sensors. This kind of sensors can be easily implemented in a real industrial scenario without the need to modify the workwear and the work equipment of the operators who have to work in collaborative robotics workspaces.

The first algorithm is an evolution of the collision avoidance algorithms based on the artificial potential fields technique. This is a well-known technique that permits to avoid collisions between robots and dynamic obstacles modifying online and in real time the planned trajectory of the robot. Fast evasive movements are obtained with this approach and can be used with fast obstacles, as the human operators. The drawback of the artificial potential fields is that the directions of the evasive motions are not known a priori, because they depend on local conditions. For this reason, the directions of the evasive movements cannot be predicted and some of these directions can be perceived as unsafe by the human operators. To overcome this problem, a novel trajectory conditioning technique is presented in this work. This technique permits to force the robot to move in well-defined directions while avoiding collisions with the human workers. To obtain these forcing actions, attractive elements with well-defined geometries and poses are associated to parts of the human body. Three attractive elements have been presented in Chapter 2. In this way the operator who works with the collaborative robot knows these directions and this can reduce his/her mental stress.

The second algorithm is a hand-over path planner algorithm with several features that permit to perform fast and fluent hand-over actions between robots and human workers. Several hand-over algorithms presented in literature consider only the case of robot-to-human hand-over. Few works consider the case of human-to-robot and bidirectional hand-over. Furthermore, only few of them consider moving targets and the pose of the hand of the operator instead of the only position. The ones that face these problematics are unfortunately too slow, undermining the fluency of the hand-over, or use wearable tracking systems, that limit the applicability of the algorithms in a real industrial scenario. The hand-over algorithm presented in this work strongly improves former results, as it permits to obtain fast, reactive, bidirectional hand-over actions based on markerless vision systems.

The algorithms have been tested in a simulation environment and in an experimental set-up. The simulation environment has been presented and permits to a worker to interact with a simulated cobot. The movements of the human operator are acquired by Microsoft Kinect v2 sensor and his/her movements are reproduce by a kinematic model of the human body in a simulated collaborative robotics workspace. The logic that permits to the kinematic model of the body to reproduce the movements of the operator given the joints position vectors evaluated by Kinect v2 sensor has been explained. The simulated workspace presents a dummy of the operator and a graphical representation of a cobot and was designed using the software CoppeliaSim. The kinematic/dynamic model of two collaborative robots have been developed. The cobots are a KUKA LBR iiwa 14 R820 and a Universal Robots UR3. The models of the cobots and of the human body have been developed in MathWorks environment. The scene in CoppeliaSim reproduces the movements of the human body and of the cobot calculated in MathWorks environment. Information related to the calculated movements of the dummy and of the cobot are sent to CoppeliaSim scene by Remote API functions. The "Minimum distance calculation" tool by CoppeliaSim permits to calculate the distances between the links of the robot and the parts of the human body and these data are inputs of the control algorithms.

The experimental set-up has been built up to test the control algorithms and develop collaborative robotics applications. The movements of the human operator are acquired by two Kinect v2 sensors. This solution permits to reduce the problems related to the occlusions of one sensor and have reliable information from the vision system. The collaborative robot is a UR3 and the developed control architecture can control the cobot with a frequency of 62.5 Hz. Three PCs are used to handle the two Kinect sensors and the cobot. One of these PCs controls the robot sending *URScript* commands as strings to the URController. The PCs and the URController communicate with each other via TCP/IP Ethernet network.

The results of tests conducted in the simulation environment and in the experimental set-up show the effectiveness and the performances of the control algorithms. The collision avoidance algorithm with trajectory conditioning technique forces the robot to move in certain directions while avoiding collisions. The conditioning action was demonstrated with static and dynamic obstacles. In all the tests, the cobot moved along the desired directions. The features of the hand-over algorithm have been clearly shown in the tests and the differences between this hand-over algorithm and ones that consider only the position of the hand have been highlighted.

Some developments can be implemented in the control algorithms and in the experimental set-up. For what concern the control algorithms, it is under development an algorithm that can predict the position of the human hand and of the other parts of the human body and it could be useful integrating this algorithm in the robotics system. In this way, the fluency of the hand-over algorithm can be improved reducing the waiting times of the human operator. It could be useful also in the collision avoidance algorithm, because it can permit to anticipate the evasive movements and maintain a larger safety distance between the cobot and the human operator. Furthermore, the collision avoidance algorithm presented in this work used the skeleton representation of the human body given by Kinect as input of the distance calculation algorithm. The skeleton is obtained by Kinect software from the point cloud representation of the framed environment. It could be considered the option of developing a collision avoidance algorithm that uses directly the point cloud representation of the collaborative workspace, in order to make possible the use of other kinds of sensors such as Lidar. For what concern the experimental set-up, it is important to substitute the Kinect v2 sensors with other markerless vision systems. In fact, Microsoft has stopped the production of Kinect v1 and v2, and it recently launched a new markerless vision sensor called Azure Kinect [103]. This sensor could be an optimal substitute of the Kinect v2 sensors used in the experimental set-up. Other actions that could improve the performances of the control architecture can be reducing the number of PCs used in the experimental set-up and writing all the algorithms in a compiled programming language. In fact, the algorithms are now written in Matlab, that is an interpreted language. Writing the algorithms with fast compiled languages permits to reach higher control frequency of the system.

Finally, a most relevant future work should be the development of a collaborative robotics application that uses the control algorithms here presented and could be implemented in real production lines.

References

[1] International Federation of Robotics (IFR). (2019). *World Robotics 2019 Industrial Robots - Statistics, Market Analysis, Forecasts, Case Studies.*

[2] Colgate, J. E., Edward, J., Peshkin, M. A., Wannasuphoprasit, W. (1996). Cobots: Robots for collaboration with human operators.

[3] https://www.interactanalysis.com/press-releases/cobot-market-to-account-for-30-of-total-robot-market-by-2027-interact-analysis/

[4] Wixcey, N. (2017). Made-to-order: The rise of mass personalization. *The Deloitte Consumer Review.*

[5] European Agency for Safety and Health at Work. (2019). *Work-related musculoskeletal disorders: prevalence, costs and demographics in the EU.*

[6] Bauer, W., Bender, M., Braun, M., Rally, P. E. T. E. R., Scholtz, O. (2016). Lightweight robots in manual assembly—best to start simply. *Frauenhofer-Institut für Arbeitswirtschaft und Organisation IAO, Stuttgart*.

[7] ISO 10218-1. (2011). Robots and robotic devices – Safety requirements for industrial robots – Part 1: Robots. *Norm ISO 10218-1.*

[8] ISO 10218-2. (2011). Robots and robotic devices–safety requirements for industrial robots–Part 2: robot systems and integration. *Norm ISO 10218-2.*

[9] ISO, I. (2016). TS 15066: 2016: Robots and robotic devices–Collaborative robots. *Geneva, Switzerland: International Organization for Standardization*.

[10] https://ifr.org/case-studies/collaborative-robots (last access 23/03/2020).

[11] Hirzinger, G., Albu-Schaffer, A., Hahnle, M., Schaefer, I., Sporer, N. (2001, May). On a new generation of torque controlled light-weight robots. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)* (Vol. 4, pp. 3356-3363). IEEE.

[12] Colgate, E., Bicchi, A., Peshkin, M. A., Colgate, J. E. (2008). Safety for physical human-robot interaction. In *Springer handbook of robotics* (pp. 1335-1348). Springer.

[13] Weitschat, R., Vogel, J., Lantermann, S., Höppner, H. (2017, May). End-effector airbags to accelerate human-robot collaboration. In *2017 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2279-2284). IEEE.

[14] Pratt, G. A., Williamson, M. M. (1995, August). Series elastic actuators. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent*

Robots and Systems. Human Robot Interaction and Cooperative Robots (Vol. 1, pp. 399-406). IEEE.

[15] Vanderborght, B., Albu-Schäffer, A., Bicchi, A., Burdet, E., Caldwell, D. G., Carloni, R., Garabini, M. (2013). Variable impedance actuators: A review. *Robotics and autonomous systems*, *61*(12), 1601-1614.

[16] Navarro, B., Cherubini, A., Fonte, A., Passama, R., Poisson, G., Fraisse, P. (2016, May). An ISO10218-compliant adaptive damping controller for safe physical human-robot interaction. In *2016 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 3043-3048). IEEE.

[17] Magrini, E., De Luca, A. (2016, October). Hybrid force/velocity control for physical human-robot collaboration tasks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 857-863). IEEE.

[18] Akkaladevi, S. C., Heindl, C. (2015, November). Action recognition for human robot interaction in industrial applications. In *2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS)* (pp. 94-99). IEEE.

[19] Potter, L. E., Araullo, J., Carter, L. (2013, November). The leap motion controller: a view on sign language. In *Proceedings of the 25th Australian computer-human interaction conference: augmentation, application, innovation, collaboration* (pp. 175-178).

[20] Hernoux, F., Béarée, R., Gibaru, O. (2015, April). Investigation of dynamic 3D hand motion reproduction by a robot using a Leap Motion. In *Proceedings of the 2015 Virtual Reality International Conference* (pp. 1-10).

[21] Tsarouchi, P., Matthaiakis, A. S., Makris, S., Chryssolouris, G. (2017). On a human-robot collaboration in an assembly cell. *International Journal of Computer Integrated Manufacturing*, *30*(6), 580-589.

[22] El Makrini, I., Merckaert, K., Lefeber, D., Vanderborght, B. (2017, September). Design of a collaborative architecture for human-robot assembly tasks. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1624-1629). IEEE.

[23] Stenmark, M., Nugues, P. (2013, October). Natural language programming of industrial robots. In *IEEE ISR 2013* (pp. 1-5). IEEE.

[24] Chan, K. Y., Yiu, C. K., Dillon, T. S., Nordholm, S., Ling, S. H. (2012). Enhancement of speech recognitions for control automation using an intelligent particle swarm optimization. *IEEE Transactions on Industrial Informatics*, *8*(4), 869-879.

[25] Rozo, L., Bruno, D., Calinon, S., Caldwell, D. G. (2015, September). Learning optimal controllers in human-robot cooperative transportation tasks with position and force constraints. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1024-1030). IEEE.

[26] Peternel, L., Babič, J. (2013). Learning of compliant human–robot interaction using full-body haptic interface. *Advanced Robotics*, *27*(13), 1003-1012.

[27] Krüger, J., Lien, T. K., Verl, A. (2009). Cooperation of human and machines in assembly lines. *CIRP annals*, *58*(2), 628-646.

[28] Neto, P., Mendes, N. (2013). Direct off-line robot programming via a common CAD package. *Robotics and Autonomous Systems*, *61*(8), 896-910.

[29] Billard, A. G., Calinon, S., Dillmann, R. (2016). Learning from humans. In *Springer handbook of robotics* (pp. 1995-2014). Springer, Cham.

[30] Argall, B. D., Chernova, S., Veloso, M., Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and autonomous systems*, *57*(5), 469-483.

[31] Landi, C. T., Ferraguti, F., Secchi, C., Fantuzzi, C. (2016, October). Tool compensation in walk-through programming for admittance-controlled robots. In *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society* (pp. 5335-5340). IEEE.

[32] Landi, C. T., Ferraguti, F., Sabattini, L., Secchi, C., Fantuzzi, C. (2017, May). Admittance control parameter adaptation for physical human-robot interaction. In *2017 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2911-2916). IEEE.

[33] Makris, S., Karagiannis, P., Koukas, S., Matthaiakis, A. S. (2016). Augmented reality system for operator support in human–robot collaborative assembly. *CIRP Annals*, *65*(1), 61-64.

[34] Michalos, G., Karagiannis, P., Makris, S., Tokçalar, Ö., Chryssolouris, G. (2016). Augmented reality (AR) applications for supporting human-robot interactive cooperation. *Procedia CIRP*, *41*, 370-375.

[35] Gammieri, L., Schumann, M., Pelliccia, L., Di Gironimo, G., Klimant, P. (2017). Coupling of a redundant manipulator with a virtual reality environment to enhance human-robot cooperation. *Procedia CIRP*, *62*, 618-623.

[36] Matsas, E., Vosniakos, G. C. (2017). Design of a virtual reality training system for human–robot collaboration in manufacturing tasks. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, *11*(2), 139-153.

[37] Di Gironimo, G., Patalano, S., Tarallo, A. (2009). Innovative assembly process for modular train and feasibility analysis in virtual environment. *International Journal on Interactive Design and Manufacturing (IJIDeM)*, *3*(2), 93-101.

[38] Michalos, G., Makris, S., Tsarouchi, P., Guasch, T., Kontovrakis, D., Chryssolouris, G. (2015). Design considerations for safe human-robot collaborative workplaces. *Procedia CIrP*, *37*, 248-253.

[39] Hernoux, F., Nyiri, E., Gibaru, O. (2015, April). Virtual reality for improving safety and collaborative control of industrial robots. In *Proceedings of the 2015 Virtual Reality International Conference* (pp. 1-6).

[40] Tinós, R., Terra, M. H., Bergerman, M. (2007). A fault tolerance framework for cooperative robotic manipulators. *Control Engineering Practice*, *15*(5), 615-625.

[41] Hentout, A., Messous, M. A., Bouzouia, B. (2015). Fault-tolerant multi-agent control architecture for autonomous mobile manipulators: Simulation results. *Computers & Electrical Engineering*, *43*, 238-256.

[42] Wilbert, A. D., Behrens, B., Dambon, O., Klocke, F. (2012, October). Robot assisted manufacturing system for high gloss finishing of steel molds. In *International Conference on Intelligent Robotics and Applications* (pp. 673-685). Springer, Berlin, Heidelberg.

[43] Levratti, A., De Vuono, A., Fantuzzi, C., Secchi, C. (2016, July). TIREBOT: A novel tire workshop assistant robot. In *2016 IEEE international conference on advanced intelligent mechatronics (AIM)* (pp. 733-738). IEEE.

[44] Ang, M. H., Lin, W., Lim, S. Y. (1999). A walk-through programmed robot for welding in shipyards. *Industrial Robot: An International Journal*.

[45] Ang, M. H., Wei, L., Yong, L. S. (2000, April). An industrial application of control of dynamic behavior of robots-a walk-through programmed welding robot. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)* (Vol. 3, pp. 2352-2357). IEEE.

[46] Grahn, S., Langbeck, B., Johansen, K., Backman, B. (2016). Potential advantages using large anthropomorphic robots in human-robot collaborative, hand guided assembly. *Procedia CIRP*, *44*, 281-286.

[47] Unhelkar, V. V., Shah, J. A. (2015, March). Challenges in developing a collaborative robotic assistant for automotive assembly lines. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts* (pp. 239-240).

[48] Ademovic, A., Lacevic, B. (2014, June). Path planning for robotic manipulators via bubbles of free configuration space: Evolutionary approach. In *22nd Mediterranean Conference on Control and Automation* (pp. 1323-1328). IEEE.

[49] Ademovic, A., Lacevic, B. (2016, May). Path planning for robotic manipulators using expanded bubbles of free c-space. In *2016 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 77-82). IEEE.

[50] Qureshi, A. H., Ayaz, Y. (2015). Intelligent bidirectional rapidly-exploring random trees for optimal motion planning in complex cluttered environments. *Robotics and Autonomous Systems*, *68*, 1-11.

[51] Nieto, J., Slawinski, E., Mut, V., Wagner, B. (2010, March). Online path planning based on rapidly-exploring random trees. In *2010 IEEE International Conference on Industrial Technology* (pp. 1451-1456). IEEE.

[52] Wei, K., Ren, B. (2018). A method on dynamic path planning for robotic manipulator autonomous obstacle avoidance based on an improved RRT algorithm. *Sensors*, *18*(2), 571.

[53] Rosenstrauch, M. J., Pannen, T. J., Krüger, J. (2018). Human robot collaboration-using kinect v2 for ISO/TS 15066 speed and separation monitoring. *Procedia CIRP*, *76*, 183-186.

[54] Malm, T., Salmi, T., Marstio, I., Montonen, J. (2019). Dynamic safety system for collaboration of operators and industrial robots. *Open Engineering*, *9*(1), 61-71.

[55] Byner, C., Matthias, B., Ding, H. (2019). Dynamic speed and separation monitoring for collaborative robot applications–Concepts and performance. *Robotics and Computer-Integrated Manufacturing, 58*, 239-252.

[56] Tsai, C. S., Hu, J. S., Tomizuka, M. (2014, September). Ensuring safety in human-robot coexistence environment. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 4191-4196). IEEE.

[57] Ragaglia, M., Zanchettin, A. M., Rocco, P. (2018). Trajectory generation algorithm for safe human-robot collaboration based on multiple depth sensor measurements. *Mechatronics, 55*, 267-281.

[58] Zanchettin, A. M., Rocco, P., Chiappa, S., Rossi, R. (2019). Towards an optimal avoidance strategy for collaborative robots. *Robotics and Computer-Integrated Manufacturing, 59*, 47-55.

[59] Wang, Y., Ye, X., Yang, Y., Zhang, W. (2017, November). Collision-free trajectory planning in human-robot interaction through hand movement prediction from vision. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)* (pp. 305-310). IEEE.

[60] Wang, Y., Sheng, Y., Wang, J., Zhang, W. (2017). Optimal collision-free robot trajectory generation based on time series prediction of human motion. *IEEE Robotics and Automation Letters*, *3*(1), 226-233.

[61] Casalino, A., Bazzi, D., Zanchettin, A. M., Rocco, P. (2019, May). Optimal Proactive Path Planning for Collaborative Robots in Industrial Contexts. In *2019 International Conference on Robotics and Automation (ICRA)* (pp. 6540-6546). IEEE.

[62] Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. In *Autonomous robot vehicles* (pp. 396-404). Springer, New York, NY.

[63] Flacco, F., Kröger, T., De Luca, A., Khatib, O. (2012, May). A depth space approach to human-robot collision avoidance. In *2012 IEEE International Conference on Robotics and Automation* (pp. 338-345). IEEE.

[64] Safeea, M., Neto, P. (2018, September). Human-robot collision avoidance for industrial robots: A V-REP based solution. In *Transdisciplinary Engineering Methods for Social Innovation of Industry 4.0: Proceedings of the 25th ISPE Inc. International Conference on Transdisciplinary Engineering, July 3–6, 2018* (Vol. 7, p. 301). IOS Press.

[65] Rubio, F., Llopis-Albert, C., Valero, F., Suñer, J. L. (2016). Industrial robot efficient trajectory generation without collision through the evolution of the optimal trajectory. *Robotics and Autonomous Systems*, *86*, 106-112.

[66] Liu, G., Song, C., Zang, X., Zhao, J. (2018). Reactive execution of learned tasks with real-time collision avoidance in a dynamic environment. *IEEE Access*, *6*, 57366-57375.

[67] Long, Z. (2019). Virtual target point-based obstacle-avoidance method for manipulator systems in a cluttered environment. *Engineering Optimization*, 1-17.

[68] Kim, J., Park, J., Hwang, Y. K., Lee, M. (2004, December). Advanced grasp planning for handover operation between human and robot. In *2nd international conference on autonomous robots and agents* (pp. 13-15).

[69] Lopez-Damian, E., Sidobre, D., DeLaTour, S., Alami, R. (2006). Grasp planning for interactive object manipulation. In *Proc. of the Intl. Symp. on Robotics and Automation*.

[70] Eguíluz, A. G., Rano, I., Coleman, S. A., McGinnity, T. M. (2017, December). Towards robot-human reliable hand-over: Continuous detection of object perturbation force direction. In *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)* (pp. 510-515). IEEE.

[71] Han, Z., Yanco, H. (2019, March). The effects of proactive release behaviors during human-robot handovers. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)* (pp. 440-448). IEEE.

[72] Huber, M., Lenz, C., Wendt, C., Färber, B., Knoll, A., Glasauer, S. (2013, August). Increasing efficiency in robot-supported assemblies through predictive mechanisms: An experimental evaluation. In *2013 IEEE RO-MAN* (pp. 503-508). IEEE.

[73] Strabala, K., Lee, M. K., Dragan, A., Forlizzi, J., Srinivasa, S. S., Cakmak, M., Micelli, V. (2013). Toward seamless human-robot handovers. *Journal of Human-Robot Interaction*, *2*(1), 112-132.

[74] Cakmak, M., Srinivasa, S. S., Lee, M. K., Kiesler, S., Forlizzi, J. (2011, March). Using spatial and temporal contrast for fluent robot-human hand-overs. In *Proceedings of the 6th international conference on Human-robot interaction* (pp. 489-496).

[75] Admoni, H., Dragan, A., Srinivasa, S. S., Scassellati, B. (2014, March). Deliberate delays during robot-to-human handovers improve compliance with gaze communication. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction* (pp. 49-56).

[76] Cakmak, M., Srinivasa, S. S., Lee, M. K., Forlizzi, J., Kiesler, S. (2011, September). Human preferences for robot-human hand-over configurations. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 1986-1993). IEEE.

[77] Aleotti, J., Micelli, V., Caselli, S. (2012, September). Comfortable robot to human object hand-over. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication* (pp. 771-776). IEEE.

[78] Mainprice, J., Gharbi, M., Siméon, T., Alami, R. (2012, September). Sharing effort in planning human-robot handover tasks. In *2012 IEEE RO-MAN:*

The 21st IEEE International Symposium on Robot and Human Interactive Communication (pp. 764-770). IEEE.

[79] Pandey, A. K., Ali, M., Warnier, M., Alami, R. (2011, June). Towards multi-state visuo-spatial reasoning based proactive human-robot interaction. In *2011 15th International Conference on Advanced Robotics (ICAR)* (pp. 143-149). IEEE.

[80] Sisbot, E. A., Alami, R. (2012). A human-aware manipulation planner. *IEEE Transactions on Robotics*, *28*(5), 1045-1057.

[81] Parastegari, S., Abbasi, B., Noohi, E., Zefran, M. (2017, September). Modeling human reaching phase in human-human object handover with application in robot-human handover. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 3597-3602). IEEE.

[82] Wang, W., Li, R., Diekel, Z. M., Chen, Y., Zhang, Z., Jia, Y. (2018). Controlling object hand-over in human–robot collaboration via natural wearable sensing. *IEEE Transactions on Human-Machine Systems*, *49*(1), 59-71.

[83] Sidiropoulos, A., Psomopoulou, E., Doulgeri, Z. (2019). A human inspired handover policy using Gaussian Mixture Models and haptic cues. *Autonomous Robots*, *43*(6), 1327-1342.

[84] Koene, A., Remazeilles, A., Prada, M., Garzo, A., Puerto, M., Endo, S., Wing, A. M. (2014). Relative importance of spatial and temporal precision for user satisfaction in human-robot object handover interactions. In *Third International Symposium on New Frontiers in Human-Robot Interaction*.

[85] Prada, M., Remazeilles, A., Koene, A., Endo, S. (2014, September). Implementation and experimental validation of dynamic movement primitives for object handover. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 2146-2153). IEEE.

[86] Pan, M. K., Knoop, E., Bächer, M., Niemeyer, G. (2019). Fast Handovers with a Robot Character: Small Sensorimotor Delays Improve Perceived Qualities. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[87] Nemlekar, H., Dutia, D., Li, Z. (2019, May). Object transfer point estimation for fluent human-robot handovers. In *2019 International Conference on Robotics and Automation (ICRA)* (pp. 2627-2633). IEEE.

[88] Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G. (2010). *Robotics: modelling, planning and control*. Springer Science & Business Media.

[89] https://www.coppeliarobotics.com/ (last access 23/03/2020).

[90] https://optitrack.com/products/v120-trio/ (last access 23/03/2020).

[91]https://www.coppeliarobotics.com/helpFiles/en/legacyRemoteApiOverview.htm (last access 23/03/2020).

[92] https://www.kuka.com/en-gb/products/robotics-systems/industrial-robots/lbr-iiwa (last access 23/03/2020).

[93] https://www.universal-robots.com/products/ur3-robot/ (last access 23/03/2020).

[94] Hartenberg, R. S., Denavit, J. (1955). A kinematic notation for lower pair mechanisms based on matrices.

[95] J. J. Craig, Introduction to Robotics, MA, Reading:Addison Wesley, 1986.

[96] https://www.universal-robots.com/how-tos-and-faqs/faq/ur-faq/parameters-for-calculations-of-kinematics-and-dynamics-45257/

[97] Corke, P. (2017). *Robotics, vision and control: fundamental algorithms in MATLAB® second, completely revised* (Vol. 118). Springer.

[98] https://robotiq.com/products/2f85-140-adaptive-robot-gripper (last access 23/03/2020).

[99] Yeung, K. Y., Kwok, T. H., Wang, C. C. (2013). Improved skeleton tracking by duplex kinects: A practical approach for real-time applications. *Journal of Computing and Information Science in Engineering*, *13*(4).

[100] https://www.universal-robots.com/how-tos-and-faqs/how-to/ur-how-tos/remote-control-via-tcpip-16496/ (last access 23/03/2020).

[101] https://www.universal-robots.com/download/?option=62055#section61793 (last access 23/03/2020).

[102] https://assets.robotiq.com/website-assets/support_documents/document/2F-85_2F-140_UR_PDF_20200211.pdf?_ga=2.117098886.1883704758.1584355603-1454715834.1573461499 (last access 23/03/2020).

[103] https://www.microsoft.com/en-us/p/azure-kinect-dk/8pp5vxmd9nhq?activetab=pivot:techspecstab (last access 23/03/2020).

[104] Sell, J., O'Connor, P. (2014). The xbox one system on a chip and kinect sensor. *IEEE Micro*, *34*(2), 44-53.

[105] Zennaro, S., Munaro, M., Milani, S., Zanuttigh, P., Bernardi, A., Ghidoni, S., Menegatti, E. (2015, June). Performance evaluation of the 1st and 2nd generation Kinect for multimedia applications. In*2015 IEEE International Conference on Multimedia and Expo (ICME)* (pp. 1-6). IEEE.

[106] Sarbolandi, H., Lefloch, D., Kolb, A. (2015). Kinect range sensing: Structured-light versus Time-of-Flight Kinect. *Computer vision and image understanding*, *139*, 1-20.

[107] Yang, L., Zhang, L., Dong, H., Alelaiwi, A., El Saddik, A. (2015). Evaluating and improving the depth accuracy of Kinect for Windows v2. *IEEE Sensors Journal*, *15*(8), 4275-4285.