

UNIVERSITY OF MODENA AND REGGIO EMILIA

DEPARTMENT OF SCIENCES AND METHODS FOR ENGINEERING

DOCTOR OF INDUSTRIAL INNOVATION ENGINEERING

XXXV CYCLE

Safety in collaborative robotics and human-robot interaction

Author:

Mattia Bertuletti

Supervisor:

Cesare Fantuzzi

PhD Coordinator:

Franco Zambonelli

December 2022

To my family

Abstract

The advent of the collaborative robotics brought new interaction possibilities with the automation of the processes. The physical barriers removing, which confined the robot in restricted zone, or the allowed physical interaction between human and robot opened the doors to new ways of the integration of the machines into the factories.

This thesis analyzes the robot integration in industrial application, in which some human safety-oriented controls are identified and treated in context where human and robot collaborate or coexist.

The European and international standards regulate these kind of applications, where human can collaborate with the machine in several ways; however, at the expense to machine performance reduction and, thus, of productivity. The limits of the *Power and Force Limiting* (PFL) and *Hand-Guiding* (HG) interaction method, of the standard ISO/TS 15066, are the performances reduction of the machines, namely velocities or forces reduction, during the whole work-cycle.

The thesis goal is to identify control systems able to maximize the robot performance while fulfilling to ISO 10218 and ISO/TS 15066 regulations, limiting the robot performances only when the safety conditions really force.

List of Figures

1.1	R2D2 and C3PO of Star Wars fame	1
1.2	Example of robot uses. 1.2(a) is an example of a robot vacuum cleaner. 1.2(b) is an example of industrial robot installation inside manufacturing industry. 1.2(c) is an example of collaboration between human and a collaborative robot	2
1.3	The CE mark	5
1.4	European Machinery Directive: the harmonised standards	6
1.5	Workspace representation for EN ISO 10218-2	7
1.6	The four collaboration modality identified by ISO 10218 and ISO/TS 15066	8
1.7	The speed and separation monitoring graphical representation: speed and separation distance	11
1.8	Force and Pressure graphical representation, according to ISO/TS 15066. F_T and p_T are the transient contact force and pressure, while F_S and p_S are the quasi-static contact force and pressure	12
1.9	Biomechanical limits of human body parts. Page 24 of [53]	13
1.10	Effective masses and spring constants for the body model. Page 27 of [53]	13
2.1	Definition of the safe operating region (yellow area). If the robot moves with a velocity that guarantees that the relative velocity with respect to the human lies inside the yellow area, the trajectory is safe.	20
2.2	Definition of the end-effector and human body part positions and velocities.	21
2.3	Approximation of the safe set through a set of ellipses.	25
2.4	Paraboloid-shaped CBFs defined on the set of ellipses that approximate the safe set.	26
2.5	Robot trajectory (nominal in solid blue, executed in dashed blue) and obstacle trajectory (dashed red).	29
2.6	Experiments setup and obstacle approaching	29
2.7	Cartesian robot positions: nominal trajectory (dashed line) and ZCBF constrained trajectory (solid line).	30
2.8	Distance (red solid line), relative velocity (green solid line), PFL relative velocity bound (black dotted line) and maximum robot velocity (magenta dashed line).	31
3.1	Overall control architecture.	39
3.2	Geometry representation of two separate capsules and of the minimum distance between them.	47
3.3	Example of operator's capsule and capsule-based geometry model of a 6 DoF offset wrist manipulator. In 3.3(a) - 3.3(c) the minimum distance condition is respected, while in 3.3(d) - 3.3(f) it is violated.	48
3.4	X-Y plane representation of the fixed relative pose (dashed blue) between robot's end-effector (solid yellow) and operator's capsule (dashed green), and of the varying relative pose (dashed black) between robot's base and operator's capsule in three different robot configurations.	49

3.5	Geometric representation of the relative velocity between the end-points of a generic robot link (solid grey bar) and the operator's capsule. Capsules are pictured as dashed black lines. Capsules' defining points are represented as solid black dots, and clearances are highlighted with solid black arrows. Finally, robot link endpoints' velocities are depicted as blue arrows, while the corresponding projections towards the operator capsule are displayed as red arrows.	52
3.6	Gaiotto GA-OL manipulator (on the right, pictured in blue) and auxiliary turning table (on the left, pictured in grey).	53
3.7	CAD representation of the mechatronic handle mounted on the end-effector of the GA-OL manipulator.	55
3.8	System environment and software architecture.	56
3.9	Experimental setup: 3.9(a) linear trajectory with enumerated targets "A" to "F", 3.9(b) floor-mounted toilet, 3.9(c) wall-hung toilet.	63
3.10	Approach orientation of labeled targets during the first experiment (3.10(a) and 3.10(b)) versus approach orientation during the second experiment (3.10(c) and 3.10(d)).	64
3.11	Adaptation of admittance parameters in presence of detected oscillations.	67
3.12	Violation of TCP linear velocity limit and corresponding variation of scaled STO signal, with magnification between 1.70 s and 2.10 s.	68
3.13	Violation of joint #02 velocity limit with TCP linear velocity and corresponding variation of scaled STO signal, with magnification between 13.30 s and 13.72 s.	68
3.14	Violation of joint acceleration limit with TCP linear velocity and corresponding variation of scaled STO signal, with magnification between 12.30 s and 12.70 s.	69
3.15	Violation of the relative human-robot minimum distance with TCP linear velocity and corresponding variation of scaled STO signal, with magnification between 59.25 s and 59.75 s.	70
3.16	Violation of the relative human-robot maximum velocity with TCP linear velocity and corresponding variation of scaled STO signal, with magnification between 3.50 s and 3.80 s.	70
3.17	Approximated distribution of maximum recorded TCP linear speed values grouped by testers and by admittance parameter sets. Median value is depicted as a solid orange line, while average value is represented as a green triangle.	71
3.18	Comparison between approximated distribution of maximum TCP Speed in GA-2000 programs (labeled as "Mechanical") and during the selected experiments (labeled as "Proposed"). Median value is depicted as a solid orange line, while average value is represented as a green triangle.	72
3.19	Approximated distribution of ratio between program time and programming duration. Median value is depicted as a solid orange line, while average value is represented as a green triangle.	73
3.20	Evaluation of Motion Indicators for each Parameter Set. For each indicator, the box-whiskers plot on the left shows the approximated distribution of the rates assigned by the testers grouped by the active parameter set, by highlighting the median (solid orange line) and the mean (green triangle). Then, the histograms display the normalized frequency of the assigned rates versus the active parameter set.	75
3.21	Evaluation of general indicators. For each indicator, the box-whiskers plot on the left shows the approximated distribution of the rates assigned by the testers, by highlighting the median (solid orange line) and the mean (green triangle). Then, the histogram on the right displays the normalized frequency of the assigned rates.	76
4.1	Example of 6 DOF Articulated arm	85

4.2	GA-OL simplified representation with joint five value at 90°	86
4.3	Translational manipulability ellipsoid for wrist singularity configuration with $\theta_5 = 90^\circ$	87
4.4	Overall control architecture with EDLS singularity avoidance strategy	88
4.5	Robot trajectory passed through the wrist singularity	89
4.6	Singular values and Reciprocals with Fast Asymmetric damping parameters	91
4.7	Singular values and Reciprocals with Slow symmetric damping parameters	92
4.8	σ_6 singular value reciprocal optimized with EDLS computed with Fast Asymmetric (red line) and Slow Symmetric (blue line) parameters	93
4.9	Wrist joint positions and Singularity distance funtion with Fast Asymmetric damping parameters	94
4.10	Wrist joint positions and Singularity distance funtion with Slow Symmetric damping parameters	94
5.1	Overall structure of the Robish system for automated picking of urban waste.	99
5.2	Software architecture which handles the overall system.	100
5.3	U-NET simplified application example architecture	102
5.4	Example of U-NET Implementation for Robish system	103
5.5	Example image dataset for Robish model	104
5.6	Example of labeling of an image of Robish dataset	105
5.7	Local reference system of the bin geometrical model	106
5.8	Example of geometrical model estimated on image after the bin detection	106
5.9	Data exchange between robot and control system (server and robot communication process)	108
5.10	Pipeline of a simple bin emptying from the robot	109
5.11	Check pose practical scheme with robot TCP and Bin Reference systems	110
5.12	Example of the DCS Cartesian position check output on Roboguide Fanuc software	111
5.13	Semplification of the case study of singularity avoidance	112
5.14	Laser pointer used to compensate the pose estimation error	113
5.15	Robish system with external safety devices remarks	118
5.16	Visibility scheme of the operator inside the truck cabinet, exploiting the right side mirror.	119
5.17	Closed circuit monitor of one cameras and the internal led light signaling	120
5.18	Scheme of the transformer connection with respect to the ground	120
5.19	Isolation monitoring device	121
5.20	Dead man switch signal control on safety control unit	121
5.21	Hand brake pneumatic scheme	122
5.22	General hardware architecture of Robish	123
5.23	Protective cover scheme of robot R2000iC/210F	124
5.24	HDR processing result starting from a 7 images acquired with different exposure times	126
5.25	Small bin on buffer position	126
5.26	Handle release action	126
5.27	Emptying without handle lock	126

List of Tables

2.1	Parameters setting for simulations and experiments	28
3.1	Admittance control parameter sets considered during the experimental validation campaign. For the sake of clarity, linear parameters are labelled with the subscript “p”, while orientation-related quantities are labelled with the “o” subscript. . . .	65
4.1	Fast Asymmetric damping parameters	90
4.2	Slow Symmetric damping parameters	90
5.1	The most relevant norms and directives used to risk assessments	116
5.2	DFK33GX174 technical data	123
5.3	M112FM06 technical data	123
5.4	Percentage success results for tests conducted by developers	127
5.5	Percentage success results for tests conducted by ecological operator	128

List of Publications

Journal papers

- F. Ferraguti, M. Bertuletti, C. T. Landi, M. Bonfè, C. Fantuzzi, and C. Secchi, “A control barrier function approach for maximizing performance while fulfilling to iso/ts 15066 regulations,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5921–5928, Oct 2020
- F. Ferraguti, M. Bertuletti, M. Gambazza, and M. Ragaglia, “High-velocity walk-through programming for industrial applications,” *Robotics and Computer-Integrated Manufacturing*, vol. 81, p. 102505, 2023
- F. Ferraguti, M. Bertuletti, N. Battilani, and C. Fantuzzi, “A robotic system for automated door-to-door urban waste collection,” To be submitted

Patents

- —, “Metodo per il controllo in autoapprendimento di un sistema robotizzato chiuso e relativo impianto di lavorazione,” Patent request 102 021 000 024 899
- F. Ferraguti, M. Bertuletti, M. Gambazza, M. Ragaglia, and C. Fantuzzi, “Metodo per il controllo di sicurezza in autoapprendimento di un sistema robotizzato chiuso e relativo impianto di lavorazione,” Patent request 102 021 000 024 905

Contents

Abstract	v
List of Figures	vii
List of Tables	xi
List of Publications	xv
Table of Contents	xvii
1 Introduction	1
1.1 Collaborative Robotics and Industrial applications	2
1.2 Safety requirement and normatives	5
1.2.1 ISO/TS 15066	7
1.3 Contribution and Thesis Outline	14
2 A human-robot interaction scenario regulated by "Power and Force Limiting" while Maximizing robot performance	15
2.1 Introduction	15
2.2 Power and Force Limiting according to ISO/TS 15066	17
2.3 Definition of the safe operating region	19
2.4 A ZCBF-Based Framework for Human-Robot PFL Operations	22
2.5 Experiments	27
2.6 Conclusion	32
3 Optimization of the "Hand-Guiding" interaction method with High-Velocities and an Industrial Robot	33
3.1 Introduction	33
3.2 Problem statement	37
3.3 General Architecture	38
3.4 Functional Logic Design	40
3.5 Safety Logic Design	44
3.6 Implementation	52
3.7 Experimental Results	60
3.8 Conclusion and Future Developments	74
4 Singularity Avoidance Implementation on Walk-through Programming for Industrial Applications	79

4.1	Introduction	79
4.2	Related works	82
4.3	Problem Statement	84
4.4	Implementation of E-DLS	87
4.5	Experiments and Results	89
4.6	Conclusion	93
5	A no-physical interaction human-robot coexistence situation integrated into a Robotic System for Automated Picking of Urban Waste	95
5.1	Introduction	95
5.1.1	State of the Art	96
5.1.2	Contribution and Organization of This Chapter	98
5.2	Overall Architecture of the Automated Picking System	99
5.2.1	Software architecture	100
5.3	Vision System for Recognizing the Bins	101
5.3.1	Detection of the instances of bins in the image	101
5.3.2	Pose estimation	105
5.4	Robotic and Gripping system	107
5.4.1	Bin manipulation	110
5.4.2	Bin position feedback control	112
5.4.3	Weight estimation	113
5.5	Safety system	115
5.5.1	Risk analysis and reduction process	115
5.5.2	Risk summary	117
5.5.3	External safety devices	117
5.6	Implementation and Validation	122
5.6.1	Overview of hardware system	122
5.6.2	Protective cover	124
5.6.3	Image acquisition: High Dynamic Range filter	124
5.6.4	Small bin handle release	126
5.6.5	Validating tests and results	127
5.7	Conclusions	128
5.7.1	Future Developments	130
	Conclusions and Future Directions	131

Chapter 1

Introduction

When we think to the "Robotics" or "Robot" term, the firsts images in mind are the sentient robot of the "Star Wars" cinematographic saga 1.1.

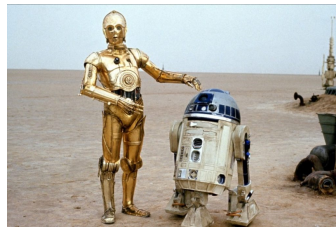


Figure 1.1: R2D2 and C3PO of Star Wars fame

The authors of the brand always wanted transmit the image where human and robot could live together, where robot helps the human with the hardest daily tasks. Leaving aside the narrative of the science fiction work, the years in which "Star Wars" was shown on small and large screens were premature to convince the people a technological conceptual and development, like the one imagined, was praticable. Instead, focusing on the robotic aspect, already in the 60s the firsts industrial robots were born. They were designed to carry out simple, but hard and fast, tasks. Therefore, taking into account the George Lucas' fictional universe and the industrial robotics development until today, it is no longer so science fiction to think that human and robot could live and collaborate together during everyday life.

Thinking about today's world, it is useful to point out how robots have become part of our lives. E.g., several robots vacuum cleaner populate the electronic store shelves or ours houses. Indeed, a repetitive and boring task like to do the home cleaning is perfectly designed to a robot 1.2(a), instead of to compute repetitive and hard task like assembly an electromechanical device. E.g., Fanuc, one of the most leader on producing industrial robots, uses his robots to assembly new robots 1.2(b). These are only few examples of how robot operates next to us and for us.

One aspect to take into consideration, when a machine works next to the human, is that the machine is not a human, therefore it is important to think to the human health. In order to guarantee the safety of the human operator, since when industrial robots were created and installed inside production facilities, robot were confined inside the physical fences. Indeed, the

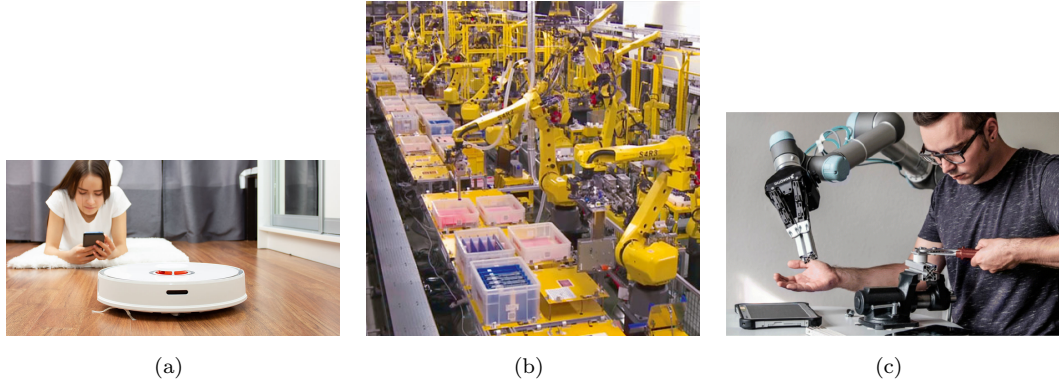


Figure 1.2: Example of robot uses. 1.2(a) is an example of a robot vacuum cleaner. 1.2(b) is an example of industrial robot installation inside manufacturing industry. 1.2(c) is an example of collaboration between human and a collaborative robot

parameters of the perimeter protections and the safety requirements are discussed and treated by the normative UNI EN ISO 10218-2 [51]. Of course, this solution allowed to protect the human from accidents caused by a collision between the mechanical arm and himself, but several disadvantages there are: the physical fences are bulky and, usually, the available space inside a factory is very valuable. Moreover, a robot confined in a pre-defined region with barriers causes a poor programming flexibility of itself. For these reasons and thanks to the technologic development on sensorization field, the physical fences were substituted by virtual fences, created by robot position sensors of laser barriers. The virtual barriers guarantee more flexibility inside an industrial area, in particular when a layout redefinition is required.

Today, the principle challenge is to continue on develop robotic technologies, in which human and robot could collaborate with completely safety and ergonomics, but optimizing his performance in speed and strength. Indeed, figure 1.2(c) shows how a robot could help a human in a particular task. In order to guarantee the safety of the human, a collaborative robot (introduced in 1.1) is used to bring an object on the human hand. Thanks to the robot sensorization and perception, the human can quietly work.

1.1 Collaborative Robotics and Industrial applications

In industrial manufacturing there is an increasing need for flexible machines that are able to work autonomously and can be adapted to changing production conditions quickly and efficiently. To protect persons against the dangers posed by their speed, movement, and force, robots usually work behind a barrier guards. However, if close interaction between human and robot is required, this effective standard method of physically separating the person at risk from the source of danger cannot be employed. For this reason, alternative methods must be applied to reduce risk. The interaction of humans with robots is a generic concept that can be characterized and classified based on two interaction parameters: space and time.

- No common space and no common time in which the human and robot move: *Not interactive situation*
- No common space but human and robot are working at the same time: *Coexistence situation*
- Human and robots share a common space but at different times: *Cooperative situation*
- Humans and robots are working in the same space at the same time: *Collaborative situation*

A no-interactive situation between human and robot implies a coexistence of these two figures. Even in industrial robot applications in which no human intervention is required during the production process, it will still be necessary for an operator to enter into the robot's workspace, e.g., for the purpose of maintenance work. In applications of this type, the workspace must be fenced off and the access doors must be interlocked. The interlock must ensure that hazardous robot functions are shut down whenever an operator enters the hazardous area and must be maintained as long as a person remains present inside the hazardous area or the access doors are opened.

In cooperative application scenarios operator and robot complete the necessary stages of the process in the same workspace at different times. Here too, technical safety measures are required. Depending on the robotic cell setup, it may be appropriate to use opto-electronic protective devices such as safety light curtains or safety laser scanners.

In others applications, it is necessary for humans and robots to interact in the same workspace at the same time. In these scenarios, which we call collaborative, the force, speed, and movement paths of the robot must be restricted. If they are available, inherent safety measures or additional safety measures such as limiting torque through the performance of the drives or safety-related parts of the system controller can be used to minimize risk.

Therefore, the collaborative situations inside the industrial scenario brought out the concept of **Collaborative Robotics**, from which some advantages stand out:

- Shared workspace: The removal of the physical barriers, no space reserved for robots. Moreover, a cost reduction could be reached.
- High-level perception: Thanks to the sensorization, the robot reacts to what hear and/or see. Then a smart rescheduling of the task is possible.
- Human-robot synergy: Synergy between the complementary skills of human and robot and Carry out operations that neither human nor robot would be able to perform

The concept of the collaborative robotics, and then the advent of industry 4.0, accelerated the technologic development bringing to the market the **Collaborative Robot** or *Cobot*. Cobots are smaller and lighter than industrial robots. They are characterized by high flexibility and able to do more tasks. Cobot's idea was born in 1994 by a General Motors initiative, where

the company tried to integrate the robotics into the automotive sector: the goal was to create autonomous machinery able to interact in complex manner with the human. Firsts cobots didn't have driving force, while the human, through the use of peripherals or control panels, managed the movements. Collaborative robots gained driving force in the 2004, when KUKA launched on the market LBR3 model. Universal Robot UR5 realized in the 2008 and this was the first cobot able to do his job near to the human operator with completely safety for him. Since 2008, flexible collaborative robot epoch started.

Generally the cobot features could be summarize in:

- Integrated sensors: perceiving an external force which, if excessive, leads to the interruption of the robot's movement.
- Compliant behavior: in the event of a collision, the external force is detected and the joint reacts by moving in the opposite direction to avoid damage.
- Over-voltages detection: a safety stop can be generated at the software level if a current spike is detected.
- Ease of programming (manual guidance)
- Flexibility and dexterity

These new technologies, instead of the traditional robots, are not suitable for moves high payloads or high speed task. However, cobots could be used in several industrial applications:

- Pick and Place: the task imply to move an object from a position to another position. Typically, a gripper mounted on the end-effector flange help the robot to bring the object and, during the robot motion, holds its position. In order to reduce cycle time, this task can be done next to a human operator.
- Machine tending: the piece load and unload from an automatic machine is a common task. Assuming a constant refueling of raw parts, a cobot could infinite work with the machine enslavement. Moreover, the collaborative robot, could easily adapt with respect to the production mix.
- Packaging and palletizing: the operation of ordering a pallet and moving the objects on it is a repetitive and tiring task. The cobot use on this operation could synchronize the whole production process, increase the operations velocities and improve the quality package.
- Sanding, polishing, deburring and rectification: even if a traditional robot can play the same operations, a cobot allows to have more flexibility if a frequent reprogramming is necessary.
- Injection printing: the advantage of the cobot use on injection printing operation is the space reduction of the robot installation.

- Assembly
- Quality control

1.2 Safety requirement and normatives

Since firsts industrial application of the robotics, imagining the robot as dangerous machines, the safety of the humans operators was the basic theme. Thus, in the world, a research of safety requirement standardization starts. Taking into account the European scope, the directives must be implemented at national legislative level in all the states members of the European Union. Basically the directives are implemented with 1:1 ration in laws or regulations at national level and, then laws and regulations must comply with them.

In order to attest a product, or a system integrated, a mark introduced. The CE mark 1.2 on a product indicates that the manufacturer or importer of that product affirms its compliance with the relevant EU legislation and the product may be sold anywhere in the European Economic Area (EEA). It is a criminal offence to affix a CE mark to a product that is not compliant or offer it for sale [23]. The CE mark means that the machine complies with the directives ap-

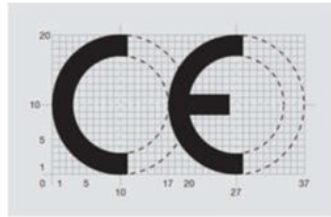


Figure 1.3: The CE mark

plicable to the machine itself. The Machinery Directive 2006/42/CE [17] defines the essential safety requirements to which the machines must follow during their design, manufacture and operation. This before that machines entered into the market. Moreover, Machinery manufactured in conformity with a harmonised standard, the references to which have been published in the Official Journal of the European Union, shall be presumed to comply with the essential health and safety requirements covered by such a harmonised standard. The standards are not mandatory, but their adoption has a consequence of reversing the burden of proof. Indeed, with the event of dispute, the company is required to prove its innocence and who disputes incorrect adoption of the standard must prove the guilt of that company.

The International Organization for Standardization is an independent, non-governmental organization, whose membership consists of different national standards bodies.[46] As of 2022, there are 167 members representing ISO in their country, with each country having only one member. The organization develops and publishes international standards in all technical and nontechnical fields other than electrical and electronic engineering, which are the responsibility of the International Electrotechnical Commission. Today the ISO norms are more than 800 and they are divided in three categories:

- Type A: basic safety norms, which provide design concepts and principles and general aspects could be applied to the machines.
- Type B: they treat a safety aspect or a kind of protection could be applied on various machinaries. Type B norms are themselves divided in type B1 and B2 norms.
- Type C: they treat the detailed safety requirements for a particular machine or a set of machines with processes shared.



Figure 1.4: European Machinery Directive: the harmonised standards

One of the main reference norm for machine safety is the UNI EN ISO 12100. Published in 2010 [52], this type A norm, in order to do a correct *risk analysis and evaluation*, describes the step to follow. The steps are logical steps, defined to correctly identify the dangers and, consequently, evaluate machine associated risks as long as alive. The protection measures and risk reduction could be integrated (and it is also preferable) during the machine design process. The ISO 12100 takes into account a process design where a robot must be integrated and a tolerable risk level must be reach.

The type B norms are intended to support the main procedure of the ISO 12100. In particular the type B1 norms treat the specific aspects of the safety, e.g. the safety minimum distance, surface temperature of noise. While, type B2 norms provided the requirements of protection design and realization and provided the guidelines to test and validate the safety device with desired performances.

Moreover, type C norms support the ISO 12100, but their aim is to define the machines limits and the dangers they generate.

In the July 2011, the EN ISO 10218-1 *Robots and robotic devices – Safety requirements for industrial robots. Part 1: Robots* [50] is published. This is a type C norm for industrial robots; however, the norm principles could be extended to other kinds of robot. The aim of the norm is to provide the requirements for safety design, protective measures and information to safety use the robot inside and industrial environment. Some examples that the norm treats are the robot stop function, the velocity control and the motion space limitation. Indeed, for the stop function the norm assert that *Every robot must have a protective stop function and an emergency stop function independently. Each stop function must provide the connection of external protective*

devices.. For the velocity limitation, usually, a robot is equipped by a velocity reduced mode, where his cartesian velocity is programmable; typically the reduced velocity is set to the cartesian velocity of the robot teaching mode, namely to 250 mm/s . Finally, a robot must have a space limitation device to prevent the robot moves over the programmed bounds. This function is necessary when the robot workspace area is greater than the protected area, instead of to prevent collision with objects or people.

Moreover, the ISO 10218-1 introduced the minimum performances of the safety robotic system and they defined them like *Performance Level (PL)*. The PL is design on a 5-level ladder, from *a* to *e* with increasing reliability.

Still in 2011, the EN ISO 10218-2 *Robots and robotic devices – Safety requirements for industrial robots. Part 2: Robot systems and integration*[51] is published. This is the second part of the type C norm ISO 10218 and it treats the dangers and the minimum safety requirements caused by an industrial robot integration inside the production lines. Moreover, the norm describes the identified dangers and provides the requirement to delete or reduce them. The ISO 10218-2 is the reference normative for robotic islands, where motion space are defined 1.5:

- Max space 1.5(a): space part where robot is able to reach without limiting devices;
- Workspace 1.5(b): space that robot could cover during program executing;
- Protected space 1.5(c): delimited space from perimeter barriers.

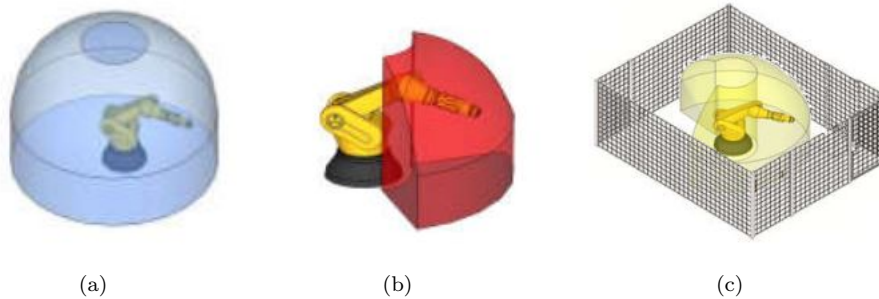


Figure 1.5: Workspace representation for EN ISO 10218-2

The motion space must be limit through hardware or software system compliant with the standard UNI EN ISO 10218-1. The perimeter barriers constitutes *spacers barriers*, whose onyl purpose is to prevent or limit the access to a dangerous zone. The plant layout could be designed to guarantee the operator work outside the perimeter barriers. Moreover, the norm provided the requirements for maintenance, stocked energy management and for plant power supply.

1.2.1 ISO/TS 15066

In the 2016, the ISO-TS 15066 *Robots and robotic devices – Collaborative robots* [53] is published. This norm defined the guidelines for collaborative applications and support or integrate the safety standard and requirements of [50] and [51]. Thanks to the new ways of work with robot 1.1, the

technical specification 15066 updates the ISO 10218 norms, defining new safety requirements and risk levels for these operations.

The safety norms ISO 10218-1/2 and ISO/TS 15066 identify four collaboration modality 1.6 and, depending on the application, they could be independently or at the same time used.

- **Safety-rated Monitored Stop (SMS):** The robot is stopped during interaction with the operator in the collaboration space. This status is monitored and the drive can remain energized.
- **Hand-guiding:** The safety of the human-robot collaboration is assured by the robot being guided manually under control at an appropriately reduced speed.
- **Speed and separation monitoring (SSM):** The speed and movement paths (trajectory) of the robot are monitored and adjusted based on the speed and position of the operator in the safeguarded space.
- **Power and force limiting (PFL):** Physical contact between the robot system (including the workpiece) and the operator can take place either intentionally or unintentionally. The necessary safety is achieved by limiting the power and force to values at which injuries and risk are not to be expected. It is required to have robots designed specifically for this operating mode. ISO/TS 15066 includes biomechanical load limits which must not be exceeded should the robot collide with body parts.

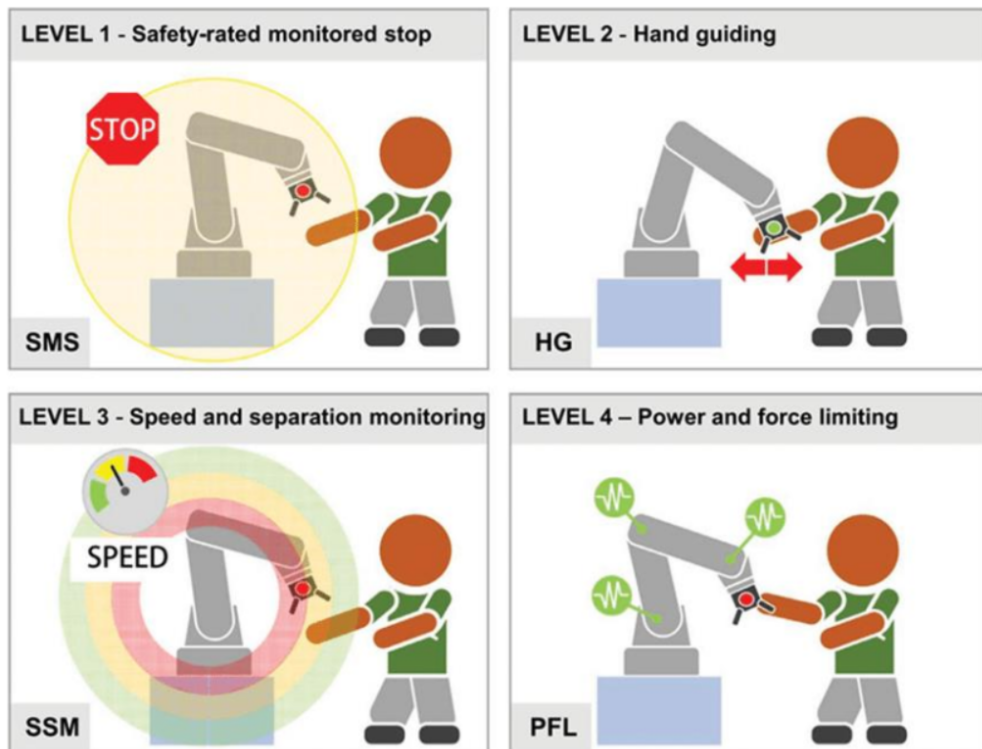


Figure 1.6: The four collaboration modality identified by ISO 10218 and ISO/TS 15066

The **Safety-rated Monitored Stop (SMS)** is used to cease robot motion in the collaborative workspace before an operator enters the collaborative workspace to interact with the robot system

and complete a task (e.g. loading a part onto the end-effector). If the operator is not present in the collaborative workspace, the robot may operate non-collaboratively in the collaborative workspace. This is a conservative mode because the collaboration is limited with a robot in a stand-still position. According to ISO 10218-1, the motion interruption of the robot is done through a category stop 2, namely without powering off the motors.

When the safety-rated monitored stop feature is used an operator shall be permitted to enter the collaborative workspace only under the following conditions:

- When the robot system or other hazards are not present in the collaborative workspace.
- When the robot system is in the collaborative workspace and is in a safety-rated monitored stop (stop category 2) in accordance with ISO 10218-1:2011, 5.4. The safety-rated monitored stop shall remain active at all times when an operator is in the collaborative workspace.
- When the robot system is in the collaborative workspace in a protective stop in accordance with ISO 10218-1:20 11, 5.4 and 5.5.3.

Matched with one or more manual device, the **Hand-guiding (HG)** mode allowed to the operator to transmit motion command to the robotic system. The task is carried out by manually actuating guiding devices located at or near the robot end-effector. Before the operator could enter in the workspace area, the robot has to perform a SMS. Only when the human takes the control of the robotic system, the robot motion is allowed. When the operator leaves the manual device, the robot executes again a SMS and, when the operator leaves the workspace, can resume the non collaborative operation.

In order to execute the HG in safety way, a velocity monitoring function, like the **Speed and separation monitoring (SSM)** must be present. Thanks to this function, the robot and operator may move concurrently in the collaborative workspace. During robot motion, the robot never gets closer to the operator than the protective separation distance. When the separation distance decreases to a value below the protective separation distance, the robot stops. When the operator moves away from the robot, the robot can resume motion automatically. When the robot reduces its speed, the protective separation distance decreases correspondingly. The maximum velocities and separation distances allowed on an application can be variables or constants and they can be determined through the risk evaluation. In particular, the minimum separation distance can be calculated by 1.1 from ISO 13855 with some modification for SSM function.

$$S_p(t_0) = S_h + S_r + S_s + C + Z_d + Z_r \quad (1.1)$$

S_p : minimum safety distance at time t_0 ;

S_h : distance contribution caused by operation motion;

S_r : safety distance contribution, taking into account reaction time of robotic system;

S_s : travelled space from robot after the system commanded a stop;

C : introduction distance, namely the distance that a human limb can travel before the sensors identify it;

Z_d : operator position uncertainty;

Z_r : robot position uncertainty;

$S_p(t_0)$ allow to dinamically compute the separation distance, allowing to the robot to vary the velocity during the application. For the computation of S_h , S_r and S_s the references equation are 1.2, 1.3 and 1.4.

$$S_h = \int_{t_0}^{t_0+T_r+T_s} v_h(t) dt \quad (1.2)$$

$$S_r = \int_{t_0}^{t_0+T_r} v_r(t) dt \quad (1.3)$$

$$S_s = \int_{t_0+T_r}^{t_0+T_r+T_s} v_s(t) dt \quad (1.4)$$

T_r : reaction time of robot, namely the time to detect the operator position and to activate the protective stop;

T_s : time to robot stops after the deceleration starts.

The velocity and separation distance trend are graphed in 1.7. When the minimum distance S_p is outdated, a time T_r elapses before the robot starts the deceleration with nevasive velocity v_s . The robot is completely stopped at the time $t_0 + T_r + T_s$ and the human limb distance must be greater the sum of $C + Z_d + Z_r$.

The most complex collaborative operation is **Power and Force Limiting (PFL)**, where physical contact between the robot (including the workpiece) and an operator can occur either intentionally or unintentionally. The contact events can occur in several ways:

- Intentionally contact situation.
- Accidental contact situation, usually caused by a proper compliance with the working procedures.
- Contact situation caused by a failure.

Moreover, the ISO/TS 15066 locate two kind of contact:

- Transient contact: contact between an operator and part of a robot system, where the operator body part is not clamped and can recoil or retract from the moving part of the robot system. The contact time is very short and depends on the robot inertia, human inertia and the relative velocity of both.

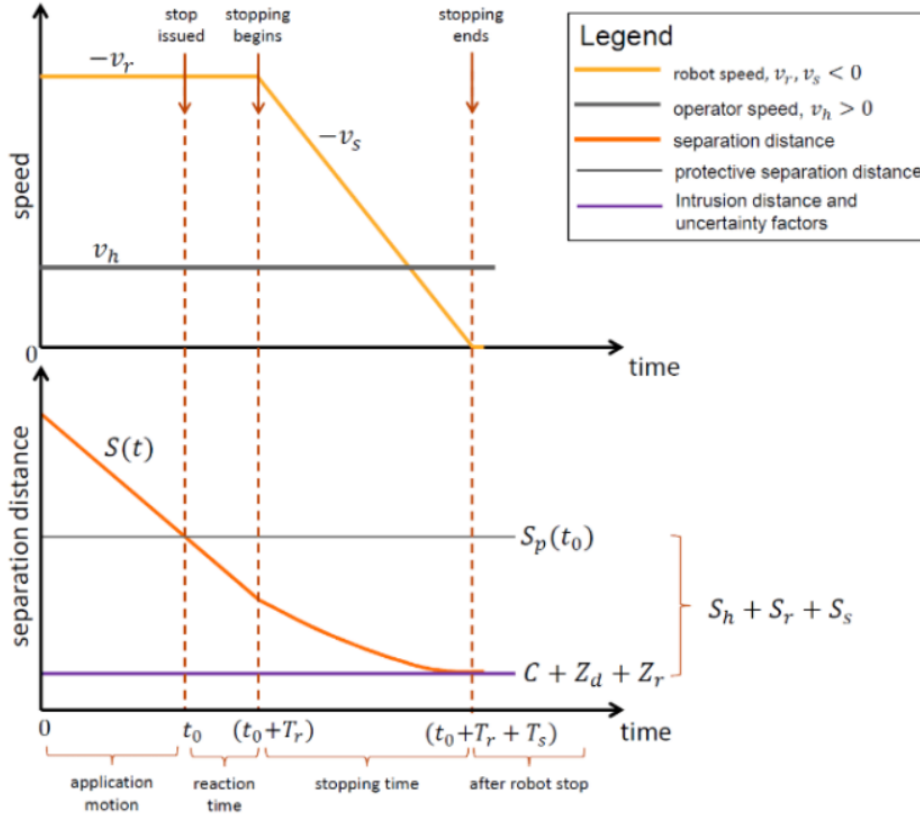


Figure 1.7: The speed and separation monitoring graphical representation: speed and separation distance

- Quasi-static contact: contact between an operator and part of a robot system, where the operator body part can be clamped between a moving part of a robot system and another fixed or moving part of the robot cell. In this situation, the robot should apply a constant pressure and force to the human limb for a prolonged time interval.

The technical specification explains that the robotic system must be designed considering that the transient and quasi-static contact are rare and avoidable. When the contact is not avoidable, there are some methods to the risk reduction:

- Active methods: actions like force and speed limiting.
- Passive methods: in order to limit the impact force, they consider the mechanical design of the robotic system:
 - contact surface increasing, e.g. rounded corners or angles and smooth and compliant surfaces.
 - bump force reduction, increasing the energy transfer time or inserting deformable components and paddings.

Generally, force and pressure are the only difference, in safety terms, between transient and quasi-static contact. Indeed, the quasi-static contact has safety bounds closer than the transient because they imply a locking of a human limb.

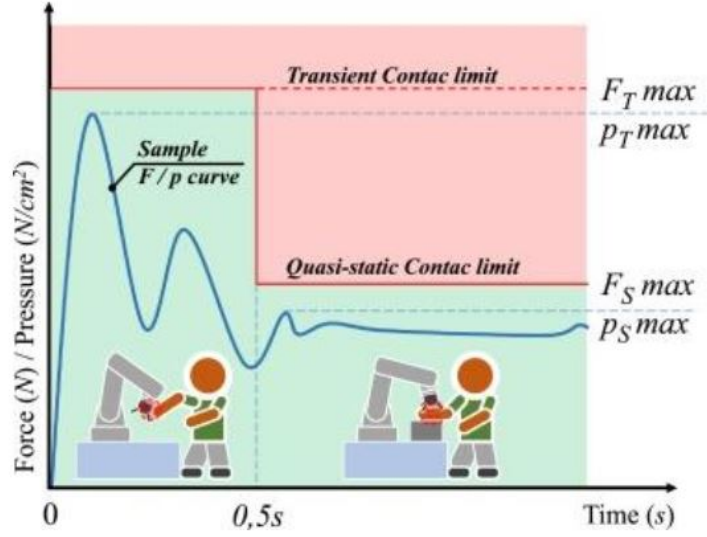


Figure 1.8: Force and Pressure graphical representation, according to ISO/TS 15066. F_T and p_T are the transient contact force and pressure, while F_S and p_S are the quasi-static contact force and pressure

As shown in 1.8, for quasi-static contacts, the threshold values of force and pressure are recorded 0.5 seconds after the impact. Before, the safety values are the same for both contacts. The maximum values of force and pressures identified by the normative are collected in 1.9. 29 body areas had been defined and, for each of them, the values of force and pressure are reported. The values of the transient contact are always twice than the quasi-static ones. Moreover, the contacts between face, skull and front are allowed only with quasi-static contacts.

Thanks to the maximum values of force and pressure, the maximum value of the transfer energy is computed as 1.5.

$$E = \frac{F_{max}^2}{2k} = \frac{A^2 p_{max}^2}{2k} \quad (1.5)$$

A : contact area between robot and body part region.

k : elastic constant relative to body limb, reported in the norm 1.10.

Body region		Specific body area	Quasi-static contact		Transient contact	
			Maximum permissible pressure ^a p_s N/cm ²	Maximum permissible force ^b N	Maximum permissible pressure multiplier ^c P_T	Maximum permissible force multiplier ^c F_T
Skull and forehead ^d	1	Middle of forehead	130	130	not applicable	not applicable
	2	Temple	110		not applicable	
Face ^d	3	Masticatory muscle	110	65	not applicable	not applicable
Neck	4	Neck muscle	140	150	2	2
	5	Seventh neck muscle	210		2	
Back and shoulders	6	Shoulder joint	160	210	2	2
	7	Fifth lumbar vertebra	210		2	2
Chest	8	Sternum	120	140	2	2
	9	Pectoral muscle	170		2	
Abdomen	10	Abdominal muscle	140	110	2	2
Pelvis	11	Pelvic bone	210	180	2	2
Upper arms and elbow joints	12	Deltoid muscle	190	150	2	2
	13	Humerus	220		2	
Lower arms and wrist joints	14	Radial bone	190	160	2	2
	15	Forearm muscle	180		2	
	16	Arm nerve	180		2	
	17	Forefinger pad D	300		2	
Hands and fingers	18	Forefinger pad ND	270	140	2	2
	19	Forefinger end joint D	280		2	
	20	Forefinger end joint ND	220		2	
	21	Thenar eminence	200		2	
	22	Palm D	260		2	
	23	Palm ND	260		2	
	24	Back of the hand D	200		2	
	25	Back of the hand ND	190		2	
	26	Thigh muscle	250	220	2	2
Thighs and knees	27	Kneecap	220		2	
Lower legs	28	Middle of shin	220	130	2	2
	29	Calf muscle	210		2	

Figure 1.9: Biomechanical limits of human body parts. Page 24 of [53]

Body region	Effective spring constant	Effective mass
	K N/mm	m_H kg
Skull and forehead	150	4,4
Face	75	4,4
Neck	50	1,2
Back and shoulders	35	40
Chest	25	40
Abdomen	10	40
Pelvis	25	40
Upper arms and elbow joints	30	3
Lower arms and wrist joints	40	2
Hands and fingers	75	0,6
Thighs and knees	50	75
Lower legs	60	75

NOTE Mass values for thighs, knees and lower legs are set to the full body weight, since these body parts are not free to recoil or retract from impact while the operator is standing.

Figure 1.10: Effective masses and spring constants for the body model. Page 27 of [53]

1.3 Contribution and Thesis Outline

Collaborative robotics opens new ways of interaction between robot and human. In order to design the collaboration with human and machine, collaborative or industrial robots can be integrated inside the manufacturing. In both cases, the sensorization and the perception of the environment play a fundamental rule. The support to the technical design and integration of the robot inside the robotic application is performed by the safety standards. In 1.2, the common standards ISO 10218-1, ISO 10218-2 and the specification ISO/TS 15066 has been described.

However, the robot works to support and assist the operator, it can relieve him/her of tedious, heavy and repetitive tasks, that are easy to be performed by the robot. In this way, the robot enhance the human operator skills and increase the ergonomics of the human work. Indeed, the operator, thanks to his adaptation and flexibility features, is irreplaceable.

The contribution of this thesis is to analyze three different use cases where a robot is integrated inside the operator workflow with the aims to improves markedly the human health. In addition to the description of the technical implementation of the human-robot interaction scenario, the safety aspects will be treated in detail.

Thanks to the technical specification ISO/TS 15066, the guidelines could regulate the collaborative situation significantly limiting performance of the robot. Indeed, the impact caused by a collision with the robot are correctly regulated through the PFL method. However, in this case the robot velocities and forces are limited considering the worst part of the body with which the robot can collide. Given this, a new way to optimize the performance while guarantee the safety of the human during a transient or quasi-static contact will be define.

Moreover, the technical specification introduces an alternative way of interaction between human and robot, namely the Hand Guiding. Usually, the HG method is regulated with the velocity and forces limitation of the robot. This translates into a system not optimized at all for industrial application, where high velocities are required. The Hand-Guiding interaction method allow to the operator to moves heavy parts, moving the physical effort almost everything on the robot and, then, improving the ergonomics of the operator work. However, in some cases, high-velocities during the HG interaction method are required by the industrial application. Thus, an high-velocity hand-guiding method will be proposed while operator safety is guaranteed. Thanks to the innovation of the particular way of robot programming with an industrial robot, just mentioned, two patent requests, [36] and [35] have been prepared and sent.

In common industrial operations and cases, human operators are constantly put under physical strain, especially for the overloading of the superior art of the human body, namely left and right arms. In this way, a robot can help the operator to relive the damages and, then, improve the human health. Indeed, in the last part of the thesis, an innovative robotic integration where robot have to pick and move the urban bin waste will be described. In this case human and robot coexists on the same system, the operator commands the missions to complete and the robot works to satisfy its. In order to guarantee the safety of the system and the people, while the robot moves, the operator must supervises the robot motion and the environment.

Chapter 2

A human-robot interaction scenario regulated by "Power and Force Limiting" while Maximizing robot performance

As long as productivity and flexibility are required from specific production processes, the integration of a robot to help the operator with his work is recommended. In this case, if the number and variability of pieces to product are not too excessive, the collaboration between human and robot allows to save money and improve the ergonomics of the operator works [96]. However, even if the human and robot activities are not computed in synergy, during their work it is possible that the two actors share the workspace. The ISO/TS 15066 proposes a PFL method to guarantee the human safety in a case like the one mentioned. Taking into account the worst part of the human body, the velocity and force of the robot are limited and, even if a contact between operator and robot happens, the energy transferred during the impact causes no harm. However, this method can restrict too much the robot performances even when a collision is not going to happen. For this reason, according to PFL method, in this section the problem to optimize the robot performances during space sharing between human and robot will be analyzed.

2.1 Introduction

In collaborative robotics, humans and robots interact in industrial scenarios without traditional safeguarding, under specific conditions. As a consequence of the spreading of human-robot collaboration technologies in the industry, great importance has been attributed to robot safety standards, which have been updated to address new co-working scenarios [32]. In particular, International ISO 10218-1 and ISO 10218-2 [50, 51] safety standards have identified specific applications and criteria where collaborative operations can occur. More recently, the technical specification ISO/TS 15066 [53] has been introduced to specify safety requirements for collabo-

rative industrial robot systems and the work environment and to supplement the requirements and guidance on collaborative industrial robot operations given in ISO 10218-1 and ISO 10218-2. The guidelines defined in the technical specification ISO/TS 15066 are detailed in 1.2.1.

In common scenarios, SSM is usually adopted, while PFL is used in tight collaborative operations setting a low collaborative robot speed. In the first case, the result is a frequently stopped robot, especially when dealing with small shared spaces, since collisions are not allowed. In the second case, instead, the result is a conservative system, where the worst collision case is always considered and the robot is moved very slowly.

The objective of this chapter is to provide a control method overcoming the limitations of both SSM and PFL operations, by embedding in an optimization-based controller the requirements of ISO/TS 15066 on safe human-robot contacts and by allowing the robot to move at higher speeds when the risk of such contacts is reduced, due to the human-robot separation distance. This objective will be obtained exploiting the theoretical framework of Control Barrier Functions [13], that provides a formal way to constrain the behavior of a controlled system into any properly specified safe set.

Related works on industrial human-robot collaborations have already addressed the definition of metrics and control methods based on separation distance and robot velocity with respect to a human operator [106]. In particular, [76] describes a trajectory generation algorithm for safe human-robot collaboration based on the solution of an optimization problem and the calculation of human predicted occupancy, a volume in which the human is expected to move within a given time horizon. However, the primary objective of these works is the avoidance of any human-robot collisions. The ISO/TS 15066 PFL requirements are instead explicitly addressed for robot motion planning or, better, re-planning in [108], however the properties of the re-planning method are not formally proved, and PFL requirements are also addressed in [100], whose aim is to define a software architecture that can abruptly switch (on demand) from SSM to PFL, rather than overcoming the limits of both modes. Finally, an analysis tool is presented in [65], whose objective is to analyze the robot dynamics from a task-dependent point of view, to evaluate the safety performance of the robot design. However, the proposed tool does not suggest any methodology for robot control design.

For these reasons, in this chapter we will face the definition of a control design framework based on the concepts of Zeroing Control Barrier Functions (ZCBFs, [13]), to enforce ISO/TS 15066 PFL velocity bounds, while obtaining faster operations when human-robot distance allows to relax the conservative constraint of the ISO technical specification. To the best of our knowledge, our proposal is the first application of ZCBFs as a design tool for safe human-robot collaboration compatibly with the requirements of ISO/TS 15066. In particular, the contributions of this chapter are:

- the real-time computation of the safe set of the system on the basis of the current distance and relative velocity between robot and human, embedding the ISO/TS 15066 PFL concepts in a non conservative way;

- the application of ZCBF-based optimization as a safety filter that can constrain the behavior of the robotic system in the mentioned safe set, whatever is the nominal and potentially unsafe controller adopted for robot trajectory tracking;
- the practical validation of the ZCBF-based control framework on an experimental setup, including a Universal Robot UR5 manipulator and depth camera system for human body tracking.

In Section 2.2 we will first specify the relevant ISO/TS 15066 PFL provisions, then in Sections 3.5 and 2.4 we will present the proposed control approach to address them. Finally, Section 2.5 will describe the practical validation and the results.

2.2 Power and Force Limiting according to ISO/TS 15066

In the technical specification ISO/TS 15066, the Power and Force Limiting (PFL) collaborative operation allows a physical contact between the robot system and an operator, either intentionally or unintentionally, under certain circumstances. In particular, the ISO/TS 15066 provides a guidance on how to assess safety of collaborative robots in PFL operations, suggesting calculations based on force limits on the basis of the pain sensitivity thresholds when contacts between the human and the robots occur. The maximum permissible force is based on the lowest energy transfer criteria that could result in a minor injury, such as a bruise, equivalent to a severity of 1 on the Abbreviated Injury Scale (AIS) established by the Association for the Advancement of Automotive Medicine¹. Adherence to the limits will prevent the occurrence of skin or soft tissue penetrations that are accompanied by bloody wounds, fractures or other skeletal damages. The force limits are used, in the Annex A of the ISO/TS 15066, to establish the related bounds that have to be set on the robot velocity for movements inside the collaborative workspace, depending on the transfer of energy that results from an hypothetical human-robot contact. In particular, the maximum permissible energy transfer can be calculated as a function of the maximum contact force value, assuming that the human body absorbs the energy of an impact as a linear spring, as in 1.5, where F_{max} and k are the maximum contact force and the stiffness constant of the body region interested by the hypothetical contact with the robot. Indeed, both the force limits and the stiffness constants vary depending on the different body areas that are involved and their values are detailed and tabulated in the Annex A of the ISO/TS 15066, for each body region.

Once the energy transfer limit value for the contact scenario is computed according to (1.5), it can be used to identify the maximum velocity at which the robot would be able to move through the collaborative workspace, while maintaining the force value below the threshold limit. Let us define v_{rel} as the relative velocity between the robot and the interested human body region. It is important to remark that in the ISO/TS 15066 and, therefore, in the rest of the chapter, by focusing on said relative velocity the analysis of robot behavior is reduced

¹<https://www.aaam.org/abbreviated-injury-scale-ais/>

to a single (translational) degree of freedom, namely the local axis connecting the human body part of interest to the closest part of the robot. In order to derive a relationship between v_{rel} and the contact force F , the ISO/TS 15066 assumes that, in case of fully inelastic contact, the total kinetic energy involved in the hypothetical impact is transferred to the human body area according to its stiffness. Thus, it results the following energy balance:

$$E = \frac{F^2}{2k} = \frac{1}{2}\mu v_{rel}^2 \quad (2.1)$$

where μ is the reduced mass of the two-body system (i.e. the system composed by the robot and the human body area), that can be computed as:

$$\mu = (m_h^{-1} + m_r^{-1})^{-1} \quad (2.2)$$

being m_h the effective mass of the human body area and m_r the effective mass of the robot. To compute the latter, the ISO/TS 15066 suggests the following simplified formula:

$$m_r = (M/2) + m_l \quad (2.3)$$

where M is the total mass of the moving parts of the robot and m_l is the effective payload of the robot system, including the tool and workpiece. Thus, from (1.5) and (2.1) it results that human safety in PFL operations enforces a constraint on the relative velocity between the robot and a given human body part. Recalling that such a relative velocity is positive when the separation distance is increasing (i.e. there is no risk of collision) and negative when the distance is decreasing, the constraint applies only in the latter case and can be stated as follows:

$$|v_{rel}| \leq v_{PFL} = \frac{F_{max}}{\sqrt{\mu k}} \quad (2.4)$$

where we have denoted with the term v_{PFL} the maximum permissible value on v_{rel} in PFL operations, according to the ISO/TS 15066.

Definition of an Adaptive Safe Operating Region

The objective of the proposed control system is enforce the safety constraints on human-robot PFL operations by guaranteeing that the robot behavior, described in terms of distance and relative velocity with respect to a selected human body area, is confined into an adaptive safe operating region. In particular, we aim to reduce the robot velocity along the direction of motion pointing to the human body, depending on the distance between the robot itself and such a human body, until the limit of v_{PFL} is reached and maintained. However, we do not want to enforce v_{PFL} all the time. Indeed, if the human-robot distance is sufficient to apply a deceleration along the robot to human direction and reach a relative velocity lower than v_{PFL} before the human-robot contact is really imminent, than the system would still be safe. In the following, we will first describe how to map these concepts into the definition of an adaptive safe operating region, while in the following section we will show how to apply geometrical considerations to formalize a Zeroing Control Barrier Function and a related optimization problem for the computation of safe robot velocities.

2.3 Definition of the safe operating region

As suggested by the ISO/TS 15066, the safety constraints on PFL can be analyzed considering a single translational degree of freedom: the distance between two points of interest on the human body and on the robot structure, respectively, and the relative velocity between the same points. Therefore, we will define the safe operating region in terms of the relationship between these two variables, that we will denote with d and v_{rel} respectively.

To embed the concepts of the ISO/TS 15066, we start by considering that when the human is far from the robot or human and robot are moving away from each other (i.e. the relative velocity is positive), the robot can safely move at a high velocity. Hence, the upper bound of the relative velocity can be defined as the maximum robot velocity in the Cartesian space, that we denote as v_{max} . This bound depends on the capabilities of the mechanical structure. When the human approaches and the relative velocity is negative, instead, the robot velocity has to be modified, so that its component along the direction of motion pointing towards the human satisfies the safety bound of (2.4) (i.e. v_{PFL}) before a contact could happen. This means that we have to analyze the deceleration capability of the robot, again in the direction of interest for the computation of the relative velocity. In general, assuming that the robot can constantly apply a maximum acceleration/deceleration of a_{max} in that direction of motion, the distance required to reduce the relative velocity from its current value to zero is:

$$d = \frac{1}{2} \frac{v_{rel}^2}{a_{max}} \quad (2.5)$$

It can be noted that the previous equation also describes a quadratic curve on a plane with d on the abscissa and v_{rel} on the ordinate. With this in mind, we can consider that a safety-related deceleration aiming to reduce the absolute value of the relative velocity from v_{max} to zero, that is a transition on the d/v_{rel} plane along the curve defined by (2.5), could end when a value of v_{PFL} is reached. Therefore, if we draw on the d/v_{rel} plane the quadratic curve (2.5), an horizontal line described by a constant v_{PFL} at any distance d and an horizontal line at constant v_{max} , we obtain a pictorial view of the safe operating region for ISO/TS 15066 compatible PFL operations. Such a view, focused on negative values of the relative velocity, which are those of interest for human safety, is given in Fig. 2.1 for a generic and abstract case. In the figure, the overall safe area is highlighted in yellow. The robot motion can result in a velocity towards the human up to v_{max} (red line) when the distance is large. If the distance becomes smaller than the one at which the curve at constant maximum deceleration (blue line) intersects the line at constant v_{max} , the upper bound on the relative velocity must be enforced by the blue line itself. Finally, if the distance is such that the v_{PFL} can be enforced without accelerating away from the human, the bound of (2.4) (i.e. the magenta line on the d/v_{rel} plane) is actually applied on v_{rel} .

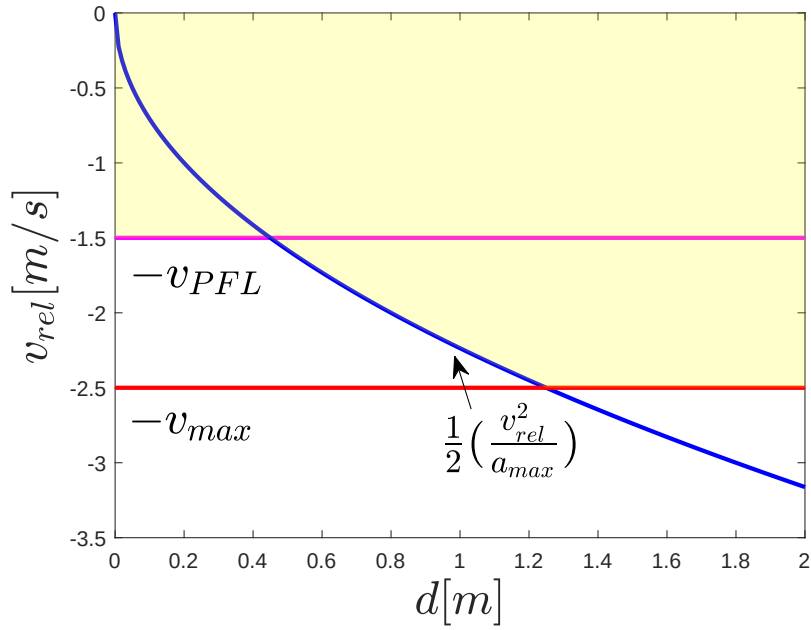


Figure 2.1: Definition of the safe operating region (yellow area). If the robot moves with a velocity that guarantees that the relative velocity with respect to the human lies inside the yellow area, the trajectory is safe.

Human-Robot distance and relative velocity evaluation

To precisely compute distance and relative velocity between a human and a robot, we will consider in the following the case of point-like objects, that is sufficient to describe the collisions between the robot end-effector and a specific part of the human body. Therefore, we will refer to Fig. 2.2 as the case of study and we will denote with $x_h \in \mathbb{R}^3$ the human body part position and with $x_e \in \mathbb{R}^3$ the position of the robot end-effector. The following computations could be extended, to achieve whole-body safety, to all the pairs composed by a robot link and a human link as proposed in [37].

Hence, we define the distance between the end-effector position and the human hand position and as:

$$d = \|x_e - x_h\| \quad (2.6)$$

and the relative velocity as:

$$v_{rel} = \dot{d} = \frac{(x_e - x_h)^T (v_e - v_h)}{\|x_e - x_h\|} \quad (2.7)$$

It is worth to note that, without loss of generality, we could also subtract from (2.6) the (constant) radii of two spheres, centered respectively in x_e and x_h , to have an additional margin on the safety constraint.

Considering again Fig. 2.2, we also compute acceleration and velocity limits for the robot with respect to the direction of motion between the human body part and the robot end-effector,

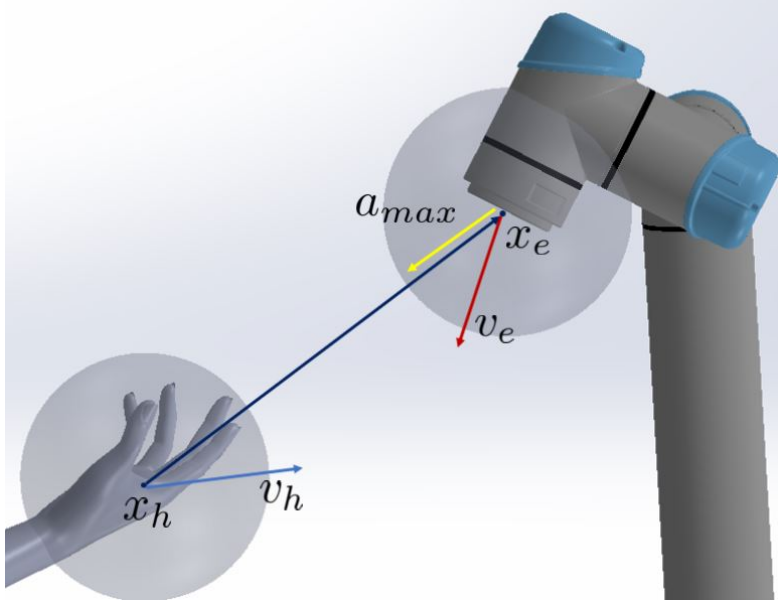


Figure 2.2: Definition of the end-effector and human body part positions and velocities.

which can be defined as the following unit vector:

$$u_s = \frac{(x_e - x_h)}{\|x_e - x_h\|} \quad (2.8)$$

We recall that the velocity and acceleration in the operational space of a robot with n degrees of freedom depend on its joint velocities and accelerations, $\dot{q} \in \mathbb{R}^n$ and $\ddot{q} \in \mathbb{R}^n$ respectively, as follows:

$$\dot{x}_e = J_t \dot{q} \quad (2.9)$$

$$\ddot{x}_e = \dot{J}_t \dot{q} + J_t \ddot{q} \quad (2.10)$$

where $J_t \in \mathbb{R}^{3 \times n}$ is the translational part of the robot's Jacobian. Then, we assume that the dynamic capabilities of the robot are expressed in joint space by symmetrical (i.e. minimum values are the opposite of the maximum ones) bounds, that we denote as $\dot{q}_{max} \in \mathbb{R}^n$ and $\ddot{q}_{max} \in \mathbb{R}^n$. As a consequence, robot velocity and acceleration bounds along the direction of u_s , corresponding to the values of v_{max} and a_{max} appearing in Fig. 2.1, can be computed as follows:

$$v_{max} = |u_s^T J_t| \dot{q}_{max} \quad (2.11)$$

$$a_{max} = u_s^T \dot{J}_t \dot{q} + |u_s^T J_t| \ddot{q}_{max} \quad (2.12)$$

where $|\cdot|$ is the element-wise absolute value of a vector.

It is important to remark that all of the previous results depend on the current robot configuration. Therefore, the safe operating region described in Fig. 2.1 is inherently time-varying and must be computed adaptively in real-time, according to the robot kinematics. Moreover, one should note that the minimum velocity value is actually $-v_{max}$, while the minimum acceleration is obtained by changing the sign of the second term in (2.12). Theoretically, even the value of a_{max} could result negative. However, this can hardly happen for collaborative robots, since

their joint velocities are generally limited. Indeed, we never detected this condition during our experiments.

2.4 A ZCBF-Based Framework for Human-Robot PFL Operations

To guarantee that robot velocity never exceeds the safety boundaries, we exploit Control Barrier Functions, in particular the Zeroing Control Barrier Functions (ZCBFs). In Section 2.4 we will provide the fundamentals of ZCBFs (for more details, the reader is addressed to [77]). Then, in Sections 2.4 and 2.4 we will exploit the ZCBFs to formalize a framework to enforce ISO/TS 15066 PFL velocity bounds.

Background on Zeroing Control Barrier Functions

Let us consider a control affine, nonlinear system as:

$$\dot{\chi} = f(\chi) + g(\chi)u \quad (2.13)$$

where $\chi \in \mathbb{R}^n$ is the system state and $u \in \mathcal{U} \subset \mathbb{R}^m$ is the control input, with $\mathcal{U}$ the set of admissible control values. Functions f and g are locally Lipschitz continuous. In our previous work [37], we exploited the Reciprocal Barrier Functions (RBF), where the barrier function was defined as a non-negative function that grows to infinity for states that approach to safety constraints, while it decreases moving away from them. However, unbounded functions values may be not desirable in practical applications, to avoid numerical issues. For this reason, in this chapter we replaced the RBF with the Zeroing Barrier Functions, where the barrier function vanishes on the set boundary.

Formally, a smooth function $h(\chi)$ is defined in order to find an appropriate Control Barrier Function (CBF) that constraints the system state to lay inside the safe region, i.e. $h(\chi) > 0$, whereas $h(\chi) \leq 0$ indicates a violation of the constraint. Thus, the safety region and its boundaries are defined by the set of admissible states $\mathcal{C}$:

$$\begin{aligned} \mathcal{C} &= \{\chi \in \mathbb{R}^n \mid h(\chi) > 0\}, \quad h : \mathbb{R}^n \rightarrow \mathbb{R} \\ \partial\mathcal{C} &= \{\chi \in \mathbb{R}^n \mid h(\chi) = 0\} \end{aligned} \quad (2.14)$$

Then, the goal becomes to design a proper control law u that allows to guarantee the forward invariance of the set $\mathcal{C}$ for all future times, i.e. if $\chi(0) \in \mathcal{C}$ then $\chi(t) \in \mathcal{C}$ for all $t \geq 0$. This objective can be reached exploiting the formulation of the CBFs and relating the system control input with the function $h(\chi)$, in order to enforce a proper constraint on the time derivative of the CBF. More precisely, a Zeroing Control Barrier Function is defined as follows.

Definition 2.4.1. ([13], Def. 5)

Given a set $\mathcal{C} \subset \mathbb{R}^n$ defined by (2.14) for a continuously differentiable function $h : \mathbb{R}^n \rightarrow \mathbb{R}$, the

function h is called a Zeroing Control Barrier Function defined on set $\mathcal{D}$ with $\mathcal{C} \subseteq \mathcal{D} \in \mathbb{R}^n$, if there exists an extended class $\mathcal{K}$ function ² α such that

$$\sup_{u \in \mathcal{U}} [\mathcal{L}_f h(\chi) + \mathcal{L}_g h(\chi)u + \alpha(h(\chi))] \geq 0, \quad \forall \chi \in \mathcal{D} \quad (2.15)$$

The ZCBF h is said to be locally Lipschitz continuous if α and the derivative of h are both locally Lipschitz continuous.

It is worth to note that the previous definition of ZCBF forces:

$$\dot{h}(\chi) \geq -\alpha(h(\chi)) \quad (2.16)$$

Once we define an appropriate ZCBF, a relationship with the control value u is derived (i.e. provided that $\mathcal{L}_g h(\chi) \neq 0$, i.e. the ZCBF has relative degree of one with respect to the control input, within the admissible set $\mathcal{C}$), yielding to the set of admissible control values $\mathcal{K}_h(\chi)$:

$$\mathcal{K}_h(\chi) = \{u \in \mathcal{U} \mid \mathcal{L}_f h(\chi) + \mathcal{L}_g h(\chi)u - \alpha(h(\chi)) \geq 0\} \quad (2.17)$$

By applying a control input $u \in \mathcal{K}_h(\chi)$, we guarantee that the set $\mathcal{C}$ is forward invariant.

Multiple barrier functions can also be composed with logical "AND/OR" operators as described in [103], introducing the concept of Piecewise Barrier Function (PBF) and relying on the definitions of piecewise smooth functions and of the so-called B-derivative to compute the Lie derivatives in (2.15).

Definition of CBF candidates

In Section 2.3 we defined a safe operating region in terms of a sub-region of the distance-relative velocity plane. The safe operating region can be expressed as a safe set within the ZCBF framework. In particular, we need to find a CBF candidate $h : \mathbb{R}^2 \rightarrow \mathbb{R}$, such that $h > 0$ inside the safe region. Then, we will plug the geometrical expressions of distance and relative velocity obtained in Section 2.3 into the CBF candidate, to obtain the constraint on robot control input that keeps the system inside the safe region by guaranteeing a safe robot velocity. It is worth to note that a mandatory theoretical assumption to guarantee safe robot operations (i.e. the forward invariance of the previously described safe set) would require that the robot motion belongs to the safe set in its initial state. In practice, satisfying this assumption is not particularly challenging (e.g. the robot generally starts its task from rest, when the human is outside of its workspace).

To mathematically describe the safe set and be able to define CBFs over it, we first define the system state χ in terms of distance and relative velocity, as follows:

$$\chi = \begin{bmatrix} d & v_{rel} \end{bmatrix}^T, \quad \dot{\chi} = \begin{bmatrix} v_{rel} & \dot{v}_{rel} \end{bmatrix}^T \quad (2.18)$$

²A continuous function $\alpha : [-b, a) \rightarrow [-\inf, \inf)$ is said to belong to extended class $\mathcal{K}$ for some $a, b > 0$ if it is strictly increasing and $\alpha(0) = 0$.

According to the definition of relative velocity in (2.7), its time derivative becomes:

$$\dot{v}_{rel} = \frac{(x_e - x_h)^T (\dot{v}_e - \dot{v}_h)}{\|x_e - x_h\|} + \frac{\|v_e - v_h\|^2}{\|x_e - x_h\|} - \frac{[(x_e - x_h)^T (v_e - v_h)]^2}{\|x_e - x_h\|^3} \quad (2.19)$$

Now, we can observe that if we assume that the control system is able to act on the robot acceleration as a control input, i.e. $u = \dot{v}_e = \ddot{x}_e$, the dynamics of the state χ can be easily rewritten in the general form of (2.13) (we omit this passage since it is not necessary in the following computations). Moreover, this assumption also implies that the specification of a CBF as a function of the state χ would already satisfy the requirement on the relative degree between such a function and the controllable input of the system.

To obtain a CBF with a tractable expression in terms of the state vector χ and satisfying the properties of (2.14), especially considering the particular shape of the safe set, we first approximate the safe region described in Section 2.3. In particular, we choose to cover the entire yellow area of Fig. 2.1 with an arbitrary number of intersecting ellipses. Ellipses are chosen because they can be also viewed as the base (i.e. the zero level) of a paraboloid. Then, paraboloids with an elliptic base are good candidates for ZCBFs: they are positive inside the ellipse and zero on the boundary; their quadratic form and the related derivatives can be easily manipulated in real-time (i.e. the safe region of Fig. 2.1 has to be adapted online, at each control cycle). Each ellipse is specified in the d/v_{rel} plane by the coordinates of its center and by its widths, so that its equation is the following:

$$\frac{(d - D_i)^2}{a_i^2} - \frac{(v_{rel} - V_i)^2}{b_i^2} = 1 \quad (2.20)$$

where i is the index of the ellipse, D_i and V_i are the coordinates of its center and a_i and b_i are its half-widths.

The ellipse-based approximation of the safe set is depicted in Fig. 2.3.

It is important to note that the number of ellipses used to compute this approximation is relevant. Of course, the highest is number of ellipses the more accurate is the approximation. On the other hand, the number should not be too large to affect the real-time adaptive computation of the safe set and the ZCBF-based constraints. Moreover, we should also recall that such ellipses necessarily cover both negative and positive values of the v_{rel} axis. For the reasons remarked in Section 2.3, only the negative ones are of interest. Theoretically, the upper bound on v_{rel} in the opposite quadrant of the one shown in Figs. 2.1 and 2.3 (i.e. human and robot moving away from each other) is irrelevant. However, a (large) constant value for this bound has to be defined, to properly compute the ellipse-based approximation of the safe set.

Now, the formulation of the final ZCBF is obtained. From each ellipse in the set approximating the safe region, we build a corresponding ZCBF as follows:

$$h_i(\chi) = \begin{cases} 1 - \frac{(d - D_i)^2}{a_i^2} - \frac{(v_{rel} - V_i)^2}{b_i^2} & \text{if } \chi \text{ inside } i\text{-th ellipse} \\ 0 & \text{otherwise} \end{cases}$$

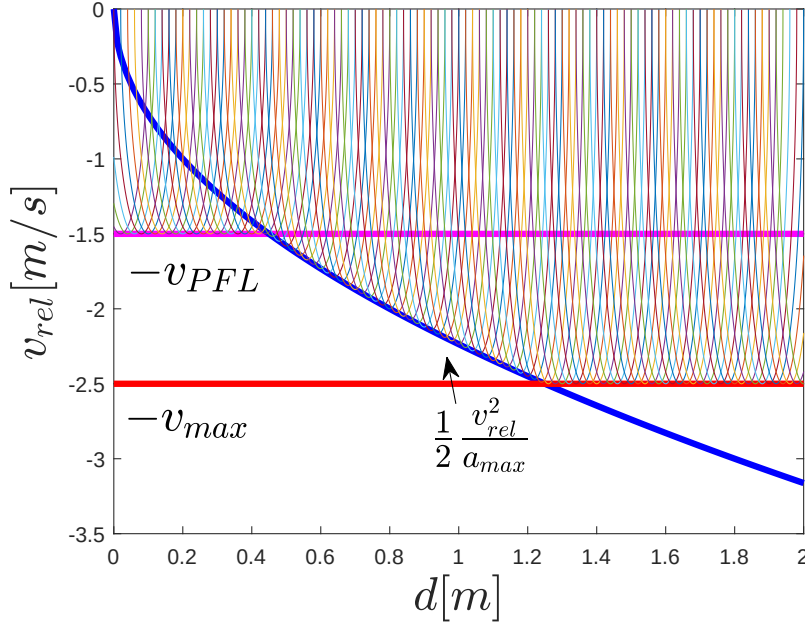


Figure 2.3: Approximation of the safe set through a set of ellipses.

where i , D_i , V_i , a_i and b_i are the same as in (2.20).

As a result, the 3D view of the paraboloids associated to safe set approximation of Fig. 2.1 can be depicted as shown in Fig. 2.4. Notice that in this plot both positive and negative values of v_{rel} are shown, being the latter those affected by the safety constraint (i.e. positive relative velocities are bounded by the v_{max} of the robot).

Each barrier function would guarantee that the control input keeps the system state inside of it. However, since the overall safe set is approximated by the intersection of the zero levels of paraboloid-shaped barrier functions, the overall ZCBF must be defined as the piecewise smooth logical "OR" of such functions, as described in [103]:

$$h(\chi) = \sum_{i=1}^n h_i(\chi) \quad (2.21)$$

where n is the total number of ellipses used in the approximation. We recall that the derivative of such a ZCBF is meant to be the B-derivative defined in [103]. However, in the following we will keep the standard notation of time derivatives, for the sake of simplicity.

To adopt the previous function as a ZCBF, we must be able to guarantee, by means of the control input design, that the constraint given by (2.16) is always satisfied. This means that we must compute the time derivative of the barrier function and enforce such a constraint on the control input computation. Considering the time derivative of the i -th paraboloid (if χ is inside the i -th ellipse):

$$\begin{aligned} \dot{h}_i(\chi) &= -2 \frac{(d-D_i) \dot{d}}{a_i^2} - 2 \frac{(v_{rel}-V_i) \dot{v}_{rel}}{b_i^2} \\ &= -2 \frac{(d-D_i) v_{rel}}{a_i^2} - 2 \frac{(v_{rel}-V_i) \dot{v}_{rel}}{b_i^2} \end{aligned} \quad (2.22)$$

and rewriting the time derivative in terms of robot/human position, velocity and acceleration,

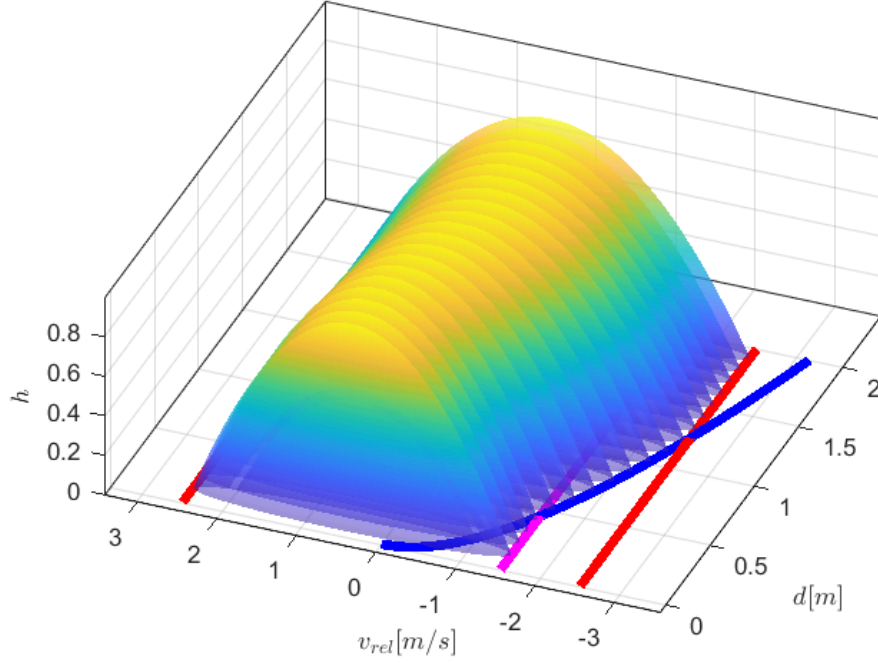


Figure 2.4: Paraboloid-shaped CBFs defined on the set of ellipses that approximate the safe set.

we obtain:

$$\begin{aligned} \dot{h}_i(\chi) = & -2 \frac{(d-D_i) v_{rel}}{a_i^2} - 2 \frac{(v_{rel}-V_i)}{b_i^2} \\ & \cdot \left[\frac{(x_e-x_h)^T (\dot{v}_e-\dot{v}_h)}{\|x_e-x_h\|} + \frac{\|v_e-v_h\|^2}{\|x_e-x_h\|} - \frac{[(x_e-x_h)^T (v_e-v_h)]^2}{\|x_e-x_h\|^3} \right] \end{aligned} \quad (2.23)$$

Since we consider as input the robot acceleration (i.e. $u = \dot{v}_e = \ddot{x}_e$), we can separate its contribution in (2.23) as $\dot{h}_i(\chi) = \beta_i(\chi) + \varphi_i(\chi)u$ with:

$$\begin{aligned} \beta_i(\chi) = & -2 \frac{(d-D_i) v_{rel}}{a_i^2} - 2 \frac{(v_{rel}-V_i)}{b_i^2} \\ & \cdot \left[-\frac{(x_e-x_h)^T \dot{v}_h}{\|x_e-x_h\|} + \frac{\|v_e-v_h\|^2}{\|x_e-x_h\|} - \frac{[(x_e-x_h)^T (v_e-v_h)]^2}{\|x_e-x_h\|^3} \right] \end{aligned} \quad (2.24)$$

$$\varphi_i(\chi) = -2 \frac{(v_{rel} - V_i)}{b_i^2} \frac{(x_e - x_h)^T}{\|x_e - x_h\|} \quad (2.25)$$

Finally, computing:

$$\varphi(\chi) = \sum_1^n \varphi_i(\chi) \quad \beta(\chi) = \sum_1^n \beta_i(\chi) \quad (2.26)$$

the set of admissible control values (2.17) for the robot can be stated as the one satisfying:

$$\varphi(\chi)u \geq -\beta(\chi) - \alpha(h(\chi)) \quad (2.27)$$

Recalling that $u = \ddot{x}_e$ and substituting the expression of the robot Cartesian acceleration in terms of joint velocities and accelerations previously recalled, i.e. (2.10), in (2.27), we can rewrite the

constraint that keeps the end-effector inside the safe set in terms of joint space variables, which will become useful for the practical implementation of the control law:

$$\varphi(\chi)J_t\ddot{q} \geq -\varphi(\chi)\dot{J}_t\dot{q} - \beta(\chi) - \alpha(h(\chi)) \quad (2.28)$$

In the next section, we finally describe how to obtain the safe control input at joint level, assuming that a nominal controller (e.g. PD for trajectory tracking) computes a desired, but potentially unsafe, robot acceleration $\ddot{x}_e^{des}$.

Optimization problem

The objective of the Control Barrier Functions is to act as a safety filter, by modifying the nominal control input only when safety is threatened. Hence, the resulting controller is minimally invasive: it allows to precisely follow the desired input when the system velocity is under the ISO/TS limit, while it modifies the behavior when the limit is not respected.

To achieve this behavior, the safety filter is deployed as an optimization problem described as:

$$\begin{aligned} \underset{\ddot{q}}{\text{minimize}} \quad & \|\ddot{x}_e^{des} - \dot{J}_t\dot{q} - J_t\ddot{q}\|^2 \\ \text{subject to} \quad & -\varphi(\chi)J_t\ddot{q} \leq \varphi(\chi)\dot{J}_t\dot{q} + \beta(\chi) + \alpha(h(\chi)) \\ & \|\ddot{q}\|_\infty < \bar{\ddot{q}} \\ & \|\dot{q} + \ddot{q} \cdot \Delta t\|_\infty < \bar{\dot{q}} \end{aligned} \quad (2.29)$$

where $\ddot{x}_e^{des}$ is the desired (or nominal) control input, $\ddot{q}$ is the commanded robot acceleration in the joint space and $\dot{q}$ is the actual joint velocity. To make the motion feasible, joint acceleration and velocity bounds $\bar{\ddot{q}}$ and $\bar{\dot{q}}$ are added to the constraints

2.5 Experiments

The validation of the proposed approach has been performed considering a Universal Robots UR5 manipulator, both for simulations and practical experiments. For the latter, the setup is completed by an Intel Realsense D415 depth camera and a human tracking software based on the NuiTrack library. The control software architecture is implemented using OROCOS and ROS frameworks and is similar to the one presented in [37]. The major differences are related to the CBF-based optimization problem, which is the novel contribution of this chapter. The optimization problem is solved online using C++ code generated by CVXGEN software.

In both simulations and experiments, the parameters in Table 2.1 were considered for the PD feedback controller (i.e. computing $\ddot{x}_e^{des}$), for the calculation of the ISO/TS 15066 PFL values and for the ZCBF-based optimization.

Table 2.1: Parameters setting for simulations and experiments

<i>PD Controller gain matrices:</i>	$\mathbf{K_P} = \text{diag} (60, 60, 60)$ $\mathbf{K_D} = \text{diag} (20, 20, 20)$
<i>ISO/TS 15066 PFL:</i>	
F_{max} (human arm):	60 <i>N</i>
k (human arm):	40000 <i>N/m</i>
m_h (human arm):	2 <i>Kg</i>
m_r (robot):	10 <i>Kg</i>
$\dot{q}_{max}$	(1.8, 1.8, 1.8, 1.8, 1.8, 1.8) <i>rad/s</i>
$\ddot{q}_{max}$	(40, 40, 40, 40, 40, 40) <i>rad/s</i> ²
<i>ZCBF-based Optimizer:</i>	
n (approximating ellipses)	100
$\bar{\dot{q}}$:	1.8 <i>rad/s</i>
$\bar{\ddot{q}}$:	40 <i>rad/s</i> ²
$\alpha(h(\chi))$:	$\gamma \cdot h(\chi)$

Simulations

The analysis of the results obtained in the Matlab/Simulink environment allowed to evaluate the effects of parameters tuning on the trajectories executed by the robot, in the ideal case of a pure kinematic system (i.e. it is assumed that joint velocity commands are instantaneously executed by the manipulator). Moreover, the trajectory of an obstacle emulating the human arm is deterministic and repeatable, while the real tracking of a human body by means of a depth camera is much more noisy and less reliable, even with state-of-the-art technologies. In particular, we focused on the influence of the parameter γ (in the function $\alpha(h(\chi)) = \gamma \cdot h(\chi)$ of our ZCBF) on the behavior of the system inside the safe set. Indeed, the ZCBF constraint expressed by (2.16) implies that with smaller values of γ the state of the controlled systems is kept at larger distances from the boundary of the safe set $\mathcal{C}$ and vice versa. In our case, staying away from the boundary of the safe set would also mean that the nominal trajectory, which is expected to achieve high velocities in absence human obstacles, would be executed quite differently after the ZCBF optimization. Conversely, larger values of γ would allow the robot to track more accurately the nominal trajectory, while avoiding impacts with the human body only if they are not safe from a PFL perspective.

The previous considerations are confirmed by the 3D plots shown in Fig. 2.5(a) and 2.5(b), obtained with values of $\gamma = 1$ and $\gamma = 15$, respectively. In both figures, the robot is drawn in its

initial position, while the obstacle in its final one, and arrows depict the direction of motion from start to end of the trajectories. As said before, the value of $\gamma = 1$ provides higher deviations from the nominal trajectory, while with $\gamma = 15$ the robot can almost reach the boundary of the safe set and, therefore, can touch the obstacle with a relative velocity close to v_{PFL} . For each simulation, the robot used is the kinematic model of UR5 collaborative robot, while the obstacle is represented by a sphere with radius of $0.15m$. Moreover, the results are processed for a set of data computed for $2.5s$ with time step of $0.006s$.

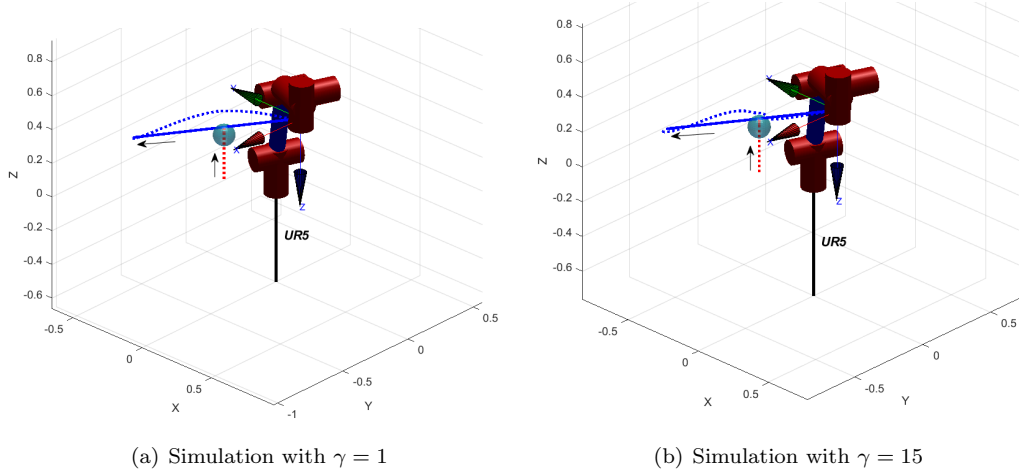


Figure 2.5: Robot trajectory (nominal in solid blue, executed in dashed blue) and obstacle trajectory (dashed red).

Experiments

Experiments with a real UR5 robot and a depth camera for human tracking reflected the features observed from simulation tests. The setup is shown in figure 2.6(a).

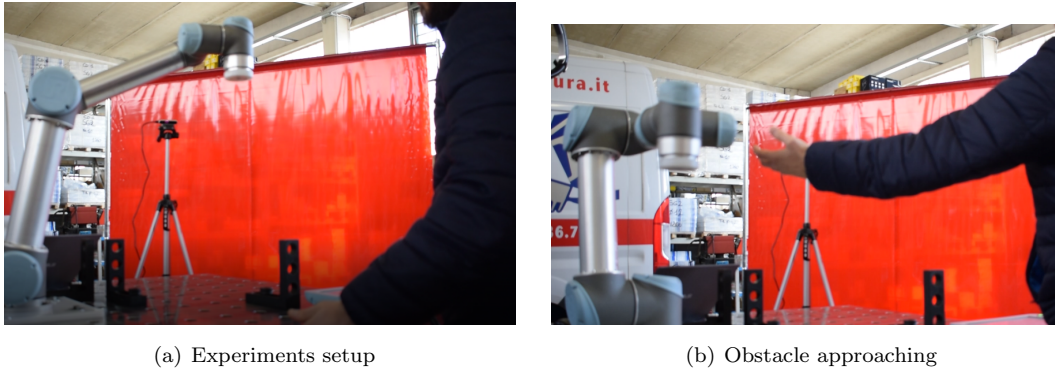
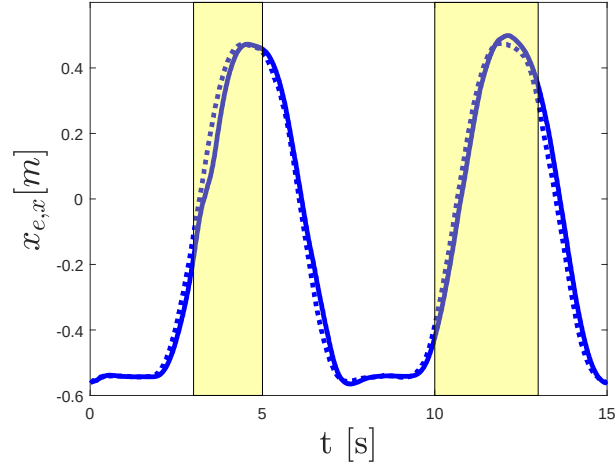
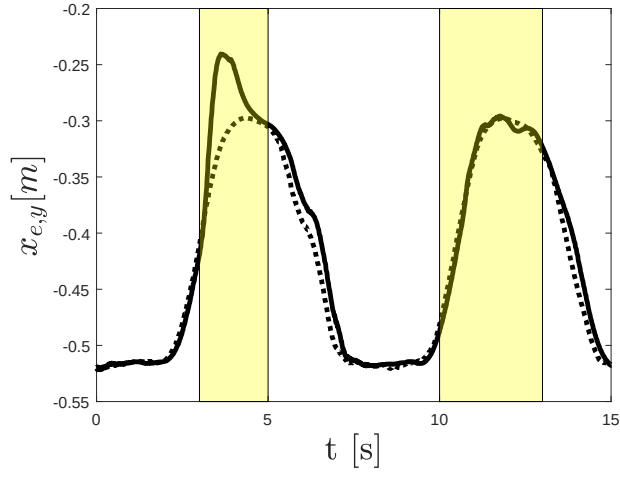


Figure 2.6: Experiments setup and obstacle approaching

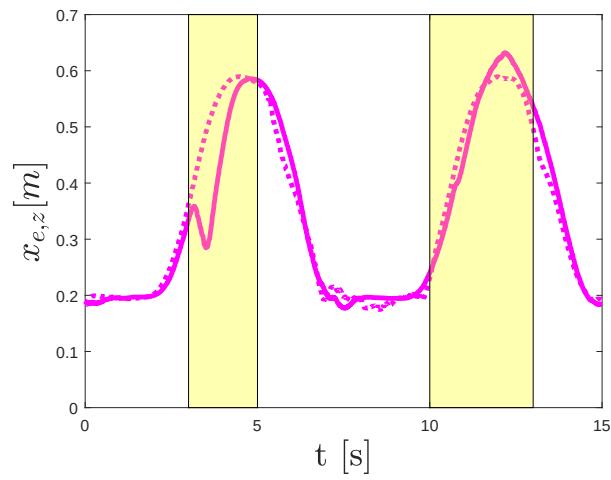
The accompanying video clip and figure 2.6 show the experiments described in the following, obtained with the $\gamma = 1$ setup. The key aspects that distinguish the practical application are related to the limited update frequency of the control system, mainly due to the frame rate of the camera system (i.e. 60 Hz), and to the joint velocity tracking performance of the low-level controller of the UR5 (i.e 125 Hz). However, the latter turned to be quite satisfactory, provided that the maximum joint acceleration prescribed by the manufacturer (i.e. 40 rad/s^2) is set.



(a) x-axis



(b) y-axis



(c) z-axis

Figure 2.7: Cartesian robot positions: nominal trajectory (dashed line) and ZCBF constrained trajectory (solid line).

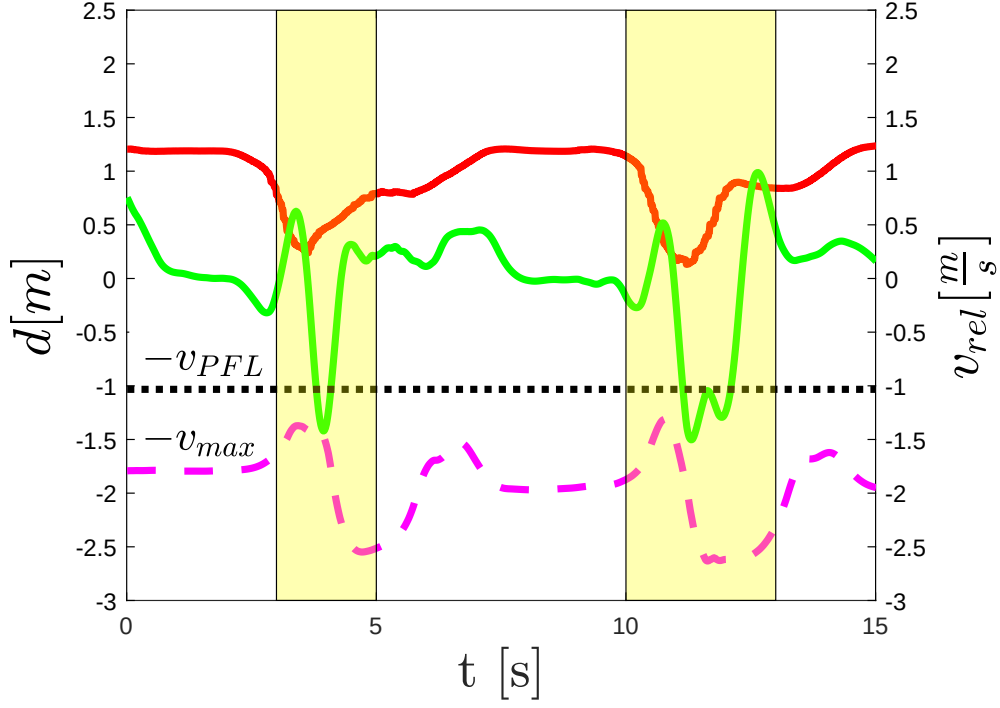


Figure 2.8: Distance (red solid line), relative velocity (green solid line), PFL relative velocity bound (black dotted line) and maximum robot velocity (magenta dashed line).

The plots in Fig. 2.7 show the robot end-effector position, in the three Cartesian coordinates, during two consecutive repetitions of a cyclic trajectory. The dashed lines show the positions tracked in absence of obstacles, while the solid lines are those obtained in presence of a human arm (like figured in 2.6(b)) along the robot trajectory. Yellow boxes highlight the time periods in which the human operator was closer to the robot. In the first cycle, it can be noted that the presence of the operator causes larger deviations from the nominal trajectory, compared to the second cycle.

The plot in Fig. 2.8 shows the distance between the robot end-effector and the human arm, their relative velocity, the v_{PFL} and the v_{max} bounds, being the latter adaptively computed on the basis of the robot configuration and, therefore, time-varying. From Fig. 2.8, it can be noted that the relative velocity changes more abruptly in the first cycle, which is due to the fact that the human arm acceleration, estimated by a Kalman filter as described in [37], was higher than in the second cycle. Since this acceleration directly affects the constraint of (2.27), this fact justifies also the previously mentioned higher deviation from the nominal trajectory. As can be seen in the accompanying video, the relative velocity returns to positive values mainly because the human retracts the hand. Finally, it is worth to note that the relative velocity overcomes the v_{PFL} during some transients, but in such transients the system is still confined in the safe set. On the other hand, a static graphical view of the latter property is difficult to obtain, since the safe set itself is time-varying.

2.6 Conclusion

In this chapter the ISO/TS 15066 is modeled as a dynamic safe set and a control barrier function approach is exploited for enforcing safety without unnecessarily limiting the velocity of the robot. The proposed approach has been successfully validated by simulations and by experiments. Future work aims at introducing human gestures prediction in the definition of the dynamic safe set and of the control barrier function in order to plan the best safe behavior for the collaborative cell. Moreover, the proposed ZCBF framework will be integrated with a Model Predictive Control (MPC) in order to consider the optimization problem in the long term.

Chapter 3

Optimization of the ”Hand-Guiding” interaction method with High-Velocities and an Industrial Robot

A robot can be seen as a support instrument, with the aim of making human work easier and more ergonomic than without. Several industrial processes, like painting, welding, quality control, or others, can be thought of as repetitive sequences of action where, precisely, the process is repeated after a first teaching phase. Try to imagine different production lots, where, for each lot, the operator must learn the work process for the first piece. If only the operator is authorized to complete the processes of the whole lots, the productivity would be lower than a machine and the experience gained in the process would be dedicated to only completing the manually work.

For this situation, the ISO/TS 15066 provides the Hand-guiding interaction method where the human could exploit the manually movements to teach a machine on how to correctly complete his work. Unfortunately, the standards severely limit the robot performances during the human interaction; the velocity is controlled at an appropriately reduced speed. Indeed, for many industrial applications, even during the teaching phase, the high velocity of the process is important to see a real result. Therefore, for these cases, the hand-guiding method would be too limiting.

In this chapter will be analyzed a new hand-guiding method to exploit the teaching phase of robot with high velocities, according to the safety standard which regulate the human-robot physical interaction.

3.1 Introduction

In the recent years the paradigm for robot usage has radically changed, moving from the idea of complete workspace segregation (achieved through physical barriers) to a scenario in which robots

and humans share the same workspace and even collaborate side by side [67], [27]. In this context, robots are becoming key elements in increasing manufacturing competitiveness, since continuous and fruitful physical human-robot interaction (pHRI) can definitely help small-to-medium sized companies achieve the higher production flexibility they need in order to cope with limited volumes and rapidly changing product requirements. Nevertheless, widespread adoption of robotic technologies in small-to-medium sized businesses is still undermined by certain well-known factors, among which the inherently complex and time-consuming nature of robot programming surely plays a crucial role [71].

Recently, methods based on virtual reality for programming a robot have been proposed [89], while traditional methods for robot programming typically consist in either using the robot teach pendant or in simulating the robot task inside an offline programming environment. In the first case, not only the operator needs to learn how to properly use the teach pendant, but the inherent point-to-point programming style is really efficient only for simple movements. In the second case, the knowledge of the platform-specific programming language (or alternatively of a 3D CAD program) is absolutely necessary, thus requiring the human operator to climb a steep learning curve.

In order to overcome these shortcomings, programming strategies usually defined as “walk-through programming” (also referenced as “lead-through programming”, or “manual guidance”) have been proposed [15], with practical applications ranging from spraying [41] to welding [16]. Examples of industrial applications where force control and manual guidance have been demonstrated can be found in [107], [68], [14], [81], [104]. These programming strategies are characterized by the fact that the human operator grabs the robot end-effector and manually leads it to the desired locations, without any prior knowledge of the specific robot programming language and of the functionalities offered by the specific teach pendant. During the teaching phase, the robot controller records either the via-points or the entire trajectory imposed by the human operator, so the manipulator will be able to replay the desired motion thereafter.

From the control point of view, the first applications of this programming paradigm were completely passive and relied on backdrivable actuators. Mechanical compensation was typically provided, by means of either hydraulic or pneumatic cylinders, in order to help the operator lift and move the robot. An example of this technology is represented by the Gaiotto GA-2000 manipulator [3]. More recently, active control strategies have been introduced, where the interaction forces exerted by the human operator on the robot are measured by a sensing system, whose output is used by the controller to enforce the desired motion.

Usually, the mentioned sensing system consists of a Force/Torque (F/T) sensor mounted on the robot wrist flange. Since the tool is typically attached after the F/T sensor, its dynamics must be compensated in order to directly estimate the forces and torques due to the interaction. Moreover, when using admittance-controlled robots like in the case of walk-through programming implementations, instability can arise when the user physically interacts with the robot.

In this chapter, we present a system which satisfies the following requirements:

- the system is safe for the user who interacts with the robot. Any kind of instability is detected and avoided, the tool dynamics is compensated and a safety assessment is performed.
- the system is hardware-independent regarding the F/T sensor and the robot vendor and allows the integration with a generic closed robotic controller.
- the system adapts its behaviour to any kind of user who wants to interact with the robot.
- an unexperienced user can program the robot without any prior robotics knowledge.
- the robot controller is able to record high-velocity continuous trajectories.

Related Works and Novel Contributions

Usually, walk-through programming architectures rely on two key elements: a sensor system and an admittance (or impedance) control algorithm [97]. The sensor system is responsible for measuring the interaction forces/torques exerted by the human operator on the manipulator. This goal can be achieved by either exploiting direct joint torque sensing (which is either not available or affected by poor data frequency and/or resolution), or by mounting a dedicated load-cell on the robot's end-effector. Obviously, the need of advanced sensing capabilities necessarily entails both cost and hardware robustness issues. Robots endowed with native torque sensing are usually characterized by high costs and limited payload and, even though the absence of sensors at joint level can be compensated with a single load-cell, these kind of sensors are typically affected by robustness and durability problems. Indeed, typically load-cells suffer of very common problems like zero temperature drift, field thermal drift, hysteresis and repeatability errors. In particular, regarding walk-through programming, among these errors, the temperature and thermal drifts are relevant and may cause the manual guidance to wrongly interpret forces/torques values [58]. In order to mitigate these issues, sensor-less approaches [75] based on model-based reconstruction of interaction forces and torques [24, 64] have also been investigated. Moreover, approaches to estimate the contact forces and torques based on the evaluation of motor current have been proposed and applied to force control [101]. That said, even though the practical feasibility of sensor-less walk-through programming has been demonstrated, to the best of authors knowledge applications of such techniques are still limited to robots characterized either by small size/inertia or moving at collaborative speed, even if some demonstrative cells started to appear in the opposite direction [9], [82]. A deeper evaluation of the performance of model-based interaction forces observers should be realized in order to check the feasibility of sensor-less walk-through programming approaches when applied to medium-to-high payload manipulators moving at high velocities. In this chapter, we consider the installation of a load-cell on the robot's end-effector. Since typically the tool to be used for the industrial task is attached on the wrist, directly after the load-cell, its dynamics need to be compensated in order to provide to the control loop the components due to the user interaction only. In this chapter, we exploited the results obtained in [58] and [87] to directly compute and eliminate the non-contact forces

due to the load dynamics and we included the tool compensation in the control architecture to obtain a walk-through programming technique, based on admittance control, that allows the use of the tool in the teaching phase, since its dynamics is compensated.

Moving to the admittance control algorithm, this element is responsible for translating the force/torque measurements into corresponding displacements in either joint Space (in case torque sensors are available on each joint) [45], Cartesian Space (in case a load-cell is used) [19], or both [10, 11]. Frequently, robots endowed with joint-level torque sensing are also provided with native impedance or admittance controllers. However, traditional manipulator controllers rarely offer such advanced functionality [*joint-level torque sensing*], thus requiring either some sort of interface between the admittance control algorithm and the trajectory interpolation layer or a completely “open” controller architecture [20] that allows to bypass the interpolation stage and send joint position (and possibly velocity) reference values directly to the axis low-level controllers. In principle, the possibility to directly feed the real-time low-level control loops with the admittance control algorithm output makes the open controller solution preferable, especially when walk-through is used to record continuous high-velocity trajectories. That said, this control approach is still not really widespread in the robotics industry and a few examples of open controller architectures exist at the moment.

Another issue when using admittance-controlled robots is that instability can arise when interacting with stiff environments. This can be due to several factors such as the robot structural dynamics, the limitations of the actuators, and the implementations of low-level controllers [28]. During the interaction, the user can behave in a stiff way and, thus, give rise to instability. Instability induces, among other undesired effects, a deviation of the robot from the desired admittance behaviour. Furthermore, it produces high-amplitude oscillations of the end-effector, undermining the user safety during the interaction. Thus, it is very important to promptly detect the rise of oscillations and then act to restore a stable and safe behaviour. An effective method for recovering stability consists of modifying the admittance dynamics parameters like, for example, increasing the damping [93] or adapting the parameters depending on the human dynamics behaviour [38]. An adaptive admittance control method for manual guidance has been proposed in [78], but the instability issues due to the parameters variation have not been addressed.

Given the aforementioned considerations and assuming that a sensor-based stable solution interfaced with a closed controller is needed in order to perform walk-through programming in actual industry scenarios, another critical issue that still needs to be addressed is the unavoidable trade-off between safety and performance. As a matter of fact, the increasing interest towards pHRI has led to new safety standards, which directly address collaboration scenarios. In particular, in safety standards [47, 48] specific criteria have been defined in order to identify when and where collaborative operations can occur, while in [49] specific safety requirements for collaborative industrial robotic systems have been introduced. In particular, the ISO/TS 15066 specifies restrictions regarding the transferred energy, the biomechanical limits and the robot velocity allowed during transient and quasi-static contact. Among these requirements, the most

relevant one with respect to the presented implementation of walk-through programming applied to industrial robots that need to execute continuous trajectories at very high velocities is the Cartesian velocity limit of 250 mm/s . Indeed, though this limitation may be suitable for scenarios in which only via-points need to be memorized, it may prevent walk-through programming to be used when continuous trajectories performed at high-velocity need to be recorded. For instance, spraying robots cannot be manually guided at low Cartesian velocities, since their motion needs to be synchronized with the spraying system set-points that cannot be kinematically scaled with respect to time.

In principle, solutions based on teleoperation [86, 88] could be considered as a safer alternative to walk-through programming, since they do not require pHRI. Nevertheless, the lack of direct interaction needs to be compensated by adding telepresence features such as haptic feedback [54] and remote vision [91], thus leading to significant equipment cost increases that can be justified only when working in highly dangerous contexts such as building demolition, decommissioning of nuclear power plants, disaster recovery, etc.

Finally, in [40] a walk-through programming architecture has been proposed, which is based on a Cartesian space admittance control algorithm interfaced with an open controller architecture. In addition, the proposed solution was provided with an additional algorithm (originally proposed in [87, 30]) that enables estimation and compensation of the dynamic effect caused by the presence of an eventual tool attached to the manipulator's end-effector. However, the proposed approach did not consider the destabilizing effects due to the interaction with a human operator. Moreover, in the robotics industry the possibility to rely on open controller architectures is not really widespread.

With respect to the previously discussed state-of-the-art, in this chapter a novel walk-through programming architecture will be introduced. In particular, this new approach differs from the aforementioned contributions since it allows to:

- achieve high velocities while guaranteeing the stability of the system and record high-velocity continuous trajectories;
- guarantee the human operator's safety;
- realize straightforward integration with a generic closed robotic controller;
- demonstrate the feasibility of the approach thanks to an extended experimental validation with professional employees using an industrial manipulator.

3.2 Problem statement

Several approaches exist to implement a reliable physical human-robot interaction, exploiting control strategies that make the robot compliant [97]. As introduced in Section 3.1, typically in the industrial scenario the walk-through-programming is obtained by implementing the admit-

tance control, since it allows to exploit the existing non-backdrivable manipulator's structure. The following main issues have to be considered and solved in order to obtain a reliable and safe industrial system:

- **Static and dynamic effects of the tool.** In several industrial applications, the teaching phase of walk-through programming has to be performed using the real tool in order to see the final result of the application (e.g. in a spraying task the amount of paint has to be directly monitored in order to verify the coverage of the whole surface). As a consequence, both the static and the dynamic effects of the load have to be compensated into the control loop. Indeed, the forces/torques that have to be tracked in the admittance control should embody only the forces and torques applied by the human operator when moving the robot end-effector to walk through the trajectory.
- **Instability issues.** When using admittance-controlled robots, instability can arise when interacting with stiff environments. Since the human behaviour dynamically changes characterized by a time-varying impedance, it can behave in a stiff way and thus produce instability effects. Possible undesired robot behaviors have to be detected and a strategy for restoring the desired stable behavior needs to be implemented.
- **Safety constraints.** Safety regulations provide constraints and requirements that need to be satisfied and may prevent walk-through programming to be used when continuous trajectories performed at high-velocities need to be recorded. However, in industrial applications such as spraying, the robot programming has to be performed at high-velocities without any kinematic scaling. A control architecture which satisfies the safety regulations but allows the execution of trajectories at high-velocities needs to be developed.

As a matter of fact, the proposed control architecture allows to program an industrial manipulator via walk-through at high velocities, while guaranteeing the operator's safety. Moreover, it allows straightforward integration with a generic closed robotic controller.

3.3 General Architecture

Walk-through programming consists of two different phases:

- *teaching phase:* the human operator physically guides the robot end-effector to teach the trajectory to be executed, while the robot controller records all the significant poses of the trajectory itself;
- *execution phase:* the robot plays the continuous trajectory back.

The overall scheme of the proposed control architecture is shown in Fig. 3.1. The robot is admittance-controlled, in order to allow a compliant behavior during the interaction. Since the teaching phase is executed by keeping the real tool (e.g. the spray gun) directly attached at

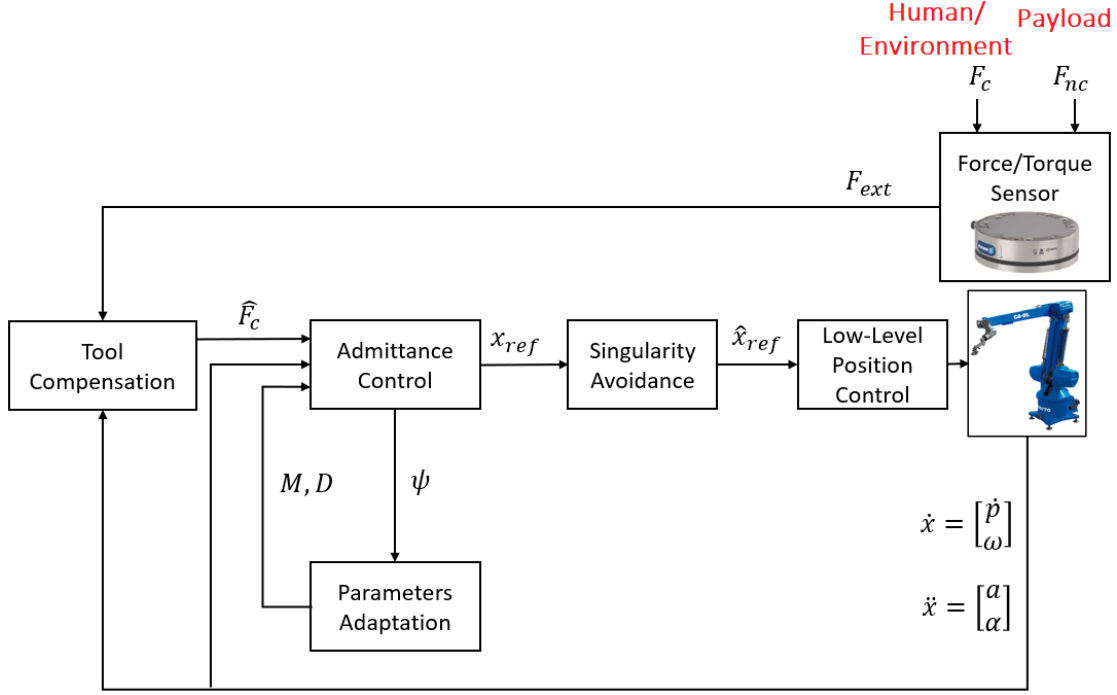


Figure 3.1: Overall control architecture.

the robot end-effector, we implemented a tool compensation algorithm to compensate the non-contact forces and torques F_{nc} applied by the dynamics of the tool and estimate the contact forces/torques $\hat{F}_c$ due to the interaction with the human operator from the raw measures of the F/T sensor (F_{ext}). Moreover, a low-pass filter is added to compensate external impact forces (≥ 5 Hz). The algorithm of tool compensation requires the knowledge of the robot linear acceleration a , angular acceleration α and angular velocity ω . If these data are not directly available, they can be estimated by using Kalman filters starting from the current pose of the robot Tool Center Point (TCP) x , as suggested in [30]. The admittance control, with input $\hat{F}_c$, provides the reference Cartesian pose x_{ref} which the position-controlled robot must follow. Since the robot interacts with the human operator, instabilities may arise in the control loop, depending on the dynamic behavior of the human operator. For this reason, we implemented a strategy for detecting rising oscillations in the robot's behavior and adapting the parameters of the admittance control to restore the desired stable behavior. The algorithm of parameters adaptation adapt the inertia M and damping D parameters of the admittance control depending on the detection of a rising oscillation by means of a detection index ψ , which is computed by the admittance controller itself.

The choice of a Cartesian-space reference pose is due to the fact that the vast majority of industrial robot controllers does not provide an open controller interface. This means that it is not possible to force the robot to follow a specific trajectory by streaming in real-time joint positions and velocity reference signals to its controller. Consequently, software architectures which generate a position reference command expressed in the Joint-space, as the one proposed in [40], could have limited application for the majority of industrial manipulators.

In order to allow straightforward integration with a generic closed robotic controller, in the control

architecture pictured in Fig. 3.1 the position reference commanded to the robot controller consists of a Cartesian-space displacement. As a matter of fact, several industrial robot controllers allow the user to lock the robot TCP position to a Dynamic Target Reference Frame (DTRF), that can be moved with respect to the world-base frame during the execution of a program. Consequently, in order to successfully deploy the proposed system on a standard industrial controller it is sufficient to:

1. create a Dynamic Target Reference Frame (DTRF);
2. set the initial pose of the DTRF to the robot TCP pose;
3. synchronize the pose of the robot TCP with the DTRF;
4. launch the Admittance Control block;
5. modify the pose of the DTRF according to the displacement computed by the Admittance Control block.

In this scenario, the robot controller (or more specifically the trajectory interpolation algorithm) will turn Cartesian-space reference poses into Joint-space reference angles and pass these references to the low-level position controller, that works independently on each joint. As sketched in Fig. 3.1, a Singularity Avoidance block is included in order to keep the robot from reaching singular configurations where an error would surely be raised either by the interpolation algorithm (due to Jacobian matrix singularity) or by the safety controller (due to joint velocity limits violation). More specifically, a typical solution to the singularity avoidance is represented by the DLS Jacobian originally introduced in [102].

As a result, the proposed architecture ensures that the robot follows the trajectory imposed by the human operator (through the F/T sensor and the Admittance Control) by exploiting only default controller functionalities and without the need of an open controller interface.

3.4 Functional Logic Design

This section presents the theoretical framework beyond the implementation of the tool compensation algorithm and the admittance control parameters adaptation strategy. As introduced in Sections 3.3, the proposed theoretical framework allows to solve the issues presented in Section 3.2 related to the rise of unstable behaviors and to the need of compensating the static and dynamic effects of the tool.

Consider the Euler-Lagrange formulation of the dynamic model of an n -DOFs manipulator, expressed in the task space:

$$\Lambda(x)\ddot{x}(t) + \Gamma(x, \dot{x})\dot{x}(t) + F_g(x) = F_\tau(t) + F_{ext}(t) \quad (3.1)$$

where $x(t) = f(q(t))$ is the TCP pose (i.e. position and orientation), obtained from the joint values $q(t) \in \mathbb{R}^m, m \geq n$, through the direct kinematic function $f(\cdot)$. $\Lambda(x) \in \mathbb{R}^{n \times n}$ is the positive

definite inertia matrix, $\Gamma(x, \dot{x}) \in \mathbb{R}^{n \times n}$ is the matrix describing the Coriolis and centrifugal effects and $F_g(x) \in \mathbb{R}^n$ is the gravity term. $F_\tau(t) \in \mathbb{R}^n$ is the torque due to the controlled joint torques $\tau \in \mathbb{R}^m$, while $F_{ext}(t) \in \mathbb{R}^n$ are the external forces and torques applied to the robot end-effector.

Let $x_d(t)$ be a desired configuration for the robot TCP and $\tilde{x}(t) = x(t) - x_d(t)$ be the pose error. The goal of the admittance control is to regulate the robot in such a way that the relationship between the motion of the robot $\tilde{x}(t)$ and the force applied by the environment $F_{ext}(t)$ is given by the following mass-spring-damper system:

$$M_d \ddot{\tilde{x}}(t) + D_d \dot{\tilde{x}}(t) + K_d \tilde{x}(t) = F_{ext}(t) \quad (3.2)$$

where M_d , D_d and K_d are, respectively, the desired inertia, damping and stiffness n -dimensional symmetric and positive definite matrices. The elastic force in model (3.2) attracts the reference pose towards the desired one. However, in case of walk-through programming, the robot is manually driven by the operator and thus the desired pose can be omitted. Thus, a mass-damper system can be implemented and the admittance control law becomes:

$$M_d \ddot{\tilde{x}}(t) + D_d \dot{\tilde{x}}(t) = F_{ext}(t) \quad (3.3)$$

As already introduced in Section 3.3, we will compensate the destabilizing effects due to the interaction with the human operator by adapting the parameters of the admittance control. As a consequence we will implement the following, time-varying, interaction model:

$$M(t) \ddot{\tilde{x}}(t) + D(t) \dot{\tilde{x}}(t) = F_{ext}(t) \quad (3.4)$$

where $M(t) \in \mathbb{R}^{n \times n}$ and $D(t) \in \mathbb{R}^{n \times n}$ are time-varying inertia and damping symmetric and positive-definite matrices such that $M(0) = M_d$ and $D(0) = D_d$.

Compensation of static and dynamics effects of the tool

The external force $F_{ext}(t)$ in (3.4) is assumed to be measured by a 6-DOF F/T sensor attached to the robot wrist flange. For ease of notation we hereafter omit the dependency of $F_{ext}(t)$ from t . Typically, in manual guidance and walk-through programming the handle for moving the robot and the tool for performing the task are attached to the robot end-effector, directly after the F/T sensor. As a consequence, they exert a non-contact effect on the sensor, with both static (i.e. gravity) and dynamics (i.e. inertial, centrifugal/Coriolis) terms. Thus, the external force measured by the F/T sensor contains both the components related to the non-contact forces and torques F_{nc} due to the load dynamics, and the components given by the contact forces and torques F_c arising from the interaction between the robot and the human operator or the environment.

$$F_{ext} = F_{nc} + F_c \quad (3.5)$$

Since in the admittance model (3.4) the term F_{ext} is the force that has to be tracked, it should embody only the force applied by the human when moving the robot through the trajectory. Thus, the non-contact forces and torques must be computed and eliminated. To this aim,

following the preliminary results presented in [40], we compute the non-contact forces and torques $F_{nc} = [f_{nc}, \tau_{nc}]$ by exploiting the Newton-Euler formulation and the motion of a rigid body due to external forces and torques:

$$\begin{aligned} f_{nc} &= m a - m g + \alpha \times m c + \omega \times (\omega \times m c) \\ \tau_{nc} &= I \alpha + \omega \times I \omega + m c \times a - m c \times g \end{aligned} \quad (3.6)$$

where $m \in \mathbb{R}$ is the mass of the tool, $c = [c_x, c_y, c_z]$ is its center of mass, $I \in \mathbb{R}^{3 \times 3}$ is the inertia tensor, $g = [g_x, g_y, g_z]$ is the gravity vector, $a \in \mathbb{R}^3$ is the linear acceleration, $\omega \in \mathbb{R}^3$ and $\alpha \in \mathbb{R}^3$ are the angular velocity and acceleration, respectively.

Equations (3.6) can be grouped and rewritten as follows:

$$\begin{bmatrix} f_{nc} \\ \tau_{nc} \end{bmatrix} = V(a, \alpha, \omega, g) \varphi \quad (3.7)$$

in order to show the dependency of the non-contact forces and torques from the inertial vector φ which can be defined as:

$$\varphi = [m, mc_x, mc_y, mc_z, I_{xx}, I_{xy}, I_{xz}, I_{yy}, I_{yz}, I_{zz}]^T \quad (3.8)$$

The matrix $V(a, \alpha, \omega, g)$ is a 6×10 matrix which elements depend on the linear velocity, linear and angular acceleration and the gravity vector. The explicit definition of $V(a, \alpha, \omega, g)$ can be found in [87].

Typically, strain gage-based devices present a force drift which is a slowly time-varying and typically temperature-dependent contribution that may deteriorate the compensation results [98]. Since the matrix $V(a, \alpha, \omega, g)$ does not consider the thermal drift, we directly include the compensation of this term into the compensation equations. In particular, the F/T sensor is zeroed when the tool is attached and in an initial static condition with a known orientation. Then, a pseudo-gravitational term $F_{g_{init}}$, defined as follows, is added to subsequent readings in order to eliminate the gravitational forces and the associated torques.

$$F_{g_{init}} = V(0, 0, 0, g_{init}) \varphi \quad (3.9)$$

The offset zeroing operation is repeated periodically during the task pauses, to cope with temperature variations and performance modifications. By including this term, the computation of the contact forces and torques can be rewritten as follows:

$$\hat{F}_c = F_{ext} - \hat{F}_{nc} + F_{g_{init}} \quad (3.10)$$

where $\hat{F}_c$ and $\hat{F}_{nc}$ are the estimate of the related forces and torques, since a perfect knowledge of mass and inertia parameters cannot be achieved. However, the uncertainties included in these quantities can be compensated by applying threshold-based logics, as shown in [87].

Admittance parameters adaptation for stable physical human-robot interaction

The contact forces and torques $\hat{F}_c$ computed in (3.10) are the forces and torques that have to be tracked in the admittance model (3.4), since they are due only to the interaction of the human with the robot when moving it through the trajectory. Thus, during the interaction, the robot is coupled with the human operator, whose dynamics (e.g. change of compliance of the arm) can cause deviations from the desired behavior. These deviations may produce robot oscillating motions of high amplitude and frequency, making the interaction unsafe for the user [25]. In order to detect the rising oscillations and then recover the desired behavior we exploited the results presented in [38], where a strategy for adapting the parameters of the admittance control for a stable physical human-robot interaction is proposed. In particular, in order to online detect the rise of an oscillatory behavior during the manual guidance, we considered the following heuristic:

$$\psi\left(\dot{\hat{x}}(t), \ddot{\hat{x}}(t)\right) = \|\ddot{\hat{x}}(t) + R_d(t)\dot{\hat{x}}(t)\| \leq \varepsilon \quad (3.11)$$

where $\varepsilon \in \mathbb{R}^+$ is a detection threshold that has to be appropriately tuned, while the vectors $\dot{\hat{x}}(t) \in \mathbb{R}^n$ and $\ddot{\hat{x}}(t) \in \mathbb{R}^n$ are the tracking error derivatives scaled with respect to bounds on the robot velocity and acceleration. The term $R_d(t)$ is the damping-to-inertia ratio matrix:

$$R_d(t) = M^{-1}(t)D(t) \quad (3.12)$$

Namely, when (3.11) is not satisfied, the robot is considered to be deviating from the interaction model imposed by the admittance control and oscillations are increasing. With a slight abuse of notation, we hereafter use $\psi(t)$ to indicate the value of $\psi\left(\dot{\hat{x}}(t), \ddot{\hat{x}}(t)\right)$ at time t .

The detection index $\psi(t)$ has been characterized in [38] as a log-normal distribution and this allows to accurately tune the detection threshold ε in (3.11). For details related to the procedure for tuning the detection threshold the reader is addressed to [38].

Once increasing oscillations have been detected according to the heuristic (3.11), the parameters of the admittance control are adapted to recover a stable behavior of the human-robot interaction. In particular, the inertia and damping parameters are increased when required due to the detection of increasing oscillations, but the desired interaction model is restored when such oscillations disappear due, e.g., to a relaxation of the operator's arm.

In the following, the equations for adapting the parameters of the admittance control will be provided in the discrete-time domain since they have to be implemented on a real robotic system. Thus, we will consider a set of time intervals $[t_i, t_{i+1}]$, with $i = 1, 2, \dots$, within which the parameter adaptation takes place.

According to the passivity and stability proofs demonstrated in [38], each component of the desired inertia matrix can be passively increased as follows

$$m_j(t_{i+1}) - m_j(t_i) = \frac{2\alpha_j(T(t_i) - \delta)}{\dot{\mathcal{X}}_j^2} \quad \forall j = 1, \dots, n \quad (3.13)$$

where $T(t_i)$ is the energy stored into a virtual energy reservoir called energy tank. In particular, the energy dissipated by the system is stored into the tank and can be reused for implementing

any desired control action in a passivity-preserving way (see [38] for further details). The value δ is a lower bound for the energy level in order not to empty the tank, while α_j are the components of a vector of weights used to distribute the energy available in the tank on all the DOFs of the robot motion. Finally, the terms $\bar{\mathcal{X}}_j$ are component-wise upper bounds on the robot velocity limits.

In practical cases, the value of the maximum allowed inertia variation computed in (3.13) could bring to an excessively large inertia variation. For this reason, we define an upper-bound $\Delta M = \text{diag}\{\bar{m}_1, \dots, \bar{m}_n\}$ on the allowed inertia variation as follows

$$m_j(t_{i+1}) - m_j(t_i) \leq \bar{m}_j \quad \forall j = 1, \dots, n \quad (3.14)$$

and we compute the following inertia increase:

$$s_j = \min \left\{ \frac{2\alpha_j(T(t_i) - \delta)}{\bar{\mathcal{X}}_j^2}, \bar{m}_j \right\} \quad \forall j = 1, \dots, n \quad (3.15)$$

The single components computed in (3.15) are exploited to fill the matrix of inertia variation as follows:

$$S_i = \text{diag}\{s_1, \dots, s_n\} \quad \forall i = 1, 2, \dots \quad (3.16)$$

Finally, the inertia variation is computed as

$$M(t_{i+1}) = M(0) + \sum_{p=0}^k S_p \beta^{(t_{i+1} - (\tau_p + \Delta t))} \quad (3.17)$$

where β ($0 < \beta \leq 1$) is a forgetting factor that allows to gradually restore the desired interaction model (3.3). Indeed, the presence of the forgetting factor β in the second term in the right-hand side of (3.17) makes the effect of each inertia increase negligible after a certain amount of time. In particular, this time is larger for higher values of β . Finally, the damping is updated preserving a constant damping-to-inertia ratio $R_d(t) = R_d(0)$, since this is a recommended practice in physical human-robot interaction [60]:

$$D(t_{i+1}) = R_d(0)M(t_{i+1}) \quad (3.18)$$

3.5 Safety Logic Design

Walk-through programming is without a doubt one of the clearest examples of physical human-robot interaction. With the introduction of human-robot collaboration technologies, an even greater relevance is attributed to robot safety standards, which have been updated to address co-working scenarios as walk-through programming. In particular, industrial settings need to comply with strict safety requirements, given by the standards ISO 10218-1 [47] and ISO 10218-2 [48] and the most recent technical specification ISO/TS 15066 [49].

This work focuses on the development of a strategy that enables human operators to record continuous trajectories performed at high velocity both at the joint and at the Cartesian space level. Consequently, the fact that mechanical hazards are the most significant ones to be taken into account from the safety point of view comes at no surprise. More specifically, among the

various potential consequences of the several mechanical hazards listed by [47], we took into consideration two main scenarios: human-robot impact, and entanglement/trapping of a generic operator's body part.

In order to obtain an effective trade-off between safety and productivity, in this chapter we propose to design a safety controller that works in parallel with respect to the previously described functional logic and that manages the safety in the teaching phase by following a two-fold approach:

- **Basic Safety Functions:** upper bounds on both joint and Cartesian space dynamic quantities are established and the safety controller checks if these bounds are overcome during the teaching phase;
- **Dynamic Human-Robot Monitoring:** the safety controller monitors in real-time the relative distance and the relative velocity of the robot with respect to the human operator. Threshold values are established in order to identify situations characterized by a high risk of impact and/or entanglement.

In both cases the safety controller checks a logic condition. If both conditions are true, the programming phase can proceed, while if at least one condition is evaluated as false, an emergency stop is issued by the safety controller and the manipulator is stopped.

Basic Safety Functions

As already mentioned, basic safety functions can be implemented in order to force an emergency stop whenever a specific dynamic quantity (at both joint or Cartesian space level) exceeds a prescribed threshold value. In this work three different kind of basic safety functions have been considered. Each one corresponds to check dynamic quantities against maximum allowed positive values, defined as follows:

- maximum allowed Cartesian Linear Speed at TCP $\|\dot{x}_p^{UB}\| \in \mathbb{R}$, where x_p and $\dot{x}_p$ correspond to the positional portion of x and $\dot{x}$, respectively;
- maximum allowed Joint Velocities $\dot{q}^{UB} \in \mathbb{R}^m$;
- maximum allowed Joint Accelerations $\ddot{q}^{UB} \in \mathbb{R}^m$.

As far as velocities are concerned, the TCP linear speed threshold depends on the specific task and it corresponds to the maximum speed needed to correctly record the continuous trajectory that will be directly converted into a robot program.

Moving to joint velocity bounds, suitable threshold values can be established by taking into account a reference configuration of the manipulator $\bar{q} \in \mathbb{R}^m$ and by computing, for each joint, the angular velocity $\dot{q}_k^{UB}$, with $k = 1, \dots, m$, that would result in the maximum allowed TCP

linear speed $\dot{x}_p^{UB}$ (which is always positive) via multiplication by the geometric Jacobian matrix $J(\cdot)$.

$$\dot{q}_k^{UB} \leftarrow \min \left(\dot{q}_k^{Max}, \frac{\dot{x}_p^{UB}}{\sqrt{\sum_{w=1}^3 (J(\bar{q}))_{w,k}^2}} \right) \quad (3.19)$$

Saturation with respect to absolute joint velocity limits ($\dot{q}_k^{Max}$) must be taken into account to guarantee that the safety check is effectively enforced. In this way, this safety function acts as a redundant check with respect to TCP speed monitoring.

Finally, once joint velocity bounds have been defined, the corresponding accelerations bounds can be computed on the basis of the robot stopping time Δt . Once again, saturation with respect to absolute joint acceleration limits ($\ddot{q}_k^{Max}$) must be taken into account to guarantee that the safety check is effectively enforced.

$$\ddot{q}_k^{UB} \leftarrow \min \left(\ddot{q}_k^{Max}, \frac{\dot{q}_k^{UB}}{\Delta t} \right) \quad (3.20)$$

For the sake of completeness, joint velocities and accelerations are checked against the discussed maximum values by means of their absolute value, so that either positive and negative values are properly checked. Consequently, the comprehensive logic condition computed by the basic safety functions can be expressed as follows:

$$\begin{aligned} \|\dot{x}_p\| &\leq \|\dot{x}_p^{UB}\| && \wedge \\ |\dot{q}_k| &\leq \dot{q}_k^{UB} \quad \forall k = 1, \dots, m && \wedge \\ |\ddot{q}_k| &\leq \ddot{q}_k^{UB} \quad \forall k = 1, \dots, m \end{aligned} \quad (3.21)$$

Dynamic Human-Robot Monitoring

As introduced before, in addition to the basic safety functions, the proposed safety controller is endowed with the capability to monitor in real-time the relative distance and the relative velocity of the robot with respect to the human operator. This way, the controller can identify situations characterized by a high risk of impact/clamping and issue an emergency stop accordingly.

In order to perform these functionalities, a strategy to model the space occupancy of both the operator and the robot is needed. To this regard, previous contributions in the field of safe human-robot interaction [74, 76] have already explored the application of capsule-based geometry models to the problem of modelling the space occupancy of complex objects, proving that this approach can be extremely efficient in approximating structured geometries. For the sake of clarity, a capsule consists in the convex hull of a sphere of given radius (also named “clearance”), which is translated along a segment. For instance, Fig. 3.2 shows two distinct capsules. The first one on the left is defined by points $P_{1,0}$, $P_{1,1}$ and radius r_1 , while the second one on the right is defined by $P_{2,0}$, $P_{2,1}$ and radius r_2 . As exemplified in the figure, the minimum distance

between the two capsules d can be obtained by computing the minimum distance d_1 between the segments $\overline{P_{1,0}P_{1,1}}$ and $\overline{P_{2,0}P_{2,1}}$ (as thoroughly explained in [29]), and then subtracting the two clearances r_1 and r_2 . Clearly, whenever $d \leq 0$, the capsules are colliding.

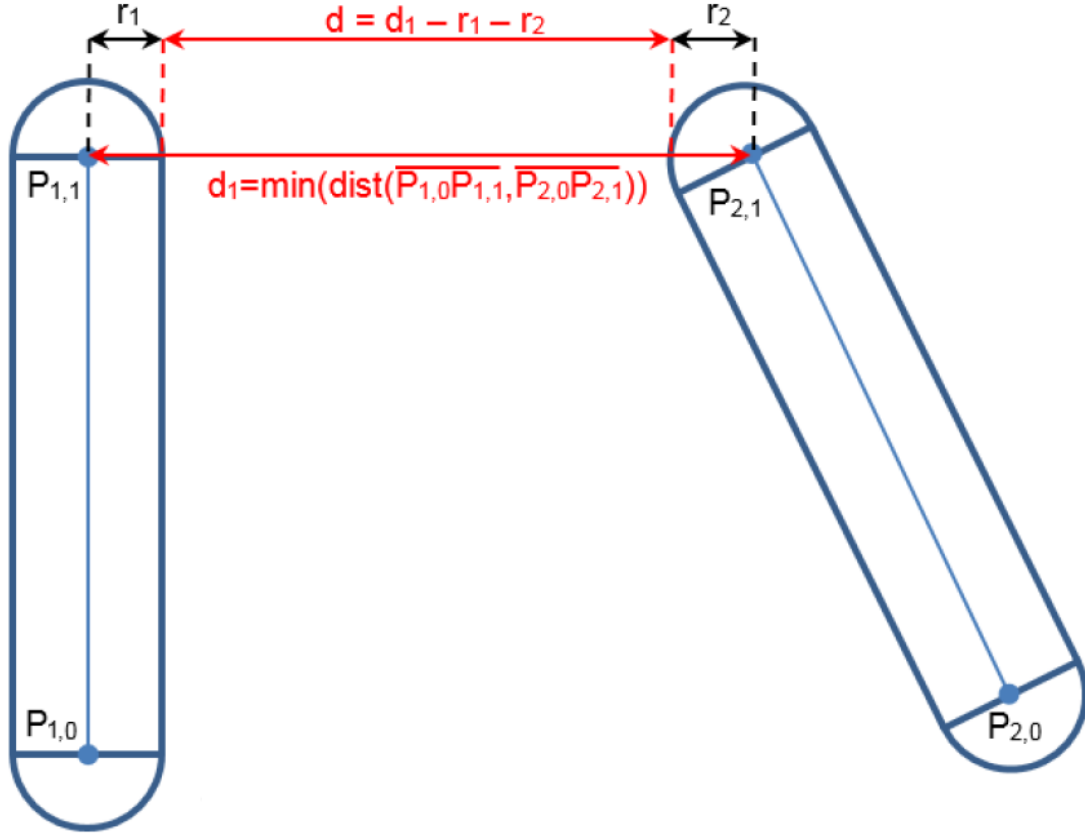


Figure 3.2: Geometry representation of two separate capsules and of the minimum distance between them.

In this work, capsules are used to model the space occupancy of the robot and of the human operator, as it is shown in Fig. 3.3. As far as the robot is concerned, a separate capsule is defined for each link and clearances are assigned on the basis of the links' geometry. Moving to the human operator, his/her space occupancy can be modelled by means of a single vertically oriented capsule, whose radius and height can be parametrized according to the anthropometric features of the specific operator.

To this regard, Fig. 3.3 depicts two distinct scenarios. In the first one the operator's capsule and the robot link capsules coexist without violating the safety checks, while in the second one a clear violation of the minimum distance check is displayed.

In addition, the operator's capsule is rigidly attached to the robot's end-effector according to the geometry of the handle that allows he/she to move the robot. For the sake of clarity, Fig. 3.4 shows the position of the operator capsule with respect to the robot's end-effector (dashed blue) and to the robot's base (dashed black) in three different configurations.

In principle, a more complex model could be taken into account for the human operator, by considering all the pairs composed by a robot link and a human link that can be computed dynamically as proposed in [39]. To this aim a tracking system that allows to detect and monitor

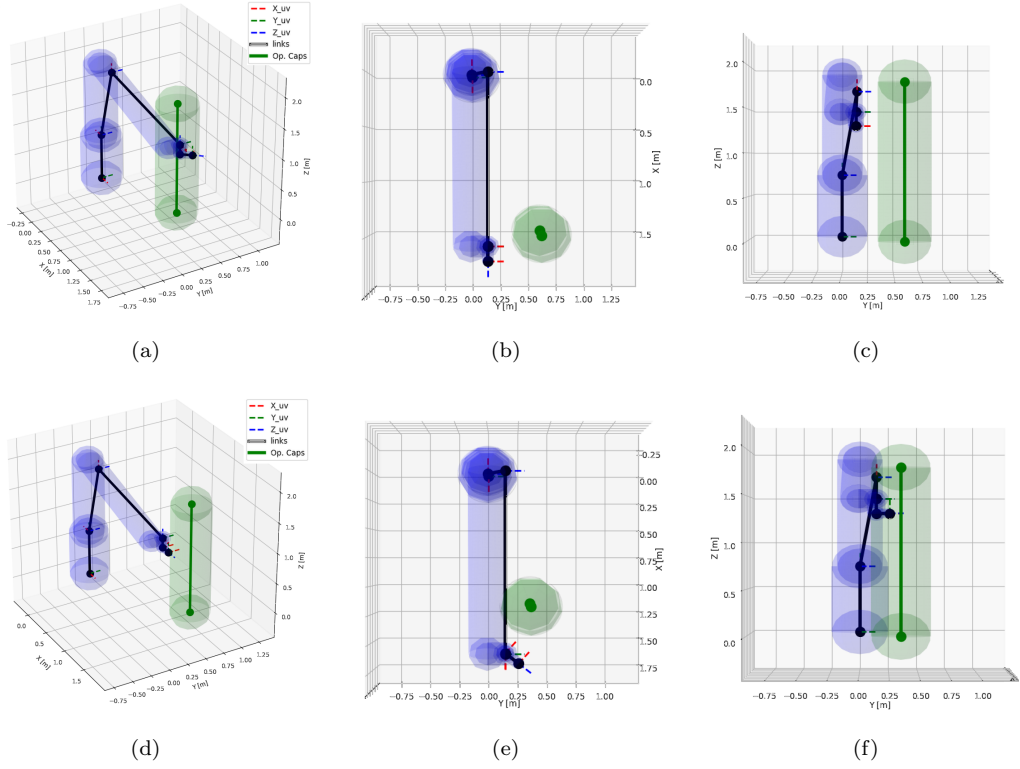


Figure 3.3: Example of operator’s capsule and capsule-based geometry model of a 6 DoF offset wrist manipulator. In 3.3(a) - 3.3(c) the minimum distance condition is respected, while in 3.3(d) - 3.3(f) it is violated.

the current position of each link of the human operator is required to be integrated, by following the strategies proposed in [74],[76] and [39]. However, the main reason behind our choice lies in the fact that we are interested in preventing collisions by ensuring that there is a sufficient separation distance between the robot and the operator, rather than precisely identifying the operator’s body part involved in a collision. In addition, given the nature of the task, some body parts (i.e. hands, wrists, and possibly forearms) would always result in collision (or at least in close proximity) with the robot, thus making this level of detail not necessary.

During the execution of the teaching phase, the safety controller updates in real-time the position of the capsules, computes the relative distance between each capsule and the operator’s one (like it is shown in Fig. 3.2), and checks that the minimum relative distance stays above a parameter threshold. By setting this threshold equal to the distance that the robot can cover at the maximum allowed linear speed $\dot{x}_p^{UB}$ during the time needed to enforce an emergency stop, the safety controller is able to prevent collisions between the robot and the operator. A pseudo-code formalization of this procedure is given in Algorithm 1, where:

- *minDistTh* represents the minimum allowed relative distance between the robot capsules and the operator’s one;
- *robRad* is a vector containing the parametric radii of the robot capsules;
- *opRelP* corresponds to the relative pose between the operator’s capsule and the robot’s end-effector;

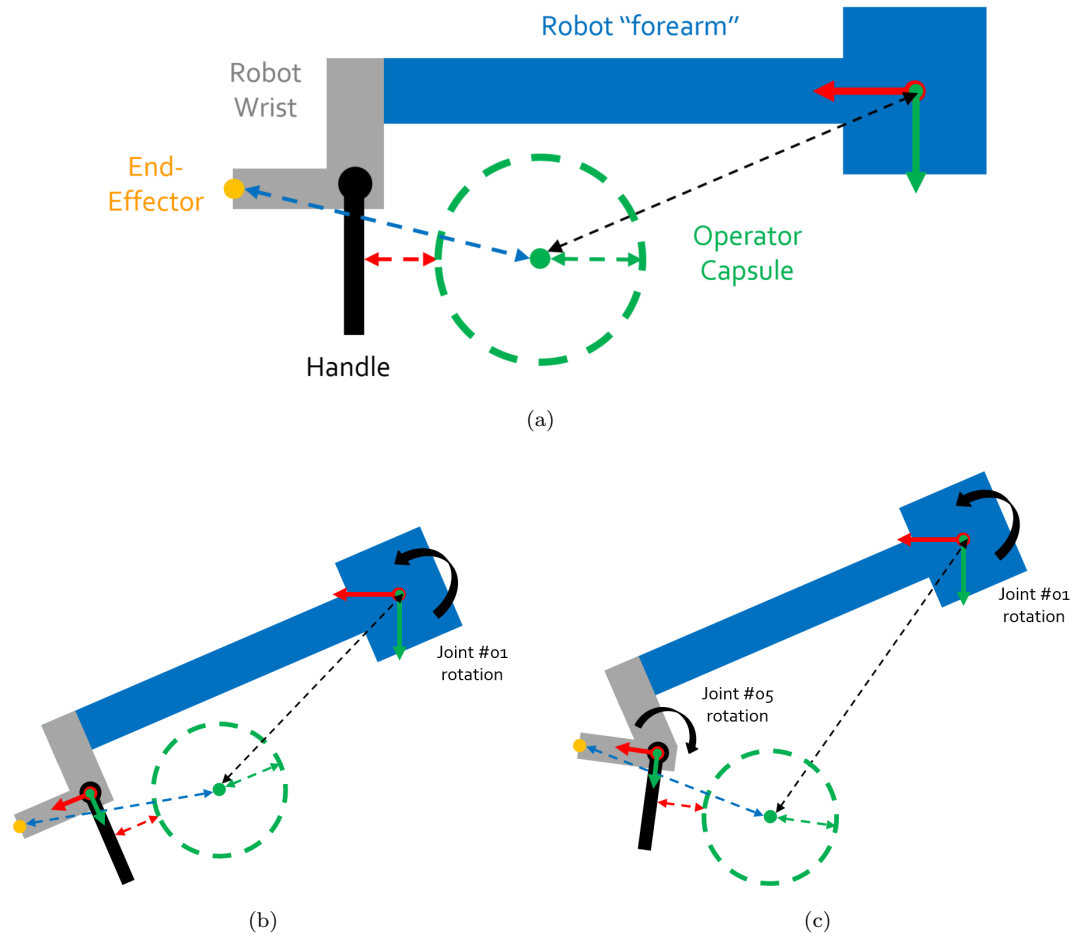


Figure 3.4: X-Y plane representation of the fixed relative pose (dashed blue) between robot's end-effector (solid yellow) and operator's capsule (dashed green), and of the varying relative pose (dashed black) between robot's base and operator's capsule in three different robot configurations.

- *opRad* is the parametric radius of the operator capsule;
- *opH* is the parametric height of the operator capsule;
- “RobotCaps” is a function that returns the set of capsules associated to robot links and to the tool on the basis of the joint configuration;
- “OpCap” is a function that outputs the operator capsule on the basis of the listed parameters and of the joint configuration of the robot according to Figs. 3.4 and 3.5.;
- “MinDist” is the function implementing the minimum distance calculation pictured in Fig. 3.2;
- *relDistOk* consists in the result of the procedure.

For the sake of clarity, at first the minimum distance is initialized at positive infinite (line 3). Then, robot and operator capsules are instantiated (lines 4-5). The following for-loop (lines 6-11) computes for each robot capsule the minimum distance with respect to the operator’s capsule (line 7), and updates the overall minimum distance if needed (lines 8-10). Finally, the conclusive if-block (lines 12-17) checks if the minimum distance condition (lines 12) is either satisfied or violated, and outputs the value of *relDistOk* accordingly.

Algorithm 1 Minimum Relative Distance Calculation

```

1: Parameters: minDistTh, robRad, opRelP, opRad, opH
2: procedure MINRELDISTCALC( q )
3:   minDist  $\leftarrow$  +Inf
4:   robotCaps  $\leftarrow$  RobotCaps( q, robRad )
5:   opCap  $\leftarrow$  OpCap(q, opRelP, opRad, opH)
6:   for all robCap  $\in$  robotCaps do
7:     rcDist  $\leftarrow$  MinDist( opCap, robCap )
8:     if rcDist < minDist then
9:       minDist  $\leftarrow$  rcDist
10:    end if
11:  end for
12:  if minDist  $\geq$  minDistTh then
13:    relDistOk  $\leftarrow$  1
14:  else
15:    relDistOk  $\leftarrow$  0
16:  end if
17: end procedure

```

Having covered in detail the strategy that allows to estimate the relative distance between the robot and the operator, let us focus on the monitoring of the relative velocity between them. The usage of relative human-robot velocity in combination with relative human-robot distance for safety purposes has been proposed as a safety metric by several contributions in the field of safe human-robot interaction, like for instance [73]. In this work, relative human-robot velocity is monitored in order to identify situations where at least a single robot link is moving towards the operator with so much speed, that an emergency stop may not be able to prevent a collision. To

this purpose, a reasonable choice for the threshold value is represented by the maximum allowed linear speed $\dot{x}_p^{UB}$.

Once again, the capsule-based geometry models of both the manipulator and of the operator represent the starting point. More specifically, the relative velocity between the robot and the operator is defined as the maximum relative velocity between the robot capsules' end-points and the operator's capsule.

Let us consider Fig. 3.5, where the solid grey bar represents the i -th robot link, surrounded by the corresponding capsule $robCap_i$. On the other hand, the operator capsule $opCap$ is pictured in the same way around its axis, passing through $opCap.P$. For both capsules clearances $robCap_i.r$ and $opCap.r$ are also highlighted. By means of kinematic calculations, the linear velocities vel_{P_0} and vel_{P_1} can be easily computed, given both joint angles and joint velocities. Then, by projecting velocity vel_{P_0} (vel_{P_1}) along the direction that connects point $robCap_i.P_0$ ($robCap_i.P_1$) to the axis of the operator capsule, we can compute the relative velocity of that specific point with respect to the operator. It is worth mentioning that the result of this projection is a signed velocity value, that is positive if the link end-point is moving towards the operator, and negative when it is moving away from the operator.

It is also worth mentioning that, as stated in [74], linear velocity varies linearly between the link end-points. As a result, either point $robCap_i.P_0$ or $robCap_i.P_1$ is necessarily the link point characterized by the maximum linear velocity. Since the projection with respect to the operator's capsule consists in a linear combination of the linear velocity coordinates, once again we can state that either point $robCap_i.P_0$ or $robCap_i.P_1$ is necessarily the link point characterized by the maximum relative velocity with respect to the operator, thus making it not necessary to check intermediate points.

Finally, please note that point $robCap_i.P_1$ of the i -th link corresponds to point $robCap_i.P_0$ of the $(i + 1)$ -th link. In addition, since the robot base (which typically corresponds to point $robCap_i.P_0$ of the first robot link) is normally still, the maximum relative velocity can be obtained by projecting the linear velocities of each end-point $robCap_i.P_1$ of the robot links.

Thus, a pseudo-code formalization of this procedure is given in Algorithm 2, where:

- *maxVelTh* represents the maximum allowed relative velocity between the robot capsules and the operator's one (positive value);
- "LinearVel" is a function that outputs the linear velocity of a specific robot capsule end-point given both joint angles and joint velocities;
- "MinDistPoint" is a function that returns the nearest point of the operator capsule with respect to a given point in the Cartesian space.

More specifically, at first the maximum relative velocity is set to zero (line 3). Then, robot and operator capsules are instantiated (lines 4-5). The following for-loop (lines 6-15) iterates along the robot capsules and computes the maximum relative velocity. For each capsule, the linear

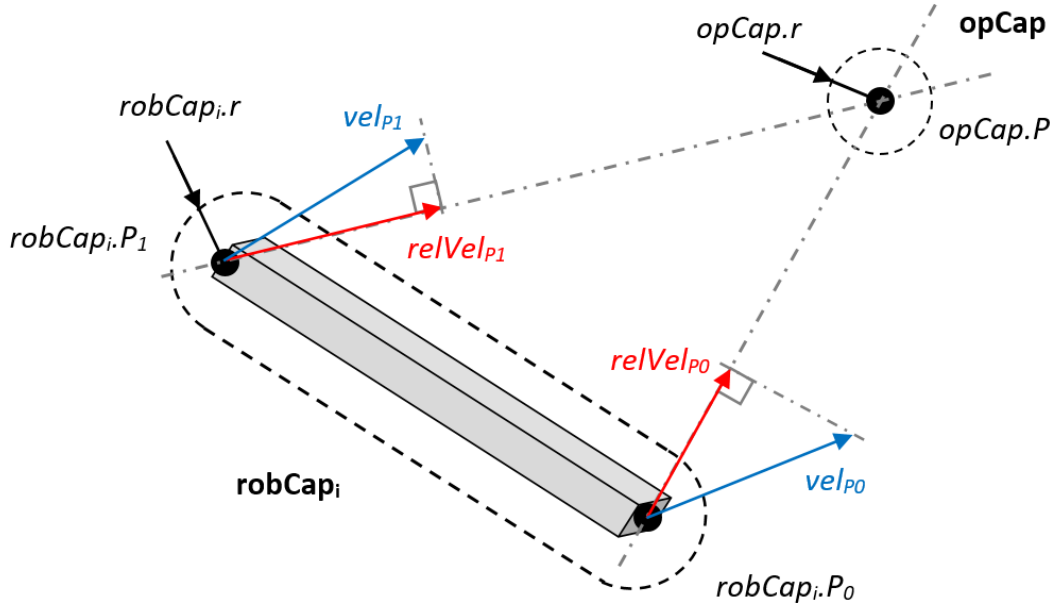


Figure 3.5: Geometric representation of the relative velocity between the end-points of a generic robot link (solid grey bar) and the operator's capsule. Capsules are pictured as dashed black lines. Capsules' defining points are represented as solid black dots, and clearances are highlighted with solid black arrows. Finally, robot link endpoints' velocities are depicted as blue arrows, while the corresponding projections towards the operator capsule are displayed as red arrows.

velocity of the end-point vel_{P_1} is computed (line 7). Then, as explained before, the direction connecting the robot capsule end-point to the operator's capsule is identified (lines 8-10), and the relative velocity is obtained by projecting vel_{P_1} along such direction (line 11). At the end of each iteration, if needed, the maximum relative velocity is updated (lines 12-14). Finally, the conclusive if-block (lines 16-20) checks if the minimum distance condition (line 16) is either satisfied or violated, and outputs the value of $relVelOk$ accordingly.

As a result, the comprehensive logic condition checked by the dynamic human-robot monitoring block can be expressed as follows:

$$relDistOk = 1 \quad \wedge \quad relVelOk = 1 \quad (3.22)$$

3.6 Implementation

In order to validate the proposed control architecture, we implemented the high-speed walk-through programming strategy described in the previous sections in a practical industrial use case. In particular, the strategy has been implemented on the Gaiotto GA-OL manipulator shown in Fig. 3.6, which is a 6 DoF industrial robot designed for spraying applications [3]. Typically, the GA-OL robot is endowed with an auxiliary turning table, on top of which the object to be sprayed is loaded and rotated for providing high manipulability to the robot.

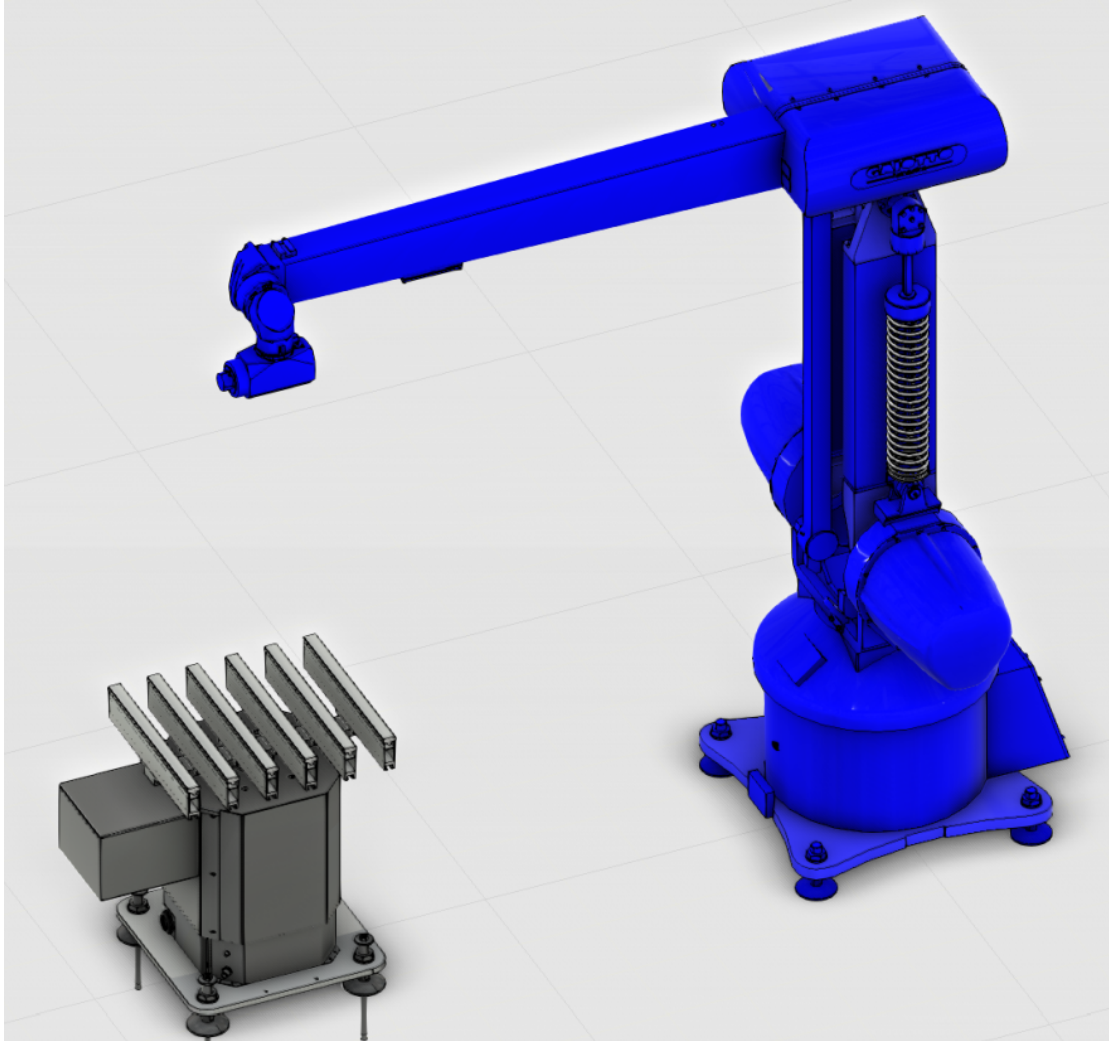


Figure 3.6: Gaiotto GA-OL manipulator (on the right, pictured in blue) and auxiliary turning table (on the left, pictured in grey).

Algorithm 2 Maximum Relative Velocity Calculation

```
1: Parameters:  $maxVelTh$ ,  $robRad$ ,  $opRelP$ ,  $opRad$ ,  $opH$ 
2: procedure MAXRELVELCALC(  $q$ ,  $\dot{q}$  )
3:    $maxVel \leftarrow 0$ 
4:    $robotCaps \leftarrow RobotCaps( q, robRad )$ 
5:    $opCap \leftarrow OpCap(q, opRelP, opRad, opH)$ 
6:   for all  $robCap \in robotCaps$  do
7:      $vel_{P_1} \leftarrow LinearVel( robCap.P1, q, \dot{q} )$ 
8:      $opCap.P \leftarrow MinDistPoint( opCap, robCap.P1 )$ 
9:      $uV \leftarrow opCap.P - robCap.P1$ 
10:     $uV \leftarrow uV / \|uV\|$ 
11:     $relVel_{P_1} \leftarrow vel_{P_1} \cdot uV$ 
12:    if  $relVel_{P_1} > maxVelTh$  then
13:       $maxVel \leftarrow relVel_{P_1}$ 
14:    end if
15:  end for
16:  if  $maxVel \leq maxVelTh$  then
17:     $relVelOk \leftarrow 1$ 
18:  else
19:     $relVelOk \leftarrow 0$ 
20:  end if
21: end procedure
```

Hardware architecture

The low-level robot control is enclosed in a logical hardware architecture including motion control, safety control and motor actuators. In particular, the walk-through programming strategy has been implemented into a Programmable Logic Controller (PLC) Intel Celeron 3955U 2.0 GHz, safety control integrated, EtherCAT master interface, RT-Ethernet Slave.

In order to measure the external forces and torques required in the walk-through programming, a Schunk F/T sensor Axia 80 has been mounted on the robot end-effector.

A mechatronic Human-Machine Interface (HMI) has been designed and developed to allow the user to guide the end-effector of the robot to walk through the desired trajectory (Fig. 3.7). The mechatronic handle is provided with two knobs, mounted with an angle of 90 degrees between each other. Each knob is equipped with a three-state dead-man switch that must be maintained by the user to ensure safety and with two functional buttons to enable the spraying, to activate/deactivate auxiliary functions, and also to send relative position commands to the auxiliary axis. In order to guarantee a conscious use of the instrument and for safety reasons, the user must keep both the dead-man buttons pressed at the same time to enable the robot motion.

The ergonomics of the handle allows to apply the principal and linear movements with the right hand, while the left hand allows to correct the orientation of the gun and the Tool Center Point of the robot.

According to the safety guidelines, the right shoulder of the human is misaligned with respect to the longest link of the robot (i.e. the third link for the Gaiotto GA-OL manipulator).

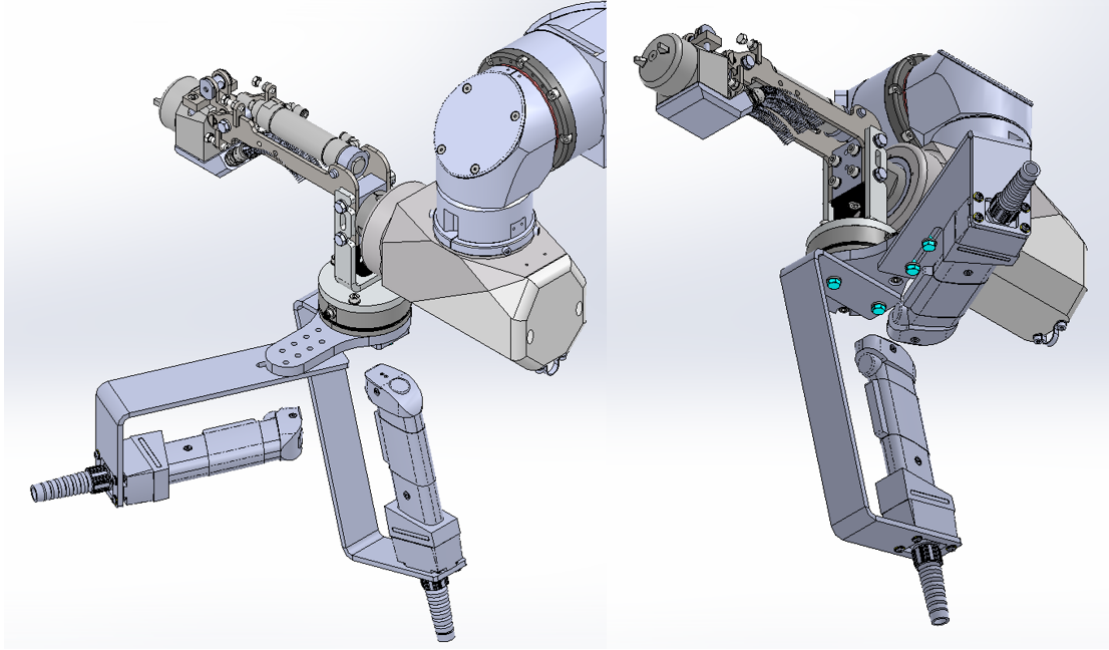


Figure 3.7: CAD representation of the mechatronic handle mounted on the end-effector of the GA-OL manipulator.

Software architecture

The software architecture integrated in the PLC is based on Codesys, according to the standard IEC 61131-3. Moreover, it is possible to implement and program standard control tasks and call C/C++ modules directly from the IEC runtime system to object-oriented extensions. In particular, the PLC incorporates the Functional Logic and the Safety Logic described in the previous sections, which communicate by means of virtual digital input/output, as shown in Fig. 3.8. The main program in the PLC Functional Logic is composed by the following tasks, programmed according to the IEC standard:

- Task **walk-through**: This task implements the walk-through programming strategy introduced in Sections 3.4 and 3.4 and provides the output pose x_{ref} which the position-controlled robot must follow. Both tool compensation and parameters adaption are implemented as C++ modules that are called directly by the walk-through task. Details about the implementation of walk-through programming will be provided in Section 3.6.
- Task **singularity avoidance**: Since the reference set-point is provided in the Cartesian space, the singularity avoidance task is programmed to keep the robot from reaching singular configurations during the motion. Details on the strategy implemented for avoiding singularities will be provided in Section 3.6.
- Task **reference tracking**: this task implements the tracking of the DTRF. Implementation details are provided in Section 3.6.
- Task **IO handle**: This task manages the digital input/output that come from the mechatronic handle.

On the other hand, the Safety Logic implements the safety functions as described in Section 3.5.

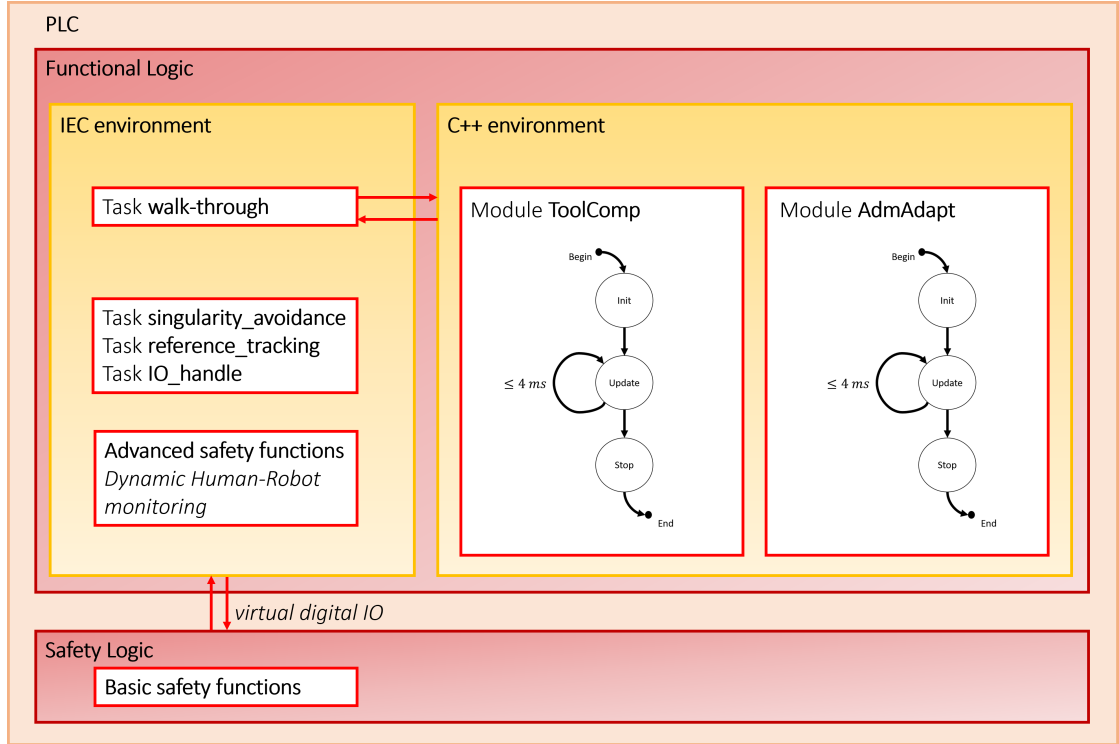


Figure 3.8: System environment and software architecture.

Implementation of walk-through programming

The walk-through programming strategy has been implemented by using two main C++ modules, that are called by the walk-through task and are structured as shown in Fig. 3.8. In particular, the *ToolComp* module implements the tool compensation described in Section 3.4, while the *AdmAdapt* module implements the parameters adaptation in admittance control described in Section 3.4.

At the beginning, some variables like, e.g., the inertia and damping starting values $M(0) = M_d$ and $D(0) = D_d$, are initialized thanks to the **init** function. Then, the **update** function runs periodically at 4 ms . This function is used for implementing the equations of the control algorithms described in Section 3.4 in order to compute the desired robot pose that has to be executed by the low-level position control. The resulting desired pose is then returned to the main PLC program task that commands the execution of the movement through the reference tracking task. Finally, the **stop** function cleans all the programming variables.

Section 3.4 provided a strategy for estimating the contact forces and torques F_c from the forces and torques F_{ext} measured by the F/T sensor. Then, Section 3.4 provided a possible way for adapting the parameters of the admittance control to recover a stable behavior of the human-robot interaction when oscillations are detected.

In the following, we provide the procedures for implementing the tool compensation and the

parameters adaptation strategies into the C++ environment of the robot controller. The procedure implemented for compensating the static and dynamic effects of the tool is provided in Algorithm 3. The input required by the algorithm are given by the mass of the tool m , its center of mass c , the inertia tensor I , the gravity vector g and the forces and torques measured by the F/T sensor F_{ext} . At first, the inertial vector and the pseudo-gravitational term are computed, according to (3.8) and (3.9), respectively (lines 3-4). Then, the matrix $V(a, \alpha, \omega, g)$ is computed according to [87] (line 6) and it is used for estimating the non-contact forces and torques due to the load dynamics, as defined in (3.7) (line 7). Finally, the contact forces and torques are estimated in line 8, according to (3.10).

The procedure implemented for adapting the parameters in the admittance control that allows the parameters to increase only when required due to the detection of increasing oscillations, but then restores the desired interaction model when such oscillations disappear, is provided in Algorithm 4. The variation of the damping is performed according to a constant damping-to-inertia ratio ($R_d(t) = R_d(0), \forall t \geq 0$). Thus, at first the damping-to-inertia ratio is computed according to (3.12) and based on the desired inertia matrix $M(0)$ and damping matrix $D(0)$ (line 3). Then, the output of the admittance control is computed starting from the interaction model (3.4) (lines 5-6-7). Then, the tracking error derivatives scaled with respect to bounds on the robot velocity and acceleration are computed in lines 8-9. Then, at each time instant, the detection index defined in (3.11) is computed (line 10).

If oscillations are increasing, (3.11) is not satisfied and the oscillatory behavior is detected in line 11, where ε is the detection threshold that can be tuned following the strategy proposed in [38]. In this case, the parameters of the admittance control must be adapted and the new components of the inertia matrix and the matrix of inertia variation are computed according to (3.15) and (3.16), respectively (lines 13-15). Finally, the inertia variation is computed in line 17 according to (3.17) and the damping is updated preserving the constant damping-to-inertia ratio in line 18.

Algorithm 3 Tool Compensation

```

1: Parameters:  $m, g, I, c, F_{ext}$ 
2: procedure TOOLCOMP(  $x, \dot{x}, \ddot{x}, F_{ext}$  )
3:    $\varphi \leftarrow \text{Array}(m, c, I)$ 
4:    $F_{g_{init}} \leftarrow V(0, 0, 0, g_{init})\varphi$ 
5:   for all  $F_{ext}$  do
6:      $V \leftarrow \text{Matrix}(\dot{x}, \ddot{x}, g)$ 
7:      $\hat{F}_{nc} \leftarrow V \cdot \varphi$ 
8:      $\hat{F}_c \leftarrow F_{ext} - \hat{F}_{nc} + F_{g_{init}}$ 
9:   end for
10: end procedure

```

Singularity avoidance

The singularity avoidance functionality is implemented by means of a dedicated task within the IEC environment, as shown in Fig. 3.8. It is worth mentioning that the kinematics of the GA-OL

Algorithm 4 Admittance Control and Parameters Adaptation

```
1: Parameters:  $D(0), M(0), \varepsilon, \Delta M, \beta, \Delta t$ 
2: procedure ADMADAPT(  $x, \dot{x}, \ddot{x}, \hat{F}_c, \dot{\chi}, \ddot{\chi}$ )
3:    $R_d(0) \leftarrow M^{-1}(0) \cdot D(0)$ 
4:   for  $i = 0$  to  $+\infty$  do
5:      $\ddot{x}_{ref} \leftarrow M^{-1} \cdot [\hat{F}_c - (D \cdot \dot{x}_{ref})]$ 
6:      $\dot{x}_{ref} \leftarrow \dot{x}_{ref} + \int \ddot{x}_{ref}$ 
7:      $x_{ref} \leftarrow x_{ref} + \int \dot{x}_{ref}$ 
8:      $\dot{\hat{x}} \leftarrow (\dot{x} - \dot{x}_{ref}) / \dot{\chi}$ 
9:      $\ddot{\hat{x}} \leftarrow (\ddot{x} - \ddot{x}_{ref}) / \ddot{\chi}$ 
10:     $\psi \leftarrow \text{Norm}(\ddot{\hat{x}} + R_d(0) \cdot \dot{\hat{x}})$ 
11:    if  $\psi \geq \varepsilon$  then
12:      for  $j = 1$  to  $6$  do
13:         $s_j \leftarrow \min \left\{ \frac{2\alpha_j(T(t_i) - \delta)}{\dot{\chi}_j^2}, \bar{m}_j \right\}$ 
14:      end for
15:       $S_i \leftarrow \text{diag}\{s_1, \dots, s_6\}$ 
16:    end if
17:     $M \leftarrow \text{InertiaAddTerms}(M, S, \Delta t, \beta)$ 
18:     $D \leftarrow R_d(0) \cdot M$ 
19:  end for
20: end procedure
```

manipulator presents two distinct singularities, both characterized by notable values of the fifth joint:

$$q_5 = \pm \frac{\pi}{2} \quad (3.23)$$

From a geometrical point of view, these singularities can be defined as hyperplanes in the Joint Space of the GA-OL, thus preventing traditional approaches (like for instance the aforementioned DLS-Jacobian [102]) to be effective. As a matter of fact, hyperplanes defined by equation (3.23) completely partition the Joint Space of the GA-OL manipulator, making it impossible to traverse from one subset to another without encountering a singular configuration, since any possible perturbation of the original trajectory would eventually result in a slightly different singular configuration. Nevertheless, this is the main reason behind the introduction of the auxiliary turning table pictured in Fig. 3.6, as mentioned in the introduction of Section 3.6. An analysis conducted on more than 60 programs realized using GA-OL manipulators equipped with the auxiliary turning table showed that q_5 was always comprised between -83 deg and $+87$ deg. Consequently, it is possible to effectively prevent the GA-OL to reach singular configurations during the teaching phase by simply enforcing the following position limits on the fifth joint:

$$q_5 \in \left(-\frac{\pi}{2}, +\frac{\pi}{2} \right) \quad (3.24)$$

In the neighborhood of the limit condition a sigmoid function is used to continuously scale robot velocities, so that: i) the manipulator stops before reaching the singular configuration, ii) it can

only be moved in the opposite direction with respect to the singularity.

Tracking of the reference frame

Similarly to the singularity avoidance, also the DTRF functionality is assigned a dedicated tasks within the IEC environment (see once again Fig. 3.8). More specifically, the DTRF is defined as a 6 DoF Cartesian frame, whose coordinates with respect to the world-base frame are updated at each time step according to a specific input or time-law, which in our case corresponds to the pose output x_{ref} of the admittance filter (see Algorithm 4).

As already introduced in Section 3.3, a convenient way to realize admittance control on a closed industrial controller is to instantiate a DTRF, initialize its pose with the very same Cartesian coordinates of the robot TCP, and then synchronize its pose with respect to the admittance filter output, thus forcing the robot to follow the desired trajectory. A pseudo-code implementation of the robot program that performs such functionality is given in Algorithm 5.

Algorithm 5 Teaching Program and tracking of the DTRF

```

1: global variables:  $x$ ,  $x_{ref}$ ,  $DTRF$ ,  $StartTeach$ ,  $StopTeach$ 
2: procedure TEACHING PROGRAM( )
3:    $DTRF \leftarrow x$ 
4:   Wait(  $StartTeach$  )
5:   SetRefSys(  $DTRF$  )
6:   while ( not(  $StopTeach$  ) ) do
7:      $DTRF \leftarrow x_{ref}$ 
8:     Lin(  $0_6$  )
9:   end while
10: end procedure

```

Basically, the program instantiate the DTRF and sets its pose to the current pose of the robot TCP x (line 3). Then, as soon as the starting signal $StartTeach$ becomes true (line 4), the active reference system is set to the instantiated DTRF (line 5). At this point, the program enters the while loop (lines 6 - 9), within which: i) the coordinates of the DTRF are updated according to the output of the admittance filter x_{ref} (line 7), ii) a linear motion towards the null pose 0_6 is commanded to the robot (line 8), so that it follows the DTRF. Finally, as soon as the stopping signal $StopTeach$ becomes true, the execution of the program is terminated.

Safety Logic

Finally, moving to the safety controller, the implementation of the safety logic lies between the safety environment and the IEC environment, as depicted in Fig. 3.8. More specifically, the basic safety functions (see subsection 3.5) are implemented within the “Safety Logic” block, while the more advance functionalities (see subsection 3.5) are executed within the scope of the “IEC environment”.

Once the outputs of of algorithms 1 and 2 are computed, and the status of condition (3.22) is determined, the IEC environment sends these data to the Safety Logic layer by means of

“virtual digital IO”, i.e. boolean variables that are exchanged between the two domains via shared memory.

Finally, the Safety Logic collects all the information needed to compute the Safe Torque Off (STO) signal, that cuts-off the electric power to the axis motors (thus preventing them to develop torque) whenever it is de-activated by the Safety Logic.

As far as parameters are concerned, the following values have been used:

- $\|\dot{x}_p^{UB}\| = 1.30 \text{ m/s}$ - maximum linear speed at TCP, due to spraying task requirements¹;
- $\dot{q}^{Max} = [200, 200, 200, 300, 300, 300] \text{ deg/s}$ - maximum GA-OL joint velocities;
- $\ddot{q}^{Max} = [450, 450, 450, 450, 450, 450] \text{ deg/s}^2$ - maximum GA-OL joint accelerations;
- $robRad = [0.25, 0.20, 0.10, 0.075, 0.05, 0.05, 0.075] \text{ m}$ - robot and tool capsules radii;
- $opRelP = [0.000, 0.000, 0.250] \text{ m}$ - operator capsule position with respect to GA-OL end-effector. Relative orientation is considered null;
- $opRad = 0.200 \text{ m}$ - operator capsule radius;
- $opH = 1.80 \text{ m}$ - operator capsule height;
- $stopT = 0.35 \text{ s}$ - robot stopping time;
- $minDistTh = 0.455 \text{ m}$ - minimum allowed distance between operator and robot capsules. It is obtained by multiplying $stopT$ by $\dot{x}_p^{UB}$;
- $maxVelTh = 1.30 \text{ m/s}$ - maximum allowed relative velocity between operator and robot capsules. Given $stopT$ and $minDistTh$, it can be set equal to $\dot{x}_p^{UB}$.

The operator capsule radius and height have been empirically selected as the medium value between the real characteristics of the testers involved in the experimental campaign. A sensor could be integrated for measuring the effective anthropometric characteristics and the exact values could be inserted into the variables $opRad$ and opH .

3.7 Experimental Results

An extensive experimental evaluation has been performed in order to:

- verify the effectiveness of the proposed control architecture, in terms of adaptation of the admittance parameters and functionalities of the safety controller;
- evaluate the performance of the implementation and compare the results with a commercially available solution with back-drivable actuators and mechanical compensation systems for passive walk-through programming (Gaiotto GA-2000 [3]);

¹The value of the velocity limit has been selected as the maximum value of the TCP reached by the robot in 40 different programs executed on a GA-2000 manipulator, which is equipped with mechanical compensation systems for passive walk-through programming.

- estimate the usability of the proposed architecture in actual industrial scenarios.

In the following, the experimental protocol is described (Section 3.7). Then, each goal of the validation and the experimental results are discussed within dedicated subsections (Sections 3.7, 3.7 and 3.7).

Experimental Protocol

The experimental campaign conducted in Gaiotto Automation SpA, has been designed in the following way:

- **Testers:** 7 Gaiotto employees with different level of experience in the development and commissioning of the GA-2000 robot [3] installations have been selected within the service and the technical department. Testers were divided in two groups: high-skilled testers (4 people), and medium-skilled testers (3 people);
- **Experiments:** 4 different kind of experiments have been considered. More in depth: i) execution of a linear trajectory defined by a series of enumerated targets while keeping a fixed orientation, see Fig. 3.9(a), ii) execution of the same linear trajectory, with variable orientation, iii) simulated glazing of a floor-mounted flush toilet, see Fig. 3.9(b), and iv) simulated glazing of a wall-hung flush toilet, see Fig. 3.9(c).
- **Admittance Parameters:** 3 different parameter sets of the admittance control have been considered, see Table 3.7. The intermediate set “P1” has been experimentally tuned so that no significant oscillations arise, before being eventually damped by the adaptation algorithm. The aggressive (i.e. less stiff) set “P0” is computed by reducing P1 values by 33%, while the conservative (i.e. more stiff) set “P2” results from a 50% increase of P1 values. For each set of parameters, nominal and maximum values are given for both inertia and damping. In addition, also the corresponding value of the instability detection threshold is listed.

As far as the selected experiments are concerned, to better explain the difference between the first one and the second one, Fig. 3.10 shows how the targets are approached in terms of relative orientation of the TCP with respect to the labeled targets. Basically, during the first experiment testers have been required to keep the TCP as aligned as possible with respect to the targets (Fig. 3.10(a) and 3.10(b)), while during the second they have been asked to point the tool from the outside in (Fig. 3.10(c) and 3.10(d)).

Moving to the testers, the four high-skilled ones have been asked to perform 2 test per experiment and parameter set, for a total of 36 tests per person, and 144 tests overall. On the other hand, the 3 medium-skilled testers have been asked to repeat the first experiment twice per each parameter set, for a total of 6 tests per person, and 18 overall. In conclusion, 162 tests were performed and recorded.

This protocol was designed in order to allow operators to acquaint themselves with the proposed walk-through programming solution by starting from simpler test-cases and moving to more complex ones. As a matter of fact, during the first experiment they are required to basically control only the positional DoFs of the robot TCP, while during the second they need to exploit also orientation DoFs. Finally, during the third and the fourth experiment high-skilled operators were faced with two actual programming scenarios referring to the two most common cases in the context of industrial ceramic glazing.

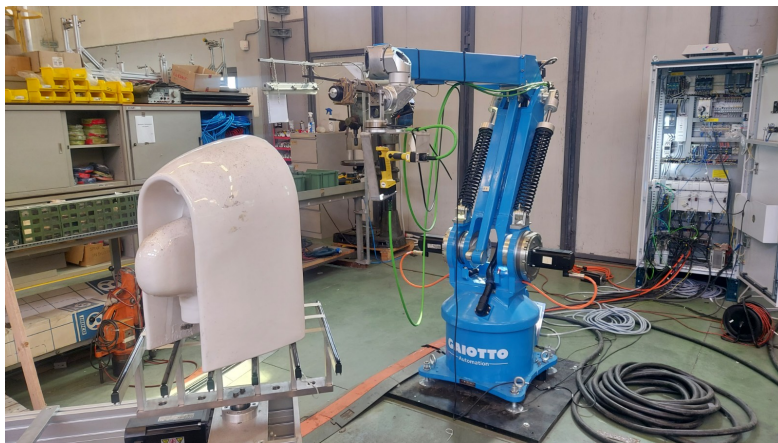
The accompanying video clip shows the execution of the four experiments. In order to show the parameters adaptation, the video report as an example the execution of the first experiment performed by three out of the seven users with the three different set of parameters. The first part of the video shows the teaching phase, while in the last part of the clip, the reproduction of the trajectory of the fourth experiment is executed.



(a)

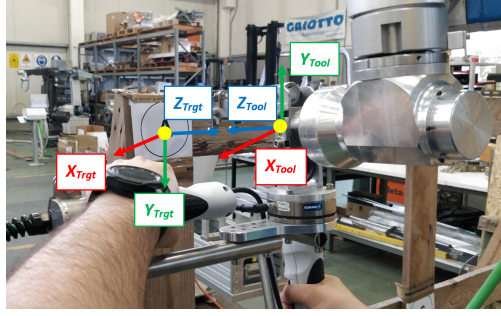


(b)

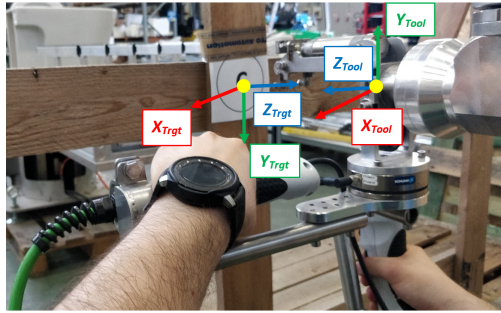


(c)

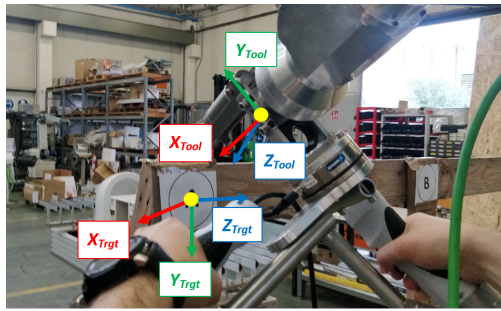
Figure 3.9: Experimental setup: 3.9(a) linear trajectory with enumerated targets “A” to “F”, 3.9(b) floor-mounted toilet, 3.9(c) wall-hung toilet.



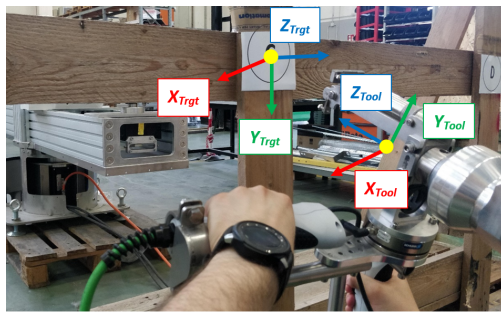
(a)



(b)



(c)



(d)

Figure 3.10: Approach orientation of labeled targets during the first experiment (3.10(a) and 3.10(b)) versus approach orientation during the second experiment (3.10(c) and 3.10(d)).

Table 3.1: Admittance control parameter sets considered during the experimental validation campaign. For the sake of clarity, linear parameters are labelled with the subscript “p”, while orientation-related quantities are labelled with the “o” subscript.

Parameters Set		Inertia				Damping			Instab. Threshold	
Label	Description	Linear [kg]		Rotational [kg m ²]		Linear [kg/s]		Rotational [kg m/s]	ε	
		M_p (0)	M_p^{Max}	M_o (0)	M_o^{Max}	D_p (0)	D_p^{Max}	D_o (0)		D_o^{Max}
P0	aggressive	10.00	20.00	1.00	2.00	40.00	80.00	4.00	8.00	0.0075
P1	intermediate	15.00	30.00	1.50	3.00	60.00	120.00	6.00	12.00	0.0392
P2	conservative	20.00	40.00	2.00	4.00	80.00	160.00	8.00	16.00	0.0435

Verification

From the verification point of view, a series of experiments have been performed in order to demonstrate the effectiveness of the proposed architecture, whose results are hereby presented and discussed.

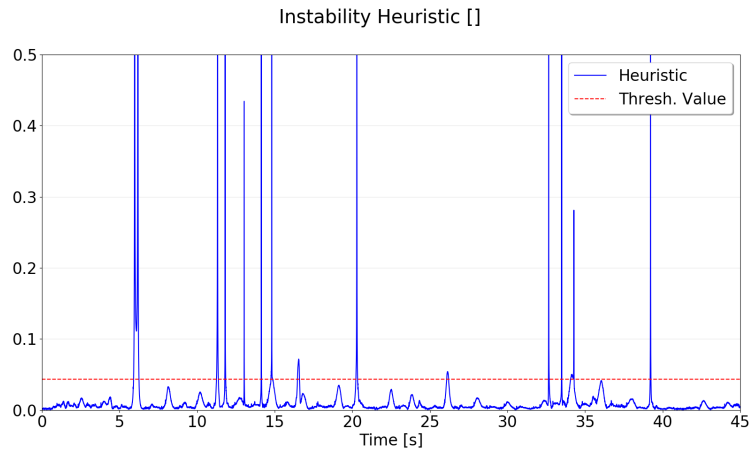
Let us start from Fig. 3.11, that shows how the adaptation of the admittance parameters works. In the example, the parameter set “P2” was active. In Fig. 3.11(a) the detection index ψ defined in the heuristic (3.11) is compared with the threshold value ε , which is equal to 0.0435 (see Table 3.7). Whenever the detection index overcomes the detection threshold, oscillations are rising and an adaptation is required. In order to ease comprehension of the data, a logical value labeled as “Instability” has been set to *True* whenever the heuristic overcomes the threshold, and *False* otherwise. Figures 3.11(b) and 3.11(c) show the linear inertia, and the rotational damping. Whenever instability is detected, the algorithm immediately increases the value of the parameters in order to rapidly stabilize the system. Then, as soon as the desired behavior is restored, the forgetting factor allows the parameters to slowly decrease, thus allowing faster and less fatiguing motions. The instabilities that are detected in Fig. 3.11 are due to the behavior of the human operator that, during the interaction, can behave in a stiff way. Indeed, the interaction with stiff environments can produce instability when using admittance-controlled robots. This can be due to several factors such as the robot structural dynamics, the limitations of the actuators, and the implementations of low-level controllers (see [28] for further details).

Moving to the safety controller, let us show how the violation of the conditions defined by the basic safety functions results in disabling the STO and stopping the robot. Figure 3.12(a) shows that as soon as the TCP linear velocity limit ($\dot{x}_p^{UB} = 1.30\text{ m/s}$) is exceeded, the STO signal (properly scaled) is disabled and the robot stops. In addition, the magnification shown in Fig. 3.12(b) also proves that the robot completely stops within the declared stopping time equal to 0.35 s.

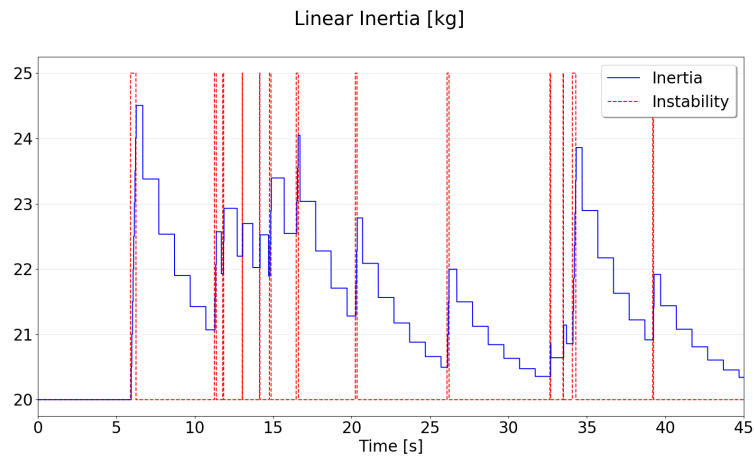
A similar situation is shown in Fig. 3.13(a), where joint #02 velocity violates the imposed limit ($\dot{q}_2^{UB} = \pm 60.00\text{ deg/s}$), thus making the safety controller disable the STO and stop the machine. Figure 3.13(c) shows the contextual behaviour of the TCP linear velocity, that rapidly decreases to zero right after the violation occurs. Magnifications are shown in Figs. 3.13(b) and 3.13(d).

Furthermore, Fig. 3.14(a) refers to the third basic safety function, i.e. the joint acceleration limit, showing that as soon as joint #03 violates the corresponding limit ($\ddot{q}_3^{UB} = \pm 171.43\text{ deg/s}^2$) the safety controller disable the STO (as usual, properly scaled) and the TCP linear velocity rapidly decreases to zero (Fig. 3.14(c)). Magnifications are shown in Figs 3.14(b) and 3.14(d).

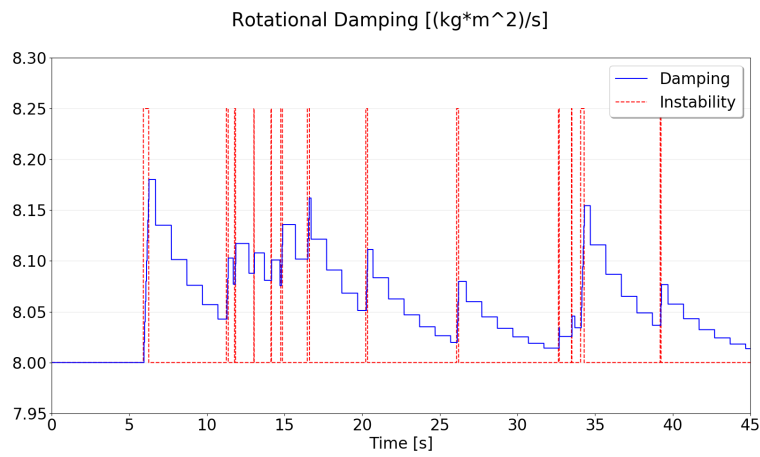
Finally, Figs. 3.15 and 3.16 shows violations related to the dynamic human-robot monitoring algorithms. Figure 3.15(a) shows how the violation of the minimum relative distance condition triggers the disabling of the STO by the safety controller, while in Fig. 3.15(c) TCP linear velocity is depicted. Magnifications are shown in Figs. 3.15(b) and 3.15(d). The very same behaviour characterizes the safety controller response to a relative velocity violation, as it is



(a)



(b)



(c)

Figure 3.11: Adaptation of admittance parameters in presence of detected oscillations.

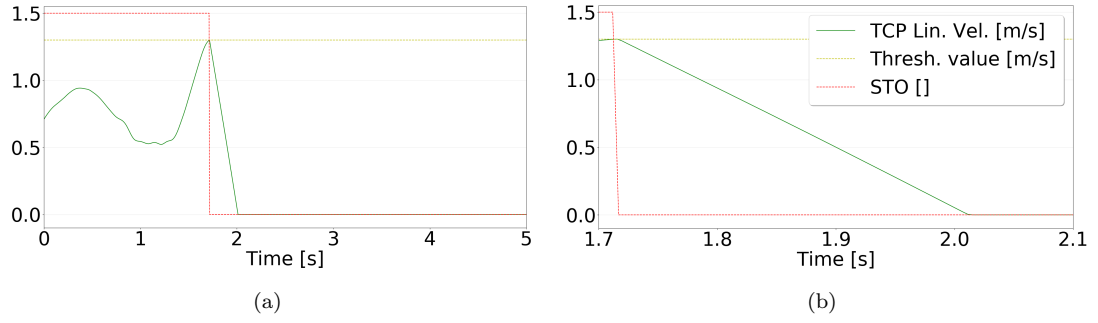


Figure 3.12: Violation of TCP linear velocity limit and corresponding variation of scaled STO signal, with magnification between 1.70 s and 2.10 s.

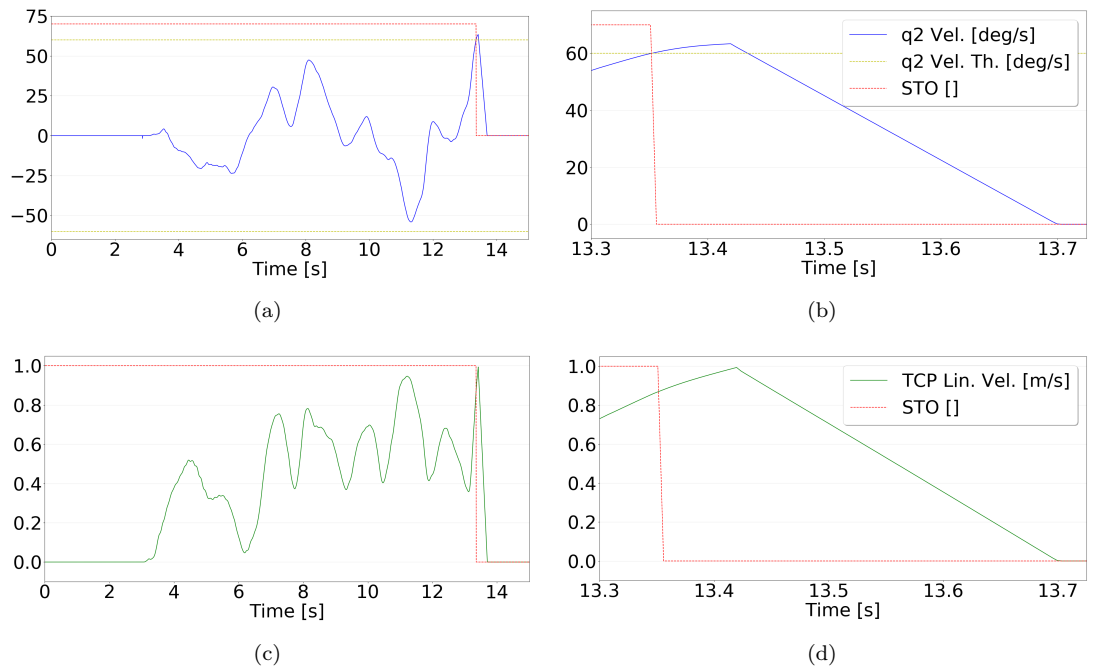


Figure 3.13: Violation of joint #02 velocity limit with TCP linear velocity and corresponding variation of scaled STO signal, with magnification between 13.30 s and 13.72 s.

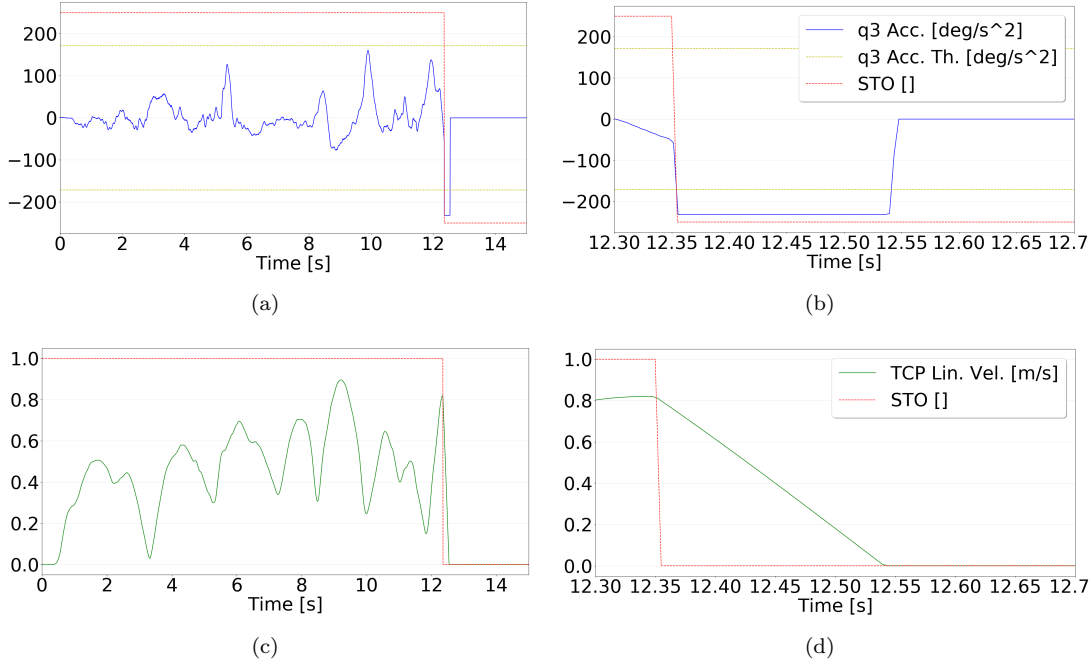


Figure 3.14: Violation of joint acceleration limit with TCP linear velocity and corresponding variation of scaled STO signal, with magnification between 12.30 s and 12.70 s.

shown in Fig. 3.16.

Performance Evaluation

In order to evaluate the performance, two main indicators have been selected: the maximum TCP linear speed reached during the programming phase, and the ratio between the programming time and the program duration. First, let us consider the maximum TCP speed. This metric is extremely significant, since the capability to record high-speed trajectories is one of the key factors that prevent state-of-the-art walk-through programming solutions to be deployed on robots whose motion needs to be synchronized with task-related set-points, like for instance spraying robots.

In this regard, Fig. 3.17 shows the approximated distribution of the maximum recorded TCP linear speeds with respect to testers (Fig. 3.17(a)), and admittance parameter sets (Fig. 3.17(b)). Starting from the testers, these data clearly show that the four most skilled ones (labels “T0” - “T3”) were able to reach maximum velocities that are significantly higher with respect to the three least skilled testers (labels “T4” - “T6”). Even though tester *T4* outperformed tester *T3* looking at median and average values (orange line and green triangle, respectively), the approximated 99.7-percentile of the distribution (represented as the upper whisker) reached by *T3* is still quite higher than the one recorded by *T4*. The reason behind the choice of the 99.7-percentile as the metric for determining the maximum TCP speed lies in the fact that, supposing a normal distribution $\mathcal{N}(\mu, \sigma)$ of the collected data, we can state that 99.7 of the data will lie within $\mu \pm 3\sigma$. Consequently, we can state that the value $\mu + 3\sigma$ (represented by the upper whisker in the box plots) is a reasonable approximation of the maximum of the distribution, while

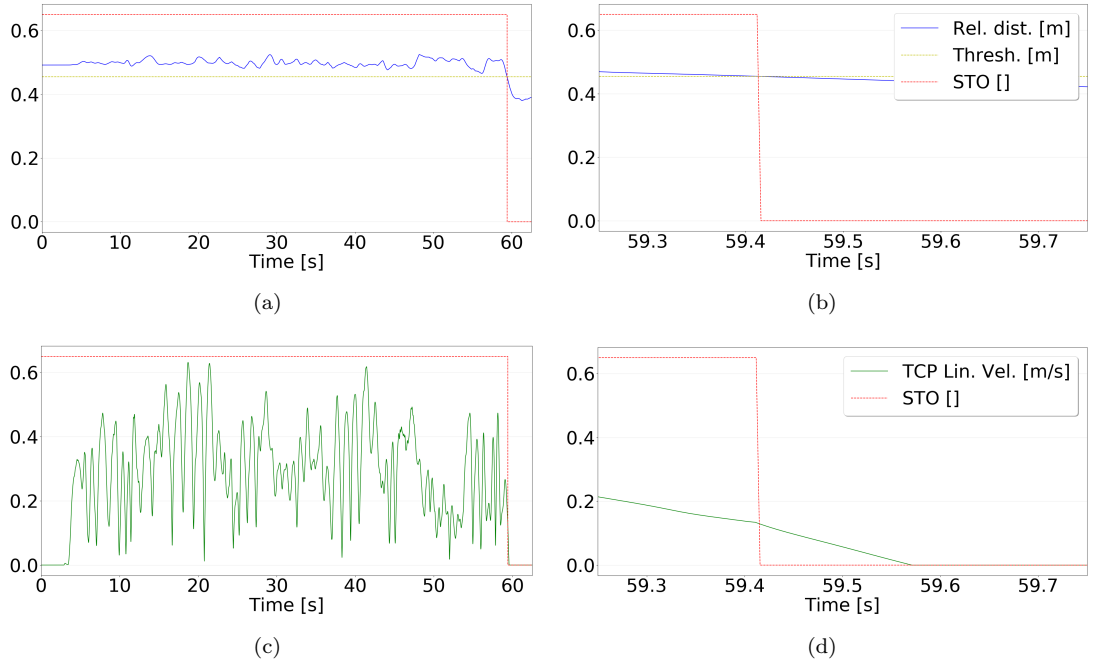


Figure 3.15: Violation of the relative human-robot minimum distance with TCP linear velocity and corresponding variation of scaled STO signal, with magnification between 59.25 s and 59.75 s.

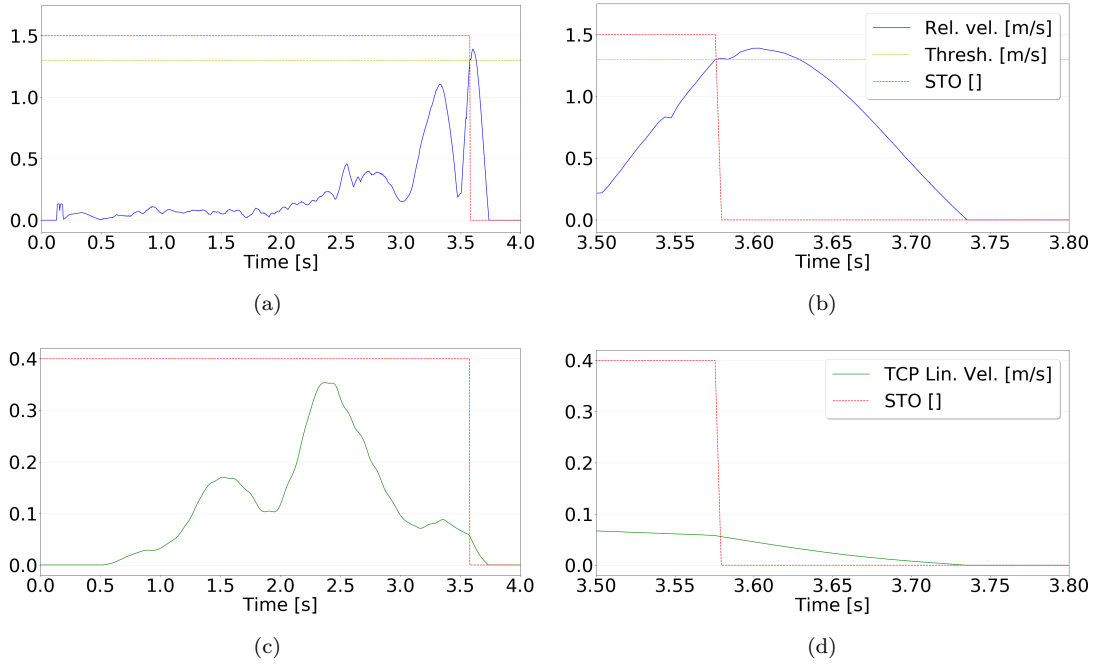


Figure 3.16: Violation of the relative human-robot maximum velocity with TCP linear velocity and corresponding variation of scaled STO signal, with magnification between 3.50 s and 3.80 s.

all the data lying beyond this value are marked as outliers. In addition, Fig. 3.17 also shows how the proposed solution enables both high-skilled and relatively low-skilled users to easily reach velocity values that lie far beyond the limit fixed for collaborative applications (250.0 mm/s), thus demonstrating the suitability of the approach for tasks that require the execution of high-velocity continuous trajectories in the learning phase.

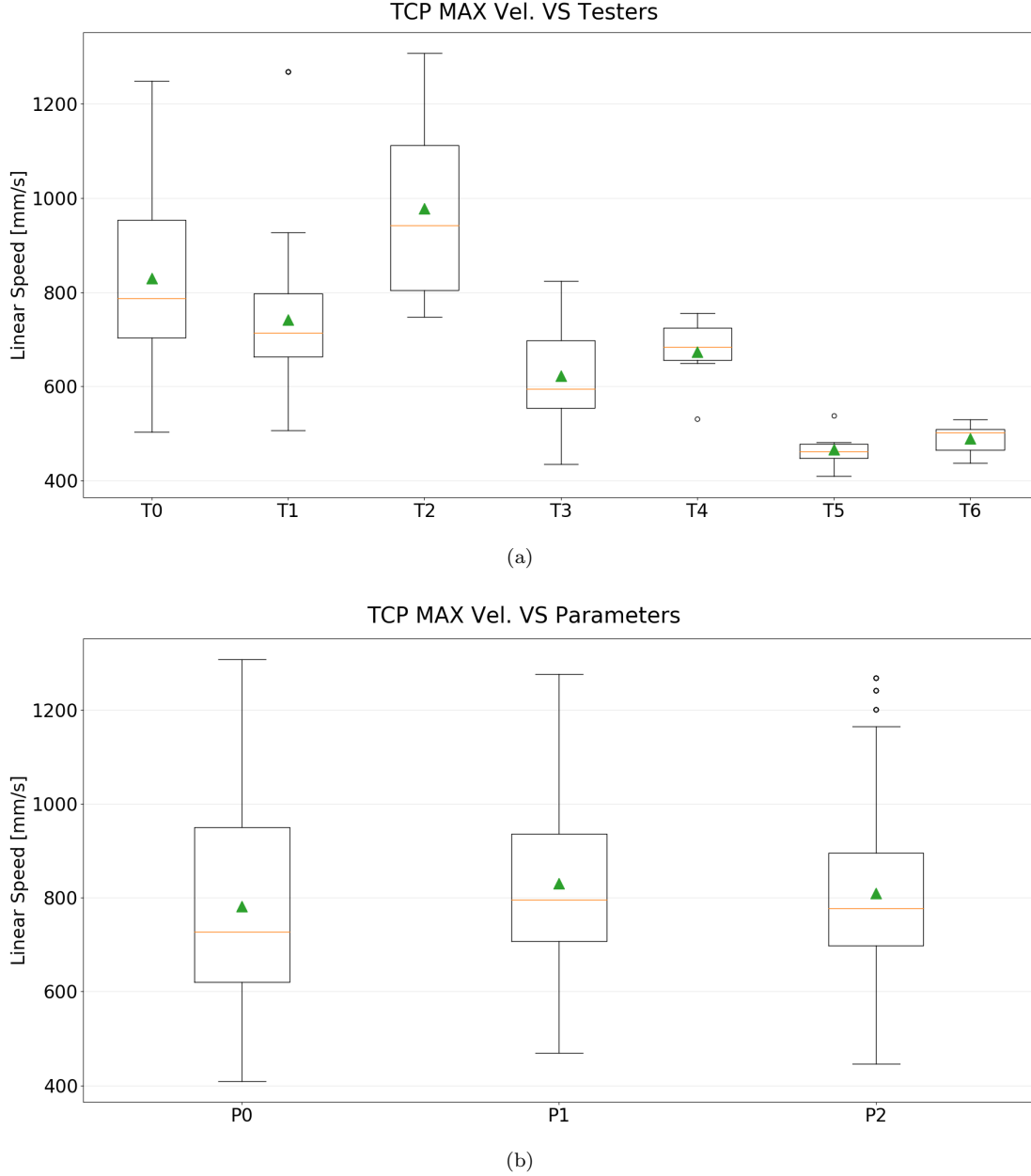


Figure 3.17: Approximated distribution of maximum recorded TCP linear speed values grouped by testers and by admittance parameter sets. Median value is depicted as a solid orange line, while average value is represented as a green triangle.

Moving to the admittance parameters, a trend does not emerge as far as median and average values are concerned. Nevertheless, it can be clearly seen that the approximated 99.7-percentile decreases when moving from aggressive admittance parameters to more conservative ones.

Moreover, in order to draw a significant comparison between the proposed solution and walk-

through programming approaches based on mechanical solutions, a benchmark has been computed on 40 different programs recorded on a GA-2000 manipulator [3]. As already mentioned, this manipulator is equipped with back-drivable actuators and mechanical compensation systems that allow passive walk-through programming. Due to its complex mechanics, a manipulator that provides passive walk-through is very expensive compared to standard industrial manipulators. Figure 3.18 shows a comparison between the previously mentioned benchmark and the approximated distribution of maximum linear TCP speed values registered during the experimental validation campaign. For the sake of clarity, due to the results shown in Fig. 3.17(a), we decided to compute the approximated distribution of the proposed solution by considering only the experiments performed by the four more skilled testers.

Even though the mechanical solution outperforms the proposed one in terms of median and average maximum speed, our strategy is characterized by a higher value of the approximated 99.7-percentile, which can be considered also the maximum value of the distribution, provided that we exclude outliers. In conclusion, we can state that the proposed solution is able to guarantee a performance level that is at the same time comparable with respect to the very much expensive existing mechanical solution, and far superior than the one reachable by collaborative solutions, whose speed limit is set to 250 mm/s according to [49].

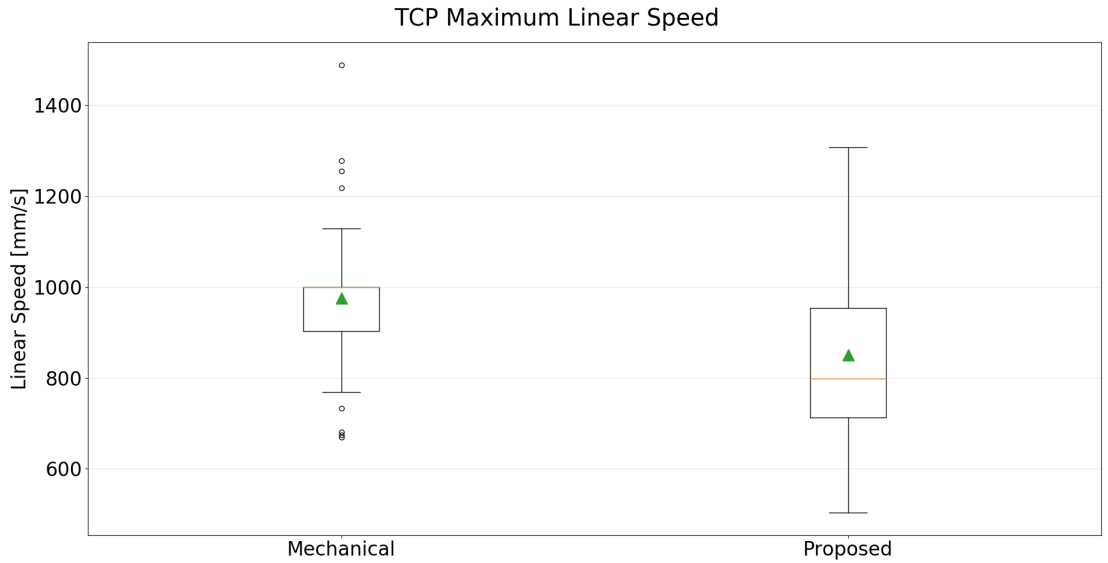


Figure 3.18: Comparison between approximated distribution of maximum TCP Speed in GA-2000 programs (labeled as “Mechanical”) and during the selected experiments (labeled as “Proposed”). Median value is depicted as a solid orange line, while average value is represented as a green triangle.

Finally, moving to the ratio between the program duration and the programming time, Fig. 3.19 shows the approximated distribution of the ratio between the programming time and the duration of the resulting registered program. For the sake of clarity, in this context programming time is defined as the time needed to complete an experiment, taking into account possible repetitions in case functional or safety errors occur. That said, data are grouped by the different experiments due to the significant differences among themselves and outliers have been removed from the

shown box plot to improve the quality of the visualization. In conclusion, we can state that the proposed solution guarantees an adequate level of efficiency, since the time needed to record 1.00 s of program trajectory is slightly less than 2.00 s on average, and slightly less than 3.25 s in the worst-case. Moreover, median values are all comprised in the interval [1.00, 1.25] s, proving a very high degree of efficiency in absence of errors that entail repetitions. In conclusion, we can state that the proposed solution allows even operators with low skills in robotics to quickly program a manipulator.

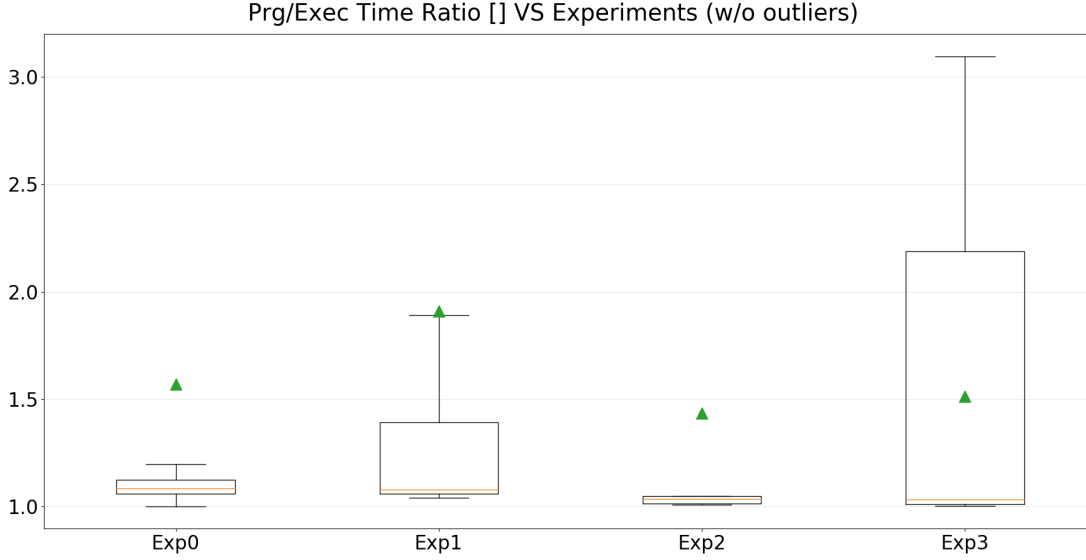


Figure 3.19: Approximated distribution of ratio between program time and programming duration. Median value is depicted as a solid orange line, while average value is represented as a green triangle.

Usability Evaluation

In order to properly evaluate the usability of the proposed walk-through programming architecture in actual industrial scenarios, the selected testers have been asked to answer a questionnaire right after the tests. The questionnaire is composed by two distinct parts. At first, testers are asked to rate indicators related to the different parameter sets: i) motion accuracy, ii) motion smoothness, iii) controller responsiveness (defined as the absence of perceivable “lag” between exerted forces and motion), and iv) physical effort. Then, they evaluate more general indicators: i) usefulness, ii) perceived safety, iii) perceived comfort, iv) induced stress, and v) learning curve speed.

Finally, a global evaluation score is also requested. Testers have been requested to fill in scores ranging from “0” (meaning either “null” or “extremely negative”), to “10” (meaning either “extremely high” or “extremely positive”). Answers have been weighted by the number of test performed by each tester: 36 for the high-skilled, and 6 for the medium-skilled.

Taking into account the results of the first part, Fig. 3.20 clearly shows that the physical effort needed to move the robot significantly increases when the admittance filter becomes more

stiff 3.20(d). A similar trend, even though less evident, characterizes the evaluation of the motion accuracy 3.20(a), while the opposite trend is shown by the controller responsiveness 3.20(c), at least as far as its media is considered. Finally, no significant trends emerges from the data regarding the motion smoothness 3.20(b). Unsurprisingly, collected data suggest that an aggressive setup of the admittance filter improves performance at the expenses of accuracy, while a very conservative parameter set demands a significant effort from the operator. Even though the choice of the optimal parameter might be highly subjective, we can state that in general an effective and efficient parameter set can be found by tuning the admittance parameters in such a way that no significant oscillations arise during the programming phase.

Moving to the second part of the questionnaire, the results of the general evaluation of the proposed solution are displayed in Figure 3.21. More specifically, a very high overall score has emerged with respect to usefulness of the proposed solution 3.21(a), perceived safety 3.21(b), and learning curve speed 3.21(e). Please also notice that the low score regarding to induced stress 3.21(d) corresponds to a positive evaluation by the testers. A slightly lower, yet still positive, score emerges on the perceived comfort 3.21(c) too, which is probably influenced by the occasional occurrence of oscillations. Finally, on the basis of the overall global evaluation we can state that the proposed architecture represents a promising solution not only from the theoretical point of view, but also from the application perspective.

3.8 Conclusion and Future Developments

The proposed solution aims at allowing human operators to program industrial manipulators, equipped with closed control systems, by directly teaching high-speed trajectories, while guaranteeing the operator's safety. The proposed framework relies on an adaptive admittance filter that converts the interaction forces measured by a force/torque sensor to TCP displacements and adapts its inertia and damping parameters so that any kind of instability is detected and avoided. In addition, a dedicated safety controller has been developed to monitor the kinematic configuration of the manipulator and stop its motion whenever the risk of a collision with the human operator becomes too high. Finally, the integration with the robot controller is designed in such a way that this architecture does not need a open-controller interface, thus enabling its integration with the vast majority of the industrial controllers available on the market.

The implementation of the proposed walk-through programming framework on an industrial manipulator designed for spraying applications is extensively detailed and the results of a thorough experimental validation are presented. Then, a comparison with respect to passive walk-through programming techniques is discussed. As a result, the authors can state that the proposed architecture successfully achieves a fruitful trade-off between safety and productivity, by guaranteeing the operators' safety and the possibility to directly record high-velocity trajectories.

As far as future developments are concerned, the authors foresee the development of a sensor-less version of the proposed solution, by integrating a dynamic model of the manipulator, and

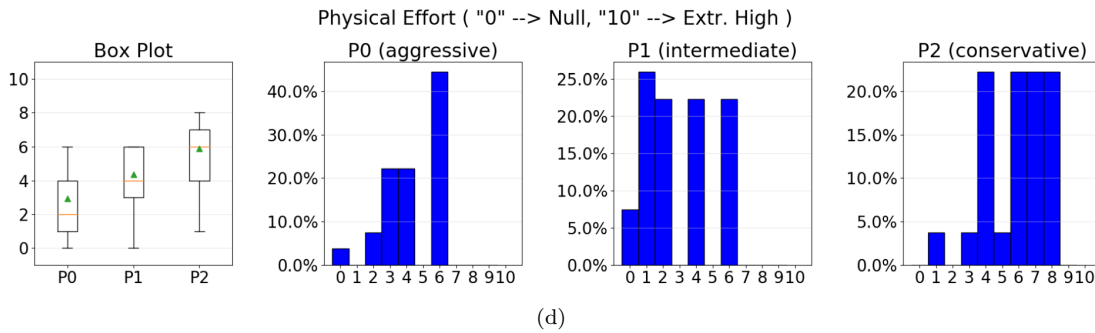
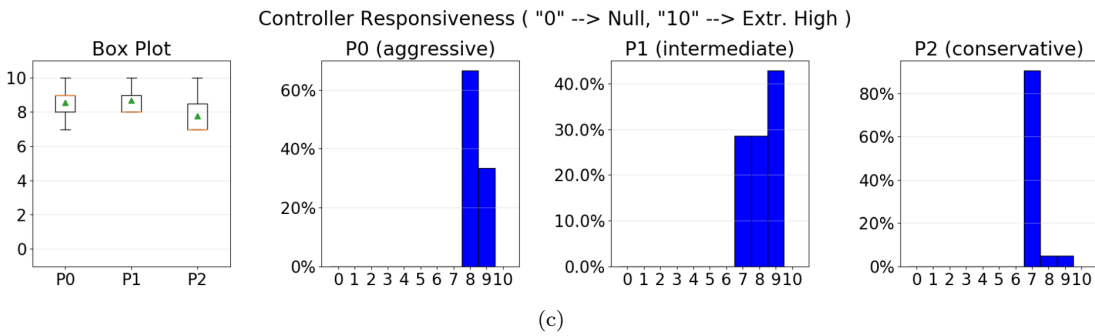
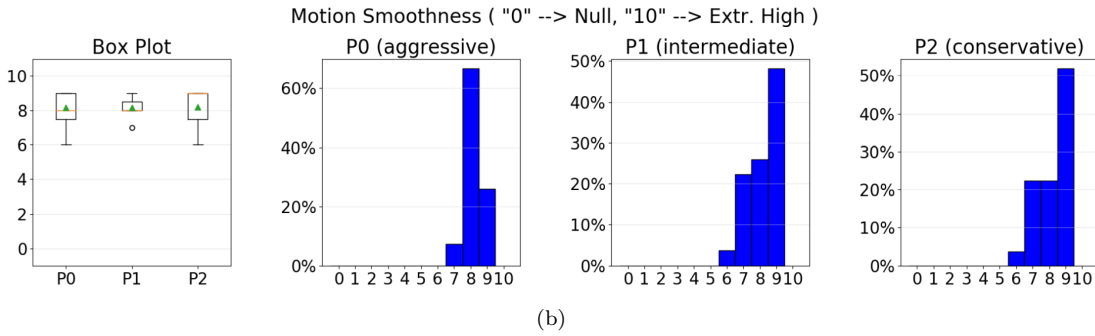
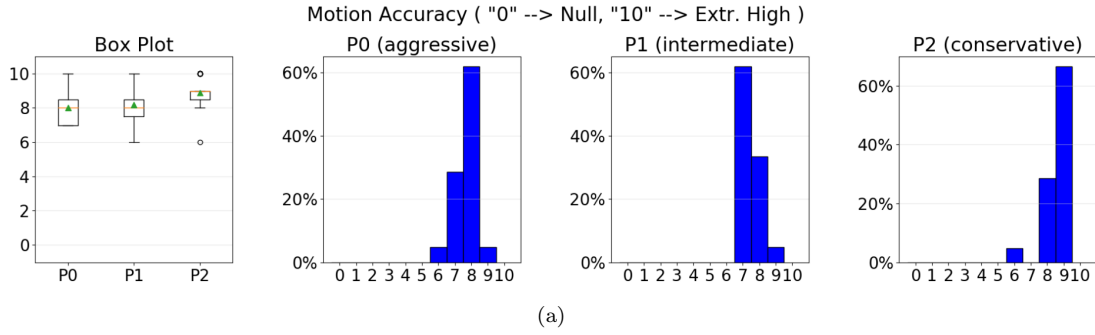


Figure 3.20: Evaluation of Motion Indicators for each Parameter Set. For each indicator, the box-whiskers plot on the left shows the approximated distribution of the rates assigned by the testers grouped by the active parameter set, by highlighting the median (solid orange line) and the mean (green triangle). Then, the histograms display the normalized frequency of the assigned rates versus the active parameter set.

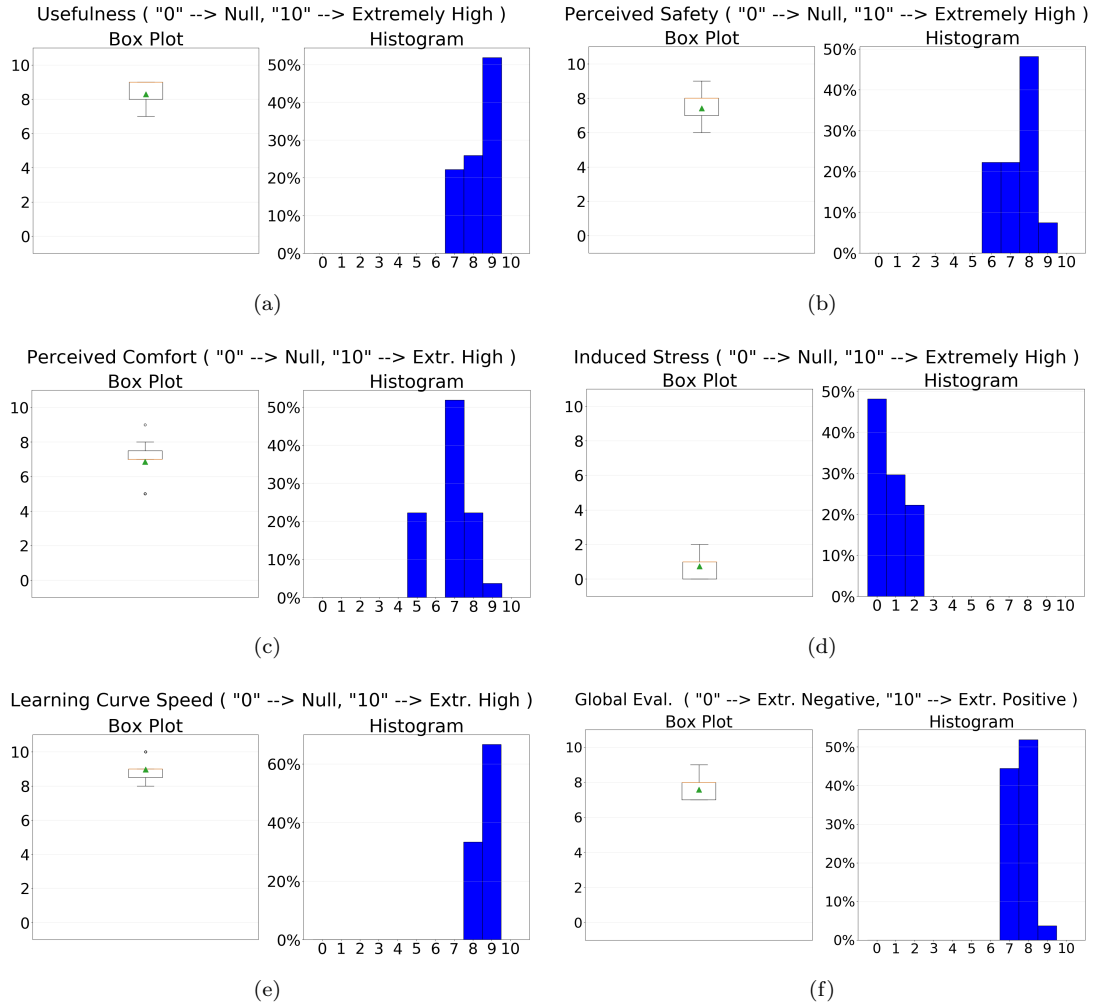


Figure 3.21: Evaluation of general indicators. For each indicator, the box-whiskers plot on the left shows the approximated distribution of the rates assigned by the testers, by highlighting the median (solid orange line) and the mean (green triangle). Then, the histogram on the right displays the normalized frequency of the assigned rates.

a broader experimental validation involving different tasks, vendors and robot kinematics. In addition, another development might be represented by the identification of a suitable kinematic redundancy resolution strategy, that might allow the controller to move the auxiliary turning table automatically with respect to the robot motion enforced by the human operator. Finally, we will integrate a tracking system to detect and monitor in real-time the current position of each link of the human operator, in order to obtain very accurate occupancy volumes.

Chapter 4

Singularity Avoidance Implementation on Walk-through Programming for Industrial Applications

4.1 Introduction

The integration of industrial robotic manipulators into the companies production processes had made an high degree of automation and productivity improvement. The programming of a robot required the specilization of many industrial operators, thus the robotics simplified heavy and repetitive tasks while generating new professional roles requests.

The motion control of a robot can be done on two main control spaces: Joint and Operational spaces. Considering that a manipulator can be schematically represented as a kinematic chain of rigid bodies connected by means of revolute or prismatic joints, the joint space denotes the space in which the $(n \times 1)$ vector of joint variables

$$q = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix} \quad (4.1)$$

is defined. On the other hand, the direct kinematics equation of a manipulator allows the position and orientation of the end-effector frame to be expressed as a function of the joint variables with respect to the base frame. The position can be given by a minimal number of coordinates with regard to the geometry of the structure, and the orientation can be specified in terms of a minimal representation (Euler angles) describing the rotation of the end-effector frame with respect to the base frame. Thus, it is possible to describe the end-effector pose by means of the $(m \times 1)$

vector, with $m \leq n$,

$$x_e = \begin{bmatrix} p_e \\ \phi_e \end{bmatrix} \quad (4.2)$$

where p_e describes the end-effector position and ϕ_e its orientation.

On the motion planning point of view, the motion of the robot on the operational space is more predictable than in the joint space, e.g. linear motion or circular motion perform remarkable trajectory on the Cartesian space. Indeed, when a robot is programmed to follow a precision path, only the motion on the operational space is possible. Unfortunately, this kind of control is affected by two main robotic issues:

- Joint limits: the joints of the kinematic chain, or the reducers and motor of the robot, usually have finite limits. In the main cases, a joint can compute more than one revolution, but for specific kinematics or mechanical design, sometimes this range could be small.
- Singularities configuration

Taking an acceptable starting and finish joint configuration for a motion on operational space, the joint limits issue can be avoided. On the other hand, the singularity issue is more complex to figure out. The differential kinematics represent the relationship between the joint velocities and the corresponding end-effector linear and angular velocity 4.3. This mapping is described by a matrix, termed geometric or analytical Jacobian, which depends on the manipulator configuration.

$$v_e = J(q)\dot{q} \quad (4.3)$$

The Jacobian is useful for finding singularities, analyzing redundancy, determining inverse kinematics algorithms, describing the mapping between forces applied to the end-effector and resulting torques at the joints (statics) and deriving dynamic equations of motion and designing operational space control schemes [83]. The kinematics singularities of a serial manipulator can be identified solving the problem:

$$\det(J) = 0 \quad (4.4)$$

Moreover, the *manipulability measure* function 4.5, with Yoshikawa's manipulability index [105], is a quality measure for redundant manipulators, which describes the distance to singular configuration. In particular, manipulability of a robot is the capacity to change the end effector's position as a function of the joint configuration. A larger manipulability measure indicates a greater range of possible motions at that configuration. The manipulability measure reduces to zero when the robot is in a singular configuration and cannot generate velocities in at least one direction.

$$\omega(q) = \sqrt{\det(J(q)J^T(q))} \quad (4.5)$$

This approach is based on analyzing the manipulability ellipsoid 4.6 that is spanned by the singular vectors of the Jacobian.

$$v_e^t (J(q)J^T(q))^{-1} v_e = 1 \quad (4.6)$$

The manipulability ellipsoid includes all possible velocities normalized for a unit input at a given robot configuration. These velocities are calculated using the robot's velocity Jacobian. If one or more ellipsoid rays are very small, these indicate that the end-effector cannot reach the velocity in the matching directions.

Taking into account the singularities, it is important to underline that the inversion of the Jacobian can represent a serious inconvenience, e.g. in 4.7. When the robot motion is near to singularity configuration, problems arise only when commanding a joint motion $\dot{q}$ by inversion of a given Cartesian motion task v_e . In this case, the inversion resolution causes a sudden increase of displacement and velocity of one or more robot joints, despite the required Cartesian displacement is small.

$$\dot{q} = J(q)^{-1} v_e \quad (4.7)$$

A solution overcoming the problem of inverting differential kinematics in the neighbourhood of singularity is provided by the *Damped Least-squares (DLS) inverse*

$$J^* = J^T (J J^T + k^2 I)^{-1} \quad (4.8)$$

where k is a damping factor that renders the inversion better conditioned from a numeric point of view.

In chapter 3, we already discussed about a new approach to physically interact with an industrial robot. One of the main unresolved issue is the identification of a singularity avoidance strategy which allowed to overcome the singularity robot configuration, namely undesired robot behaviours or machine stops. Therefore, the aim of this chapter is to describe the implementation of singularity avoidance algorithm able to provide an alternative solution in real time.

In the section 4.3, it will be explained how the classic DLS method cannot be figured out wrist singularity problem of the Gaiotto GA-OL robot.

4.2 Related works

The research of a singularity avoidance strategy pertaining to the experimental protocol adopted in 3.7 identified two set of papers. First set, [61], [69], [12] and [59], collects four solutions to find alternative configurations for a cartesian movement, while second set, [44], [22] and [26], collect three solutions to limit the robot workspace based on the singulaty configurations.

In [61], Wenxiao Lei, et al., proposed a control method to avoid the singularity of a manipulator during a tracking motion, namely during a real-time task. The method is based on the DLS method and the relation proposed between joint and cartesian velocities is

$$\dot{q} = J^+(q)\dot{x}^b + (I_n - J^+(q)J(q))y \quad (4.9)$$

where $J^+(q)$ is the pseudo-inverse of the Jacobi matrix $J(q)$, $I_n \in \mathbb{R}^{n \times n}$ indicates an unit matrix and $y \in \mathbb{R}^n$ is an arbitrary vector.

The proposed method suggests the computation of $J^+(q)$ introducing a no-constant damping term, instead like in 4.8. Indeed, the name of the method is *Adaptive Damped Least-squares*, where k^2 is substitutes by

$$\lambda = \frac{A}{\tan(c + B)} \quad (4.10)$$

where A and B are two constant terms, experimentally evalutated, while c is

$$c = \sqrt{|JJ^T|} \quad (4.11)$$

Thus, thank to λ , the equation 4.9 changes in

$$\dot{q} = J_1^+(q)\dot{x}^b + (I_n - J_1^+(q)J(q))y \quad (4.12)$$

where $J_1^+(q)$ is

$$J_1^+ = J^T(JJ^T + \lambda I)^{-1} \quad (4.13)$$

An alternative method to solve the singularity problem is proposed by Nasr [69], where the DLS method is not taken into account. This chapter studies the singularity avoidance approach, trying to minimize the error between the end-effector pose and the joint configurations. In order to do this, two main sub-approach has been analyzed:

- First approach exploits the genetic algorithms to identify the alternative configurations with respect to singularity configuration. Through the Least-square method, two objective functions are proposed, one to minimize the error in the cartesian space and one for the joint space. Both functions must be simultaneously optimized.

- The second approach divides the configuration space into three zones: SAFE, CARE and STOP. These are used to define a zone function, which modulates the speed reduction upon approaching to the singularity.

Returning to DLS-like method, in 2021 Stefano Aldini [12], et. al, proposed an innovative strategy to the singularity avoidance theme applied to a manual guidance task (as in our case 3.3). Indeed, the control system is based on the admittance control 3.1, but the Aldini scheme is slightly different with respect to the singularity handling. In order to approach the singularity problem, the authors suggest the *Exponentially Damped Least-squared (EDLS)*. The EDLS method involves the Jacobian decomposition through *Singular Value Decomposition (SVD)* strategy:

$$J = U\Sigma V^T \quad (4.14)$$

U and V are both orthonormal matrices, while Σ is a diagonal matrix with the same dimensions as J and containing the singular values σ_i , with $i = 1, \dots, 6$ in the case of a six degrees-of-freedom (DOF). Approaching to the singularity configuration the singular values approach to zero, resulting in a Jacobian matrix that degenerates when inverted. Indeed, as in 4.9, the Jacobian pseudo-inverse is important to relates joint and cartesian velocities.

Differently from 4.13, the inverse of the Jacobian matrix is obtained by SVD methods as

$$J^* = V\Sigma^*U^T \quad (4.15)$$

and the new singular values $\Sigma_{ii}^* = \frac{s_i}{\sigma_i}$, and s_i is equal to:

$$s_i = \begin{cases} 1 - \beta^{\frac{\sigma_i - \bar{\sigma}_0}{\bar{\sigma}_1 - \bar{\sigma}_0}}, & \text{if } \sigma_i \geq \bar{\sigma}_0 \\ 0, & \text{otherwise} \end{cases} \quad (4.16)$$

$\bar{\sigma}_0$ and $\bar{\sigma}_1$ are the damping parameters, their difference defines the damping rate and β is a constant small value. The values of $\bar{\sigma}_0$ and $\bar{\sigma}_1$ can be different, depending on whether the robot is approaching a singularity or moving away from it. Aldini proposed two modes of these parameters and each mode, depending if robot is moving away or towards the singularity, has two set of $\bar{\sigma}_0$ and $\bar{\sigma}_1$.

Finally, for the first set of papers, Kurtz [59], proposed a passivity-based control implemented to solve a tracking reference system problem. The control is defined by an optimization algorithm of a quadratic problem, where the manipulator dynamic is considered. In order to define the main optimization bound, as in chapter 2, the Control Barrier Functions (CBF) are used.

$$\begin{aligned}
& \underset{\tau, \ddot{q}, u_r}{\text{minimize}} && \omega_1 \|u_r - u_r^{nom}\|^2 + \omega_2 \|\bar{J}^T \tau - f^{des}\|^2 \\
& \text{subject to} && M\ddot{q} + C\dot{q} + \tau_g = \tau, \\
& && \dot{V}(\ddot{q}, u_r) \leq 0, \\
& && J_\mu \ddot{q} \geq b
\end{aligned} \tag{4.17}$$

where $b = -\dot{J}_\mu \dot{q} - \alpha_1(\mu(q) - \epsilon) - \alpha_2 J_\mu \dot{q}$

This optimization, which is a convex QP, ensures that both passivity ([70]) and singularity avoidance hold for any solution. First bound considers the dynamics of the system, second the passivity and the last, according to Exponential Control Barrier Functions (ECBF), the singularity avoidance bound. Given that the singularity avoidance constraint and the passivity constraint may be in conflict, leading to an infeasible QP, an input to the reference system u_r is introduced. The scalar positive weights ω_1 and ω_2 regulate the relative priorities of tracking the nominal input to the reference system and applying the standard Passivity-Based controller, namely $\|\bar{J}^T \tau - f^{des}\|^2$.

The second set of the research provided a collection of strategies where, in order to avoid the singularity configuration, the workspace is limited. JiWoong Han [44], suggests an hardware and software architecture for a teleoperation robotic task, where the force feedback provided by the teleoperation device is used as repulsive mechanism to avoid the singularity. In particular, in order to provide a force feedback, the manipulability ellipsoid 4.6 is applied to the teleoperation mechanism. So, the force feedback considers the motion direction of the robot and the teleoperation device. Then, Carmichael, in [22], proposes an other repulsive mechanism to the singularity given by the potential fields. Moreover, given that the repulsive forces add energy into the human-robot system, in order to guarantee the safety, these forces must be limited. Indeed, Dimeas, in [26], with respect Carmichael, inserts a boundary condition to the repulsive forces, which guarantee the system passivity.

4.3 Problem Statement

This chapter aims to find and implement a solution of the singularity for the robot GA-OL, during the walk-through programming task 3. In order to do this, first a singularity analysis for GA-OL robot is done. As described in 4.2, the kinematics singularities of a manipulator can be found by 4.4. In particular, the singularities can be classified as

- Boundary singularities: when the robot is completely extended or crouch.
- Internal singularities: inside the robot workspace and, usually, caused by two or more motion axis alignment.

The common serial manipulator singularities can be summarized in:

- Wrist singularity

- Elbow singularity
- Shoulder singularity

In order to comprehend the common singularity configurations, the figure 4.1 will be taken into account.

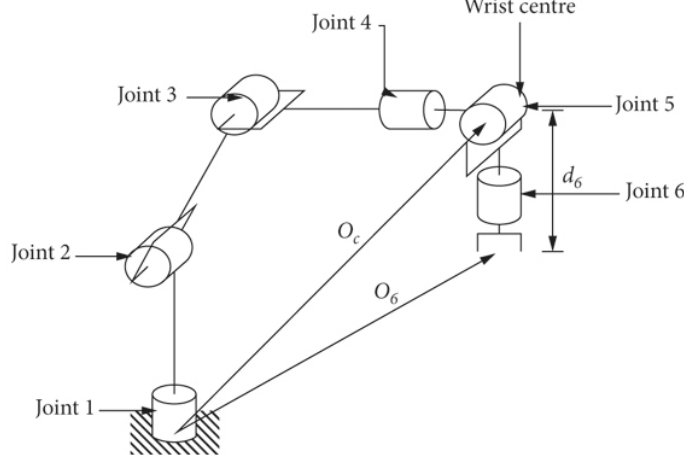


Figure 4.1: Example of 6 DOF Articulated arm

The wrist singularities are the most common singularities for anthropomorphic robot floor mounted. In particular, these are recorded when the axis of joint four and six become coincident. Then, the elbow singularity occurs when the wrist center O_c , namely the intersection point of the axis joints four, five and six, lies on the plane which passing through the joint axis two and three. Finally, the shoulder singularity is recorded when the wrist center O_c lies on the plane which passing through the axis joint one and two.

Now, according to the GA-OL kinematics chain, the three common singularities are analyzed. The wrist singularity configuration is recorded when the joint value of the joint five is $\pm 90^\circ$.

The joint configuration represented in 4.2 is $q_1 = 0.0^\circ$, $q_2 = 0.0^\circ$, $q_3 = 0.0^\circ$, $q_4 = 0.0^\circ$, $q_5 = 90^\circ$ and $q_6 = 0.0^\circ$. The analytical jacobian for this configuration is

$$J = \begin{bmatrix} -0.2850 & 0.8450 & -0.1550 & -0.1550 & -0.1400 & 0 \\ 1.6120 & -0.0000 & 0.0000 & 0.0000 & 0.0000 & 0 \\ -0.0000 & -1.6120 & -1.6120 & -0.0000 & -0.0000 & 0 \\ 0 & 0.0000 & 0.0000 & 0.0000 & -0.0000 & 0.0000 \\ 0 & 1.0000 & 1.0000 & 1.0000 & 0.0000 & 1.0000 \\ 1.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0000 \end{bmatrix} \quad (4.18)$$

While, the analytical jacobian, changing the value of $q_5 = -90^\circ$ is

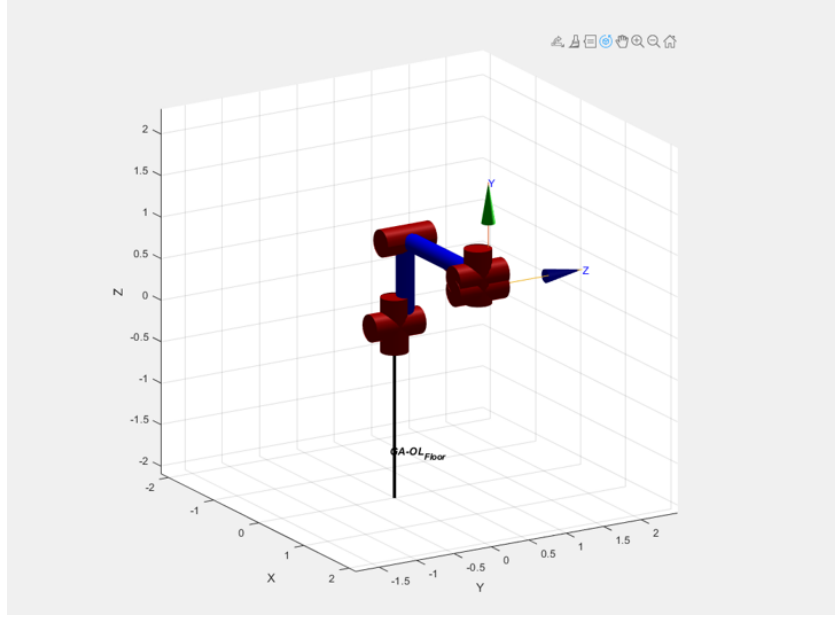


Figure 4.2: GA-OL simplified representation with joint five value at 90°

$$J = \begin{bmatrix} -0.0050 & 0.8450 & -0.1550 & -0.1550 & 0.1400 & 0 \\ 1.6120 & -0.0000 & 0.0000 & 0.0000 & 0.0000 & 0 \\ 0 & -1.6120 & -1.6120 & -0.0000 & 0.0000 & 0 \\ 0 & 0.0000 & 0.0000 & 0.0000 & -0.0000 & 0.0000 \\ 0 & 1.0000 & 1.0000 & 1.0000 & 0.0000 & 1.0000 \\ 1.0000 & 0.0000 & 0.0000 & 0.0000 & 1.0000 & 0.0000 \end{bmatrix} \quad (4.19)$$

In both cases, it is worth noting that, for each matrix, there is one linearly dependent joint, namely q_6 depend on q_1, q_2, q_3, q_4, q_5 , and the rank of the matrices is 5. Thus, the determinant of the Jacobians tend to 0. Moreover, the fourth line is always composed by zero elements. This because, when the robot is in singularity configuration, the rotation of the wrist around base frame x-axis is blocked.

For the elbow singularity, GA-OL kinematics has this kind of singularity when the joints axis configuration are: $q_1 = 0.0^\circ, q_2 = 0.0^\circ, q_3 = \pm 90^\circ, q_4 = 0.0^\circ, q_5 = 0^\circ$ and $q_6 = 0.0^\circ$. Given that the mechanical and software joint limits are more stringent, this singularity case shouldn't happen. Even for the shoulder singularity, the joint limits should avoid it, but a possible joint configuration for shoulder singularity is: $q_1 = 0.0^\circ, q_2 = -30.0^\circ, q_3 = -47.5^\circ, q_4 = 0.0^\circ, q_5 = 0^\circ$ and $q_6 = 0.0^\circ$.

Thus, the wrist singularity is the only singularity configuration that can causes problems to the walk-through robot programming in the operational space 3.1. Delving further the wrist singularity of GA-OL, a manipulability analysis has been done. According to Yoshikawa [105],

the manipulability measure 4.5 can be written as:

$$\omega(q) = \omega_A(q) \times \omega_W(q) \quad (4.20)$$

where $\omega_A(q)$ and $\omega_W(q)$ respectively represents the translational and rotational manipulability measure. Hence, separately computing the translational and rotational manipulability measures for the GA-OL wrist singularity configuration, it denotes that only the translational value is different from 0.

$$\omega_A(q) = 2.70954, \omega_W(q) \rightarrow 0 \quad (4.21)$$

Indeed, computing the manipulability ellipsoid with only the translational part of the Jacobian, the result is 4.3, while with the rotational part of the Jacobian the ellipsoid cannot be drawn.

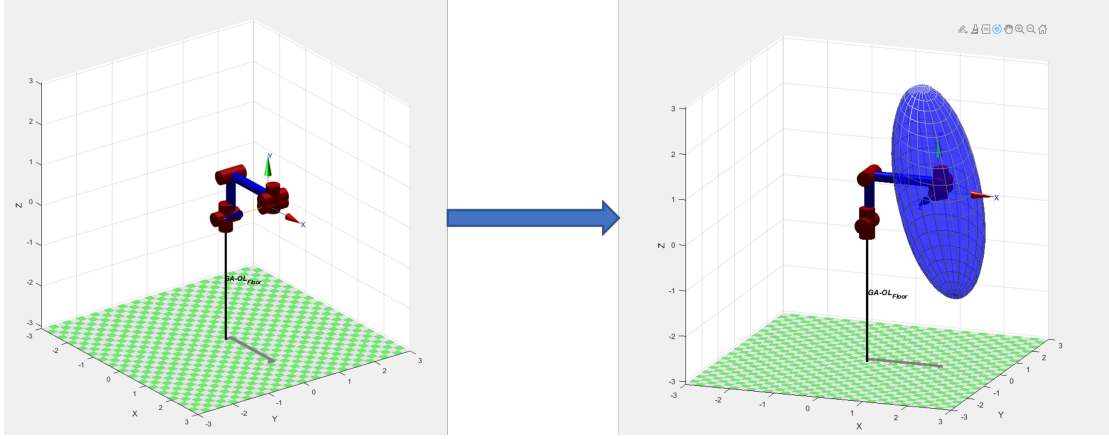


Figure 4.3: Translational manipulability ellipsoid for wrist singularity configuration with $\theta_5 = 90^\circ$

Finally, considering the results of the problem statement, the state of the art on the related works section 4.2 and the use-case affinity treated on the paper of Aldini [12], in this chapter the implementation of the singularity avoidance strategy, proposed by him, into the Gaiotto GA-OL system environment and software architecture 3.8 will be analyzed and validated. Moreover, a simple experiment while robot passing through the wrist singularity configuration, with related results, is described in section 4.5.

4.4 Implementation of E-DLS

The walk-through robot programming foresaw the computation of the target cartesian pose, provided by the C++ Module *ToolComp* and *AdmAdapt* of 3.8. After that, the target pose is used for moving a variable reference system, which the robotics control of GA-OL tries to follow. Of course, this kind of control is affected by singularity problem and the previous singularity avoidance control only limited the range of q_5 .

In order to implement the singularity avoidance strategy with *Exponentially Damped Least-square*

method, the control level of the robotic control must change.

Given the equations 4.7 and 4.8, the inverse kinematics results provide the joint velocities starting from cartesian end-effector velocities. Hence, taking into account the overall control architecture of walk-through programming method in 3.1, the output of the admittance control block is $\dot{x}_{ref}$. Moreover, as said before, the Low-level Position Control modified the pose a variable reference system and the robot follow it. But, considering that the output of EDLS method, namely of the Singularity Avoidance block, is the joint velocities $\dot{q}_{ref}$, the Low-level Position Control must directly modify the target joint positions q_{ref} , as in 4.4; q_{ref} is the time integration of $\dot{q}_{ref}$.

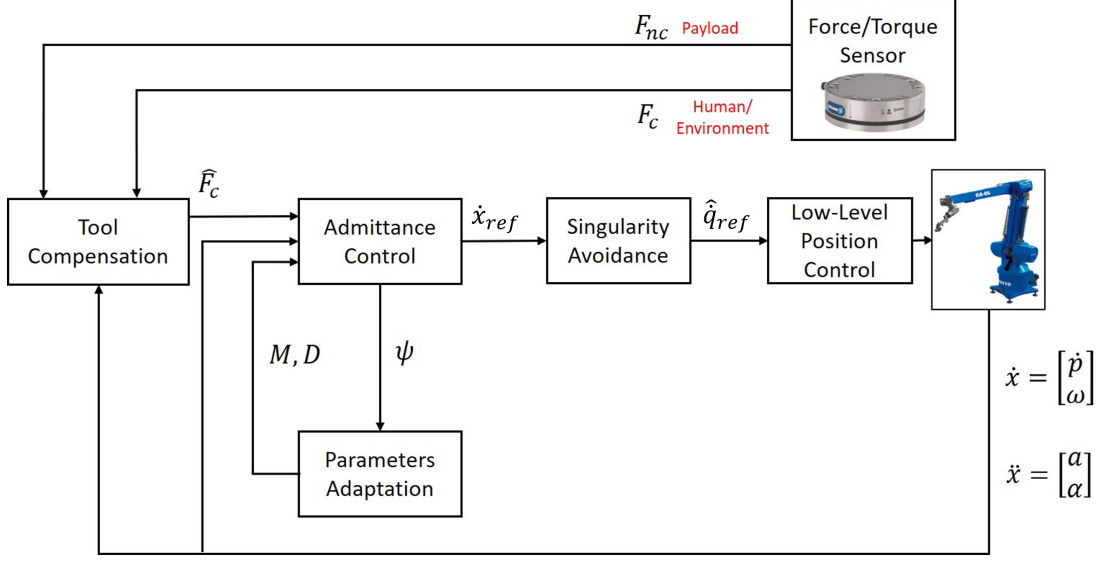


Figure 4.4: Overall control architecture with EDLS singularity avoidance strategy

The software implementation of the Singularity avoidance strategy only modifies the Task *singularity avoidance* in 3.8, while the task of the tracking reference system has been removed. The logical algorithm of the singularity avoidance strategy is described in the algorithm 6.

At each time step, where admittance algorithm provides a new cartesian position, or velocity, setpoint, the actual value of the robot Jacobian matrix is computed. Indeed, from the Jacobian matrix, the singular value decomposition method 4.14 provides the components U , σ and V (line 4). Then, before to compute new singular values, the damping parameter are based on the robot moves toward or away the singularity (line 6); the determinants of the Jacobian at the actual and previous instants are compared.

In line 18, the new singular values are computed as in 4.16 and, in line 20, the Σ^* matrix is composed inserting the new singular values on their diagonal. Finally, in order to obtain the modified and optimized Jacobian matrix, in line 21 the inverse method of SVD is applied.

Once the damped Jacobian is obtained, the joint velocities are calculated as in 4.7. Moreover, with the time integration of $\dot{q}$, q is the new joint position setpoint to apply to the robot joint. Indeed, taking figure 3.8, the Task *reference tracking* is substituted by the task which command the new setpoint joint positions.

Algorithm 6 EDLS Singularity Avoidance

```
1: Parameters:  $\det J, \det J_{prev}, J, \sigma, \sigma^*, s, U, \Sigma, V, \sigma_0, \sigma_1$ 
2: procedure EDLS(  $J$  )
3:   for all  $J_i$  do
4:      $(U, \sigma, V) \leftarrow \text{SVD}(J)$ 
5:      $\det J = \det(J)$ 
6:     if  $\det J \leq \det J_{prev}$  then
7:        $\sigma_0 = \sigma_0^b$  and  $\sigma_1 = \sigma_1^b$ 
8:     else
9:        $\sigma_0 = \sigma_0^a$  and  $\sigma_1 = \sigma_1^a$ 
10:    end if
11:     $\det J_{prev} = \det J$ 
12:    for all  $\sigma_i$  do
13:      if  $\sigma_i \geq \sigma_0$  then
14:         $s_i = 1 - \beta^{\frac{\sigma_i - \sigma_0}{\sigma_1 - \sigma_0}}$ 
15:      else
16:         $s_i = 0$ 
17:      end if
18:       $\sigma_i^* = \frac{s_i}{\sigma_i}$ 
19:    end for
20:     $\Sigma^* \leftarrow \text{Diag}(\sigma_i^*)$ 
21:     $J^* \leftarrow \text{InverseSVD}(V, \Sigma^*, U^T)$ 
22:  end for
23: end procedure
```

4.5 Experiments and Results

The complexity to implement a singularity avoidance strategy into an industrial application as a walk-through robot programming is to provide a feasible solution at each robot cycle time. For GA-OL, combined with the walk-through programming method, the experimental robot setup is similar to 3.7. Remembering the section 4.3, the most common singularity configuration for GA-OL kinematics is the wrist singularity represented in 4.2, where the joint five assume the values of $\pm 90^\circ$. On this way, the configured experiments foresee the execution a trajectory where the robot passing through the wrist singularity (figure 4.5). The walk-through program starts in wrist configuration, as in 4.5(a), where the singularity is far. Then, principally moving the wrist with respect the robot vertical axis, the joint axis five assume a value of 90° (figure 4.5(b)). Finally, the goal is to pass through the singular condition and to assume a value of joint axis five $> 90^\circ$, as in 4.5(c).

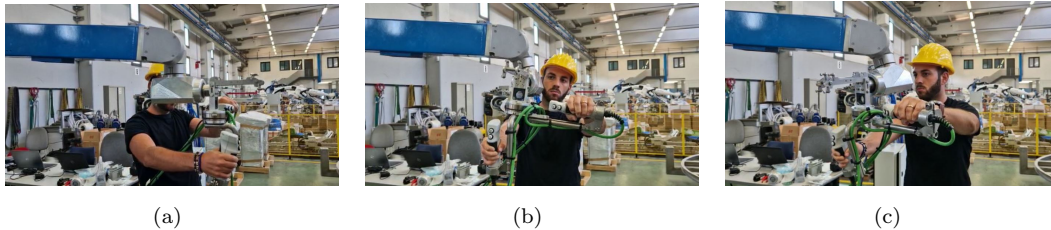


Figure 4.5: Robot trajectory passed through the wrist singularity

The experiment configured in (figure 4.5) has been executed for two sets of damping parameter of EDLS, represented in table 4.1 and 4.2, as in [12].

	$\bar{\sigma}_1$	$\bar{\sigma}_0$
toward sing.	0.35	0.25
away sing.	0.25	0.15

Table 4.1: Fast Asymmetric damping parameters

	$\bar{\sigma}_1$	$\bar{\sigma}_0$
toward sing.	0.45	0.25
away sing.	0.45	0.25

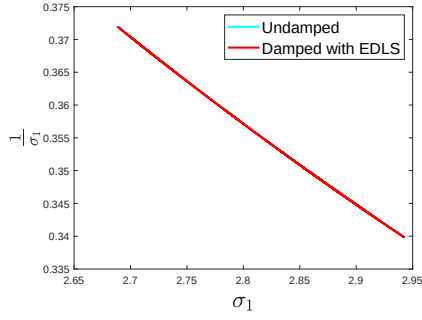
Table 4.2: Slow Symmetric damping parameters

For the first experiment, the Fast Asymmetric damping parameters are used. In figure 4.6, the results of the first experiment are shown. For each graph, the singular value, extract from the Jacobian matrix with SVD method 4.14, is compared with its reciprocal. Moreover, always for each graph, the reciprocal of the singular value optimized with the EDLS method 4.16 is compared. Considering all six subgraphs of figures 4.6 and 4.7, the singular values σ_i are represented in descending order.

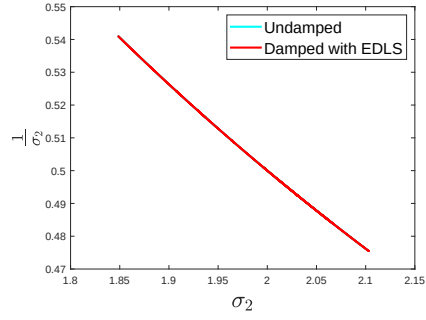
From σ_1 to σ_5 , namely in 4.6(a), 4.6(b), 4.6(c), 4.6(d) and 4.6(e), there are no differences between the normal reciprocal and the optimized reciprocal. This because, taking the equation 4.16, the i -th singular value extracted from the Jacobian matrix is always lower the the damping parameter 0, namely $\sigma_i < \bar{\sigma}_0$.

Instead, focusing on the last singular value 4.6(f), it is worth of notice how the behaviour of the reciprocal singular value with respect the same optimized is difference. For the Undamped case, namely cyan line, when the robot approaches to the singularity, the reciprocal tend to infinity while, for the Damped case with EDLS method, namely red line, the reciprocal singular value optimized diverges from the undamped line and damping then value to 0. The motivation of only σ_6 has two difference behaviour between its reciprocal and the same optimized is correleated to the Jacobian analysis in 4.3. Indeed, retriving the Jacobian matrix for the case where $q_5 = 90^\circ$ 4.18, it's worth of notice that there is one linearly dependent joint, namely q_6 depend on q_1, q_2, q_3, q_4, q_5 ; this dependency indicates joint axes that are aligned and causes singularity.

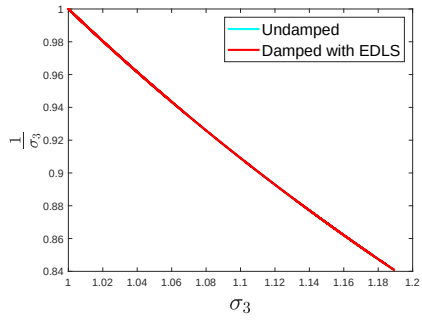
Then, in figure 4.7, the results of the same experiment with Slow symmetric damping parameters are shown. As in 4.6, only σ_6 presents two different behaviour between undamped and damped reciprocal 4.7(f). Thus, it's worth to compare the behaviours of the damped reciprocal singular value with Fast Asymmetric and Slow Symmetric damping parameters 4.8. The Fast Asymmetric parameters tries to mantein the undamped path as possible (depends on the values of $\bar{\sigma}_0$ and $\bar{\sigma}_1$). When the lower limit is very close, the slope of the curve is very steep, namely the dynamics is *fast*. On the same graph, the Slow Symmetric case is figured. In this case, the damping ratio, namely the slope, is softer then Fast Asymmetric, indeed the line blue diverges from the cyan



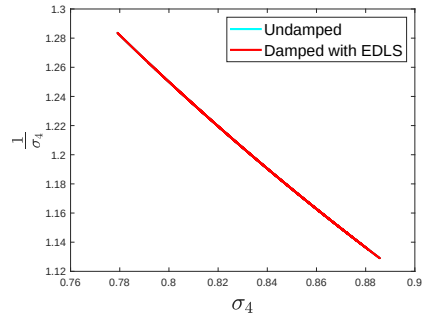
(a)



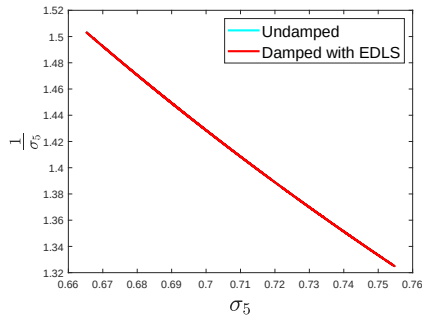
(b)



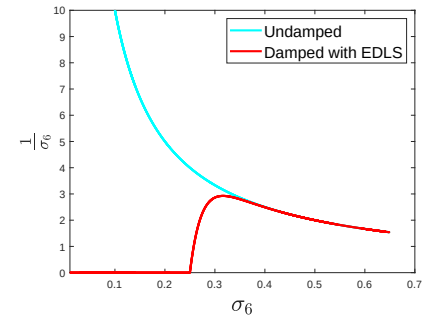
(c)



(d)

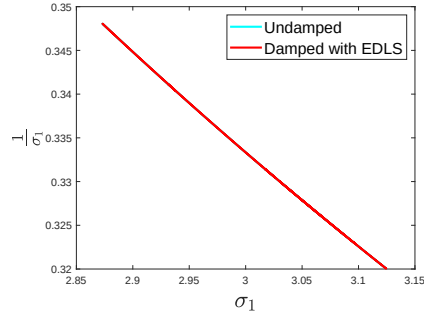


(e)

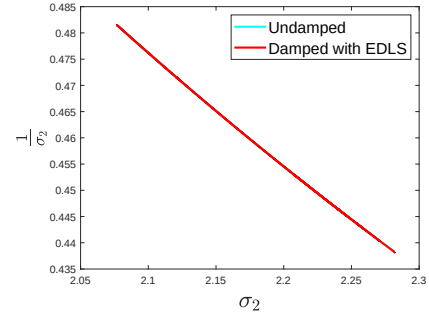


(f)

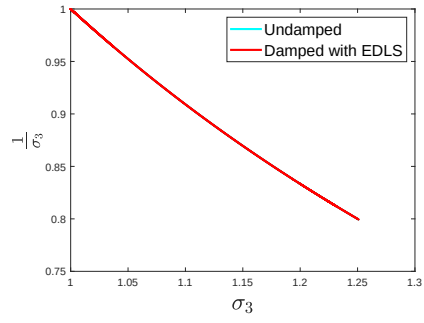
Figure 4.6: Singular values and Reciprocals with Fast Asymmetric damping parameters



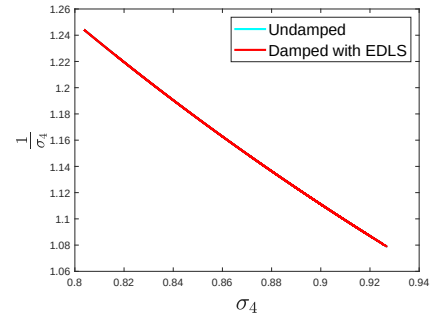
(a)



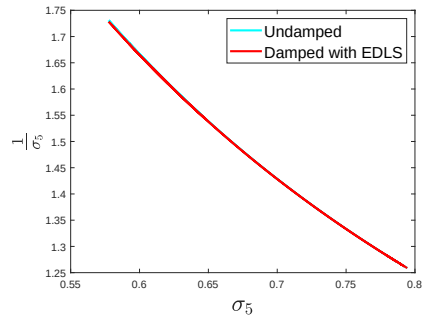
(b)



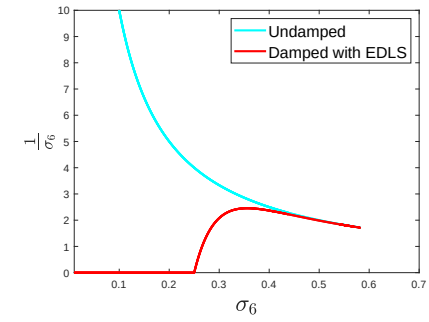
(c)



(d)



(e)



(f)

Figure 4.7: Singular values and Reciprocals with Slow symmetric damping parameters

line earlier than red line.

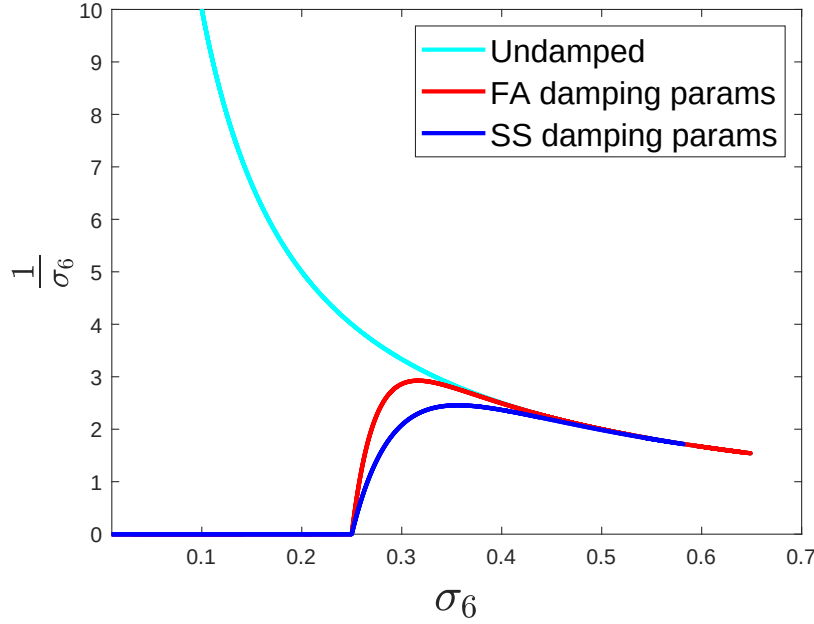


Figure 4.8: σ_6 singular value reciprocal optimized with EDLS computed with Fast Asymmetric (red line) and Slow Symmetric (blue line) parameters

Finally, for the same experiments executed with Fast Asymmetric and Slow Symmetric parameters, the Wrist joint position with respect to a singularity distance function are presented in figure 4.9 and 4.10. The singularity distance function, used to see how far we are from the robot singularity, is equation 4.5. In the graphs, the singularity distance function, later called $\omega(q)$, decreases since the q_5 , namely the value of the joint axis five, approaches to the value -90° . Moreover, focusing on the yellow patches, $\omega(q)$ assumes, for an instance, the value 0 until the axis five exceeds the singularity configuration. In the same region, the behaviour of q_4 and q_6 is so that the motion directions of these axis are blocked until $\sigma_6 < \bar{\sigma}_0$.

4.6 Conclusion

The wrist singularity is a configuration could be reached during the robot programming through hand-guiding mode. In chapter 3, the singularity avoidance strategy foresaw the limitation of the range of q_5 , so to maintain far from the singularity configuration. For industrial application use cases, for example a painting robot program recorded with walk-through programming method, the possibility to overstep the wrist singularity configuration, analyzed in 4.3, is very high. After an accurate state of the art research and a singularity analysis for robot GA-OL, the singularity avoidance strategy proposed by Aldini [12] had been chosen to implement. Then, once the implementation had been done, a set of experiments had been proposed to validate the algorithm. The results 4.5 showed how the wrist singularity configuration can be easily overcome. Moreover, choosing empirically the damping parameters σ_0 and σ_1 , it is possible to change the behaviour of the robot near the singularity configuration.

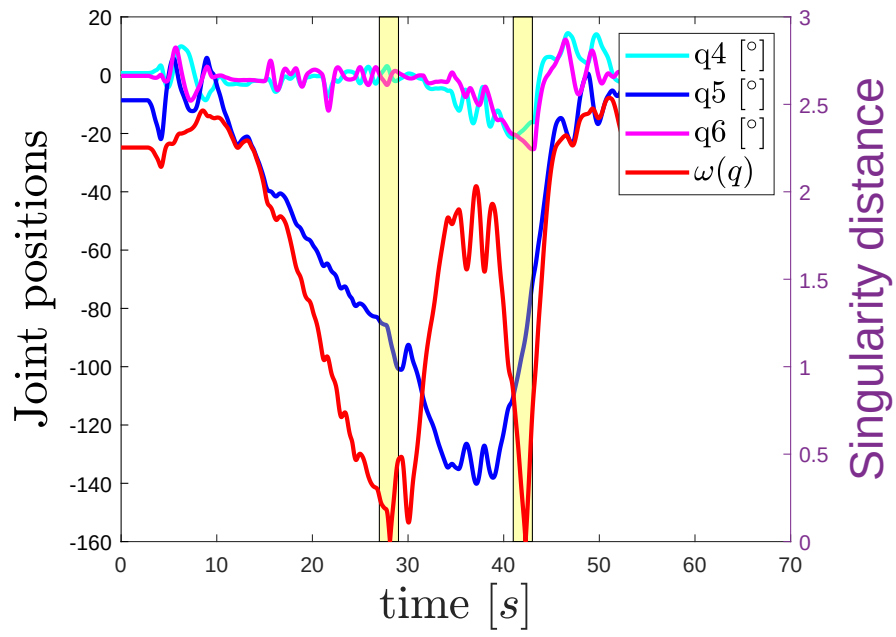


Figure 4.9: Wrist joint positions and Singularity distance function with Fast Asymmetric damping parameters

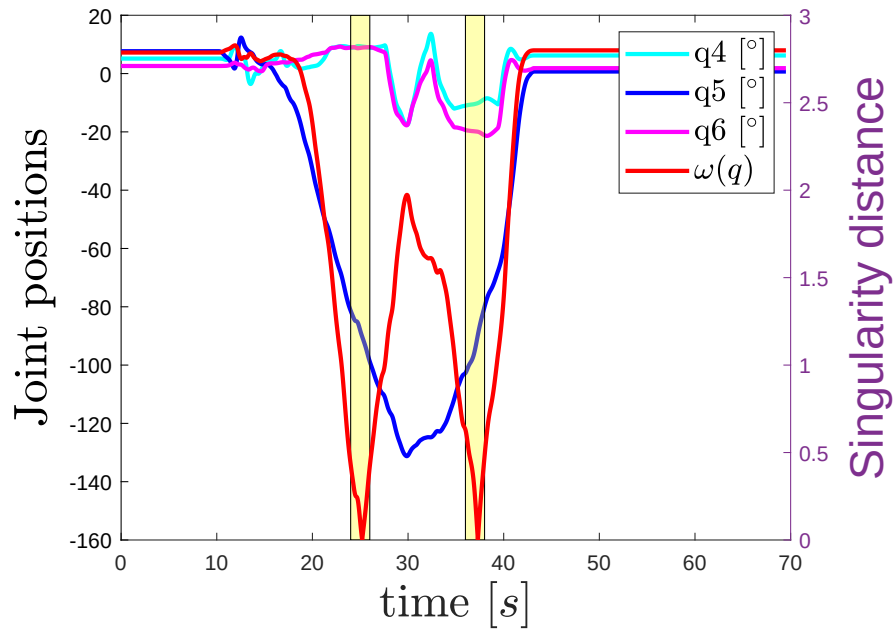


Figure 4.10: Wrist joint positions and Singularity distance function with Slow Symmetric damping parameters

Chapter 5

A no-physical interaction human-robot coexistence situation integrated into a Robotic System for Automated Picking of Urban Waste

The collaborative robotics, according with relative standards, regulated the physical interaction or shared workspace between human and robot with the aim to guarantee the operator safety. However, even a no-physical interaction between these two actors cannot be neglected. As describes in the introduction chapter 1, new technologies and safety devices allowed to break the physical barrier which isolated robot in closed and limited workspace.

The operator can interact with a machine even with remote control systems, like teach pendant. In this case, human commands actions or missions to the robot and the robot must complete them. Usually, this actions allowed to lighten the human work and improve the work style. Even if a physical interaction not happens, a safety system which prevents damage to the operator or to the environment must designed.

In this chapter, we will analyze the design and implementation of a robotic system where human and robot coexists in the same system. Operator commands mission to the robot and, while the robot moves into the environment the operator must supervise the surroundin space; so, human and robot actions are simultaneously computed. Moreover, in order to guaratee the operator and environment safety, the safety system design and implementation will be treated.

5.1 Introduction

The urban waste management and handling is an important theme that actually raises awareness into the world wide population since it is a key concept for environmental sustainability. Waste

collection is the first step into the waste management process and it consists in picking the waste and transferring it from the point of grouping to the point of treatment. The citizens must provide the waste already separated into different classes of waste (e.g. plastic, paper, aluminium, glass, vegetable waste, organic and undifferentiated), since this is an essential requirement for sustainable waste management [18]. Then, the waste management can be performed with different modalities, depending on the agreement between the local municipality and the waste management companies, and with different collection systems (e.g. curbside bins, pneumatic systems, door-to-door) [21]. In particular, door-to-door refers to the collection system where citizens insert the waste into dedicated personal bins or plastic bags and place the containers outside of their house, typically close to the street. The bins could be of different sizes depending on their capacity, ranging typically from 23-25 liters to 120-240-360 liters. Waste collection activities are strongly associated with a variety of physical, chemical and biological hazards for the waste collecting operators, with very frequent occupational accidents [72]. Non-fatal injuries in waste collection are mainly work-related musculoskeletal disorders due to the manual material handling of waste containers. Indeed, waste collection requires the operator to exit the vehicle and manually handle the containers (e.g. lifting, lowering, pushing, pulling, etc.) [21]. The ergonomics analysis provided in [21] revealed that waste collecting operators experience ergonomic issues during door-to-door waste collection related to a poor design of waste collection strategies, in terms of organization, operations, technology and equipment. Moreover, before the collection, there is a potential growth of microorganisms in the waste, which can lead to an exposure via inhalation to chemical and biological hazards for the operators (e.g. exposure to bioaerosols) [63].

5.1.1 State of the Art

As described in 5.1, most waste collection is manually made. Ecological operators drives the truck through the city streets and manually moves the bins (*e.g. or the waste bags*) towards the self-integrated empty system (if present) into the truck. Otherwise, especially for small bin, the waste are completely manually moved.

Industrial solutions

In order to limit the operator manual movements, several industrial solutions had been realized. Farid [2] realized a lateral automatic waste collection system integrated on a small truck. The operator drives the truck on the right side of the cabin and, once a bin is met, the lateral gripping system can be remotely actuated. The system is designed to grip several size of wheeled bins and mounts an hydraulic gripping system that hugs the bins during the emptying. When the operator thinks to be in a good truck position, the system can be manually actuated inside or outside the cabin. Of course, a good precision by the operator is required before to start the gripping system. Taking the Farid system as a reference, several generic disadvantages can be highlighted:

- The operator must be skilled to precisely stop the truck in the correct position
- Small bins cannot be collected
- The gripping system is manually actuated
- Obstacles, like vehicles or plants, cannot be avoided
- The gripping system increases the vehicle fuel consumption.
- Bin sets or arrays cannot be completely emptied. If bins are too close, the gripper cannot hug the bin without colliding with other bins. Then, if more than one line is present, since in the second line the bins cannot be emptied.
- Fallen bins cannot be taken from gripper
- Only wheeled bins can be emptied
- Waste weight cannot be calculated

Parallel, in order to really automatize the waste collecting procedures, other companies try to design more innovative solutions. Nord Engineering and ESA (Ecologia Soluzione Ambiente) provide respectively, [5] and [1], two semi-automatic systems (*Easy*), which manually commands are required but works with a different waste management system. In the first case, waste are collected into single big underground containers. When the truck reaches the correct position, a 3 degrees of freedom system can be actuated to grip the container. Given that the containers are placed underground, a new mechanic gripping system is designed to improve the empty phase. Moreover, an automatic compacting system and dynamic weighing system are integrated into the application. However, the Cartesian system integrates the base rotation only when the container must be open, thus it cannot be used to increase the reachability of the robot. In the second case, a vision system (composed by 7 cameras) is integrated. Even this system provides new big waste containers, but simply grounded. Similarly to Easy, a new mechanical gripping system had been studied to pick the container from the top of it. In both cases, the semi-automatic systems cannot be used to manage the "door-to-door" waste collection system. The industrial innovation, however, is not only focused on waste collection on the urban streets. If a previous differentiated collection from the user is done, then the waste recycling is easier than none. Otherwise, a mixture of different kind of waste would make it complicated a good waste recycling. But, even if a differentiation of the waste is done, there is a good probability that had not been done correctly. For these reasons, autonomous recycling system had been designed to correctly sort the waste. [8], [7] and [6] provide autonomous system to automatically sort the waste over a conveyor belt and exploiting the artificial intelligence to recognize and classify the objects.

Scientific research

Automatic waste collection, correlated with the scientific and industrial innovation on autonomous vehicle guidance, is a theme little treated on industrial research. An automated waste collecting

robot was introduced using SSDLite model and Faster RCNN for image detection. The robot movement are planned based on the waste material location and localization. First the mobile robot moves toward the waste material recognized and, then, in order to empty the waste, robotic arm engage the goal [66]. One of the best study topics correlated to automatic waste collection, especially for automatic urban waste picking, is the object pose estimation on outdoor environment. 3D vision sensor technology struggle to have good performance on outdoor environment, while 2D vision sensor, namely RGB cameras, doesn't provides the third component information. Thus, this should be estimated. A complex and articulated vision system, mounted on the top of a car, detects 3D point on moving objects, combining the information deriving from a full 360 °view 2D camera sensor with distance measurement of three laser scanner and GPS antenna [94]. Thanks to the combination of the photometric consistency with the prior information of the road, this system could map the 3D information of the environment during the car motion. Moreover, the topic of the outdoor vision system for estimates the objects poses interests the mobile robot navigation. In [56], an outdoor monocular visual odometry system, combined with a pre-trained depth estimation network and semantic segmentation network, is proposed to reduce the localization error of the examined mobile robot. Starting from an RGB-D image, the artificial intelligence systems extract the depth map and semantic information, and combining them with the robot tracking system, a pose graph optimization is produced. Guillaume Seguin (et al.), in [43], proposed a pose estimation methods for multiple people in stereoscopic movies. First, a semantic segmentation of the people is done, then, the poses of them are estimated. The most relevant topic of this method is that could be used in outdoor environment and combining images (video frames) acquired from different dislocated cameras.

5.1.2 Contribution and Organization of This Chapter

The huge effort to reorganize the waste management of people's habits and the complexity of reachability of several isolated locations guide the requirement to maintain the same waste containment system. In this chapter a new approach to collect the urban waste is described, considering the waste collection inside the single family bin for each common waste class, generally associated with different bin colors. As long as the autonomous drive will not be completely consolidated and integrated with the vehicles and Italian laws, the automatize system is designed to be supervised and controlled by the human. In order to reduce the work accidents and improve the productivity of the waste collection, a new robotic system to automatize the waste picking and emptying into the waste collection dedicated truck is described.

The system proposed integrates a robotics manipulator to guarantee the flexibility to the ecological operator while parking the truck before emptying the bins. An artificial intelligence vision system had been developed to solve the bad performance of 3D vision system on outdoor environment. The gripping system is designed to pick the colored waste bins, a human machine interface (HMI) is integrated on the truck cabin and, finally, a safety system is designed to guarantee the operator and system safety.

5.2 Overall Architecture of the Automated Picking System

The *Robotic System for Automated Picking of Urban Waste*, named **Robish**, is a robotic system specifically designed for automatizing the picking of urban waste, where waste is organized into dedicated recycling bins. The overall system is constituted by 5 main sub-systems, that are indicated with circled numbers in Fig. 5.1:

- (1) Vision system
- (2) Robotic system
- (3) Weather protective cover
- (4) Safety system
- (5) Human-Machine interface (HMI)



Figure 5.1: Overall structure of the Robish system for automated picking of urban waste.

In particular, once the truck is stopped next to the bins, the robot is moved in its initial position and an industrial camera mounted on the robot end-effector is used to acquire the image of the scene. The operator, remaining seated into the truck cabinet, can visualize the image into a tablet and can select through the HMI the desired bin to be picked up, by directly clicking on it. Then, the vision system automatically identifies the bin into the image and computes its position and orientation in a world reference frame. The robot moves to the target position with the desired orientation, picks and empties the bin, using a specifically designed gripper. Since the system must work outdoor in every weather condition and during the whole day, the robot and the electrical components are covered with a protective cover. The cover is designed to resist to extreme weather conditions. Finally, safety functions and devices are integrated to guarantee the

safety of the overall system. In the following sections, a detailed description of each sub-system will be provided.

5.2.1 Software architecture

In this section, the software architecture that handles the overall system is briefly introduced. In particular, as shown in Fig. 5.2, all the information and the data processed by the software modules are gathered into a specifically designed *Server*, developed in C++. The server communicates with the *HMI* through https requests and exchanges information with the other modules through shared memory. The vision system is composed by two modules: the *Image Acquisition* module requests the RGB camera to acquire the image of the scene, while the *Image Processing* module processes the image in order to detect the bins inside the scene, together with their position and orientation. This information is made available to the server thanks to the shared memory. On the other side, the robotic system is composed by two modules: the *Robot Driver* module interfaces the robot and the server to take the required pose (position and orientation) commands, while the *Robot Controller* module handles and executes the received motion requests and the signals received from the external devices (e.g. the signals received from the vacuum sensors mounted on the gripper). When the user interacts with the HMI and selects the bins to be picked up, the corresponding request is sent to the Server and the image is acquired by the camera and processed by the vision system. Then, the motion command to move the robot to the position and orientation of the selected bins is sent to the robotic system, which executes the motion. Finally, a database has been implemented in order to log all the activities of the system and to handle the alarms triggered by the machine during the execution of the task.

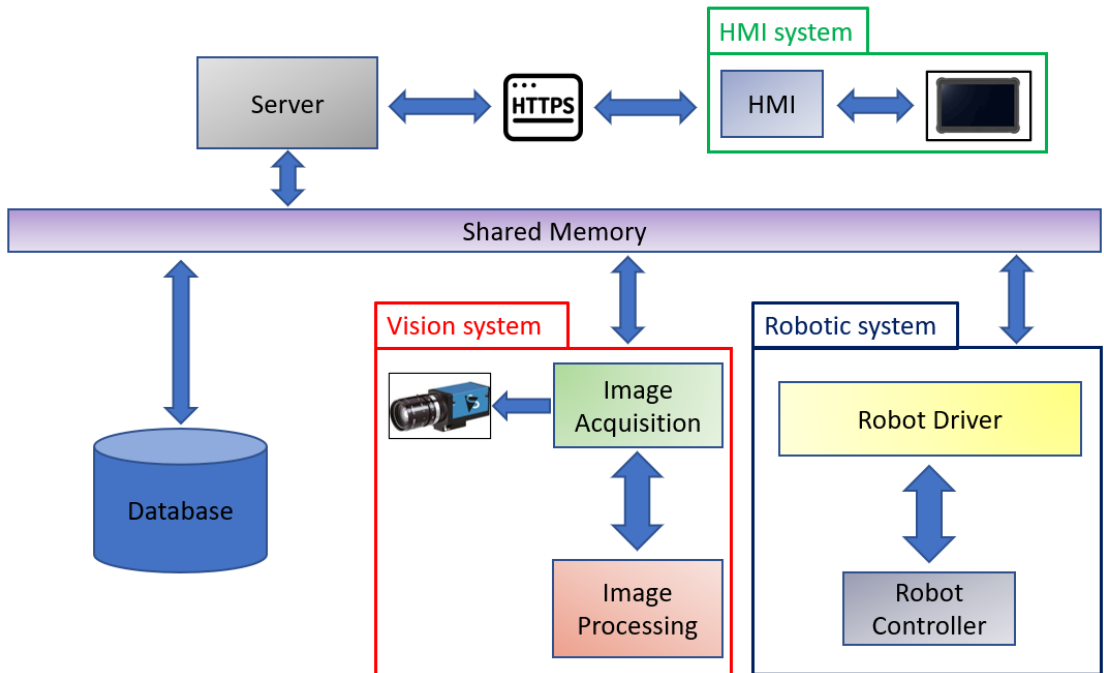


Figure 5.2: Software architecture which handles the overall system.

5.3 Vision System for Recognizing the Bins

The vision system is the core of Robish, since it allows to automatically detect in the image of the scene the bins to be picked up and estimate their position and orientation. This information is then used by the robot to pick up and empty the bins. The pipeline of the vision system is constituted by:

1. An *Image Acquisition* phase, where the image of the scene is acquired by the camera.
2. A *Bin Detection* phase, where the instances of bins are detected into the image (Section 5.3.1).
3. A *Bin's Pose Estimation* phase, where the position and orientation of the bins is estimated with respect to the camera reference frame (Section 5.3.2).

Camera calibration is required in order to transform the pose expressed in the camera reference frame into a pose expressed in the robot reference frame. Camera calibration is performed using standard computer vision techniques (e.g. [92]).

5.3.1 Detection of the instances of bins in the image

The bin detection system takes as input the image acquired by the camera and provides as a result the detected bins. In order to allow this functionality, a Convolutional Neural Network (CNN) [57] has been implemented.

Robish neural network

In the last years, deep convolutional networks have been widely used in many visual recognition tasks [79]. Their typical use is related to classification tasks, where the output to an image is a single class label. However, the desired output of standard CNNs is not suitable in the context of this chapter, since it would not include any localization information. Indeed, to the aim of the proposed system, it is required to: (i) estimate the presence of one or more instances of bin inside the scene (ii) estimate with high precision the position of each instance inside the image. In order to overcome the limitations of standard CNNs we have therefore decided to employ a Fully Convolutional Network (FCN) [62]. FCNs are neural networks, mainly used for semantic segmentation, that only perform convolution (and subsampling or upsampling) operations, without fully connected layers like standard CNNs. In particular, we implemented the U-Net network [79]. The architecture of the U-Net consists of a contracting path (down-convolution) and an expansive path (up-convolution), resulting in a U shape. Figure 5.3 shows an example of the encoding (contraction) of an input image and the decoding (expansion), providing an output image.

The contracting path is a typical convolutional network that consists of repeated application of convolutions, each followed by a rectified linear unit (ReLU) and a max pooling operation.

During the contraction, the spatial information is reduced while feature information is increased. The expansive pathway combines the feature and spatial information through a sequence of up-convolutions and concatenations with high-resolution features from the contracting path. Thanks to the *down-convolution* (compression) and the *up-convolution* (expansion) part 5.3, the completely convolutional network U-NET is characterized by an **U** structure:

- *down-convolution*: a minimal dimension of the network (*bottleneck*) is reached during the compression process. This is possible thanks to convolution and pooling operations.
- *up-convolution*: the expansion process rebuild the original dimension of the image, reintegrating progressively the dimension of the convolutive layers.

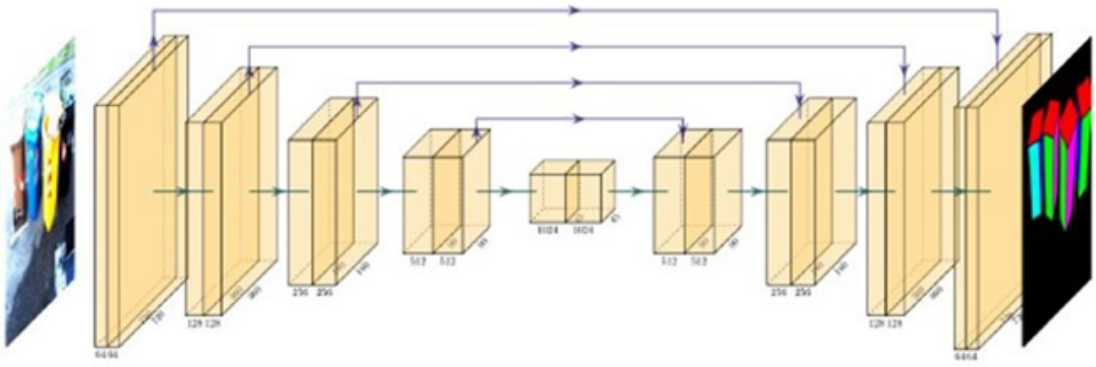


Figure 5.3: U-NET simplified application example architecture

U-NET Implementation

As described in 5.3.1, a U-NET architecture has been choice to detect bins into images. In particular, starting from 5.3, the implementation of the U-NET for Robish is described in 5.4.

The characteristics of the architecture are:

- Input resolution: a scale factor 1:4 allows to reduce computational complexity without loss of performance
- Activation function: ReLU function has been used
- Kernel initialization: HeNormal, it draws samples from a truncated normal distribution centered on 0 and standard deviation proportional to layer parameter.
- Down-convolution: each step consisting of Conv2D[3x3] + Conv2D[3x3] + MaxPool2D[2x2]. The number of down-convolution steps depends on image size.
- Up-convolution: each step consisting of Conv2D[2x2] + Merge + Conv2D[3x3] + Conv2D[3x3]. The number of up-convolution steps depends on image size.

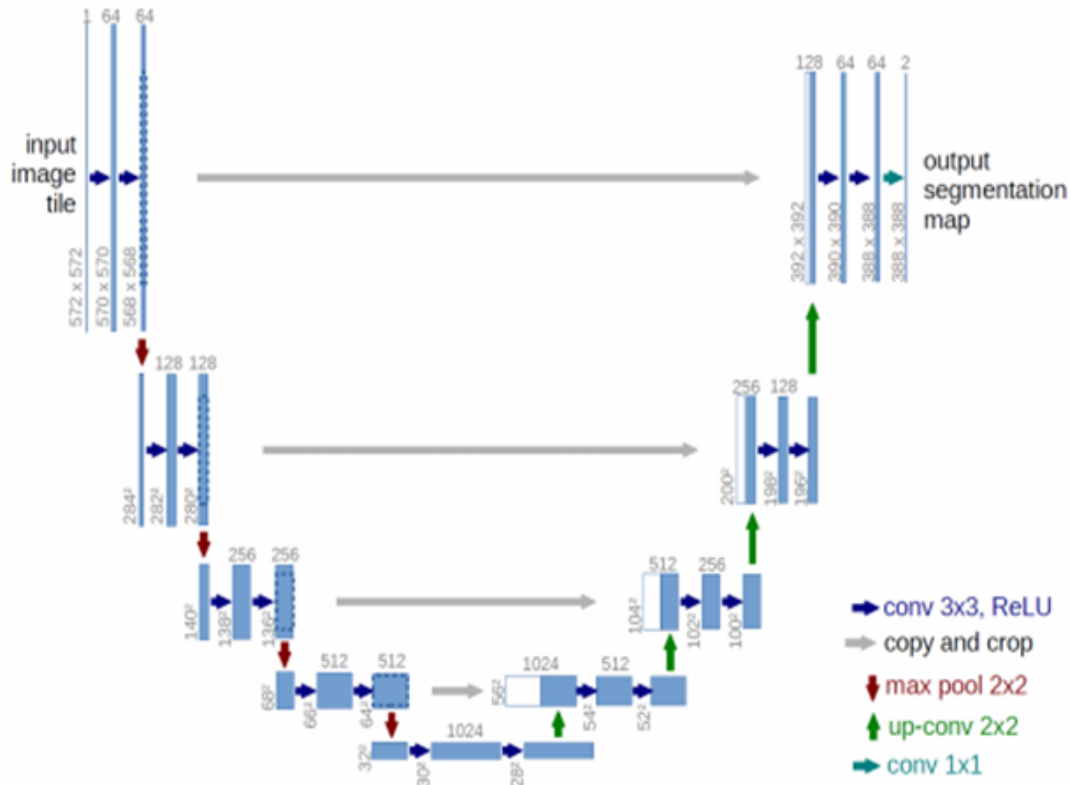


Figure 5.4: Example of U-NET Implementation for Robish system

The ReLU function, *Rectified Linear Unit*, is the most commonly used activation function in deep learning models because help a model account for interaction effects and non-linear effects. Moreover, for ReLU activation function the depth (number of layers) of the neural network architectures plays an important role [80].

In up and down convolution there are server functions used to change the dimensions of the data during the path:

- *Conv2D*: This layer creates a 2-Dimension convolution kernel that is convolved with the layer input to produce a tensor of outputs.
- *MaxPool2D*: It's a sample-based discretization process. The object it to down-sample an input regression, reducing its dimensionality and allowing for assumptions to be made about features contained in the sub-regions binned.
- *Merge*: It's the last down-convolution step, after which the up-convolution steps start.

Dataset creation and labeling

The dataset creation is the complex component before the training of the U-NET model explained in 5.3.1. The Robish dataset, used to train the neural network, is characterized by 800 images. The whole images dataset, like 5.5, composed a set of images, where several bins are contained. The urban recycling is, principally, characterized by 5 kind of waste, correlated by different color

of bins:

- Paper: yellow
- Plastic: blue
- Organic: brown
- Glass: green
- Undifferentiated/Dry: black

For each kind of waste, there are several dimension of the bins; it depends on the specific use of the bin. In this chapter, for each bin the volume of 120 Lt. and 240 Lt. are considered. Only for Organic waste, the smallest size of 24 Lt. is considered.



Figure 5.5: Example image dataset for Robish model

The images dataset are acquired combining the presence of one or more bin and taken into account sever criteria:

- Environment conditions variations: background, illumination and weather.
- Position and orientation variation of the bins
- Occlusions

Once images are acquired, labeling is required. The labeling process consists of to define labels to associate at some features inside the images. Each label characterized a class of the bin. In Robish system, the classes are the different faces of the bin and the labels are the different color masks associate to the class 5.6.

The class-color label association is:

- Top face: Red mask label
- Front face: Green mask label
- Left face: Pink mask label

- Right face: Blue mask label
- Rear face: Light blue mask label
- Bottom face: Orange mask label



Figure 5.6: Example of labeling of an image of Robish dataset

Then, the dataset size is increased through *data-augmentation* operations. The data-augmentation is a pre-process technique used to increase the number of images inside the dataset, applying random automatic transformation on them:

- Rotation of image with range of 10%
- Horizontal shift until 5% of image width
- Vertical shift until 5% of image height
- Cut until 25% of the image surface
- Zoom until 50% of image dimension

The dataset creation is the last step before run the Robish model training.

5.3.2 Pose estimation

The pose estimation process is the final process used to extract the position and the orientation of the bin in the image with respect to the camera. Once the bins are detected into the image by detection process, a reconstruction of the model into the image is required. The result of 5.3.1, is an image like 5.6, namely an image with black background and colored mask (label) found by the neural network. Thus, the pose estimation process must to group the colored label of a bin and, then, estimate the position and orientation taken into account geometrical information of the bin.

For each bin volume size, a local reference system is defined.

As seen in figure 5.7, the axes direction of the reference system has been choice from the geometry of the bin. The origin of the reference system is positioned on the geometrical centroid, the Z axis follows the depth, the X axis follows the bottom edge and, consequentially, the Y axis. Then, for each bin volume size, thanks to the known geometry, 7 different parameter are defined:

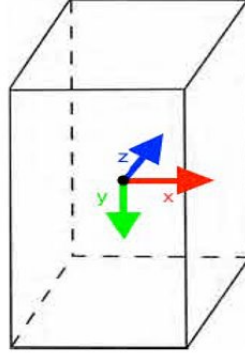


Figure 5.7: Local reference system of the bin geometrical model

- d_{cap} : the depth of the bin cap
- d_{up} : the depth of the h height, taken under the bin picture frame
- d_{down} : the depth of the bin base
- h : bin height
- w_{cap} : the width of the bin cap
- w_{up} : the width of the h height, taken under the bin picture frame
- w_{down} : the width of the bin base

The parameter just illustrated follows the example of geometrical bin model estimated in figure 5.8.

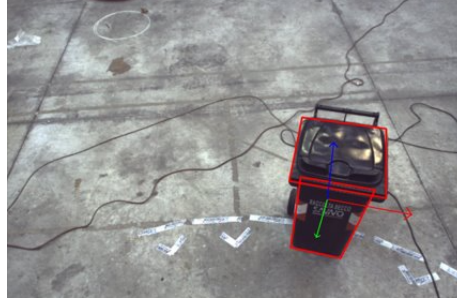


Figure 5.8: Example of geometrical model estimated on image after the bin detection

Given that the feature difference between the bins are not so evident, then to automatically dumbfound the correct volume size is not so easy. Thus, some input from the user are required to estimate the correct position of the bins. First, for each bin into the image a selection of volume size is required. Then, an input click allows to localize the colored mask to take into account during the face mask selection.

The Pipeline used to estimate the bin pose is:

- *Faces map*: starting from an input image, a value map, called *faces*, has been created 5.6. The value map contains a set of binary images, each of which is associated a color class.

This phase is useful to classify precisely the pixels inside the image. In order to implement the *Faces map* process, a coding from RGB to HSV is required.

- *User click*: the input user click allows to extract the selected pixel and, thus, the interest bin. Given a random number of bin inside the image, an algorithm compare the detected classes based on the distance with respect to the selected pixel.
- *Corner estimation*: the corner are estimated for each face. The estimation consists of two algorithm: first, filters and corner detection function of OpenCV are used. Then, an iterative optimizatoin, based on scoring (bonus/malus), is used. The score is set by motion of each corner along the possible directions: 4 cardinals and 4 diagonals directions. The output condition of the algorithm is when any corner movements increase the score. The iterative optimization algorithm is explained in 7. The parameter used to optimized the corner estimation are the list of corners of the colored face, *CornersFace*, to move inside the binary mask *Face*. In 3 a first evaluation of the score is done. The evaluation algorithm is explained in 8. The first evaluation takes into account the first corner estimation, obtained by corner detection function of OpenCV. Then, each corner is moved, by a *MotionStep*, toward one of the 8 possible directions. Thus, in order to compare the first score, another score evaluation is done. The process repeats until a best score is found.

Algorithm 7 Corner Optimization

```

1: Parameters: MotionStep, Score
2: procedure CORNEROPT( CornersFace , Face)
3:   BestScore = Evaluation(CornersFace , Face)
4:   for Corner in CornersFace do
5:     Corners = Corners + MotionStep
6:     TmpScore = Evaluation(Corners , Face)
7:     if TmpScore ≥ BestScore then
8:       Exit
9:     end if
10:  end for
11: end procedure

```

The corner evaluation algorithm 8 check the status of each pixel in the binary mask, *Face*, with respect a polygon created by 6. Then, if the pixel is contained into the polygon a positive $BONUS_{INSIDE-WHITE}$ point is added, otherwise a negative $MALUS_{OUTSIDE-WHITE}$ is decreased.

- *Solve PnP*: The *perspective-n-points* OpenCV function is used to provide the translation and rotation vector of the bin with respect to the camera reference system.

5.4 Robotic and Gripping system

The robotic system is the second component of Robish treated. Once the vision system detects and estimates the position and orientation of the bins with respect to the camera, the robot must

Algorithm 8 Corner Evaluation

```
1: Parameters:  $BONUS_{INSIDE-WHITE}$ ,  $MALUS_{OUTSIDE-WHITE}$ ,  $Score$ ,  $Polygon$ 
2: procedure EVALUATION(  $CornersFace$  ,  $Face$  )
3:    $BONUS_{INSIDE-WHITE} = 1.1$ 
4:    $MALUS_{OUTSIDE-WHITE} = 1.0$ 
5:   for  $pixel$  in  $Face$  do
6:      $Polygon = \text{CreatePolygon}(CornersFace)$ 
7:     if  $pixel$  inside  $Polygon$  then
8:        $Score = Score + BONUS_{INSIDE-WHITE}$ 
9:     else
10:       $Score = Score - MALUS_{OUTSIDE-WHITE}$ 
11:    end if
12:  end for
13:  return  $Score$ 
14: end procedure
```

transform the information into a sequence of action to pick the bins sequence and empty them.

The software architecture of the robotic system is shown in 5.9. The server store the information of the bins poses. Then, when the empty request has been received, the Server transfer the data to the robot communication process. Here, the robot communication process seems a client which wait command from the server. Moreover, the robot communication process has a state machine, with which send command to the robot.

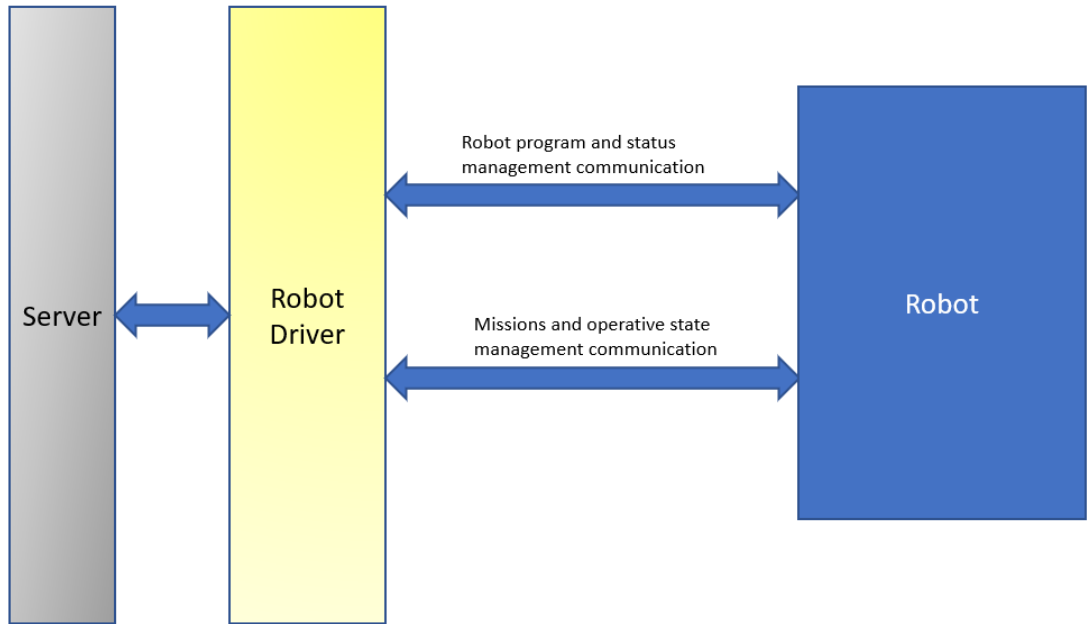


Figure 5.9: Data exchange between robot and control system (server and robot communication process)

The states of the robot are:

- Idle: the idle state consist on a wait of server command
- Move to camera position: Server ask to the robot to go toward the pose where the camera

can acquire the image.

- Move to base position: The robot must move toward the base pose, namely the pose where robot is close into the truck shape. This is pose that the robot must maintain during the truck motion on the streets.
- Check of the reachability of a pose: the robot check if a pose is reachable. Normally, this operation is executed before start to pick the bin, because the robot doesn't know the bin position a priory.
- Empty: this operation send the robot toward the bin poses and allow to empty the bin content into the caisson.

In order to empty a simple bin, a pipeline 5.10 of operations must be follow.

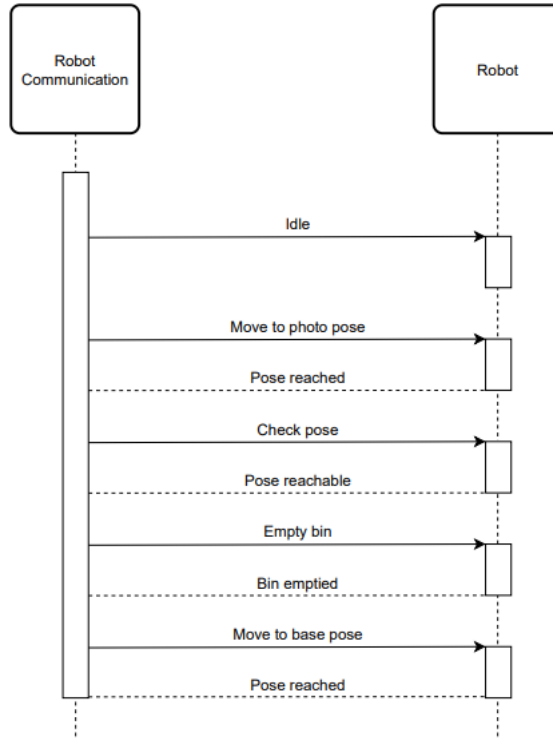


Figure 5.10: Pipeline of a simple bin emptying from the robot

The robot starts from an idle status, where he waits for a command. If the robot starts from the base position, namely the truck just stop the motion, the server ask to the robot to move toward photo position. Once vision system detect the bin, the robot must check the reachability of the pose. If the pose is not reachable, the mission goes down and the user can try with the same image and select an other bin or acquire a new image. If the bin is reachable, the server will ask to the robot to empty the bin. Finally, once the bin is emptied, and the user wants to move with the truck, the robot must be moved toward the base pose. For the check pose and empty phases, the translation and rotation of the bin, T_{bin}^{cam} and R_{bin}^{cam} , must be sent with the relative command. Of course, the robot must transform the position and orientation with respect to robot base, obtaining T_{bin}^{rob} and R_{bin}^{rob} .

5.4.1 Bin manipulation

During the empty operation, it's worth of notice that the bin manipulation from the robot is a sequence of operation planned during the software development.

- Trajectory planning : the robot movement has been realized thanks to the robot controller, which allows to move the robot on joint or Cartesian space. In order to improve the performance of robot motion, the joint movements has been used where the robot joints configuration was known a priory:
 - Base joint position: joint configuration of the robot while the truck moves;
 - Photo joint position: joint configuration of the robot while the vision system acquire the image and detect the bins;
 - Empty joints positions sequence (configuration used after the bin picked);
 - Approach joints position sequence (configuration used before the bin pick);

For this movements, the velocity and acceleration of the dynamic is set to 70% of the maximum values of the robot.

- Check pose: the check pose function allows to check the reachability of a linear motion by offsetting the robot TCP (*Tool Center Point*) position RS_{tcp}^{rob} . If the work-piece is outside the range of robot motion or if the robot cannot take the specified posture to pick up the work-piece, RS_{bin}^{rob} , 5.11 the function ensures that the motion can process the next work-piece smoothly without causing any alarm.

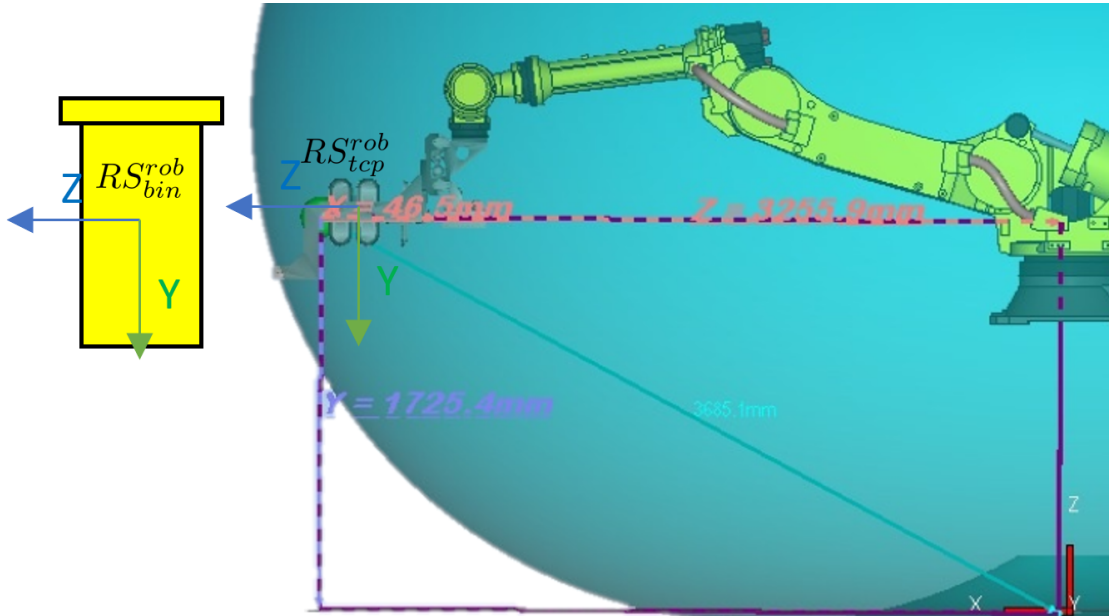


Figure 5.11: Check pose practical scheme with robot TCP and Bin Reference systems

- Safety controls : in most of the commercial robots, the safety functions are already integrated in the robotic controller; usually sold with software options. Even if the safety

system will be described in 5.5, among the various safety functions of the robot, for Robish system the most useful safety functions are:

- CHECK OF THE CARTESIAN POSITION: even if in most of robot motion 5.4.1 are in known joint robot configuration, The randomness and the complex prediction of the joint interpolation and the variable Cartesian bin position computed by vision system 5.3.2 can causing unpredictable collision between robot and other part of the truck.

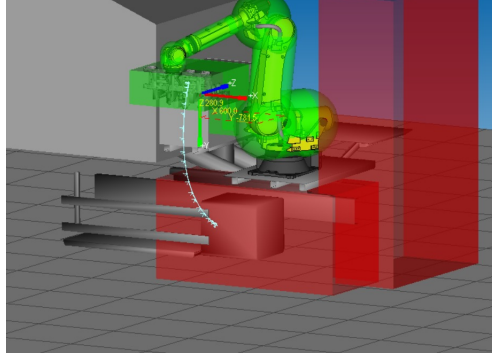


Figure 5.12: Example of the DCS Cartesian position check output on Roboguide Fanuc software

Check of the Cartesian position is a function which can creating several operating or non-accessible area and control a run-time the position of other specific area, created by the user, or the capsules of robot links with respect of them. In Robish system, a specific area to control is created around the gripper, which will be described in ?? . Thus, if a point of the gripper area, or a point of the capsules of robot links, enter inside the non-accessible areas the system stops. Eventually, the same calculations of the Cartesian positions inside of the logical software flow could prevent the stop of the system, namely a time waste.

- CHECK OF THE CARTESIAN VELOCITY: in order to guarantee the safety in the automatic operation of robots, one of the most common limitation introduced is the limitation of the Cartesian velocity of the TCP of the robot. Usually, the safety functions of commercial robots integrates the control of the linear velocity. The principle behind this theory is explained in [84].

$$v = \begin{bmatrix} \dot{p} \\ \omega \end{bmatrix}, J(q) = \begin{bmatrix} J_P(q) \\ J_O(q) \end{bmatrix}, \dot{q} = \begin{bmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_i \\ \vdots \\ \dot{q}_n \end{bmatrix} \quad (5.1)$$

where, J is the *geometrical Jacobian*, i is the $i - th$ joint and n is the number of the joint of the robot. For the Cartesian position speed check purpose, the positional part

of the velocity and, even of the Jacobian, is important, thus:

$$\dot{p} = J_P(q)\dot{q} \quad (5.2)$$

Finally, if we assume that the position and velocity of the joints are safety monitored, then the equation (5.3) guarantee the Cartesian speed limit.

$$\|\dot{p}\| \leq V_{max} \quad (5.3)$$

where, V_{max} is the maximum linear Cartesian velocity allowed by the safety robot controller.

- Singularity check : when the configuration of a robot arm causes a lost of a degree of freedom, then a singularity occurs. For most of six-axis robots, the singularity that occurs at elbow or at the boundary between "front" and "back" configurations causes regional structure degeneracy. This kind of singularity can easily be avoided by restricting the workplace of a robot.

However, the wrist singularity may happen virtually any place inside the workplace. Singularity avoidance functions 5.13 will provide a real-time solution to avoid the wrist singularity. Indeed, the values of q , such that $rank(J(q)) < m - 1$, where m is the number of robot degrees of freedom, say singularities of the manipulator.

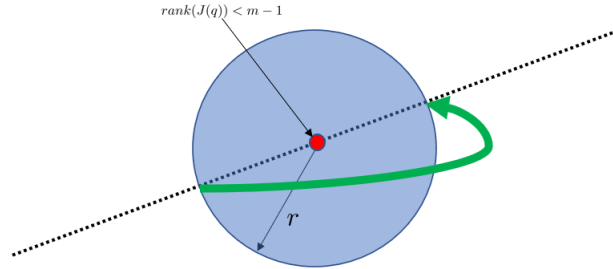


Figure 5.13: Simplification of the case study of singularity avoidance

5.4.2 Bin position feedback control

The robotic system is design to pick the bin detected by the vision system 5.3.2. Robish vision system is designed exploiting a 2D sensor camera, where the pose of the bin is estimated. Given the overall dimensions of the mechanical system and the work environment, where Robish works, the image acquisition distance with respect to the object to estimate is between 2 and 3 meters. This distance doesn't allow to reach high accuracy and, then, given the need of pick the bin with high percentage of resolution, a feedback control 5.14 is introduced. The vision system estimates the position and orientation of the bin T_{bin}^{cam} with respect to itself. Then, thanks to the estrinsics calibration between robot and camera, the pose of the bin T_{bin}^{rob} with respect to the

robot is calculated. T_{bin}^{rob} serves to plan the approach movement of the robot towards the nearest face of the bin. Once the robot moved in front of the bin, a laser pointer is used to detect the real distance d_{bin}^{laser} between bin and itself. Assuming that the bin is positioned in front of the robot, like in fig. 5.1, then the laser allow to compensate the depth error estimate from 5.3.2, obtaining T_{bin}^{*rob} .

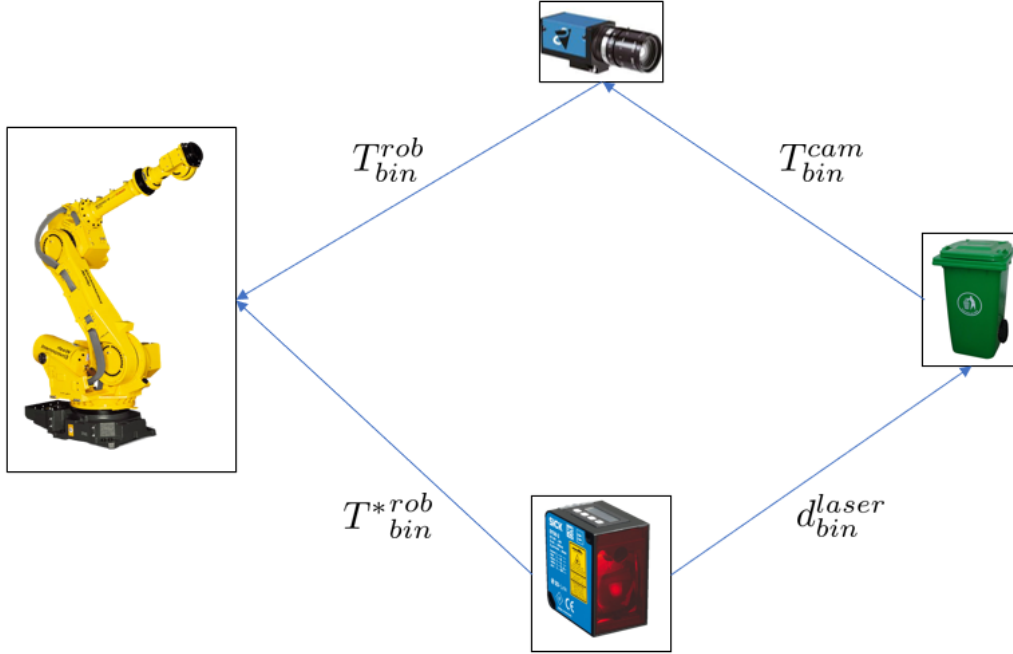


Figure 5.14: Laser pointer used to compensate the pose estimation error

5.4.3 Weight estimation

Robish integrates an add-in function of weight estimation of the content of the bin. In order to support an high-level tracking system, where it's possible to monitor the weight of the urban waste or, simply, to monitor the waste quantity inside the caisson, the robot must estimate the weight of the waste inside the bin just emptied. There are several method to estimate the weight of an object taken into account from the robot gripper. The most common way is to install a load cell between the robot flange and the gripper and, with some calculation and knowing the empty weight of the bins, is possible to estimate the urban waste just emptied. For economics and installation reasons, we opted for a full software solution, developed inside the robot controller. With the robot controller used to design the Robish system, a joints currents monitoring is allowed. In particular, the most important variable used to define the weight estimation controller is the *disturbance current*, namely a difference between the theoretical current apply to $i - th$ robot joint and the real current apply. The real current apply to the joint could be different because:

- The payload set on the robot controller is different with respect to the real payload
- Non linear disturbance effects on current

Ideally, without disturbance and if the payload of the gripper mounted on the robot flange is correctly set, the disturbance current read on each joints is 0. Normally, in order to compensate the disturbance effects, it could not be easy define a disturbance model of the current. Thus, in order to define an optimization function of the algorithm 9, a current threshold is compared with the disturbance current.

Algorithm 9 Weight Estimation Calculation

```

1: Parameters: robotPL, robotPLPrev,  $\epsilon$ ,  $\sigma$ , maxWeight, minWeight
2: procedure WEIGHTEST(  $\hat{T}$  )
3:   robotPLPrev = 0
4:    $\hat{T} = \sum_{i=1}^n |T_i|$ 
5:    $\hat{T}_{prev} = \hat{T}$ 
6:   robotPL =  $\frac{\text{maxWeight} - \text{minWeight}}{2}$ 
7:   while End = False do
8:     Computation of  $\hat{T}$ 
9:     if  $|\hat{T}| < \sigma$  then
10:      No convergence!
11:      Exit
12:     end if
13:     robotPL = robotPLPrev -  $\frac{\hat{T}}{T}$ 
14:     if  $|\hat{T} - \hat{T}_{prev}| \leq \epsilon$  then
15:       End = True
16:       Exit
17:     end if
18:     robotPLPrev = robotPL
19:   end while
20: end procedure

```

Considering that,

$$\hat{T}_i \sim T_i^* - T_{i,com} \quad (5.4)$$

where T^* is the i-th theoretical torque, T_{com} is the i-th real torque applied to the motors and i is the motor index, the function to be optimized is:

$$\hat{T} = \sum_{i=1}^n |\hat{T}_i| \quad (5.5)$$

where, n is the number of the motors of the robot. The objective function is similar to a convex problem, where the function to minimize is

$$\hat{T} \leq \epsilon \quad (5.6)$$

where ϵ is a threshold, R^+ and little, tuned with respect to motor current behaviour. Given that the expression of $\hat{T}_i$ in 5.5 is unknown, thus its derivative too, the *zero* research of the function is similar to the root research of an unknown function. The Newton-Raphson method offers a root research algorithm, considering the function derivative, then the slop of the tangent to the

curve:

$$\hat{T}_{j-1} = \hat{T}_j - \frac{f(\hat{T}_j)}{\dot{f}(\hat{T}_j)} \quad (5.7)$$

Applying 5.7 in 9 and considering a whole global zero of the function 5.4, the modification of the robot payload during the research allow to find correctly the convergence of the convex problem.

5.5 Safety system

Remembering the chapter 1, safety requirement and risk analysis of the robotic application must be identified and done. In order to design a correct and efficient safety system for the machine, a particular process analysis must be follow.

5.5.1 Risk analysis and reduction process

The safety system is system constituted by hardware and software strategies think to guarantee the safety of the users (and not only) while Robish system operates. The design of the system considers the identifiable security deficiencies, which need and improvement or an integration of the safety measures. The security deficiencies depend by the machine limits during the integration of each components, e.g. the installation of the robot on the truck platform.

In order to design a good safety system, there are, essentially, four steps to follow:

- Risk assestment: the analysis starts with the identification and the documentation of the risk entity, then to continue to the definition of the risk reduction measures. First, the machine limits must be identified. Then, determining the operation and the dangers of the machine using, determining the risks and evaluating the existents measures, it is documented if the risk is acceptable or not.
- Safety design: the Safety Desing phase has the purpose to provide a complete and detailed understanding of the concept solutions considered in the Risk Assestment phase. In particular, the existents safety function are organized and detalied to response to the risk identified. Moreover, additional measures are proposed to compensate the safety function lack for particular risks.
- Definition of Essential Safety Requirements: as reported in 1.2, the Machinery Directive 2006/42/CE is used to define the essential safety requirements that the machine must satisfy.
- Computation of Performance Level Required

In order to correctly evaluate the risk identification, a dictionary of norms and directives has been used. The most relevant of them are in 5.1.

Name	Emission date	Title
2006/42/EC	2006	Machine directive
EN ISO 10218-1	2011	Robot and robot devices - Safety requirements for industrial robots - Part 1: Robots
EN ISO 10218-2	2011	Robot and robot devices - Safety requirements for industrial robots - Part 2: Robot systems and integration
UNI EN 1501-2	2010	Refuse collection vehicles - General requirements and safety requirements - Part 2: Side loaded refuse collection vehicles
IEC 60364-4-41	2005	Low-voltage electrical installations - Protection against electric shock

Table 5.1: The most relevant norms and directives used to risk assessments

The using of the normative UNI EN 1501-2 is a clear example of the number and the type of the norm used during the risk assessment depends on the industrial application considered. Indeed, the normative UNI EN 1501-2 [95] applies to side loaded refuse collection vehicle (RCV), as defined in UNI EN 1501-1. In particular, the document deals with all significant hazards, hazardous situations and events relevant to the side loaded RCV, when it is used as intended and under conditions of misuse which are reasonably foreseeable by the manufacturer, throughout its foreseeable lifetime. Moreover, the normative is applicable to the design and construction of the side loaded RCV so as to ensure that it is fit for its intended function and can be operated, moved, cleaned (including unblocking), adjusted and maintained during its entire lifetime.

5.5.2 Risk summary

After a first evaluation of the risks, the principles danger sources are:

- **Object falling:** consists of the bin falling after a collision with respect to the caisson.
- **Unexpected motion from the robot:** e.g. if the robot moves while the truck moves.
- **Crushing:** e.g. a possible crushing between the lift device (robot) and the ground during the lowering phase.
- **Entrapment:** e.g. entrapment between the bin and the lift device (robot).
- **Insufficient visibility:** the actuation of the system without the operator observes the work area.

In order to reduce the risks evaluated, several additional measures of the risks reduction has been adopted. In particular, some intrinsic features were already integrate in one Robish component and, then, any safety measure has been adopted, because the risk was already reduced. E.g., as explained in 5.4.1, the robot controller already integrates several safety function by programming on its software and, thus, the risks caused by the unexpected motion from the robot has been solved.

5.5.3 External safety devices

Even if many risks have already been reduced by the intrinsic safety function of the Robish hardware and software components, the rest of the risks has been solved by integrating external safety devices.

Assuming that, when the robot works, the operator, and anyone, cannot enter into the operation area, external devices to guarantee this trick has been added. In figure 5.4.1, the most relevant external safety devices are remarked.

- **PHYSICAL SIDE BARRIERS:** two side barriers are lowered when the robot operates and the truck is stopped. The safety measure has been introduced to notify to external passer-by



Figure 5.15: Robish system with external safety devices remarks

to not enter. Moreover, even if not represented, a yellow flashing light and an acoustic signal are activated during the robot motion.

- **EMERGENCY STOP BUTTONS:** two stop buttons are positioned next to the Physical side barriers. One stop button is mounted on robot controller and one on the robot teach pendant. Finally, the last emergency stop button is positioned inside the truck cabinet, where the operator normally works. The emergency stop buttons prevents danger caused by unexpected robot movements or incorrect bin picking by the gripper. Even if the operator work inside the truck cabinet, the stop buttons installed next to the barriers are useful if an external people needs to stop the system operation.
- **CLOSED CIRCUIT TELEVISION SYSTEM:** the visibility guarantees by the side mirrors is not sufficient for Robish system, as the norm UNI EN 1501-2 says 5.16.

Indeed, the operator, who guides the truck, could have visibility in front of him, through the windscreen, or in the right side, through the right window. Moreover, thanks to the right side mirror, the operator visibility is increased. In figure 5.16, the green lines estimates the operator visibility, while the red lines show the work-space area which the operators could be seen. When the robot operates, in particular when he's going toward the bin, a portion of the operator visibility, guarantee by the right side mirror, is compromised. Thus, in order to complete the operator visibility during the robot operation, namely to eliminate the red lines, two closed circuit cameras has been installed. So, the operator could control the operative zones, inside the truck cabinet 5.17, during robot movements.
- **ISOLATION CONTROLLER:** Robish system install a robotic arm over a truck platform. The power source derived from a gasoline generator, which power any components. Of course, the truck has not a grounding and, thus, an **IT system** must be considered. As explained

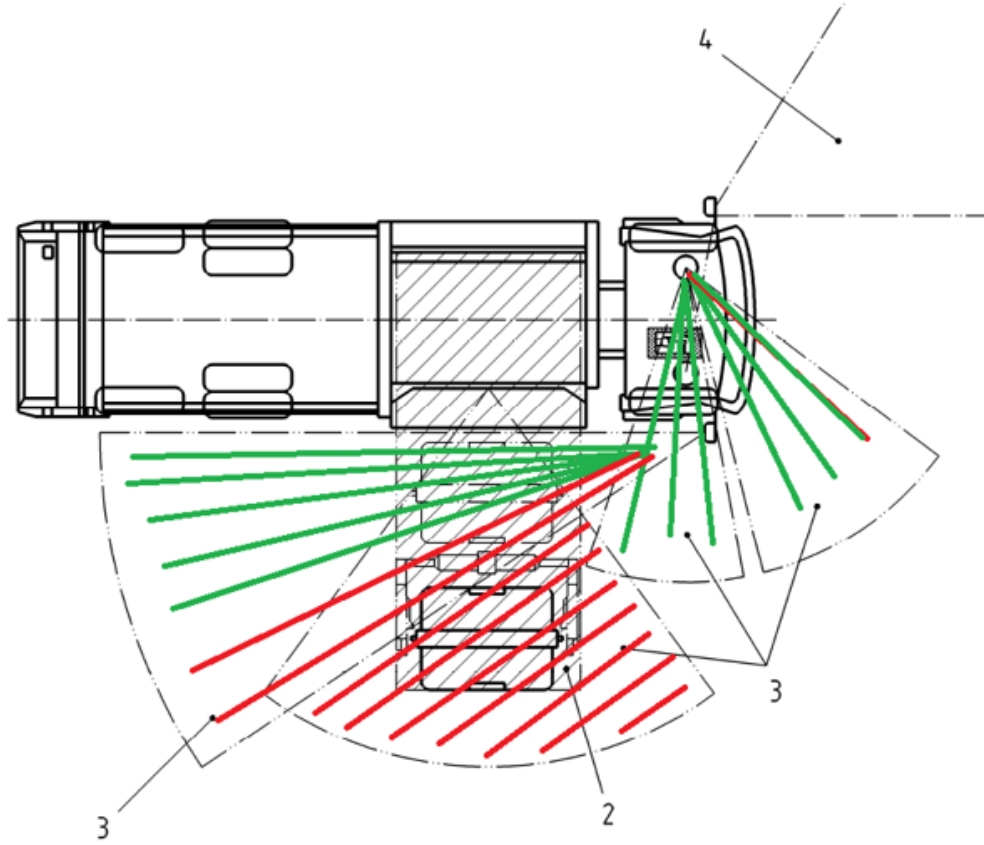


Figure 5.16: Visibility scheme of the operator inside the truck cabinet, exploiting the right side mirror.

in [42], on IT system the neutral of the transformer is not grounding.

The electric scheme 5.18 is used when the dangerous caused by the power disconnection is greater than to maintain the energy on. Indeed, in Robish system, the truck platform has not a real grounding, because the tires prevent the contact between, e.g., the robot and the ground. On IT system, the electric voltage parts must be isolated from the ground or connected with him through very high impedance R_A .

From IEC 60364-4-41, 5.1, in IT system can be used several monitoring and protective devices. In particular for monitoring devices, his aim is to notify the presence of an abnormal condition after a system failure. The isolation monitoring devices 5.19 can detect isolation deficiencies due to the resistive component.

In Robish system, if the isolation monitoring device detect an abnormal operation of the isolation system, a notification to the operator is send.

- **DEAD-MAN SWITCH:** In order to have more control in the operator attention, an *Hold-To-Run* command must be pressed to run the automatic waste bin picking. The Hold-To-Run command is represented by a dead-man switch 5.15, which the operator must press during the automatic operation. A dead-man switch is an ergonomic 3 state switch with an electric contact normally closed. The switch has 3 state operation, where



Figure 5.17: Closed circuit monitor of one cameras and the internal led light signaling

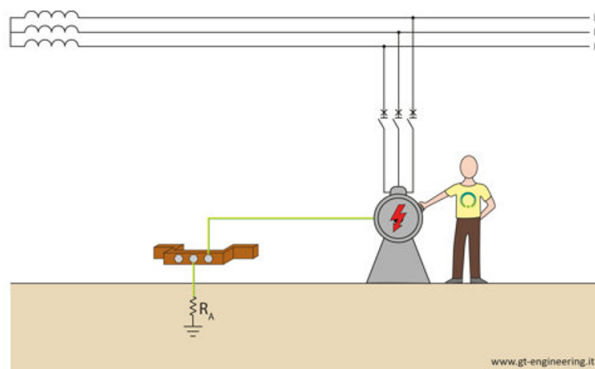


Figure 5.18: Scheme of the transformer connection with respect to the ground

- First: the switch is released
- Second: the switch is pressed in an intermediate position of his stroke
- Third: the switch has been exaggeratedly pressed, then the position of the switch has reached his maximum stroke.

Only the second condition open the contact and, then, enable the operation of the automatic behaviour.

Thus, when the first o third condition is verified during the automatic cycle, the robot safety stops and the gripper doesn't released; so avoiding the bin falling. Moreover, in order to discourage any tampering a monitoring control on enabling pressed has been added. If the dead-man switch state stays high for more than 10 minutes 5.20, the automatic cycle stops and to recover the normal function a sequence of releasing and pressing needs.

- **TRUCK CABINET PUSHBUTTON:** The last safety external device controlled by the operator is a pushbutton, installed on the truck cabinet. Besides the emergency stop button, others controls are allowed:
 - Alarm reset: the alarm reset button allow to acknowledge the normal operation status, only if all alarms solved. If the button pressed when even one alarm is up, the normal operation status are not acknowledge.



Figure 5.19: Isolation monitoring device

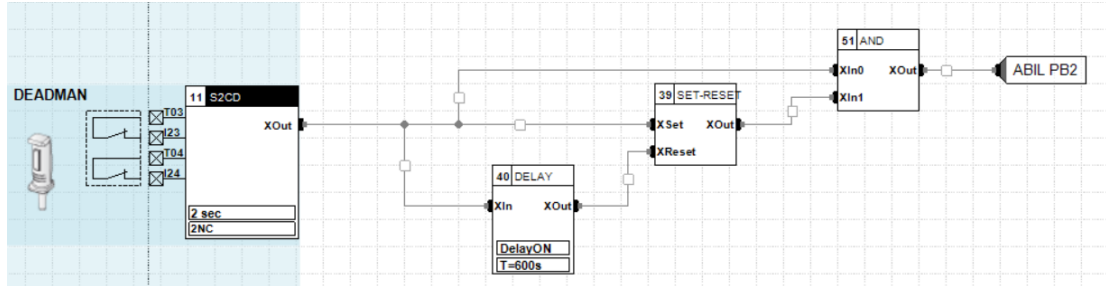


Figure 5.20: Dead man switch signal control on safety control unit

- Manual-Automatic switch: even if the system is designed to operate in automatic mode, a manual use method of the system has been thought. In this phase, e.g., the robot can be moved with safe, and slow, velocity only with manual movements (Jogging). Moreover, only in this status, the manual operation of open and close gripper allowed through the open/close gripper buttons.
- Open/close gripper buttons
- HAND-BRAKE HANDLE: finally, the last important control is demanded to the Hand-brake control. As explained in 5.5.2, the unexpected motion from the robot is a risk must be to reduce. In particular, when the truck is moving, the robot absolutely has not to move. Thus, in order to prevent this kind of risk, a control on the hand-brake has been implemented.

The truck brakes operates through compressed air. If there is no air compressed into the brake pneumatic circuit, the brakes works. Otherwise, only if air compressed flows in the brake circuit, the truck wheels are no brake. Then, order to recognize the brake action status, two manometer has been installed 5.21. Thus, once the safety signal of the hand-brake is acquired, two control must be done:

- If the hand-brake is release, so the truck could moves, the robot must be in the close position, namely the robot configuration which allow to stay in shape with truck, otherwise an alarm, with signals for operator, occurs. Moreover, even if the robot is in shape with truck, an emergency stop occurs.
- If the hand-brake is on, the robot can moves with respect the safety guidelines already explained.

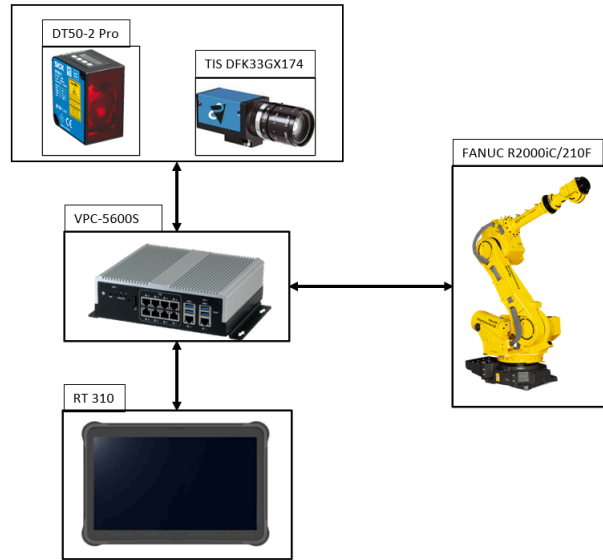


Figure 5.22: General hardware architecture of Robish

DFK33GX174	
Resolution	1920x1200
Sensor type	Sony IMX174LQJ-C
Sensor format	1/1.2 inch
Pixel size	5.86 um

Table 5.2: DFK33GX174 technical data

captured. In order to get a trade off between the maximum number of bins into the image and the acquisition distance, the optical lens characteristics are illustrated in 5.3.

The HMI tablet, RT 310, is a rugged tablet PC with a display size of 10.1 " and a pre-installed operative system windows 10 professional. With the tablet, the web-based HMI, explained in ??, is used through a simple web browser and https communication. Finally, the robot integrated to move the waste bins is the FANUC R2000iC/210F. The technical characteristics of the robot allow to move payload of 210 Kg until 2655 meters of distance. Of course, the maximum payload must be decreased by the gripper weight. Indeed, from a static and dynamic simulation, the maximum weight of the bin moved by the robot at the maximum distance is 120 Kg.

M112FM06	
Imager Size	1/1.2 inch
Focal length	6 mm
Field of view	93.1° x 59.0°

Table 5.3: M112FM06 technical data

5.6.2 Protective cover

As shown in figure 5.1, the third principle components of the system is the weather protective system. Indeed, given that the robot has been installed between the truck cabinet and the caisson, thus on external environment, the critical weather conditions had been taken into account to evaluate a complete and efficient protective cover for him. The operative weather conditions are surely variable:

- Temperature: from -20° to 60°
- From high to low humidity
- Rain, snow, hailstorm or scorching sun

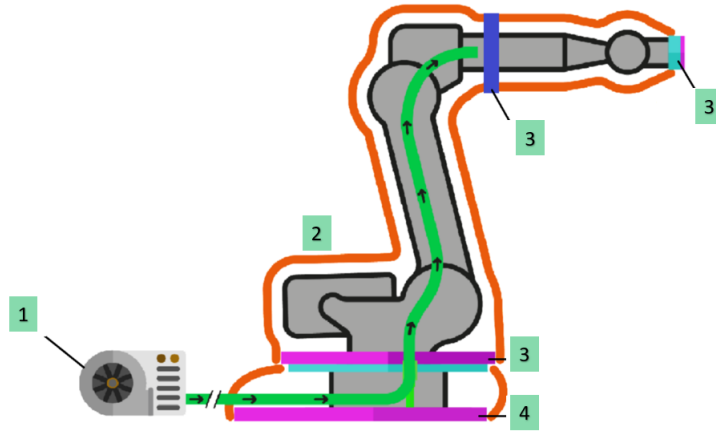


Figure 5.23: Protective cover scheme of robot R2000iC/210F

The cover solution identified is represented in figure 5.23.

- 1 Fan heater with hot battery: The fan heater feeds air into the cover. In order to monitor the exercise temperature, a temperature sensor is installed inside the cover. If a threshold temperature is reached, the hot battery warms the air feed into the cover, otherwise the air cools the components.
- 2 The protective cover: multi-layer insulating protection to limit the heat exchange. The cover is a polyamide tissue, coated by thermoplastic layer.
- 3 Mobile or fixed headband: the mobile headband allow to split the relative rotation of the tissue mounted before and after it. While, the fixed headband offers an assembly or disassembly simple solution useful for maintenance operations.
- 4 Base frame: the picture frame where robot is mounted.

5.6.3 Image acquisition: High Dynamic Range filter

The vision system, described in 5.3, is a pipeline of operation where the bin is detected into the image (*detection*) and, then, his pose is estimated (*estimation*). The detection process is

a Convolutional Neural Network model trained with a particular dataset. The most important characteristic of the dataset is that must be various in terms of background and of rotation of the bins. Of course, the most disturbing element of the trained model is the continuous variable natural lighting and reflexes, produced by the sun. Even if, during the dataset acquisition, these considerations are taken into account, there is an high level of uncertainty of the disturb reduction. Then, in order to reduce the disturbing element, an High Dynamic Range (later called HDR) filter is used during the image acquisition phase. High-dynamic-range imaging (HDRI or HDR) is a technique used in imaging and photography to reproduce a greater dynamic range of luminosity than is possible with standard digital imaging or photographic techniques [4]. When an image is acquired by a camera sensor, bright regions may be overexposed, while the dark ones may be underexposed. The most common method to obtain HDR images is to use photographs of the scene taken with different exposure values. Then, a merging between the images is made to obtain the useful HDR image. Several HDR methods are implemented and proposed. In particular, in OpenCV framework there more than one hdr fusion method to obtain a post-processed result. Robish system implements an exposure fusion methods, developed in [90] from Tom Mertens. The method avoid to calibrate the response function of the camera and blends the images in a multi-exposure sequence considering saturation and contrast. Of course, a set of exposure time of the camera sensor must be define to obtained an HDR result. The natural lighting and reflexes enormously vary during the day in a image. However considering the same weather condition, the set of exposure time to consider must be different. During the day, the natural lighting is very high and, then, the reflexes on shiny surface are very annoying. In order to avoid to define several static set exposure time to switch, a simple dynamic set exposure time creation algorithm has been proposed in 5.6.3.

Algorithm 10 Dynamic exposure time set creation algorithm

```

1: Parameters: expMax, expMin, expDelta, expSet, rate, rateMax, rateMin, iUp, iDown
2: procedure EXPOSUREDYN(expActual)
3:   if expActual < expMin then
4:     expActual = expMin
5:   else if expActual > expMax then
6:     expActual = expMax
7:   end if
8:   rate = computeRate(rateMin, rateMax, exp)
9:   expDelta = rate * expDelta
10:  iUp, iDown = setPosNegDelta(expActual, expDelta, expMin, expMax)
11:  for i in iUp do expSet = add(expActual + i * expDelta)
12:  end for
13:  for i in iDown do expSet = add(expActual - i * expDelta)
14:  end for
15:  return expSet
16: end procedure

```

The algorithm is function of the actual exposure time of the camera. The camera sensor could work in *Auto exposure mode*, where the sensor automatically set the best exposure time with respect the external lighting. If the lighting is high, the exposure time will be low, otherwise

high. When the automatic actual exposure is acquired, a comparison with respect exposure limit did. Then, taken into account an exposure delta value between one exposure value and the neighbour, a ratio value is computed at line 8 as:

$$rate = rateMin + (rateMax - rateMin) * \frac{expAct - expMin}{expMax - expMin} \quad (5.8)$$

Then, the rate value provides a real exposure delta value. Given the actual exposure value, the number of exposure set to find is defined in function 10. Finally, the algorithm returns an array of exposure time used to acquire images at different exposures, like in Figure 5.24.

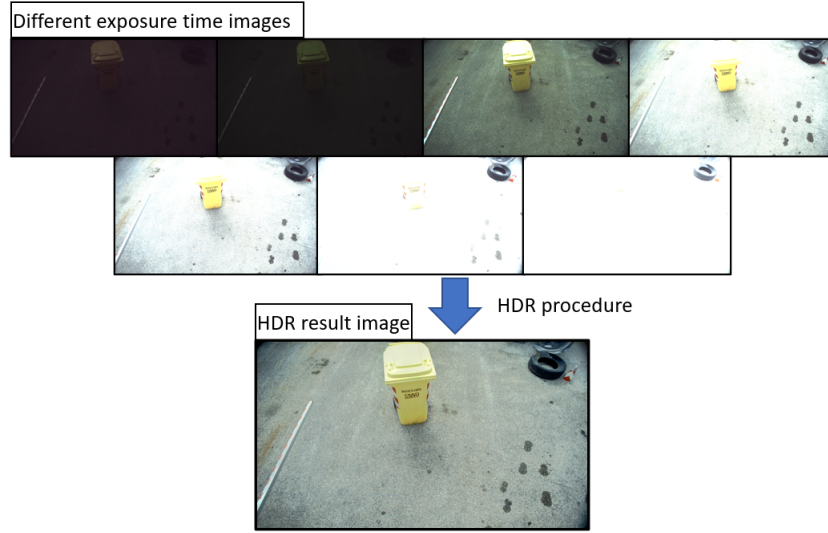


Figure 5.24: HDR processing result starting from a 7 images acquired with different exposure times

5.6.4 Small bin handle release

The small bins, emptied from Robish system, are used to collect organic waste. The bin capacity is 25 lt. and his material is polypropylene. The shape is of pyramidal trunk and the cap is equipped by stray-resistant locking system [55]. Just for locking system, an handle release



Figure 5.25: Small bin on buffer position

Figure 5.26: Handle release action

Figure 5.27: Emptying without handle lock

strategy has been prepared. Indeed, as shown in figures 5.25, 5.26 and 5.27, once the robot picked the small bin, it must moves towards an intermediate position where the bin is buffered. Here, a mechanical template lock the bin and the robot can release the handle without any vision system. Once the handle is unlocked, the robot can empty the bin and to complete his mission.

5.6.5 Validating tests and results

In order to validate the performance of the system, several external picking tests had been done. The results will be evaluated based on

- Reachability result: if the bin is reachable by the robot TCP. It depends on robot workspace range and on pick configuration of the bin.
- Vision system result: it depends on the training dataset of 5.3. If the neural network model never seen something similar to the training images, then a bad result will expected.
- Gripping and Handling result: if the vacuum system of the gripper or if the handle release operation for small bin fails.
- Robot movements result: if any joint limit occurs during linear movements or if any safety area is compromised.

with respect each bin size.

As follows, two sets of tests will be presented; first, a tests result conducted by the developer and, second, a tests result conducted by ecological operator.

- Mixed bins grouped on areas: the developer tests the system
- Mixed bins grouped on areas: the ecological operators drive the truck and uses Robish system to empty the bins

Test conducted by expert user

In order to reduce the success percentage risk based on the same bin sequence picking, the tests have been conducted mixing the bins areas composition. Moreover, the tests conducted count an amount of 59 group test, where 25 for bin size of 24 lt., 11 for 120 lt. and 23 for 240 lt.. Then, for each size the percentage success result has been extracted for reachability, vision system, gripping and handling and robot movements results.

Bin Size [lt.]	Reachability [%]	Vision Detection [%]	Gripping&Handling [%]	Robotic[%]
24	95,65217391	85,71428571	90,47619048	100
120	100	100	90	100
240	95,45454545	100	100	100

Table 5.4: Percentage success results for tests conducted by developers

Given the percentages in table 5.4, it worth of notice as the small bins have some percentage points less the full success result with respect other sizes. First, a reachability problem occurs. Indeed, the small bin normally are farther then other sized because the geometrical dimensions. The center of gravity is near to the ground, thus the bins are always near to the robot workspace

limits. Second, the vision system has difficulty to distinguish the small bin features into some image. This because the small bin faces are similar to each other and, sometimes, the perimeter is not identifiable. Finally, sometimes the precision of vision is not accurate enough to finalize the handle release inside the mechanical template. Moreover, sometimes the embossed pictogram applied on the front of the bins with size 120 lt., e.g. in figure 5.5, doesn't allow to apply a correctly the vacuum.

Test conducted by ecological operator

Then, some tests are conducted by an ecological operator. In this tests, the operator drove the truck inside a prefix path. Several areas, constituted by mixed bins, had been configured. The operator starts the tests from a start position with the truck and, then, once he reached the areas, the automatic bin picking procedure was being performed. Moreover, in order to make evident the efficiency of the collection system, the handle release phase had been removed for this tests. Again, considering the same evaluation model used in 5.6.5, a collection of the results is proposed in table 5.5. The number of tests executed to define the results are 34.

Bin Size [lt.]	Reachability [%]	Vision Detection [%]	Gripping&Handling [%]	Robotic[%]
24	100	100	100	100
120	100	100	85,71428571	100
240	76,92307692	100	100	100

Table 5.5: Percentage success results for tests conducted by ecological operator

It worst to notice that the reachability evaluation enhances for 24 lt. bins while it get worse for 240 lt. bins. Indeed, for smallest bins, in order to improve the robot positioning with respect to the bins, the operator drove better the truck system. For the biggest bins, the robot workspace not always is enough great. The vision detection system improved the result for smallest bin, because a new training of the neural model had been realized. Moreover, the weather condition was better for the vision system (little natural lighting and reflexes). Finally, the result of 85%, recorded under Gripping and Handling class for bin size of 120 lt., follows the same problem signed in 5.6.5, namely the embossed pictogram bin feature which not allowed to do vacuum. However, the percentage is lower than the other tests because, in 5.6.5, it had been conducted 25 tests versus 8 tests for those conducted by the operator.

5.7 Conclusions

The Robish system is a semi-automatic system designed for automatized the urban waste bins collection. In according to the customer, a commercial truck had been made available to design and realize a robotic system installed on it. The purpose of the system is supply to the ecological operator a robotic assistant during the waste bin collection and to relieve the operator by heavy and repetitive operations. The Robotic system 5.4, flanked by an intelligent vision system 5.3, is

able to automatically recognize the bins on the street and empty them. The ecological operator, through an innovative and easy designed Human Machine Interface, can program the waste bin collection simply presiding his truck driving position.

In parallel, the operator and passers-by safety is guarantee by the safety system 5.5, where monitoring system and safety devices are combined to support the operator monitoring and work.

The results obtained by the tests 5.6.5 are comfortable. Once the system is tuned and trained to recognize the urban bins, the picking results show a good success percentages on the several evaluation classes.

However, the tests marked some system limits:

- *Reachability*: unfortunately, the robot reachability not always satisfies the expectations. Indeed, the maximum distance reached by the robot is lower than 3 meters, because the robot always works under his origin (robot base 5.11). In a real scenario, the bins are not always positioned in range with respect to the limit. Moreover, an inverse problem could happen. If the bins are too closely with respect to the truck side, and if the vision system's able to see the bin into the image, the robot would move toward the nearest bin face, entering a prohibited security area.
- *Vision system velocity*: considering the hardware used in Robish system 5.6.1, the bin identification into the image, namely image acquisition with HDR, inference and pose estimation, uses 5,5 second; it's variable because the HDR filtering process 5.6.3. If a better performance is required, an hardware improvement, and thus a costs increment, is need.
- *Weight estimation*: the weight estimation method 9 use the embedded torque information of the robot motors to estimate the weight of the bin. The torque motor data are not free of disturbance and, sometimes, the weight estimation process could required some second. Thus, a possible and expensive solution is to install a load cell on the robot wrist. The weight information extracted should be more precise and quicker than the actual cheap solution.
- *Caisson filling*: The Robish system is design to automatize the waste bin emptying into the caisson. However, filling the caisson always in the same point or side creates a waste accumulation which not uses the whole caisson space. A solution of this issue is to integrate a compactor which moves the just emptied waste toward the rear of the caisson.
- *Path planning*: the robot software is a sequence of fixed and variable movements. Of course, to use the same motion strategy for all bins configurations (orientation) is not robust. Moreover, the bin face nearest to the robot base is not always the best face to reach with the front of the gripper, because the obstacle or other bins could be struck.

5.7.1 Future Developments

The Robish system limits can be passed improving some aspects.

The reachability limits need to plan an improvement on robot installation. Given that the robot workspace is concentrated under the robot base installation, a possible solution could be desing a robot platform positioned almost at wheel height. Moreover, the integration of a linear external axis should help to reach the most distance bins.

The vision system is an artificial intelligent based system which requires training. Each training uses several hours or days to be complete and, however, the labeling phase of the dataset is necessary 5.6. A future improvement could be define an autonomous labeling system, which automatically recognized the worst inference images and labeled the real bins on them.

Finally, the bin picking queue programming is completely managed by the operator know-how. Of course, the operator cannot always know the robot limitation with respect to the bin pose to reach. Moreover, considering the scene presented in the image, there could be several strategies to compose the queue programming and to choice the best bin face to pick. A future development is to define an optimizer which allows to define the best bin face to pick with respect to obstacle and robot limits. Then, in order to improve the system performance, the best scored bin picking queue programming must be computed by the algorithm. Moreover, an high-level path planner must can plan the robot motion considering:

- Singularity
- Joint limits
- Safety areas
- Obstacle avoidance

Conclusions and Future Directions

The robotics and, collaborative robotics, is a powerful tool used to increase the performances of industrial production lines inside the companies. Usually, more speed robot has the greater is the productivity. Of course, in order to guarantee the safety inside the installations, physical barriers to protect the operators are required. However, this kind of protections is an obstacle to the production flexibility, where frequently retooling of the machines are required. Indeed, collaborative robotics goes further to standard protection and try to propose new way of interaction with robot. On this way, not always an high velocity of the robot allows an high productivity, specially when the operator must interact with the machine (or robot).

The standard ISO/TS 15066 regulates the new ways of interaction between human and operators, at the expense to the machine performance. The Power and Force Limiting method can be overcome flanking a controller which optimized the robot velocity when the collision danger is far (2). The Control Barriers Functions approach defines a safe operating region where the human can quietly work. So, in this scenario, even if a physical contact between operator and machine is expected by the application, the machine performances are maximized.

If the physical interaction is required by the industrial application, the Hand-guiding method allowed to manually move the robot only exploiting the human input command. To guarantee operator safety, however, maximum velocities are limited according to the ISO/TS 15066. Usually, industrial robotic implementation required high-velocity processes to guarantee quality and productivity and, in this case, the Hand-guiding would be too limited. If the hand-guiding controller and the safety system are properly designed, like in chapter 3 and 4, the just mentioned limitations can be overcome.

On the other hand though, the robotics can become an important source to guarantee the health of the humans during their works, even if a physical contact is required (5).

The results achieved in this thesis show how safety standards can be exploited to overcome the too stringent limitations, while guarantee the human safety. Depending on the industrial application, the reference standard can be different but, properly design the control and safety system, the machine performances can be maximized.

However, this work, even providing satisfactory results, has always had one limit: the robotic

system cannot predict and estimate the human motion and action in a safety way. A vision system which detect the human posture with skeletal tracking approach have been introduced in chapter 2. The human art position is used to instantly change robot motion, but this information cannot be exploit to certificate a robotic system. Moreover, the only instant position of the human is not sufficient to optimize the robot behavior. Rather, it would be more performant to predict the human gesture. On this way, a technological solution or sensorization system which can provide the human position with a certain performance level would open new horizons in industrial application, where human and robot must and would coexists.

Finally, always trying to equipped the robot with the sense of the vision, a major limitation of modern vision systems is the complex and expensive research to perceive and recontruct the surround environment; especially for outdoor scenario, where the event variability is very high. In addition to guarantee the safety of the human, the perception of the environment would allowed to the robots to dynamically change their behaviours and movements based on the people and movable objects.

Bibliography

- [1] ESA - 2 Automatic System - 2AS. [Online]. Available: <https://www.ecologia.re.it/2as/>
- [2] Farid - Lateral automatic collection system - SLF. [Online]. Available: <https://faridzoellergroup.com/faridindustrie/prodotti/slf/>
- [3] “Gaiotto Automation SpA - Robotized glazing in sanitaryware production.” [Online]. Available: https://sharedcontent.sacmi.com/sharedcontent/media/Documents/Ceramics/catalogue/Gaiotto_glazing_GA2000_GA-OL_GA-F-OL_7WD_MASSCONTROL.pdf
- [4] High Dynamic Range (HDR). [Online]. Available: https://vovkos.github.io/doxyrest-showcase/opencv/sphinxdoc/page_tutorial_py_hdr.html
- [5] Nord Engineering - Lateral automatic collection system - Easy. [Online]. Available: <https://www.nordengineering.com/easy/>
- [6] Sadako Technologies - Max-AI. [Online]. Available: <https://sadako.es/max-ai/>
- [7] Waste Robotics. [Online]. Available: <https://wasterobotic.com/robots-recycling/>
- [8] Zen Robotics - Fast Picker. [Online]. Available: <https://zenrobotics.com/solutions/fast-picker/>
- [9] AIMEN Centro Tecnologico, “Hand guiding industrial robots,” <https://youtu.be/IauPQa98EmA>, 2020, [Online; accessed 02-August-2022].
- [10] A. Albu-Schaffer and G. Hirzinger, “Cartesian impedance control techniques for torque controlled light-weight robots,” in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 1, 2002, pp. 657–663 vol.1. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/1013433>
- [11] A. Albu-Schäffer, C. Ott, and G. Hirzinger, “A unified passivity-based control framework for position, torque and impedance control of flexible joint robots,” *The International Journal of Robotics Research*, vol. 26, no. 1, pp. 23–39, 2007. [Online]. Available: <https://doi.org/10.1177/0278364907073776>
- [12] S. Aldini, A. K. Singh, M. Carmichael, Y. K. Wang, D. Liu, and C. T. Lin, “Prediction-error negativity to assess singularity avoidance strategies in physical human-robot collaboration,” vol. 2021-May. Institute of Electrical and Electronics Engineers Inc., 2021, pp. 3241–3247.

- [13] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2016.
- [14] D. Andronas, Z. Arkouli, N. Zacharaki, G. Michalos, A. Sardelis, G. Papanikolopoulos, and S. Makris, "On the perception and handling of deformable objects – a robotic cell for white goods industry," *Robotics and Computer-Integrated Manufacturing*, vol. 77, p. 102358, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0736584522000461>
- [15] M. H. Ang, L. Wei, and Lim Ser Yong, "An industrial application of control of dynamic behavior of robots-a walk-through programmed welding robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 3, 2000, pp. 2352–2357 vol.3. [Online]. Available: <https://ieeexplore.ieee.org/document/846378>
- [16] M. H. Ang, W. Lin, and S. Lim, "A walk-through programmed robot for welding in shipyards," *Industrial Robot: An International Journal*, vol. 26, no. 5, pp. 377–388, 1999. [Online]. Available: <https://doi.org/10.1108/01439919910284000>
- [17] ANGLMAYER, I. (2021) Machinery directive: Revision of directive 2006/42/ec. [Online]. Available: <https://policycommons.net/artifacts/1817942/machinery-directive/2555361/>
- [18] F. Bartolacci, A. Paolini, A. G. Quaranta, and M. Soverchia, "Assessing factors that influence waste management financial sustainability," *Waste Management*, vol. 79, pp. 571–579, 2018.
- [19] L. Bascetta, G. Ferretti, G. Magnani, and P. Rocco, "Walk-through programming for robotic manipulators based on admittance control," *Robotica (ROB)*, vol. 31, pp. 1143–1153, 10 2013. [Online]. Available: http://journals.cambridge.org/article_S0263574713000404
- [20] A. Blomdell, I. Dressler, K. Nilsson, and A. Robertsson, "Flexible application development and high-performance motion control based on external sensing and reconfiguration of abb industrial robot controllers," in *IEEE International Conference on Robotics and Automation (ICRA) - Workshop on Innovative Robot Control Architectures for Demanding (Research) Applications*, 2010, pp. 62–66. [Online]. Available: <https://lup.lub.lu.se/search/publication/1bd87fa1-d136-4a06-a4e7-661ae1daecc2>
- [21] L. Botti, D. Battini, F. Sgarbossa, and C. Mora, "Door-to-door waste collection: Analysis and recommendations for improving ergonomics in an italian case study," *Waste Management*, vol. 109, pp. 149–160, 2020.
- [22] M. G. Carmichael, D. Liu, and K. J. Waldron, "A framework for singularity-robust manipulator control during physical human-robot interaction," *International Journal of Robotics Research*, vol. 36, pp. 861–876, 6 2017.

- [23] CE Marking Association (UK). (Retrieved September 23, 2020) How is the ce mark enforced? [Online]. Available: <https://www.cemarkingassociation.co.uk/how-is-the-ce-mark-enforced/>
- [24] A. de Luca and R. Mattone, "Sensorless robot collision detection and hybrid force/motion control," in *IEEE International Conference on Robotics and Automation (ICRA)*, April 2005, pp. 999–1004.
- [25] F. Dimeas, V. C. Moulianitis, C. Papakonstantinou, and N. Aspragathos, "Manipulator performance constraints in cartesian admittance control for human-robot cooperation," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 3049–3054.
- [26] F. Dimeas, "Singularity avoidance in human-robot collaboration with performance constraints," pp. 89–100, 2021.
- [27] N. Dimitropoulos, T. Togias, G. Michalos, and S. Makris, "Operator support in human-robot collaborative environments using ai enhanced wearable devices," *Procedia CIRP*, vol. 97, pp. 464–469, 2021, 8th CIRP Conference of Assembly Technology and Systems. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2212827120314918>
- [28] S. Eppinger and W. Seering, "On dynamic models of robot force control," in *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, vol. 3, 1986, pp. 29–34.
- [29] C. Ericson, *Real-Time Collision Detection*, T. Cox, Ed. Elsevier, 2005. [Online]. Available: <http://realtimecollisiondetection.net/>
- [30] S. Farsoni, C. T. Landi, F. Ferraguti, C. Secchi, and M. Bonfè, "Compensation of load dynamics for admittance controlled interactive industrial robots using a quaternion-based kalman filter," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 672–679, April 2017.
- [31] F. Ferraguti, M. Bertuletti, C. T. Landi, M. Bonfè, C. Fantuzzi, and C. Secchi, "A control barrier function approach for maximizing performance while fulfilling to iso/ts 15066 regulations," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5921–5928, Oct 2020.
- [32] F. Ferraguti, A. Pertosa, C. Secchi, C. Fantuzzi, and M. Bonfè, "A methodology for comparative analysis of collaborative robots for industry 4.0," in *2019 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2019, pp. 1070–1075.
- [33] F. Ferraguti, M. Bertuletti, N. Battilani, and C. Fantuzzi, "A robotic system for automated door-to-door urban waste collection," To be submitted.
- [34] F. Ferraguti, M. Bertuletti, M. Gambazza, and M. Ragaglia, "High-velocity walk-through programming for industrial applications," *Robotics and Computer-Integrated Manufacturing*, vol. 81, p. 102505, 2023.

- [35] F. Ferraguti, M. Bertuletti, M. Gambazza, M. Ragaglia, and C. Fantuzzi, “Metodo per il controllo di sicurezza in autoapprendimento di un sistema robotizzato chiuso e relativo impianto di lavorazione,” Patent request 102 021 000 024 905.
- [36] —, “Metodo per il controllo in autoapprendimento di un sistema robotizzato chiuso e relativo impianto di lavorazione,” Patent request 102 021 000 024 899.
- [37] F. Ferraguti, C. T. Landi, S. Costi, M. Bonfè, S. Farsoni, C. Secchi, and C. Fantuzzi, “Safety barrier functions and multi-camera tracking for human–robot shared environment,” *Robotics and Autonomous Systems*, vol. 124, p. 103388, 2020.
- [38] F. Ferraguti, C. T. Landi, L. Sabattini, M. Bonfè, C. Fantuzzi, and C. Secchi, “A variable admittance control strategy for stable physical human–robot interaction,” *The International Journal of Robotics Research*, vol. 38, no. 6, pp. 747–765, 2019.
- [39] F. Ferraguti, C. Talignani Landi, S. Costi, M. Bonfè, S. Farsoni, C. Secchi, and C. Fantuzzi, “Safety barrier functions and multi-camera tracking for human-robot shared environment,” *Robotics and Autonomous Systems*, vol. 124, p. 103388, 2020.
- [40] F. Ferraguti, C. Talignani Landi, C. Secchi, C. Fantuzzi, M. Nolli, and M. Pesamosca, “Walk-through programming for industrial applications,” *Procedia Manufacturing (Proceedings of 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017, 27-30 June 2017, Modena, Italy)*, vol. 11, pp. 31 – 38, 2017. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S235197891730330X>
- [41] G. Ferretti, G. Magnani, and P. Rocco, “Assigning virtual tool dynamics to an industrial robot through an admittance controller,” in *International Conference on Advanced Robotics (ICAR)*, 2009, pp. 1–6. [Online]. Available: <https://ieeexplore.ieee.org/document/5174700>
- [42] GT-Engineering. IEC 60364-4-41: electric shock protection. [Online]. Available: <https://www.gt-engineering.it/iec-normative-tecniche/iec-60364-4-41/sistemi-it>
- [43] Guillaume Seguin, Karteek Alahari, Josef Sivic, and Ivan Laptev, “Pose estimation and segmentation of multiple people in stereoscopic movies,” *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 37, no. 8, pp. 1643–1655, 2015.
- [44] J. Han and G.-H. Yang, *Singularity Avoidance in Teleoperation System through Force Feedback of Master Device*. IEEE, 2017.
- [45] G. Hirzinger, A. Albu-Schaffer, M. Hahnle, I. Schaefer, and N. Sporer, “On a new generation of torque controlled light-weight robots,” in *IEEE International Conference on Robotics and Automation (ICRA)*, vol. 4, 2001, pp. 3356–3363 vol.4. [Online]. Available: <https://ieeexplore.ieee.org/document/933136>
- [46] International Organization for Standardization. (Retrieved November 18, 2020) Iso - members. [Online]. Available: <https://www.iso.org/members.html>

- [47] *ISO-2011-10218-1 - Robots and robotic devices - Safety requirements for industrial robots - Part 1: Robots*, ISO Std., 2011. [Online]. Available: <https://www.iso.org/standard/51330.html>
- [48] *ISO-2011-10218-2 - Robots and robotic devices - Safety requirements for industrial robots - Part 2: Robot systems and integration*, ISO Std., 2011. [Online]. Available: <https://www.iso.org/standard/41571.html>
- [49] *ISO/TS 15066 - Robots and robotic devices – Collaborative robots*, ISO Std., 2015. [Online]. Available: <https://www.iso.org/standard/62996.html>
- [50] ISO 10218-1-2011. (since July 1, 2011) Robots and robotic devices – Safety requirements for industrial robots. Part 1: Robots. [Online]. Available: <http://www.iso.org>
- [51] ISO 10218-2-2011. (since July 1, 2011) Robots and robotic devices – Safety requirements for industrial robots. Part 2: Robot systems and integration. [Online]. Available: <http://www.iso.org>
- [52] ISO 12100-2010. (2010) Safety of machinery - general principles for design - risk assessment and risk reduction. [Online]. Available: <http://www.iso.org>
- [53] ISO TS 15066:2016. (since February 15, 2016) Robots and robotic devices – Collaborative robots. [Online]. Available: <http://www.iso.org>
- [54] J. M. Jacinto-Villegas, M. Satler, A. Filippeschi, M. Bergamasco, M. Ragaglia, A. Argiolas, M. Niccolini, and C. A. Avizzano, “A novel wearable haptic controller for teleoperating robotic platforms,” *IEEE Robotics and Automation Letters (IEEE RAL)*, vol. 2, no. 4, pp. 2072–2079, Oct 2017.
- [55] Jacoplastic. Domestic collection: model 25 lt. [Online]. Available: <https://www.jcoplastic.com/index.pl?pos=01.01.01.03&lang=en&item=3&menu=01>
- [56] Jee-Seong Kim, Chul-Hong Kim, Yong-Min Shin, Il-Soo Cho, and Dong-Il “Dan” Cho, “Outdoor monocular visual odometry enhancement using depth map and semantic segmentation,” in *Proceedings. 2020 20th International Conference on Control, Automation and Systems (ICCAS)*, 2020.
- [57] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [58] D. Kubus, T. Kroger, and F. M. Wahl, “Improving force control performance by computational elimination of non-contact forces/torques,” in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 2617–2622.
- [59] V. Kurtz, P. M. Wensing, and H. Lin, “Control barrier functions for singularity avoidance in passivity-based manipulator control,” 9 2021. [Online]. Available: <http://arxiv.org/abs/2109.13349>

- [60] A. Lecours, B. Mayer-St-Onge, and C. Gosselin, "Variable admittance control of a four-degree-of-freedom intelligent assist device," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Minnesota, USA, 2012.
- [61] W. Lei, D. Zhou, P. Li, and G. Sun, "Singularity avoidance strategies for target tracking visual servo systems," 7 2021.
- [62] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," 06 2015, pp. 3431–3440.
- [63] A. M. Madsen, M. Raulf, P. Duquenne, P. Graff, M. Cyprowski, A. Beswick, S. Laitinen, P. U. Rasmussen, M. Hinker, A. Kolk, R. L. Górný, A. Oppliger, and B. Crook, "Review of biological risks associated with the collection of municipal wastes," *Science of The Total Environment*, vol. 791, p. 148287, 2021.
- [64] E. Magrini, F. Flacco, and A. De Luca, "Estimation of contact forces using a virtual force sensor," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 2126–2133.
- [65] N. Mansfeld, M. Hamad, M. Becker, A. G. Marin, and S. Haddadin, "Safety map: A robotics safety evaluation and safe robot design," Workshop on Autonomous Robot Design at the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, may 2018.
- [66] Md. Nafis Raihan, Mahbubur Rahman, Fahad Bin Alam, and Ekra Bin Syed Mojib, "A novel approach for waste collection: Automated waste collecting robot with advanced image recognition technology and onboard robotic arm," in *Proceedings. 2020 IEEE Region 10 Symposium (TENSYP)*, 2020.
- [67] G. Michalos, P. Karagiannis, N. Dimitropulos, D. Andronas, and S. Makris, *Human Robot Collaboration in Industrial Environments*.
- [68] G. Michalos, N. Kousi, P. Karagiannis, C. Gkournelos, K. Dimoulas, S. Koukas, K. Mparis, A. Papavasileiou, and S. Makris, "Seamless human robot collaborative assembly – an automotive case study," *Mechatronics*, vol. 55, pp. 194–211, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0957415818301326>
- [69] M. Nasr, M. Marey, M. M. Abdelhameed, and F. A. Tolbah, "Using genetic algorithm for singularity avoidance in positioning tasks of a robotic arm," p. 163, 2018.
- [70] R. Ortega, J. A. L. Perez, P. J. Nicklasson, and H. J. Sira-Ramirez, *Passivity-based control of Euler-Lagrange systems: mechanical, electrical and electromechanical applications*. Springer Science & Business Media, 2013.
- [71] Z. Pan, J. Polden, N. Larkin, S. Van Duin, and J. Norrish, "Recent progress on programming methods for industrial robots," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 87–94, 2012. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0736584511001001>

- [72] O. M. Poulsen, N. O. Breum, N. Ebbenhøj, Åse Marie Hansen, U. I. Ivens, D. van Lelieveld, P. Malmros, L. Matthiasen, B. H. Nielsen, E. M. Nielsen, B. Schibye, T. Skov, E. I. Stenbaek, and K. C. Wilkins, "Sorting and recycling of domestic waste. review of occupational health problems and their possible causes," *Science of The Total Environment*, vol. 168, no. 1, pp. 33–56, 1995.
- [73] M. Ragaglia, L. Bascetta, P. Rocco, and A. M. Zanchettin, "Integration of perception, control and injury knowledge for safe human-robot interaction," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 1196–1202.
- [74] M. Ragaglia, A. M. Zanchettin, and P. Rocco, "Safety-aware trajectory scaling for human-robot collaboration with prediction of human occupancy," in *International Conference on Advanced Robotics (ICAR)*, July 2015, pp. 85–90.
- [75] M. Ragaglia, A. M. Zanchettin, L. Bascetta, and P. Rocco, "Accurate sensorless lead-through programming for lightweight robots in structured environments," *Robotics and Computer-Integrated Manufacturing (RCIM)*, vol. 39, pp. 9 – 21, 2016. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S073658451500126X>
- [76] M. Ragaglia, A. M. Zanchettin, and P. Rocco, "Trajectory generation algorithm for safe human-robot collaboration based on multiple depth sensor measurements," *Mechatronics (MECH)*, vol. 55, pp. 267–281, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0957415817301861>
- [77] M. Rauscher, M. Kimmel, and S. Hirche, "Constrained robot control using control barrier functions," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 279–285.
- [78] D. Reyes-Uquillas and T. Hsiao, "Safe and intuitive manual guidance of a robot manipulator using adaptive admittance control towards robot agility," *Robotics and Computer-Integrated Manufacturing*, vol. 70, p. 102127, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0736584521000119>
- [79] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.
- [80] J. Schmidt-Hieber, "Nonparametric regression using deep neural networks with relu activation function," *Ann. Statist.*, vol. 48, no. 4, pp. 1875 – 1897, 2020.
- [81] M. Schwenzer, S. Stemmler, M. Ay, A. K. Ruppel, T. Bergs, and D. Abel, "Model predictive force control in milling based on an ensemble kalman filter," *Journal of Intelligent Manufacturing*, 2022.
- [82] Sherlock H2020 Project, "Physical Human-Robot interaction for HRC applications," <https://youtu.be/rFuxTkktI>, 2022, [Online; accessed 02-August-2022].

- [83] B. Siciliano and J. . E. Slotine, “A general framework for managing multiple tasks in highly redundant robotic systems,” in *Fifth International Conference on Advanced Robotics ‘Robots in Unstructured Environments*, 1991, pp. 1211–1216 vol.2.
- [84] Siciliano B.,Sciavicco L.,Villani L.,Oriolo G., *Robotics: Modelling, Planning and Control*.
- [85] SN EN 60825-1:2014. (since May 15, 2014) Safety of laser products - part 1: Equipment classification and requirements. [Online]. Available: <https://webstore.iec.ch/publication/3587>
- [86] S. Tafazoli, S. E. Salcudean, K. Hashtrudi-Zaad, and P. D. Lawrence, “Impedance control of a teleoperated excavator,” *IEEE Transactions on Control Systems Technology (IEEECST)*, vol. 10, no. 3, pp. 355–367, May 2002.
- [87] C. Talignani Landi, F. Ferraguti, C. Secchi, and C. Fantuzzi, “Tool compensation in walk-through programming for admittance-controlled robots,” in *IEEE Annual Conference of the Industrial Electronics Society (IECON)*, Oct 2016, pp. 5335–5340.
- [88] M. Tanzini, J. M. Jacinto-Villegas, A. Filippeschi, M. Niccolin, and M. Ragaglia, “New interaction metaphors to control a hydraulic working machine’s arm,” in *IEEE Symposium of Safety and rescue Robotics (SSRR)*, 2016. [Online]. Available: <http://ieeexplore.ieee.org/document/7784319/>
- [89] T. Togias, C. Gkournelos, P. Angelakis, G. Michalos, and S. Makris, “Virtual reality environment for industrial robot control and path design,” *Procedia CIRP*, vol. 100, pp. 133–138, 2021, 31st CIRP Design Conference 2021 (CIRP Design 2021). [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2212827121004807>
- [90] Tom Mertens, Jan Kautz, and Frank Van Reeth, “Exposure fusion,” *Computer Graphics and Applications*, p. pages 382–390, 2007.
- [91] P. Tripicchio, E. Ruffaldi, P. S. Gasparello, S. Eguchi, J. Kusuno, K. Kitano, M. Yamada, A. Argiolas, M. Niccolini, M. Ragaglia, and C. A. Avizzano, “A stereo-panoramic telepresence system for construction machines,” *Procedia Manufacturing (PROCMAN)*, vol. 11, pp. 1552 – 1559, 2017. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S2351978917305000>
- [92] R. Tsai and R. Lenz, “Real time versatile robotics hand/eye calibration using 3d machine vision,” in *Proceedings. 1988 IEEE International Conference on Robotics and Automation*, 1988, pp. 554–561 vol.1.
- [93] T. Tsumugiwa, R. Yokogawa, and K. Hara, “Variable impedance control based on estimation of human arm stiffness for human-robot cooperative calligraphic task,” in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 1, 2002, pp. 644–650 vol.1.

- [94] Tsunetake Kanatani, Hideyuki Kume, Takafumi Taketomi, Tomokazu Sato and Naokazu Yokoya, “Detection of 3d points on moving objects from point cloud data for 3d modeling of outdoor environments,” in *Proceedings. ICIP 2013*, 2013.
- [95] UNI EN 1501-2-2010. (2010) Refuse collection vehicles - general requirements and safety requirements – part 2: Side loaded refuse collection vehicles to. [Online]. Available: <https://www.uni.com>
- [96] F. Vicentini, “La robotica collaborativa,” *Sicurezza e flessibilità delle nuove forme di collaborazione. Tecniche Nuove, Milano*, 2017.
- [97] L. Villani and J. De Schutter, *Force Control*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 161–185. [Online]. Available: https://doi.org/10.1007/978-3-540-30301-5_8
- [98] R. M. Voyles Jr, J. D. Morrow, and P. K. Khosla, “A comparison of force sensors,” *Advanced manipulators laboratory, the Robotic Institute Carnegie Mellon University*, 1994.
- [99] VTC. E-Mark Certification. [Online]. Available: <http://vtc-test.com.cn/services/show.php?lang=en&id=256>
- [100] A. Vysocký, H. Wada, J. Kinugawa, and K. Kosuge, “Motion planning analysis according to iso/ts 15066 in human–robot collaboration environment,” in *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 2019, pp. 151–156.
- [101] A. Wahrburg, J. Bos, K. D. Listmann, F. Dai, B. Matthias, and H. Ding, “Motor-current-based estimation of cartesian contact forces and torques for robotic manipulators and its application to force control,” *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 879–886, 2018.
- [102] C. W. Wampler, “Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods,” *IEEE Transactions on Systems, Man, and Cybernetics (IEEE TSMC)*, vol. 16, no. 1, pp. 93–101, Jan 1986.
- [103] L. Wang, A. D. Ames, and M. Egerstedt, “Multi-objective compositions for collision-free connectivity maintenance in teams of mobile robots,” in *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 2016, pp. 2659–2664.
- [104] Q. Wang, W. Wang, L. Zheng, and C. Yun, “Force control-based vibration suppression in robotic grinding of large thin-wall shells,” *Robotics and Computer-Integrated Manufacturing*, vol. 67, p. 102031, 2021. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0736584520302428>
- [105] T. Yoshikawa, “Translational and rotational manipulability of robotic manipulators.” IEEE, 1991, pp. 1170–1175. [Online]. Available: <http://ieeexplore.ieee.org/document/239275/>

- [106] A. M. Zanchettin, N. M. Ceriani, P. Rocco, H. Ding, and B. Matthias, “Safety in human-robot collaborative manufacturing environments: Metrics and control,” *IEEE Transactions on Automation Science and Engineering (IEEE TASE)*, vol. 13, no. 2, pp. 882–893, April 2016.
- [107] T. Zhang, Y. Yu, and Y. Zou, “An adaptive sliding-mode iterative constant-force control method for robotic belt grinding based on a one-dimensional force sensor,” *Sensors*, vol. 19, p. 1635, 04 2019.
- [108] P. Švarný, M. Tesar, J. Behrens, and M. Hoffmann, “Safe physical hri: Toward a unified treatment of speed and separation monitoring together with power and force limiting,” 11 2019, pp. 7580–7587.