Multi-modal Human-Robot Interaction to Improve Efficiency and Flexibility in a Shared Workspace

Zur Erlangung des akademischen Grades eines
Doktors der Ingenieurwissenschaften
(Dr.-Ing.)

von der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte
DISSERTATION

von
M.Sc. Gabriele Bolano

Datum der mündlichen Prüfung:	2. Mai 2023
Referent:	Prof. Dr.-Ing. Rüdiger Dillmann
Korreferent:	Prof. Dr.-Ing. Rainer Stiefelhagen

Declaration of Originality

This is to certify that the content of this thesis is the outcome of my independent and original work. I certify that the intellectual content of this thesis is the product of my own research and that all the assistance received in preparing this work and sources have been acknowledged.

Karlsruhe, *Gabriele Bolano*

March 2023 *Karlsruher Institut für Technologie (KIT)*

Credits

I take this section to thank all the people that helped or supported me in some way during the path that led me to this achievement. First of all I am sorry if I forget someone, but I am sure that you know if you are one of these people.

I would like to thank Prof. Dr.-Ing. Rüdiger Dillmann for the great support through these years and for making true the opportunity to finish this thesis.

A really huge thank you to the IDS department leader Dr.-Ing. Arne Rönnau, for always supporting me and believing in my ideas and PhD work. I really appreciate the way you pushed to tackle challenging tasks, keeping always the fun part of the job and the team spirit.

For the same reason I would like to thank Georg Heppner, for the great support in many projects and the positive attitude also during tight deadlines.

A big thanks to all the IDS team, for the great time spent together and the many hours spent together in fun (and also less fun) projects.

To Stefan Scherzinger for trying always to push my German and keeping the motivation to improve that.

To Christin Jülg for introducing me to the topics that made this thesis possible and the many interesting talks.

To Jakob Weinland for the time spent together on one of the hardest projects and being able to turn that into some fun. I will remember our great filters.

To Genessis Perez Rivera for bringing also some internationality to the team in the last period working at FZI and for sharing some of the most troublesome projects. Sorry for giving up on the last one, I did actually quite enjoyed the business trips to Munich.

To Philip Keller for the interesting exchanges of ideas and the nice trips made together.

To Lea Steffen for the random office talks and for sharing the cool conference trip to Brazil. Next time we will need to skip the waterfalls or get a better phone cover.

To Atanas Tanev for sharing long days in the office, as well as low motivation moments and for always being available for an exchange of ideas or just a relaxed talk after hours of work. Let's celebrate with a Piña Colada after this.

To the many project partners for the great time spent on tiring business trips and long integration sessions, finding anyway time to celebrate a successful project milestone together.

To the students I have supervised for supporting my research ideas and providing great efforts and results.

To Christoph Semperowitsch for having shared a very nice WG together and for always having time for a nice chat after work.

To all my friends from Italy which were not directly involved in the work which led to this thesis but did support me by just being there and by spending great moments together everytime I was back home. You always managed to bring a smile to my face and I think this is definitely something that helped me to achieve this goal anyway.

To my girlfriend Ambra for having supported me from the beginning of this path, even tough it was not easy at some points for obvious logistic reasons. Thank you for always encouraging me and for putting up with me in many situations. I know that sometimes I managed to be very annoying when I needed to face some work related issue and I appreciate how you always tried to make me focus on how to solve difficult situations. Thanks again.

A huge thank you to my family.

To Marino for being always supportive.

To my sister who always believed in what I do and has everytime shown high appreciation for my decisions.

To my mother for supporting me in every decision and making everything to make me able to achieve my goals. I know that is not easy and I am really thankful for that.

Abstract (English Version)

Nowadays, robots are able to work close to humans and execute tasks that require to share the workspace with them. Many works have investigated how to make robots able to foresee the human's motion and intentions, in order to avoid collisions and react in a dynamic way. Anyway, these methods usually rely on stopping or slowing down the robot, which is highly inefficient.

In a close collaboration is also really important that the members of the team have a clear idea of the intention and motion of each other. In research there is a lack of works on how to represent the robot intentions and motion to the human. The communication of this information has to be as intuitive as possible in order to make the human able to understand it easily and quickly, enabling him to react to that without losing the focus on his task. The use of visual information on screens enables the visualization of virtual information related to the robot motion, in order to represent its next planned trajectories and the volume that it will occupy in the near future. Anyway, this is not optimal since it requires the user to divert his attention and focus to such external devices. A clear communication of this information is crucial to make the human more comfortable in the interaction and to achieve a better efficiency in the task completion. Another fundamental aspect is also the importance of methods to program and control the robot with intuitive interfaces, in order to reduce long and complex programming phases and the need of experts. Novel methods for online programming could further improve to efficiency in the reprogramming phases, without the need to stop the application. For example, the robot trajectory could be adapted at runtime during execution, as well as the relevant parameters and workspace constraints. Once the user is able to clearly understand the robot intentions and planned motion, it is important to provide him with a way to change or agree with the robot plan, eventually adapting it to his preferences. In order to provide also this information to the robot, different channels of communication could be used to make the interaction as intuitive and easy as possible. The natural way of communication for humans is through speech and gestures, such as finger pointing, face expression, head motion and eye gaze. Providing virtual user interfaces, the user can easily communicate his intentions and needs in an intuitive way. Virtual objects can be displayed and manipulated through the use of mixed-reality in order to change and adapt the robot motion and behavior.

This work proposes different methods to to represent robot intention and motion information to the human in an intuitive way, exploring and combining different channels of communication. The use of mixed-reality allows the representation

of this information directly in the workspace, overlapping virtual objects at the needed position in the real world. The swept volume of the robot trajectory is represented directly in the workspace through the use of projections and 3D virtual objects in AR, giving the user an exact and clear idea about the robot planned motion. The use of acoustics signals, as sounds and synthesized speech, are also an intuitive and effective way to alert and inform the human about the state of the robot plan, its next goal and possible collisions. A combination of these communication methods, in order to improve the ease of the communication of the robot's intents, is proposed and evaluated.

This work investigates also novel communication interfaces and how to combine them to improve the Human-Robot interaction. Once the human receive and understand the information about the robot intentions, he can then take more intelligent decisions for the completion of his task in the shared workspace, with a consequent better efficiency and comfort in the interaction with the robot.

In order to have a better flexibility in the interaction, this work investigates and proposes methods to modify the robot plan and trajectory at runtime, as well as interfaces to define workspace constraints, which needs to be considered by the robot during execution. For example, the user might need to work alone in a predefined area of the workspace and he should be able to easily communicate to the robot to avoid a specific area without the need to stop the application or making complex changes in the robot program. In order to provide also this information to the robot, different channels of communication are proposed to make this interaction as intuitive and easy as possible. The use of intuitive and natural interfaces for robot programming and motion modification, allows inexperienced user to interact with the system, decreasing the time needed for training and programming. The improvements in the interaction are evaluated in a realistic assembly scenario, where the efficiency benefits of the communication system are measured and evaluated.

Furthermore, collision avoidance and dynamic task scheduling approaches to improve the efficiency of the collaboration are proposed. Metrics to evaluate the improvements in safety and task completion time are investigated and evaluated.

Abstract (German Version)

Heutzutage sind Roboter in der Lage, in der Nähe von Menschen zu arbeiten und Aufgaben auszuführen, die es erfordern, den Arbeitsbereich mit ihnen zu teilen. In vielen Arbeiten wurde untersucht, wie Roboter in die Lage versetzt werden können, die Bewegungen und Absichten des Menschen vorherzusehen, um Kollisionen zu vermeiden und dynamisch zu reagieren. Allerdings sind diese Methoden darauf angewiesen, den Roboter anzuhalten oder zu verlangsamen, was sehr ineffizient ist.

In der Forschung gibt es einen Mangel an Arbeiten darüber, wie die Absichten und Bewegungen des Roboters für den Menschen dargestellt werden können. In einer engen Zusammenarbeit ist es sehr wichtig, dass die Teammitglieder eine klare Vorstellung von den Absichten und Bewegungen des jeweils anderen haben. Die Kommunikation dieser Informationen muss so intuitiv wie möglich sein, damit der Mensch sie leicht und schnell verstehen kann, so dass er darauf reagieren kann, ohne den Fokus auf seine Aufgabe zu verlieren. Die Verwendung visueller Informationen auf Bildschirmen ermöglicht die Visualisierung virtueller Informationen im Zusammenhang mit der Bewegung des Roboters, um seine nächsten geplanten Trajektorien und das Volumen, das er in naher Zukunft einnehmen wird, darzustellen. Dies ist jedoch nicht optimal, da der Benutzer seine Aufmerksamkeit und seinen Fokus auf solche externen Geräte lenken muss. Eine klare Kommunikation dieser Informationen ist von entscheidender Bedeutung, um dem Menschen die Interaktion angenehmer zu machen und eine bessere Effizienz bei der Aufgabenerfüllung zu erreichen. Ein weiterer grundlegender Aspekt ist die Bedeutung von Methoden zur Programmierung und Steuerung des Roboters mit intuitiven Schnittstellen, um lange und komplexe Programmierphasen und den Bedarf an Experten zu reduzieren. Neuartige Methoden zur Online-Programmierung könnten die Effizienz bei der Neuprogrammierung des Roboters weiter verbessern, ohne dass die Anwendung angehalten werden muss. So könnten beispielsweise die Roboterbahn zur Laufzeit angepasst werden, ebenso wie die relevanten Parameter und Arbeitsraumbeschränkungen. Sobald der Benutzer in der Lage ist, die Absichten und die geplante Bewegung des Roboters klar zu verstehen, ist es wichtig, ihm die Möglichkeit zu geben, den Roboterplan zu ändern oder ihm zuzustimmen und ihn schließlich an seine Präferenzen anzupassen. Um dem Roboter auch diese Informationen zur Verfügung zu stellen, könnten verschiedene Kommunikationskanäle genutzt werden, um diese Interaktion so intuitiv und einfach wie möglich zu gestalten. Die natürliche Art der Kommunikation für den Menschen ist die Sprache und Gesten, wie Fingerzeig, Gesichtsausdruck, Kopfbewegung und Blicke. Durch die Bereitstellung von virtuellen Benutzerschnittstellen kann

der Benutzer seine Absichten und Bedürfnisse auf einfache und intuitive Weise kommunizieren. Virtuelle Objekte können durch den Einsatz von Mixed-Reality dargestellt und manipuliert werden, um die Bewegung und das Verhalten des Roboters zu verändern und anzupassen.'

In dieser Arbeit werden verschiedene Methoden vorgeschlagen, um die Absichten und Bewegungsinformationen des Roboters für den Menschen auf intuitive Weise darzustellen, wobei verschiedene Kommunikationskanäle erforscht und kombiniert werden. Die Verwendung von Mixed-Reality ermöglicht die Darstellung dieser Informationen direkt im Arbeitsraum, indem virtuelle Objekte an der benötigten Position in der realen Welt überlagert werden. Durch die Verwendung von Projektionen und virtuellen 3D-Objekten in AR wird das überstrichene Volumen der Robotertrajektorie direkt im Arbeitsraum dargestellt, wodurch der Benutzer eine genaue und klare Vorstellung von der geplanten Roboterbewegung erhält. Die Verwendung von akustischen Signalen, wie Tönen und synthetischer Sprache, ist ebenfalls ein intuitiver und effektiver Weg, um den Menschen über den Status des Roboterplans, sein nächstes Ziel und mögliche Kollisionen zu informieren. Eine Kombination dieser Kommunikationsmethoden wird verwendet, um die Kommunikation der Absichten des Roboters zu erleichtern.

In dieser Arbeit werden auch neuartige Kommunikationsschnittstellen und deren Kombination zur Verbesserung der Mensch-Roboter-Interaktion untersucht. Sobald der Mensch die Informationen über die Absichten des Roboters erhält und versteht, kann er intelligentere Entscheidungen für die Erledigung seiner Aufgabe im gemeinsamen Arbeitsbereich treffen, was zu einer besseren Effizienz und einem höheren Komfort bei der Interaktion mit dem Roboter führt.

Um eine bessere Flexibilität bei der Interaktion zu erreichen, werden in dieser Arbeit Methoden zur Änderung des Roboterplans und der Flugbahn zur Laufzeit sowie Einschränkungen des Arbeitsbereichs vorgeschlagen, die vom Roboter während der Ausführung berücksichtigt werden müssen. So kann es beispielsweise vorkommen, dass der Benutzer in einem vordefinierten Bereich des Arbeitsraums allein arbeiten muss, und er sollte in der Lage sein, dem Roboter auf einfache Weise mitzuteilen, dass er den gewünschten Bereich meiden soll, ohne dass er die Anwendung anhalten oder komplexe Änderungen am Roboterprogramm vornehmen muss. Um dem Roboter auch diese Informationen zur Verfügung zu stellen, werden verschiedene Kommunikationskanäle vorgeschlagen, um diese Interaktion so intuitiv und einfach wie möglich zu gestalten. Die Verwendung intuitiver und natürlicher Schnittstellen für die Roboterprogrammierung und Bewegungsmodifikation ermöglicht es unerfahrenen Benutzern, mit dem System zu interagieren und die für die Schulung und Programmierung benötigte Zeit zu verkürzen. Die Verbesserungen in der Interaktion werden in einem realistischen Montageszenario evaluiert, wobei die Effizienzvorteile des Kommunikationssystems gemessen und bewertet werden.

Darüber hinaus werden Ansätze zur Kollisionsvermeidung und dynamischen Aufgabenplanung vorgeschlagen, um die Effizienz der Zusammenarbeit zu ver-

bessern. Metriken zur Bewertung der Verbesserungen bei der Sicherheit und der Zeit für die Aufgabenerledigung werden untersucht und bewertet.

Contents

Acronyms

ALSA Advanced Linux Sound Architecture.

AR Augmented Reality.

CAD Computer-Aided Design.

CPU Central Processing Unit.

CUDA Compute Unified Device Architecture.

DOF Degrees of Freedom.

FPS Frames per Second.

GPU Graphics Processing Unit.

GUI Graphical User Interface.

HAT Hardware Attached on Top.

HMD Head-Mounted Display.

HRC Human-Robot Collaboration.

HRI Human-Robot Interaction.

IER Intent Error Rate.

MEMS Micro Electro Mechanical Systems.

NLU Natural Language Understanding.

OS Operating System.

PCL Point Cloud Library.

ROS Robot Operating System.

STT Speech To Text.

TCP Tool Center Point.

TTS Text To Speech.

UI User Interface.

URDF Unified Robot Description Format.

VR Virtual Reality.

Glossary

Augmented Reality is an interactive experience that combines real world and computer-generated content.

Collision Avoidance is the plan for action that the robot takes to evade a detected collision.

CUDA is a programming technique with a runtime environment of the same name, with which algorithms for GPUs can be compiled and executed on them. CUDA is developed by Nvidia exclusively for graphics cards of its own brand.

End-Effector is the tool at the end of a robotic arm with which objects are gripped or manipulated.

FlexBE is a flexible and user-friendly behavior engine for ROS.

GPU-Voxels is a CUDA based library which allows high resolution volumetric collision detection between animated 3D models and live pointclouds from 3D sensors.

Human-Robot Collaboration is the study of collaborative processes in which human and robot agents work together to achieve shared goals.

Human-Robot Interaction is the study of interactions between humans and robots.

Kinematic Configuration describes the arrangement of all movable axes of a robot with respect to each others and thus its mobility.

Multi-Modal Communication is a method of communicating using a variety of methods, including verbal language, sign language and different types of communication.

RGBD-Camera is a type of depth camera that provides both depth (D) and color (RGB) data as the output in real-time.

Robot Configuration describes the state of all moving axes of a robot and thus its body posture.

Robot Programming is programming the controller inside a robot to perform a specific task using actuators and feedback from various sensor.

Robot Safety is an aspect of occupational safety and health when robots are used in the workplace.

Robot Simulation is a method to create an application for a physical robot without depending on the physical machine, thus saving cost and time.

ROS is an open-source robotics middleware for robot software development.

Speech Recognition is an interdisciplinary subfield of computer science and computational linguistics that develops methodologies and technologies that enable the recognition and translation of spoken language.

Unity is a cross-platform game engine that can be used to create 3D games and interactive simulations. In this thesis is used to design VR and AR environments and interfaces.

Virtual Reality is a simulated experience that employs pose tracking and 3D near-eye displays to give the user an immersive feel of a virtual world.

Voxel designates a cubic volume in three-dimensional space, which is the smallest unit of the space partitioning for the GPU-Voxles maps used in this work. A voxel can take different states and be associated to one or more entities.

1 Introduction

1.1 Motivation

Today robots are able to work closely to humans and they are usually programmed to perform complex tasks. Many research works have focused on improving the robot capability of understanding its surroundings, proposing methods to make it able to perceive objects and humans, as well as predict their motion and intentions. On the other hand, the problem of understanding the robot intent and planned motion is still an under-researched topic. However, in a close collaboration it is really important that both participants understand the intentions of each other, in order plan tasks and motions in an efficient way. Human are communicating their intentions all the time using implicit and explicit cues, such as gaze and gestures, which are hard to replicate for robots with no anthropomorphic features. For this reason, it is really important to design and develop novel methods to communicate intentions for robotic manipulators. At the same time, understanding the robot intent and planned motion, the user needs intuitive interfaces to adapt the robot plan to his needs in order to fulfill the task or handle unexpected events.

The programming of robots is a complex task. Research studies have investigated how to use simulation tools to enable the programming of robots for non-experts, but this is usually limited to simple scenarios. Furthermore, the gap between simulation and real world is often a problem, since an approach to gradually and safely test in the real world the programs defined in simulation is not available. In the same way, the control and teleoperation of robots for complex manipulation tasks is difficult and current methods in literature propose intuitive interfaces with limitations in flexibility.

Another critical aspect in HRC is related to safety and efficiency. Safety concepts usually rely on stopping or slowing down the robot as soon as the user gets in its proximity. This is highly inefficient, forcing the robot to wait in an idle state without progressing in the execution of its tasks. To enable a safe and efficient HRI is important to provide methods for real-time collision detection which could be used for an efficient task scheduling based on the position of dynamic obstacles.

1.2 Research Questions

The research goals of this thesis are the following:

Research goal 1. How to program complex robotic scenarios for non-experts and interactively?

Research goal 2. How to enable a safe and efficient HRI?

Research goal 3. How to represent robot intents and motions in an intuitive way?

Research goal 4. How to use multi-modal communication to adapt the robot plan and motion at runtime in a flexible way?

1.3 Approach

The proposed approach can be subdivided in four main steps as shown in Figure 1.1 and Figure 1.2:

- A system for intuitive robot programming to make users with no robotic experience able to program complex robotic systems and simulate their behavior before executing anything on the real hardware. Intuitive gesture-based interfaces can also be used to easily control or teleoperate a robot for complex manipulation tasks.
- The robot needs to provide safety features and collision avoidance capabilities, which can be combined with a dynamic task scheduling to improve the efficiency of the interaction.
- The robots needs to communicate its intent and planned motion to the user in an intuitive and effective way. The user needs to understand this information as quickly as possible and a multi-modal communication can help the human to understand the robot plan and planned swept volume, for a better efficiency and ergonomics in the interaction.
- Multi-modal interfaces to modify the robot plan and motion at runtime, enable the user to adapt the robot trajectory execution without the need to stop the application for long and complex reprogramming phases.

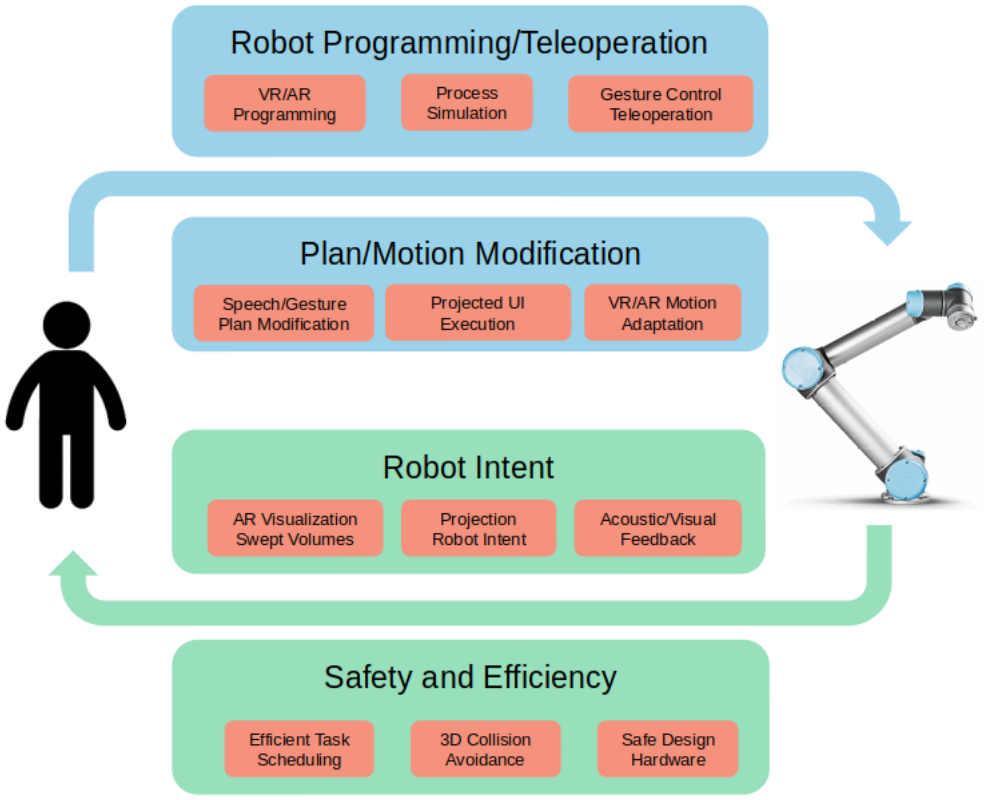


Figure 1.1: Overall overview of the proposed approach.

1.4 Key Contributions

The main scientific contributions are summarized in the following. The details of each contribution are presented and discussed in its respective chapter.

- **Programming, simulation and control of complex robotic systems for non-experts:** Methods to enable the programming and simulation of complex robotic systems for users with no programming experience are proposed. The focus is also on bridging the gap between simulation and real world, proposing a method based on three steps. The use of VR allows to define offline the desired environment and program the robot motion. AR can then be used to check the robot trajectories in the real world, for further adaptations in the real environment and to check the correctness before execution on the real hardware. Finally, the program can be directly executed on the real hardware, without the need of any further programming. Furthermore, novel approaches to control and teleoperate a robot for complex manipulation tasks using intuitive gesture-based interfaces are proposed and evaluated.
- **Safe and efficient collision avoidance:** Safety is a critical aspect in HRC. The safety effects and metrics to evaluate them, using non-safe external control

authorities in a collaborative scenario, are investigated. In order to improve the efficiency of the collaboration and avoid the usual approach that consists of slowing down or stop the robot when a possible obstacle is detected, a system for online replanning of the robot motion is proposed. The proposed method focuses on a GPU-based 3D collision avoidance system to allow collision prevention in real-time and enabling a dynamic task scheduling in order to improve the efficiency in the task completion. A set of metrics to evaluate the efficiency benefits of using such system have been proposed. Experiments have been conducted in a realistic assembly HRC scenario.

- **Communication of robot intention and motion in a multi-modal manner:** Robots are becoming really dynamic, but usually they don't provide interfaces to communicate their intentions and motions. Novel methods to communicate the robot plan and planned trajectory are proposed. The use of head-mounted displays and projection-based AR allows the representation of the swept volume of the robot directly in the workspace, enabling the user to understand the robot plan while working in close collaboration with the robot. A combination of visual and acoustic feedback is proposed for a more effective feedback and draw the attention of the user while he executes his task. With a proper understanding of the robot motion and plan, the worker can have better ergonomics during the interaction and cooperate with the robot more efficiently.
- **Multi-modal online robot plan and motion modification:** Enabling the communication and understanding of the robot intents and motions, the user might need to adapt the robot plan and trajectory execution according to his needs. A multi-modal approach to modify the robot plan and motion is proposed. Different interfaces are proposed in order to enable a flexible adaptation of the robot motion during runtime. Natural interfaces, which can be used by unexperienced operators, can reduce reprogramming phases as well as the time needed for training and the overall interaction time. The proposed methods rely on the use of projections, VR/AR, speech and gesture recognition to provide intuitive and flexible interfaces to the user.

1.5 Structure of the Thesis

The thesis is structured as follows. Chapter 2 presents the related work in the field of Human-Robot Interaction, with particular focus on robot intent communication, multi-modal and intuitive interfaces, VR/AR and robot programming. The intuitive robot programming system for simulation and evaluation of complex robotic scenarios is described in Chapter 3. The gesture-based methods for robot control and teleoperation are also presented and evaluated. The methods to enable a safe and efficient collision avoidance system are described in Chapter 4. This chapter presents the method for a dynamic task scheduling in order to improve the efficiency in a shared workspace, including relevant metrics to evaluate the

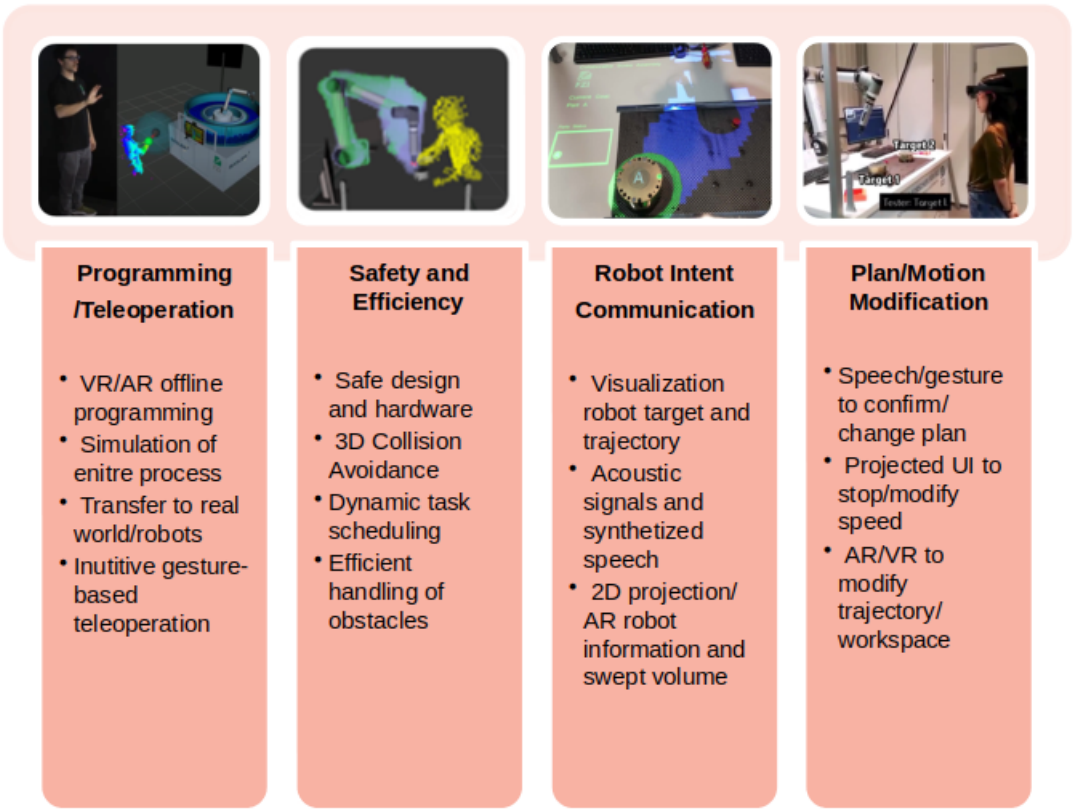


Figure 1.2: The main topics investigated in this thesis and details about the proposed approach for each step.

benefits in a collaborative task. The robot intent communication methods are described in Chapter 5. In this chapter, the multi-modal approaches to communicate the robot plan and motion to the user are presented and evaluated. The methods to adapt the robot plan and motion at runtime are described in Chapter 6. The multi-modal interfaces to enable a flexible and intuitive modification of the robot trajectories and execution parameters are presented and evaluated. Chapter 7 presents a summary of the contributions and conclusions.

2 Related Work

As robots started to work outside of fences and close to humans, Human-Robot Collaboration and Interaction have become crucial topics of research.

Many studies have investigated the different aspect of HRC. In this section are presented the relevant works in the field of HRI, with particular focus on robot programming and teleoperation, safety and collision avoidance, robot intents communication and multi-modal interfaces.

2.1 Robot Programming and Teleoperation

2.1.1 Robot Programming

Due to their complexity, many researchers have studied how to ease the programming of robotic systems, in order to make this possible for inexperienced users [60]. Another important aspect is how to speed up the process for defining the configuration of the desired robot trajectory and task workflow, without losing the precision and flexibility in the definition.

The most used method to teach robotic manipulators is the waypoints definition of trajectories using a teach pendant [12]. This device has been used since the early years of robotics, in order to easily move and program a robotic arm [20]. This method of programming allows to have a precise definition of the robot trajectories, defining the desired position of the robot in joint or Cartesian space. The drawback of this method is that it is time consuming for the definition of new configurations [78]. The joint-based teaching is a tedious process and also in Cartesian space the user needs to always check the robot reference frame used to express the position coordinates. For this reason, many robots nowadays provide a lead-through teaching mode, in order to enable the user to move them manually pressing a button on the teach pendant, which can be used then for storing the desired configuration of the manipulator in the program. The drawback of this method is that moving the robot by hand, the user needs to get in contact with the robot, which could not be allowed in some applications and it might require physical effort. Furthermore, these programming methods require the deployment of the robot for the definition of the program, which could be used in the meantime to perform the current task or for other purposes.

Simulation tools allow the user to define a program using a virtual representation of the robot without the need to get in physical contact with the real one. Furthermore, they enable the user to teach the robot offline or remotely, without the need to be on site or stop the current task executed by the robot. Virtual Reality (VR) is a promising tool that has been investigated in many works, for example to teleoperate a robot remotely, using a simulated representation of it [81], [61], [73]. The possibility to directly interact with a realistic simulation of the robot, has been also used to ease the robot programming, making this as intuitive as possible through the use of user friendly interfaces and the visualization of menus to store the needed configurations [68]. The user can for example interact with a virtual representation of the robot, dragging the desired link to the needed position. Buttons, menus and text information can be displayed at specific positions in order to enable an intuitive programming for the inexperienced user. Augmented Reality (AR) is another technology that is gaining interest for many robotic applications [42]. This is due to technological advances in recent years, which have led to application in robotics for programming [23], [58], [12]. Krings et al. [56] propose an AR-assisted approach to define different path-planning strategies for wheeled mobile robots. The method relies on the possibility to see and edit the defined paths, providing the visualization of the simulate execution. The evaluation results showed that the proposed approach is easier to use and faster than other conventional methods. In the work from Ong et al. an AR-based programming system for industrial applications in presented [76]. Through the use of a hand-held pointer, the user can define the waypoints and the path that the real robot has to follow. The approach includes the use of sensor data in order to include path planing and collision detection algorithms. Unfortunately, these methods do not allow an high precision in the definition of the robot trajectories. In fact, this relies on the accuracy of the simulation, which might lead to problems in tasks that have such requirements.

The combination of these methods can enable an efficient, flexible, intuitive and precise way of programming a robot. An offline method can be used to define quickly and intuitively the robot program. The real hardware can then be used only for small adjustments, which are needed to be performed online. In the existing literature, there is lack of work on this issue. Only few works have investigated how to integrate VR tools with existing open-source frameworks commonly used to control robotic systems [121]. Codd-Downey et al. proposed a system to integrate Unity [112] and ROS (Robot Operating System) [88] using a yaml-based communication protocol over web sockets [37]. The authors developed the system for a teleoperation task involving a mobile robot. The current state of the art is missing work on how to connect together offline and online methods for robot programming in a bidirectional way. The accuracy of the simulation environment determines the need of adaptations on the real robot and it is important that these modification can be stored and updated in the simulation environment as well. This allows further offline changes in the simulation, keeping the updates made using the real robot.

It has been also found a lack of research related to the programming of more

complex tasks, which include different robot trajectories and the operation of different types of hardware. Including the definition of different operations, as for example operating a gripper or moving a conveyor belt, can enable the user to design and program the entire robotic task, involving different hardware and defining the synchronization between the components involved. The work of Schillinger et al. presents FlexBE, a user-friendly high-level behavior engine which allows to define the desired task workflow connecting together simple building blocks [97]. These represent basic steps as the execution of a particular trajectory, the opening/closing of a gripper or the wait for a signal from a sensor.

In this thesis is proposed a VR framework to intuitively and quickly define a robot program. This includes the definition of the needed trajectories, as well as the definition of the entire task, which includes also operations involving other hardware components. The system architecture proposed aims to fill the gap between online and offline programming. In this way, the user can define easily and quickly the desired program in VR. If adjustments in the robot trajectories are needed, this can be done on the real robot, using the GUI provided which can be combined with the teach pendant. The use of a shared database between the two components, allows to change and update the program on both sides, depending on the user needs.

2.1.2 Simulation and VR/AR

Simulation tools are nowadays used for many applications. As an example, simulation environments are extensively used in robotics for programming. In this way the user can teach and visualize the robot trajectories in a safe way and without the need to move the real robot, which might require a lot of efforts and might cause collisions with real world objects due to errors in the programming of the desired motion [44], [39].

VR is a technology that in the last years has been deployed for different application in robotics, but mainly for robot programming and teleoperation [81], [61]. Many works focus on making the programming of robots easier, making the user able to move manually a virtual robot without any effort, dragging the end-effector position to the needed configuration. VR for teleoperation is used to simulate the environment in which the real robot is working, representing and updating the virtual model with the joints information coming from the real sensor data of the manipulator. In this way the user can send commands to the real robot and have a visual feedback on its current configuration, without the need to be physically on site.

VR is a powerful tool to display a completed simulated scenario, providing the user an immersive experience. However, the information displayed in the virtual environment could be perceived differently when applied in the real world. This is the main reason why AR is also spreading as a powerful and interesting technology for many robotics applications. AR allows the visualization of virtual objects and

information in the real world, attaching 2D or 3D components at the desired position and superimposing them in the real environment. Ong et al. propose an AR-based method to assist the user with little robot programming knowledge in the definition of the robot program [76]. Using an AR interface and a handheld pointer, the worker can move around in the work cell, defining 3D positions and trajectories for the real robot. In the approach proposed by the authors, the user can see the augmented information through an head-mounted display and two cameras mounted on it. The handheld device, which position is used to define the robot path, is tracked by the motion capture system OptiTrack.

Milovanovic et al. presented in their work an overview of VR and AR applications in the field of architectural design [69], [84]. For example, Tonn et al. propose in their work a SAR (Spatial Augmented Reality) application for 1:1 scale design of interiors [108]. Using two projectors and an input device, the design of doors and windows could be checked in the desired indoor environment.

In the work of Rohacz et al. [93], the use of AR as potential tool to support intralogistic planning in the automotive industry is investigated. The study showed that in literature there is a lack of scientific research for intralogistic planning and the authors proposed an AR application to plan carries and shelves. The augmented objects were displayed through a tablet, using the tracking of a flat marker positioned at the intended location.

Reif et al. pointed out that production and logistic lines require continuous adaptations and even rebuilding [90]. For this reason is fundamental to deploy a planning tool to generate realistic models before the actual realization. Many companies use 2D CAD (Computer Aided Design) tools for the planning of logistic systems. Anyway these tools are usually not intuitive and do not allow the user to change between different views. For this reason the authors highlight a possible use of VR and AR to develop new intuitive and efficient methods for this task.

In this thesis it is proposed a system that enables the planning and evaluation of complex robotic system without the need of programming skills. The use of a simulation environment in VR allows the define the desired scenario from scratch and without the need of the real equipment. Robot trajectories and interaction between different components can be defined through intuitive interfaces and verified then on the real line using an AR overlay, which can be used for further modifications of the program. Finally the simulated motions can be directly exported to the real robot for immediate execution.

2.1.3 Teleoperation

With the diffusion of robotics in everyday life, an increasing interest for intuitive control methods has arisen. The HRI topic has gained a lot of interest within the scientific community and enabling an intuitive operation is one of its key aspects [51]. For years teach pendants have been the most widely used methods

for interacting with robots. Using them, the user can program the robot moving manually the end-effector to the desired position at each stage of the robot task. This method is simple and requires no learning, but it has the drawback that the user has to be close and touch the robot and this is not always possible. In addition, once the various positions are recorded, it is difficult to make further amendments. Hence, small adjustments may require a lot of additional work [78].

A natural way to communicate between humans is done through gestures. There are many works related to the control of robots by means of wearable devices, such as gloves or electro-mechanical sensors [101], [122], [19]. These devices are used to track the position of the human hand and fingers in order to send commands to the robot [109], [115]. Fischer et al. present a comparison between two modes of teleoperation of an industrial robot [36]: using a data glove and a control object (peg). The results, related to an assembly task, show that the teleoperation by means of the control peg is more efficient in terms of speed and success rate. Lately, the use of VR and AR tools for intuitive robot control, has attracted a lot of attention. Gaschler et al. propose an intuitive robot system in an augmented reality workcell [40]. The human operator uses a hand-held pointing device in order to specify waypoints for the robot end-effector and for triggering actions. A projector is used to display an AR-based user interface on the worktable. Yamada et al. investigate a tele-robotics system for a construction machine [124]. The system consists of a servo-controlled construction robot, two joysticks for controlling the robot and a 3D virtual environment. The operator performs remote operations on the construction robot by manipulating the graphic robot directly in the virtual environment using the joysticks. The position and shape of the task objects in the virtual world are updated in real time through the use of a stereo camera fixed in the remote field. The drawback of these methods is the need of wearable devices or objects, with a consequent decrease of naturalness and efficiency in the interaction. Furthermore, they typically require expensive sensors in order to achieve a good precision. Ampornaramveth et al. propose a solution of this problem developing a vision-based gesture recognition method for HRI [46]. Hand poses and face expressions are detected in order to interact with a pet robot. The issue of controlling a robot with gestures has also been approached using vision based techniques to recognize both static and dynamic hand gestures [89]. In the work of Corradini et al. are presented different architectures for gesture-based interaction between humans and an autonomous mobile robot [30]. Ehrenmann et al. studied an intuitive way of commanding a robot by verbal and gesture commands [35]. The aim of this work was to provide a method to interact with a robot using natural and direct communication techniques fusing different communication channels. Additionally, there are works making use of skeleton tracking for robot control [113]. In the work of Vasanth et al. a Microsoft Kinect is used to track the user hand in order to send specific commands to a robot arm. The use of cameras and computer vision techniques allows the recognition of human gestures without the need of wearable devices. These methods often use a set of hand gestures that can be recognized as robot commands [87], [51]. Pater et al. present the use of pointing gestures recognition to automatically sort objects into

bins [80]. The programming by demonstration system proposed is coupled with speech and graphics in order to facilitate the HRI. Using these approaches, the user has to learn all the possible gestures that are needed to send various commands to the robot. They also usually allow just the execution of some pre-programmed tasks or the command of the robot motion only in some directions. A biologically inspired approach about target reaching with a robotic arm was presented in [114].

Service robots are becoming able to execute complex tasks and are being used for many different applications [86], [110]. They are usually programmed to execute a given task autonomously or to help the user providing information and guiding him. For example cleaning robots are used in domestic environments to clean the floor and their interaction with the user is limited in avoiding possible collisions. Other service robots are deployed for tasks in which the interaction with the human is the most important aspect. Tribel et al. propose a service robot for guiding passengers and help them in busy airports [110]. The work focuses on making the robot aware of the human social behavior, recognizing the people activities and relations.

For most of service robots, the interaction with the human is limited in the selection of the desired task, which is then performed autonomously by the robot. Many works focus on the interaction with robot in order to communicate the desired behavior through the use of GUIs, speech or gestures [105], [120]. Li et al. propose a system to communicate intentions with a non-verbal communication based on the user's eye gaze [59]. Their approach is used to infer the user's intention in order to trigger an assistive robot to provide proper service. For other tasks the user might need to adapt the robot's motion or teleoperate its TCP in order to reach custom positions. This problem has been addressed in the work from Muszynski et al. [71]. The authors studied how to control personal service robots through the use of a hand-held computer interface. The user is enabled to switch between different autonomy levels for the robot: skill control, body control and task control. The results of the experiments showed that the autonomy levels configuration increased the efficiency, while the situational awareness given by the camera view could be improved. The selection of different autonomy levels is also studied in the work of Baker et al. [17]. The authors suggest a system that allows a robot to range from fully autonomous to teleoperated. The method is designed for an urban search and rescue scenario.

Teleoperation has the advantage to make the user able to move the robot at a safe distance with precision. The research has investigated how to make this control as intuitive and easy as possible for unexperienced users [81]. Since humans are using gestures to communicate between them and show their intentions, this is an interesting communication channel that has been studied in many works [87], [89]. Some approaches consist of detecting the user's gestures in order to trigger predefined robot motions, as for example moving in one direction [19]. Wolf et al. propose a system in which a wearable device is used to capture electromyography and inertial signals [122]. These are used to send commands to the robot in order

to trigger autonomous capabilities. In this way the system gains in safety, not relying entirely on human's input. If the robot task involves the use of a tool, the tracking of a hand-held device can help the user to achieve a better precision in the control. Fischer et al. proposes a study, which consists in a comparison between the use of a control object and a data glove [36]. The results of the experiments conducted by the authors show that the control object led to fewer errors. The scaling of the teleoperated motion is a method that allows the user to have a better precision in the control, for example for surgical applications [91]. In this way, bigger user movements can be mapped into smaller robot motions, enabling the robot to have a more steady position and allowing small errors in the control performed by the human.

The drawback of most of teleoperation methods is that they usually provide just the control of the position of the robot end-effector, without any degree of freedom on the orientation. This is usually because the kinematics of the robot can be a problem for arbitrary positions and rotations. This can led the robot to being not able to reach the desired position from its current configuration or forcing him to reconfigure its joints. This could be dangerous and cause undesired collisions.

GUIs are helpful tools in order to make the user understand how to interact properly with the robot and understand its behavior through the use of visual feedback. They provide him support to achieve the intended behavior without the need of previous experience with the system.

In this thesis, an intuitive method to interact and control a service robot with different levels of autonomy is proposed. In contrast to other methods which allow only the control of the position, in the proposed system the user is able to intuitively teleoperate the robot TCP position and orientation using a set of control tools, which are tracked by a camera system. The interaction with these tools, enables the robot to grasp or place the desired one in order to safely go towards the control area. At this point the user is enabled to teleoperate the TCP motion with a scaled control, that enables him to achieve a better precision. The focus is also put on the flexibility and safety of the control, since in literature there is a lack of methods that allow the user to control the position and rotation of the robot TCP avoiding kinematics reconfigurations. In order to enable the inexperienced user to learn immediately how to interact with the robot without any teaching phase, it has been developed a GUI that supports him in the different phases of the interaction. The feedback about the robot behavior and the tracking of the tools enable him to feel more comfortable during the control and achieve a better precision.

2.2 Safety and Collision Avoidance

2.2.1 Safety

Safety is a crucial aspect in HRC. Since robots are allowed to work close to humans, not causing any harm to the workers has become a critical requirement. For this reason, every HRC application needs to go through a risk assessment, in order to identify and evaluate the possible risks which can cause harm to the human during operation. Safety measures and functions can be implemented in order to mitigate the risks and guarantee no harm to the worker. Research works focus mainly on defining methodologies to assess the risks in HRC applications [13]. For example, Askarpour et al. propose a methodology that starts from a generic modular model and customizes it for the target system in order to analyze hazards according to known standards and study the safety of the collaborative environment [15]. Iterative design methods based on model-driven risk assessment and decision support are used to identify potential hazards in the workspace and estimate the impact of the necessary safety measures in terms of costs, cycle time and flexibility [16]. In literature it is possible to find research on how to model and describe the different possible ways in which a task can be performed, providing methods to automatically explore the state space in order to detect hazard situations in early stages of the system design [116].

2.2.2 Collision Avoidance

As soon as robots started to share the same workspace with humans, the need of collision avoidance capabilities has arisen. In particular the machine needs to avoid dynamic obstacles such as moving workers, in order to prevent any harm or risk. Mobile robots relies on dynamic planners which adapt to the detected objects around the platform [127]. Laser scanners are usually used to detect the distance from obstacles, making the robot stop immediately if an object is closer than a safety distance threshold.

For robotic manipulators this is usually more challenging due to the many degrees of freedom and links. Laser scanners can be used as well to monitor areas in which the robot needs to slow down or stop if a human enters the restricted zone. Anyway, this does not allow human and robot to share the same workspace efficiently in order to perform their task simultaneously. Safety skins enable the robot to work close to the human through the use of contact sensors on the surface of the robot [50]. In this way, possible contacts are immediately detected and a stop signal can be sent in time in order to avoid harm to the worker and keep the impact forces below the safety thresholds [66]. Capacitives sensors can be used to avoid any contact, reducing the vulnerability of the skin sensors and without affecting the appearance of the robotic arm. Lam et al. propose the use of an invisible sensitive skin built inside the robot arm for collision avoidance of

a 6-DOF manipulator [57]. These methods ensure safety but are also inefficient, since they force the robot to stop when an obstacles is already in close proximity.

3D camera-based methods can enable the monitoring of the workspace in order to allow the prediction of possible collisions and perform a dynamic planning [99]. Recent approaches are using the computations of the volume that will be occupied by the robot in the near future to predict future possible collisions [64], [18]. The problem of these approaches is that they require a lot of computations in order to generate the swept volumes [118] of the robot trajectories and to check possible collisions with the live environment. This is not feasible using standard CPUs [123], [28].

The GPU-Voxels library proposed by Hermann et al., allows the fast computation of high-resolution voxels maps through the use of parallelized algorithms [48]. In this way, the live environment collected by 3D cameras can be monitored for possible collisions in real-time using the robot swept volume. This enables the robot to predict possible impacts and replan its motion dynamically, avoiding idle stops and improving efficiency in the task execution.

2.2.3 Dynamic Task Scheduling

The scheduling of robot tasks is of crucial importance to ensure efficiency in the task completion, minimizing costs and enhancing the overall productivity of the robot [103]. Many works have focused on how to dynamically schedule tasks for multi-robot systems [128]. This allows the allocation of tasks between multiple types of robot which usually have their own skills and functionalities. The communication between robots makes possible to perform auctions on the tasks, based on the resources available to each robot and optimizing a cost function which can aim at minimizing the tasks completion time or the resources needed [100].

For single robot systems, in particular manipulators, there are works on how to schedule the robot tasks based on the human actions, for a better cooperation in a collaborative assembly task [126].

Focusing in the case of human and robot sharing the same workspace to perform autonomous tasks, the are not many studies in literature on how to switch between different tasks based on the workspace state. In HRC, the environment in which the robots operate is really dynamic and an efficient scheduling of the targets priorities could improve the overall task completion time, in the same way humans prioritize tasks based on what the other workers are doing. For example, having multiple workpieces available to work on, if an obstacle is blocking the current target, a real-time replanning capability could foster the selection of targets far away from the worker, helping in increasing the distance between user and robot and consequently interference between their actions, when they are performing autonomous tasks in the same workspace.

2.3 Human-Robot Communication

2.3.1 Robot Intents Communication

Many works have explored how the robot can predict the human intentions and motions, but not so many works focused on the problem of the communication of the robot intentions. This is a crucial problem, since in an effective and efficient collaboration, each participant should be aware of the intentions of the others. In the same way, the human should clearly know and understand the plan of the robot and be able to perceive the robot planned motion. Anyway, the human use implicit and explicit cues, which are hard to use on a robot, in particular if it doesn't have anthropomorphic features.

Collaboration requires multiple entities to work together by effectively coordinating their actions to achieve a shared goal [55]. Humans are able to achieve good coordination by utilizing verbal and nonverbal cues in order to communicate their goals and intents [72]. They continuously use implicit cues to understand the intentions of other people when they work in a common workspace. Eye gaze, gestures, speech and other natural cues allow them to have an efficient and effortless cooperation. Robots usually don't provide this information, especially in the case of manipulators that don't have a human-like appearance. This can also lead to a feeling of anxiety for the user, due to the unpredictable robot motion. Making robots able to provide this information is challenging, because this is often subtle and it is based on modalities that robots usually don't have, like gaze or facial expressions.

A lot of research has been done in the field of intention predictions and communication in HRI. The main focus of these works is related to the better understanding of human motion and intentions to make the robot behave and react in an intelligent and efficient way [72], [64], [31], [95]. For example, [70] has conducted a study to predict human intentions based on the tracking of gaze information. On the other hand, not many works have focused on how to make the robot behavior transparent to the user.

Some studies have highlighted the usefulness of revealing the intentions of the robot. As an example, in [106] the authors state that people will be more likely to see the robot as being more approachable if it shows forethoughts before performing a functional action. [111] shows that humans are not only reacting but use prediction to plan their motion, as they try to acquire information to anticipate future motion of other objects.

Work has been done related to the information that a robot should provide to the users in assistive applications. The work presented in [24] is focused on a human-robot interaction experiment to investigate what type of verbal feedback people prefer in verbal updates by a service robot.

[25] presents the results of an experiment related to the human-robot social interaction. The purpose of the study was to measure the impact of certain features and behaviors on people's willingness to engage in a short interaction with a robot. The behaviors tested were the ability to convey expressions with a humanoid face and the ability to indicate attention by turning towards the person that the robot was addressing.

A module that supports engagement between a human and a humanoid robot by generating appropriate directed gaze, mutual facial gaze, adjacency pair and backchannel connection events has been developed in [49]. This module implements policies for adding gaze and pointing gestures to referring phrases, performing end-of-turn gazes, responding to human-initiated events and maintaining engagement.

In the research from Takayama et al., several techniques to improve robot readability are presented [106]. The focus of the study is the use of animation principles in order to illustrate forethought and reaction. The results showed that with this additional information people are more confident in their interpretations of the robot behavior. Furthermore the robot appears more appealing and approachable.

In [26] is presented a study to make a mobile platform plan understandable to the humans that walk around it. A beamer was mounted on an autonomous vehicle in order to cast geometrical shapes representing the direction in which the robot will move in the near future. Different shapes to represent the vehicle's intentions were studied through the evaluation of the human's gaze during a close interaction. A similar study that highlights the importance of the communication of the robot plan is presented in [119]. A wheelchair with a laser projector is used to give path information to the passenger and to the nearby pedestrians.

Research on the nonverbal communication signals that a non-humanoid robot can utilize during HRC was addressed in [27]. This work presented a study that explores how to use simple multimodal light and sound signals to request help during a collaborative task. Different frequencies and intensities of the signals were studied in order to express different levels of urgencies of the help request.

A promising method to represent robot information to the human is the use of AR tools. In this way, the displayed information can overlap directly with the real world objects, making the robot's plan understandable in a more intuitive way. This allows to easily visualize information and virtual markers exactly on the desired position. An early research from Birkfellner et al. explored the use of a head-mounted AR device for visualization in biomedical applications [21]. Other works exploit AR tools to support operators in a industrial workplaces [67]. AR is deployed to visualize the assembly process, production updates as well as video and text instructions.

Walker et al., investigated this problem for robot which do not have anthropomorphic features, proposing different approaches to represent this information for aerial robots using AR [117]. The results, showed that the proposed designs

implemented by the authors improved the task efficiency and the user perception of the robot as a teammate.

Part of the motivation of this thesis is based on the lack of research on the representation of planned motion for a highly replan-capable robot without anthropomorphic features. A more intuitive way to show this information can help the human to have a more comfortable interaction during close collaboration. Since the robot is able to change its motion based on the user's actions, the human has to be able to understand if the robot changes its target, as well as the workspace that it will occupy in the near future. This is as important as the understanding of the human motion, because in a collaborative task both the participants should know what the other partner is doing and wants to do.

In this work are proposed methods that aim at contributing to this goal by providing the human with intuitive information relative to the robot motion. This has been done using acoustic and visual channels and with AR-based interfaces. The proposed methods contribute to this goal by representing the robot motion directly in the workspace. The use of a projector and head-mounted displays allows to visualize the information at the relevant position in the real world. In this way the user is able to perceive feedback from the robot and predict its motion without losing the focus on his task. This has been done in such a way to allow an intuitive understanding of this information.

2.3.2 Dialogue

Accordingly to the IFR[1], modern collaborative environments are literally exploding and bringing the automation out of the cages where robots were confined due to safety issues. Several manufacturers have released robots able to work safely with humans. These robots can be used in production lines for collaborative assembly, welding, packaging and material handling, just to mention some possibilities [65]. A proper designed HRC with different sensors can improve the robot autonomy in a collaborative task [54].

It is clear that the success for future collaborative applications relies on a natural and intuitive HRI, with robots able to react quickly to human inputs [62]. However, for what concerns acoustic communication, the information conveyed by an acoustic input is often degraded due to noise and reverberation [63]. In this section a review of the state of art in the field of HRC and speech recognition is provided.

Natural Language Understanding

Natural Language Understanding (NLU) can be implemented with different purposes, such as avoiding intermediate keying and handwritten steps and making

[1] International Federation of Robotics

the human user able to communicate with the machine. At the same time, hands or eyes can carry out other functions; this is the case for example of medical transcription, where doctors dictate directly to the computer removing the need of a second person and decreasing visit time [34]. Other examples are forensic investigations which require identification and diarization techniques [33].

Other types of NLU applications are related to speech command functionalities and rely on a smaller vocabulary. In these applications, a speech input is transformed into one or more actions to control some functionalities for specific systems. Examples are jet fighters weapon systems, training for air traffic controllers [104], cars multimedia and navigation, control of medical devices for surgery and anaesthesiology [82].

Finally, nowadays are available systems able to perform full natural language understanding and generation with conversational chatbot characteristic. They can converse with humans by understanding and processing several aspects of speech communication (contents and emotion, like Alexa in the consumer market).

Collaborative Robotics and NLU

In some specific scenarios, robots with autonomous speech dialogue capabilities seem to gain human attention, especially for their need to appear socially natural.

One example, the humanoid robot Pepper, initially developed for business to business applications, has found several applications not only in retail but also in consumer market and academics [79]. It includes several key features, including support for social autonomy and the ability of providing a natural interaction using speech and gestures.

Another example of robot capable of interacting with humans is the humanoid robot developed by Toshiba, used at the travel fair in Berlin (2016). It was installed at an information desk in order to respond to attendees' verbal questions about events at the fair.

In hospitality, a holistic approach to HRI is the Enn-na Hotel-resort (which means "Strange/Changing Hotel" in Japanese) in Nagasaki Prefecture, Japan [77]. Although one limitation shown by this initiative was the simple and non-emotional interaction, this could be instead considered satisfactory in other fields where such expectations are not critical. In healthcare, the number of applications and research studies, are increasing. From home assistance for elder care, with robots in charge of assisting people and understanding emergency situations [129], to support in existing surgeon systems in order to control the tools with voice and acoustic signals [130], [74]. Somehow the industrial sector has remained static in the last years, perhaps due to the complex acoustic environment which affects the reliability of the the acoustic acquisition system [63]. At research level some authors have shown that speech is a possible communication channel, providing

promising results with non-trained operators [32]. In literature there are several examples of industrial robots that can collaborate with speech [83], [107]. Worth to mention is an interesting multimodal approach, which combines haptic, vocal and visual feedback with a Universal Robot [45].

In the conducted research, it has been found a lack of clear guidelines to implement proper acoustic interaction between robots and humans. Also, accordingly to the works found in literature [102], it can be summarized that the HRI community seems more focused on creating proof of concepts than detailed and validated scientific conclusions and in general speech recognition has received minor attention. In this thesis is proposed a robust platform to support state of art ASR in industrial robotics, enabling natural speech communication in multimodal HRI and providing a methodology to evaluate systematically the designed interaction.

2.3.3 Multi-Modal Robot Interfaces

Regarding effective and intuitive interfaces to convey the user's intentions, gestures and speech are the natural methods used by humans [75], [38], [125]. The research has focused on providing flexible and intuitive interfaces through the use of speech, hand gestures and head gestures [94], [92], [52].

2.4 State of the Art Summary

In this chapter, the related work in the field of HRI has been introduced. In particular, the state of the art of the research in the topics of robot programming and simulation, safety and collision avoidance, communication of robot intentions and online robot plan and motion modifications has been described. In the following, is reported a summary of the related work, with focus on the lack of research found in the investigated topics, which is the main motivation of this thesis.

- **Robot programming, simulation and intuitive robot control:** many works are investigating how to ease the robot programming for non-experts using intuitive interfaces and simulation tools. However, the focus is usually in programming single robots without an approach to program and simulate complex systems made of multiple robots and hardware components such as grippers and conveyors belts. Furthermore, the gap between simulation and real world does not allow an easy deployment of the programs defined in simulation on the real robots. A method to easily switch between simulated and real robot programming, as well as adding multiple level of testing in the real setup with the use of VR and AR tools, could enhance the time needed to define and test a program, as well as making more realistic investments before acquire expensive hardware.

- **Safety and collision avoidance:** many works have focused on how to enable a robot to safely work together with human workers, through the definition of standards and tests to validate the safety thresholds. Anyway, clear guidelines on how to guarantee safety using external control authorities, such as ROS, are missing. Furthermore, collision avoidance systems implemented in such controllers, are hard to evaluate and there is a lack of benchmarks and test routines to evaluate the safety aspects introduced by these components. In addition, collision avoidance system are usually not focusing on the application efficiency, just focusing on making the robot stop in case of possible collisions. The use of fast and real-time capable collision prediction modules, can be used to enable a more dynamic task scheduling in order to avoid idle times and perform a more efficient target selection.
- **Communication of robot intention:** the research has been focused on how to make the robot able to predict and understand what the human is planning to do, but there is a lack of research on how to enable the worker to understand the robot intention and planned path. This is anyway of crucial important to enable a more efficient and ergonomic collaboration. In literature there are works that investigates the representation of the robot planned path using beamers and AR, in particular for mobile robots. A method to intuitively represent robot feedback information for robotic manipulators without anthropomorphic features could enhance the trust in the robot and allow the worker to plan more efficiently his tasks and how to move in the workspace.
- **Online robot plan and motion modification:** many methods are focusing on how to program the robot, but the studies on how to do this at runtime, while the robot is executing its task, are limited. There is a lack on how to quickly and intuitively adapt the robot motion online, without the need to stop the application. In particular, multi-modal interfaces and the use of VR/AR tools can enable the use of virtual objects to adapt the robot trajectories and defined workspace constraints at runtime.

In the next chapter, the proposed methods for robot programming, simulation as well as intuitive and flexible robot control are presented and evaluated.

3 Robot Programming and Control

3.1 Robot Programming and Simulation

This section introduce the motivation and proposed approach for intuitive robot programming of complex systems. A VR/AR-based system is proposed to define and validate complex scenarios, without the need of programming experts. The desired setup and robot trajectories can be safely defined and simulated in a virtual environment. The different types of visualization and the easy export for the execution on real hardware enable to bridge the gap between simulation and real world, making easier to validate the defined programs before deployment. Parts of the results presented in this section are part of the work done in the German research project Sim2log VR and have been published in [7] and [6].

3.1.1 Motivation

Robotic systems are complex and commonly require experts to program the motions and interactions between all the different components. Operators with programming skills are usually needed to make the robot perform a new task or even to apply small changes in its current behavior. For this reason many tools have been developed to ease the programming of robotic systems. Online programming methods rely on the use of the robot in order to move it to the desired configurations. On the other hand, simulation-based methods enable the offline teaching of the needed program without involving the actual hardware setup. VR and AR allow the user to program a robot safely and effortlessly, using a virtual representation of it and without the need to move the real manipulator. However, online programming methods are needed for on-site adjustments, but a common interface between these two methods is usually not available. In this section a VR/AR-based framework for programming robotic tasks is proposed. The use of predefined building blocks allows the easy definition of complex scenarios composed of multiple hardware components. In this way the entire workflow of a robotic system can be simulated and also evaluated through the computation of the relevant performance indicators. The system architecture deployed allows the integration of the defined programs into existing tools for online teaching and execution on the real hardware. The proposed virtual environment enables the intuitive definition of the entire task workflow, without the need to involve the real setup. The bilateral communication between this component and the robotic

hardware allows the user to introduce changes in the virtual environment, as well into the real system. In this way, they can both be updated with the latest changes and used in a interchangeable way, exploiting the advantages of both methods in a flexible manner.

3.1.2 Method

The system proposed to simulated and program complex robotic systems is made of three main steps:

- A VR module, in which the user can define the simulated environment using predefined building blocks representing hardware components, such as robots, gripper, conveyor belts and tables. Once the simulation scenario is created, the module can be used to define the robot programs without the need of programming, simply by moving the robot TCP to the desired positions using the VR controllers and selecting the needed operations with the use of a UI.
- A AR module, that can be used to check how the defined robot trajectories and programs will look in the real world. A smartphone can be used to display a virtual overlay on the real setup, giving a better understanding of the robot reachability and correctness of the defined motions. The robot trajectories can also be further adapted in AR, giving the possibility to fine-tune the waypoints of the trajectories in the real scenario.
- A ROS module that is responsible to execute the programs defined in the simulation on the real hardware. The trajectories and operations defined in the VR/AR modules can be imported in ROS and executed on the real robots, without the need of further programming. The changes needed while executing the programs on the real robot, can be applied in this module and stored to allow further modifications of the robot program in the VR/AR components.

The three proposed steps to enable an efficient and effective simulation of a logistic robotic system are represented in Figure 3.1.

The three components share the simulation data through a database and can also communicate between each other through a direct TCP connection. In Figure 3.2 is reported the overall architecture of the framework and the communication between the components.

VR Module

The VR module main purpose is to enable the definition of complex simulation scenarios without the need of programming. The use of predefined building blocks representing different types of hardware, allows to easily spawn the needed

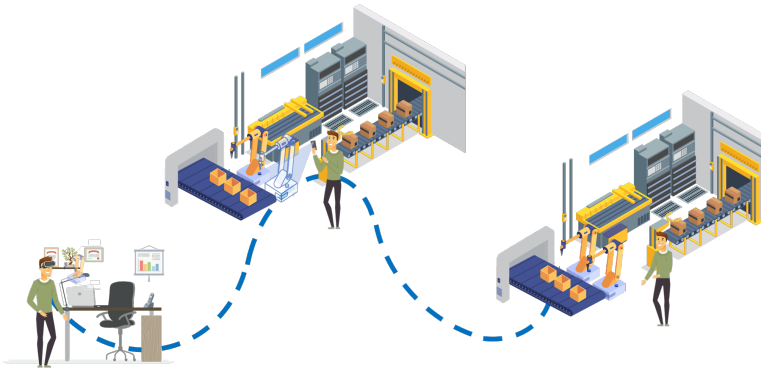
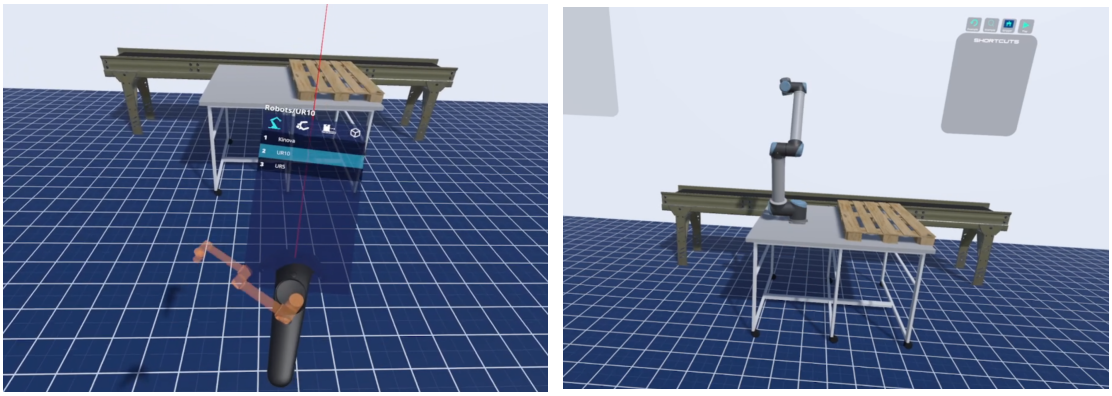


Figure 3.1: The three main steps of the proposed framework to enable the simulation of complex robotic systems: from pure simulation in VR, testing on the real setup through AR and execution on real hardware using ROS.

objects in the environment trough the use of VR controllers. A dedicated button on these latter, can be used enable/disable the catalog of available models, which are listed based on type. The system includes different types of robots such as Universal Robots and Kinova, as well as grippers and conveyor belts. Once the desired model is selected, it can be placed in the simulation environment moving it to the needed position, by simply clicking on the controller triggers to grab and release the object. Figure 3.3 represents the selection of a Universal Robots UR10 robot which is then placed on a table. To quickly move in the scene, which is crucial for large environments, a virtual laser pointer has been added to the simulated VR controllers. This can be used to point to the desired position and move the digital twin to it by clicking on the controller button.



(a) Selection of robot.

(b) Robot added to the scene.

Figure 3.3: The VR environment provides a library of building blocks to easily create complex simulation scenarios without the need of programming. In (a) a robot is selected in order to be placed on a table next to a conveyor belt. In (b) is represented the scenario after the placement of the robot in the scene.

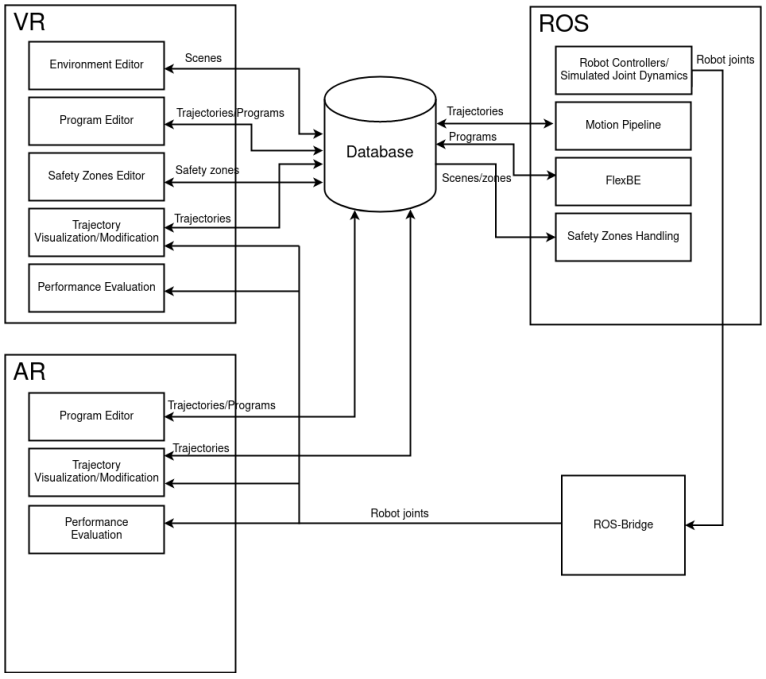


Figure 3.2: VR/AR/ROS architecture and communication between components.

The building blocks type is taken into consideration while creating and moving objects, in order to enable a coherent coupling between the different models. For example, a gripper can be coupled only to a robot TCP, making the process of positioning the item more intuitive and less prone to errors. In the same way, a robot base can be easily placed on a table surface, without the need to precisely set the proper orientation.

In order to enable the intuitive programming of the robot program in VR, a UI has been implemented in order to select the desired robot, the needed operations and test the defined motions. The visualization of the UI and an example list of instructions defined by the user are represented in Figure 3.4. A robot can be selected using the virtual laser pointer. The available operations listed in the UI are: definition of joint-based or Cartesian trajectories, enable/disable gripper, set conveyor belt speed and wait. The operation to define a trajectory allows to move the robot to the desired positions and save the needed waypoints. The robot can be moved by simply getting close to its TCP. When the VR controller get in proximity of the robot end-effector, which is specified by a distance threshold, a green sphere around the TCP appears (as represented in Figure 3.5a) which can be used to grasp and move the robot tool. By clicking on the controller trigger, the robot TCP can be moved and released to the desired position. An IK solver implemented in Unity is used to compute the robot joint configuration needed to match the 6 pose of the VR controller. In this way, the needed joints positions can be computed and used to display the robot model configuration to reach the target TCP pose. The UI shows the information about the robot configuration, such as joint values and TCP Cartesian pose, and includes buttons to add/remove a waypoint as well as sliders to specify the robot speed to execute the motion (as shown in Figure 3.5b).

For every waypoint added, a small blue sphere is added for visualization, and a blue line is displayed to show the defined end-effector path. This provides an immediate feedback to the user while he is programming the motion, in order to avoid errors in the path definition. The UI provides also a test function, in order to simulate the execution of the programmed trajectory.

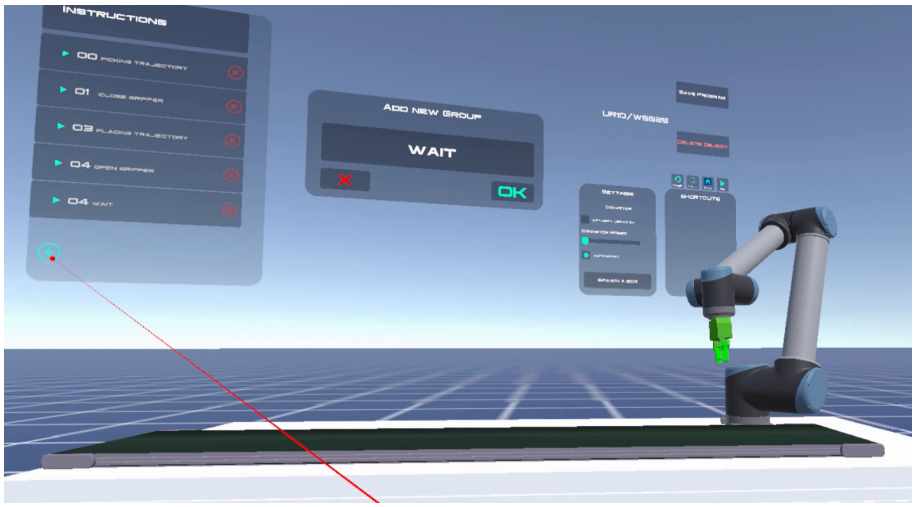
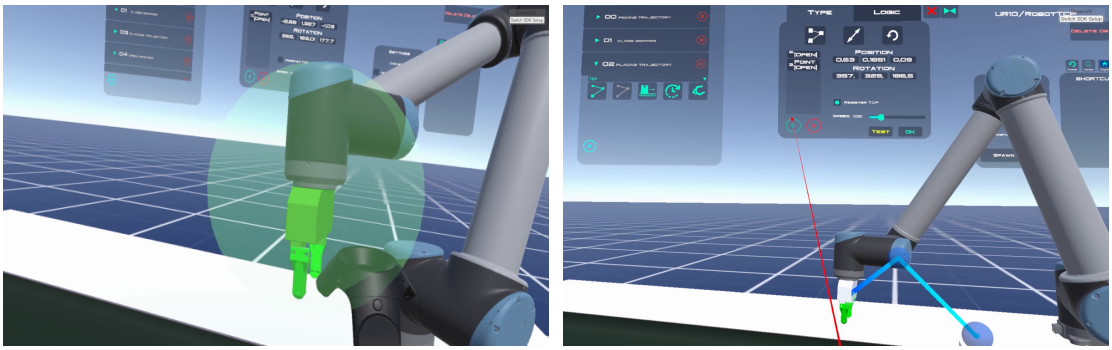


Figure 3.4: The VR environment used to intuitively program offline the robot trajectories and the task workflow. The user can specify the needed robotic operations using a menu and buttons displayed in the virtual environment.



(a) Visualization of area to grab and move the robot TCP.

(b) UI to define the program operations and trajectory waypoints.

Figure 3.5: During the definition of a robot trajectory, the robot TCP is higlighted with a green sphere (a). Moving the VR controller in this area, the user is able to intuitively drag the virtual robot, positioning its end-effector in the needed configuration. The menu available in the VR environment enables the user to define quickly and intuitively the desired robot program. Just selecting the desired operation, the entire task workflow can be programmed offline in simulation (b).

Furthermore, a specific module responsible to simulate the incoming goods in the

line has been developed. Using this component, virtual objects which simulate the moving parts in the line, can be spawned at specific position in the environment. Using the VR UI is possible to specify how many virtual objects per minutes needs to be created in order to simulate different throughput scenarios. The virtual objects are affected by the physics of the simulation, which allows them to move around transported by conveyor belts as well as be picked and placed by robots. For each simulated good, its position and information are tracked and stored to automatically compute performance metrics of the designed line. In this way, it is possible to test if all the pieces can be correctly processed and how many of them are able to reach the end of the line without being lost at some point in the process. The time needed to process them is also stored to provide information about the current cycle time and enabling the user to change the process parameters to achieve the target metrics.

AR Module

The AR module allows to visualize the virtual models and robot motions overlapping the simulated objects in the real world. The AR application can run on a pc connected to a camera or can be used with a smartphone. As shown in Figure 3.2, the component communicates with the shared database, in order to retrieve the trajectories defined in VR and enable the modifications of them, which can then be exported in order to be used in the other two modules. Figure 3.6 shows the AR visualization, in which a Universal Robots UR10 and the related trajectory defined in simulation are represented in a real environment. To apply modifications to the robot motion, the UI provides a set of buttons that allow to move the robot TCP in both position and orientation.

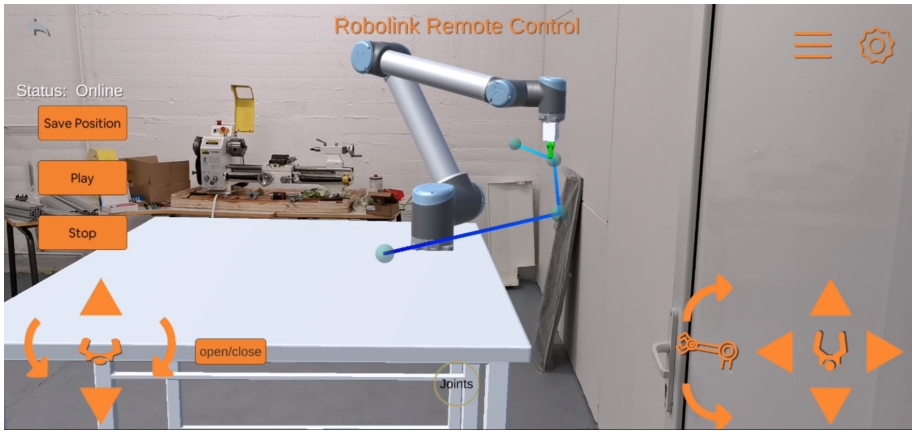


Figure 3.6: The virtual representation of the selected robot can be displayed in the real setup through the use of AR. The defined trajectories are shown and can be executed to test the robot behavior in the real environment. A simple UI can be used to apply changes to the trajectory waypoints and test the modified motions.

In the same way, it is possible to test the other robotic components with the AR overlay, providing an intuitive and effective way to test the programmed robot trajectories in the real setup and without the need to use the real hardware for that. Doing this, the robot reach can be better inspected and crucial hardware decisions can be taken before buying and deploying expensive hardware.

ROS Module

The detailed system architecture related to the ROS module responsible for the execution of the programs on the real hardware, is reported in 3.7.

The execution of the trajectories defined in VR on the real robot is done using the FZI Motion Pipeline, a ROS software stack that can be used for an easy and flexible execution of trajectories on robots with a standard ROS interface. Due to its modularity, the execution and online modification of the robot trajectories can be performed abstracting from the particular hardware interface provided by the manipulator. This component is responsible to smooth the trajectory and send it as a goal to the robot controller. The execution of Cartesian motions is also supported, providing the Cartesian waypoints to a Cartesian Controller implemented using ROS Control [96]. This controller is based on forward dynamics trade-offs to avoid singularities and have a robust and safe computation of the inverse kinematics of the manipulator for the desired configuration. The FZI Motion Pipeline provides the user with a GUI, which has been developed to ease the execution and adaptation of trajectories for inexperienced users. The trajectories available are listed through the interface and the user can use simple buttons to load, execute or modify them. If a waypoint needs to be adapted, the user can enter the desired configuration values or use the robot teach pendant to set the changed position and store it in the needed trajectory.

For the definition of a high-level task, the open source high-level behavior engine FlexBE [98] has been integrated, which is implemented in ROS as well. This is responsible to regulate the execution of the task, which is modeled as a state machine, triggering the execution of the needed trajectories and operations. FlexBE also provides a GUI which enables the user to easily define the workflow of the robotic task connecting together building blocks and without the need of programming. Once defined, the resulting behavior can be executed on the real robot using the buttons provided by the interface, which then shows the current state of the execution. The only programming involved is the definition of the basic steps of the program, which are for example the execution of a certain trajectory or the opening of a gripper. However, these need to be defined just once in order to interface with the FZI Motion Pipeline, which is then responsible to abstract to the specific hardware.

The communication between the VR/AR and the ROS components is based on the shared database. Once the user saves the desired program in the virtual environment, it can be imported and accessed by the FZI Motion Pipeline and FlexBE

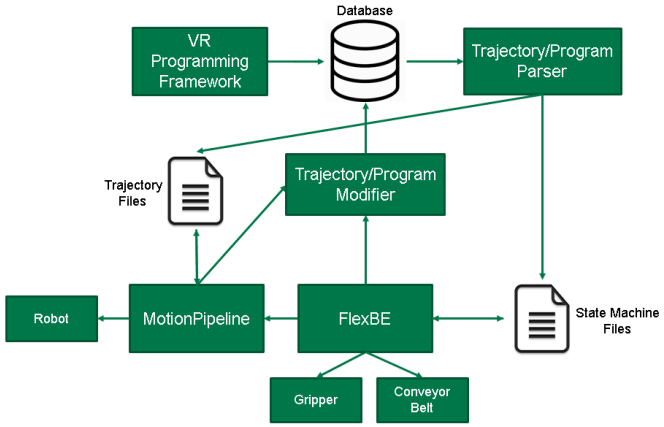


Figure 3.7: Details on the system architecture used for the ROS module, which is communicating with the VR/AR components through a shared database. From this latter the program information could be stored in files with the proper format, which are used by the Motion Pipeline and FlexBE.

modules. The same happens if changes are performed online: the modifications are uploaded in the database and the updates can be imported in the VR/AR environments to enable further offline programming phases. In this way, the user can deploy the preferred programming method depending on the requirements, being able to switch the method at any time if needed. For example the general program can be defined in the VR/AR environment and once exported to the real robot it can be adapted to reach the desired positions with more precision. On the other way round, the updated program is exported in the shared database as well, in order to allows further offline modification on the VR/AR side, keeping the program up to date on every component. The storing of the program information in the shared database has the advantage to allow the use of the offline VR programming or the online programming independently. For example, if the robot is needed in operation to execute some other task, the user can create a new program offline in the VR environment. On the other hand, if the user wants to modify a trajectory using the real robot, he can do that without the need to have the VR application up and running. For both methods, the shared database allows the two components to be always up to date and ready to be used interchangeably.

The program information stored in the database is parsed by the Trajectory/Program Parser module and then used to create the behavior files which are used by FlexBE for the execution on the real hardware components using ROS. The database has been structured to include the relevant fields for the definition of a robotic task. Every operation has an unique id used to identify it and a set of fields related to its execution. For example, the operation for the execution of a robot trajectory includes the joints configuration and the Cartesian pose of the end-effector. The needed FlexBE states to operate robot, gripper and conveyor belt are implemented to be used without any further programming, providing input parameters that can be set in the building blocks available in the GUI or extracted

from the database in order to define the needed configuration and speed.

3.1.3 Experiments

Multiple experiments have been designed to test the proposed framework.These include:

- A pick and place scenario implemented in VR, in which the user had to teach the robot to correctly pick 3 objects from different positions and place them in a box. This has been done to test the usability of the demonstrator and the time needed to program the robot for such task.
- The programming of pick and place task in VR and the execution of it on a real robot and gripper, without the need of further programming. This has been done to validate the system architecture and the possibility to export directly the programs from the simulation into the real hardware.
- The simulation of an entire line in VR, with different robots and hardware components working in the logistic process. This has been done to test the feasibility of the design and test complex scenarios.
- The simulation of a scenario in which a real robots need to interact with the virtual parts defined in the simulation as well as with an additional virtual manipulator. This has been done to validate the AR module and the interaction between simulation and real hardware, made possible by the system architecture proposed.

Pick and Place Programming Time Evaluation

The first experiment aims at testing the usability of the system and collect time measurements on how long it takes to teach the robot to perform a specific task using the proposed VR-based method. For this purpose, an application has been implemented in which the user had to program the robot to perform a pick and place task. The scene has been already set up with a virtual Universal Robots UR10 equipped with a gripper. The robot is positioned on a table, on which are located 3 cubical objects and the target box in which they need to be placed. A virtual obstacle, representing an area that the robot should avoid, has been also added to the environment, positioned between the objects and the target box. This has been done to add more complexity to the programming task, requiring the definition of trajectories with multiple waypoints to avoid collisions with the virtual obstacle. In Figure 3.8 is represented the pick and place application scenario.

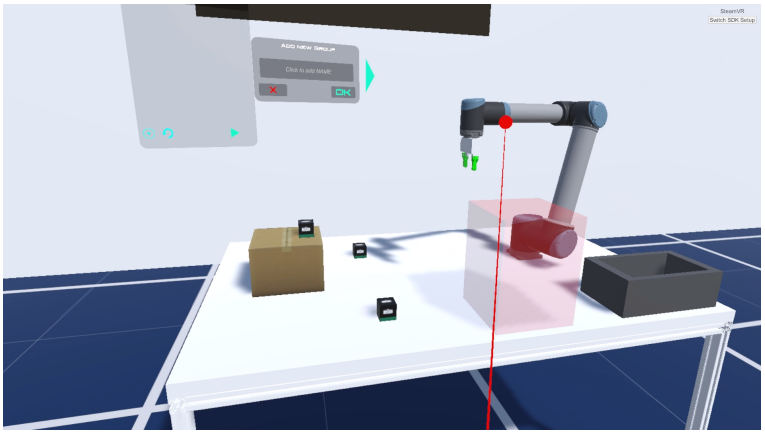


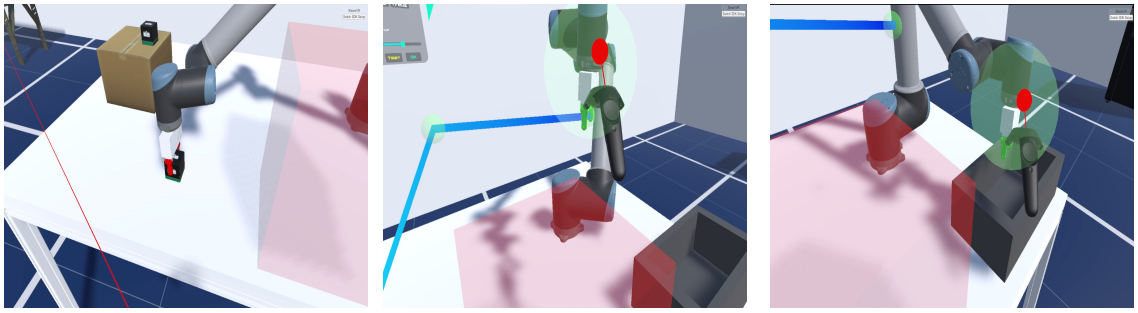
Figure 3.8: The scenario used to measure the time needed to program a robot to perform a pick and place task. The goal consists in programming the robot to pick the three objects on the table and place them in the target box. The obstacle needs to be avoided by the robot while executing the defined trajectories.

Before starting the experiment, a set of instructional slides have been shown as represented in Figure 3.9. These include the description of the task that the robot needs to be programmed for, the available controls on the VR controllers and a list of the available operations provided by the virtual UI.

As soon as the user agrees to star the game, a timer is set to keep track of the time needed to program the robot and make it complete the task. Furthermore, the time needed by the robot to perform the pick and place operation for each object has also been measured. Figure 3.10 shows an example of the trajectory programming phase. The user needs to place the robot TCP correctly to pick the object from the table and define the trajectory waypoints to avoid the restricted area and make the robot able to reach the target location above the box.

(a) Goal of the application. (b) VR controllers inputs. (c) Available operations.

Figure 3.9: Before starting the test, the user gets information about the application goal and how to use the available tools. First of all, the pick and place position are explained in (a). The VR controllers input buttons functionalities are represented in (b). Finally, the available operations available in the UI are listed in (c).



(a) Robot picking position. (b) Definition of trajectory. (c) Robot placing position.

Figure 3.10: The user has to teach the robot trajectories to make the robot pick and places the three objects. In (a) is represented the defined picking position. The definition of a trajectory to reach the target box is shown in (b). In (c) is represented the final position of the robot to place the object in the target box.

Once the user succeeds to complete the challenge, by making the robot place the three objects correctly in the box, the measured metrics and information about the defined programs are stored in a database and shown to the user in VR. Figure 3.11 depicts the visualization of the results after the completion of the task. The collected information shows that it was possible to teach and make the robot perform in only 146.52 seconds, which is a really short time to program a robot and a gripper to perform such pick and place task.

Figure 3.11: Once the three objects are successfully placed in the box, the metrics of the test are shown and stored. These include the overall time needed to perform the task, as well as the time needed by the robot to perform each pick and place operation.

Pick and Place Programming and Execution on Real Robot

In this section is presented the test to validate the programming of a robotic system consisting of a robot equipped with a two fingers gripper. The setup

deployed included a conveyor belt, in order to test the use of additional hardware components available in the framework and not only robotic manipulators. Figure 3.12 represents the hardware setup used to test the programming system.

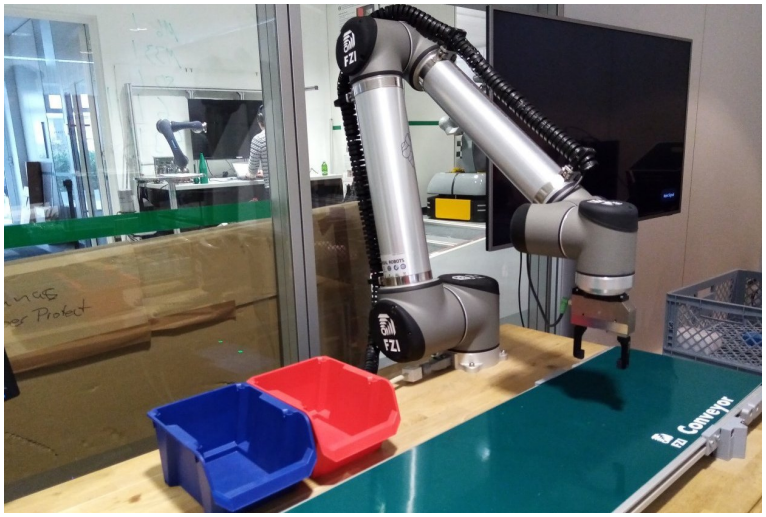
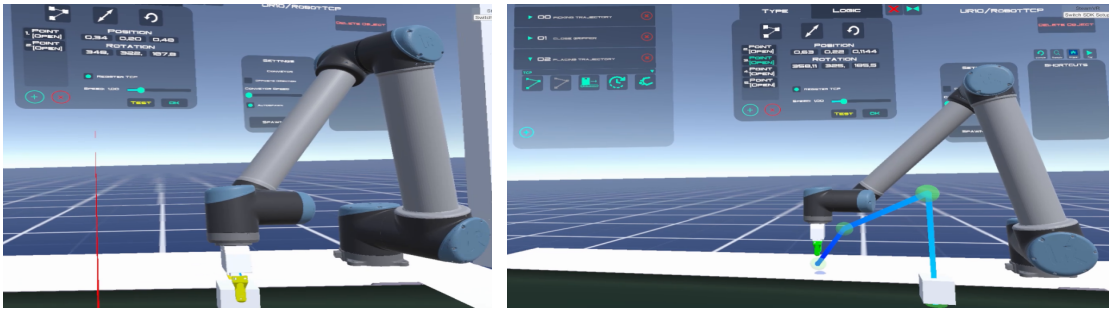


Figure 3.12: The overall hardware setup used for the evaluation experiment. This includes a Universal Robots UR10, a Weiss WSG-25 gripper and a conveyor belt. The programming task consists of the definition in VR of a robot trajectory and the operations to enable the gripper and conveyor belt. Then the program is imported on the real robot for execution and further online adaptations.

Firstly, the programming of a simple trajectory is shown in Figure 3.13. This has been done in the VR environment in order to define offline the needed waypoints in a quick way and without involving the real robot. Because of the small object to grasp, the final robot configuration to enable the picking of it should be very precise and it needs to be performed on the real equipment using the real setup. Anyway, once exported in the database, the program is automatically converted and imported on the FZI Motion Pipeline, ready to be executed on the real robot. As shown in Figure 3.14 the program defined is ready to be executed and the needed waypoints can be adjusted on the GUI using the real robot.



(a) Picking. (b) Placing.

Figure 3.13: In (a) is depicted the last waypoint of the trajectory defined in VR to pick the object on the conveyor belt. In (b) are represented the waypoints defined by the user to define the placing the object on the table.

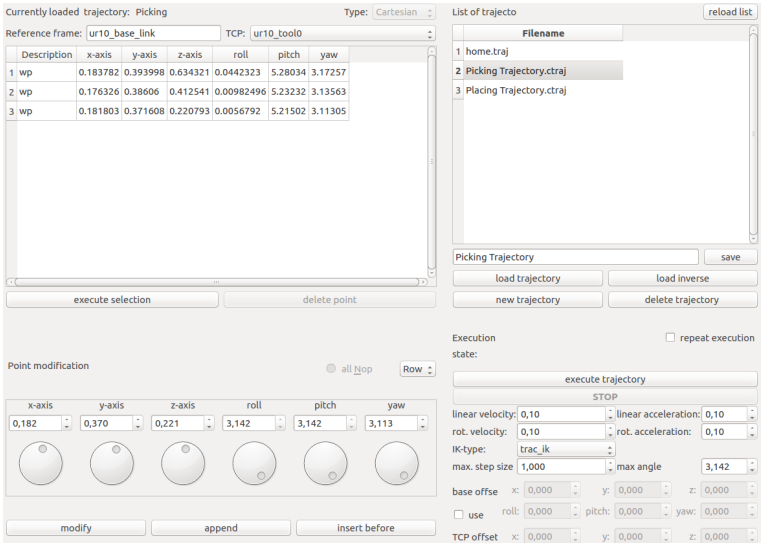


Figure 3.14: The GUI interface provided by the Motion Pipeline shows the robot trajectory defined offline within the VR environment. Using this interface the program can be executed on the real robot and the trajectory waypoints can be adapted online on the robot using the teach pendant if a more precise positioning of the manipulator TCP is needed.

In this example, the only point which needs to be defined more precisely is the last one of the trajectory. Adjusting manually the robot position with the teach pendant, the needed configuration can be defined with more precision and then stored using the graphical interface. In the same way as it was done in the VR programming, the modified trajectory is exported in the database and can be used for further offline modifications or as a basis for a more complex trajectory or program. As further test, the program has been extended to include the entire pick and place task, using the already stored trajectory for the picking operation. This has been done to show that the user can re-use the updated trajectory defined

on the real setup, to further develop the task workflow and other trajectories offline using the VR environment. In the simulation environment, the previously designed trajectory can be deployed to further extend the task definition with the missing operations. This includes the closing of the gripper for the grasping of the part, the trajectory definition to bring the object to the placing position and the opening of the gripper to release it. The addition of a wait operation of two seconds has been included before the picking action in order to have a precise grip. The speed of the conveyor belt is also modified to make it stop before the grasping and to reactivate it after that. All these different operations can be specified easily and in a quick way using the menu provided in the virtual environment. Once the task is saved through the corresponding button, the program is stored in the database and converted to the necessary trajectory files for the Motion Pipeline and to the file containing the behavior definition of the entire program workflow which is used by FlexBE. Figure 3.15 depicts the resulting task definition after being imported in the FlexBE GUI. Through this interface, the entire program can be executed on the real hardware pressing a button. In addition, the GUI shows to the user the various phases of the execution, which can be used to monitor or change the current behavior. The names of the different execution steps defined in VR are shown, making it easy to identify the different phases of the program while in execution.

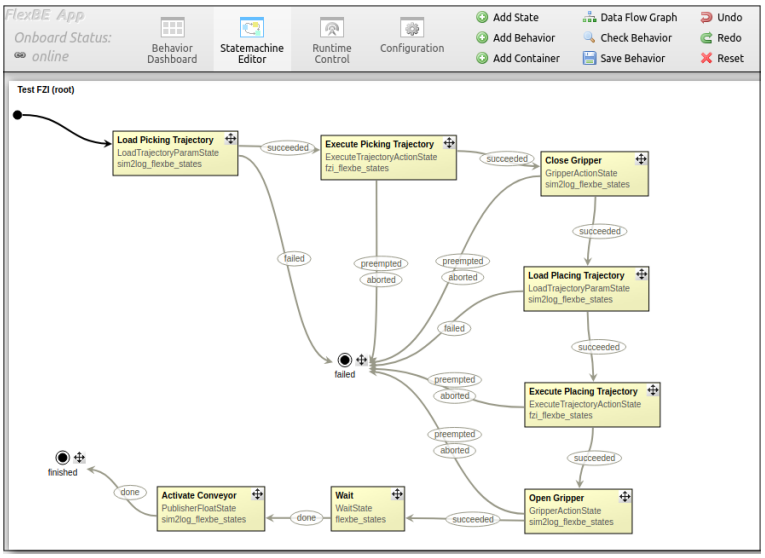


Figure 3.15: FlexBE representation of the program defined offline in the VR environment. Once imported in FlexBE the program can be executed directly on the real hardware using the FlexBE app GUI.

VR Simulation and Evaluation of a Logistic Process

A realistic logistic scenario to be defined and simulated using the VR module has been designed in order to prove the flexibility and usability of the framework. The

considered logistic process has been selected to include different components that can be used to define a large variety of different applications. This includes:

- A component to simulate the incoming goods in the line.
- A conveyor belt to transport the goods towards a robotic station.
- A robot picking the goods from the conveyor and placing them on a pallet.
- Manual transportation of the goods from the pallet towards another robotic station.
- A machine tending component which requires the robot to load the pieces into a machine in order to be processed.
- The same robot from the previous point to unload the machine and place the processed goods on a conveyor belt.
- A conveyor belt which brings the good towards a target destination.

In Figure 3.16 is represented the 2D sketch of the desired line to simulate. The different components are reported, as well as the position where the goods are fed to the line and the target destination.

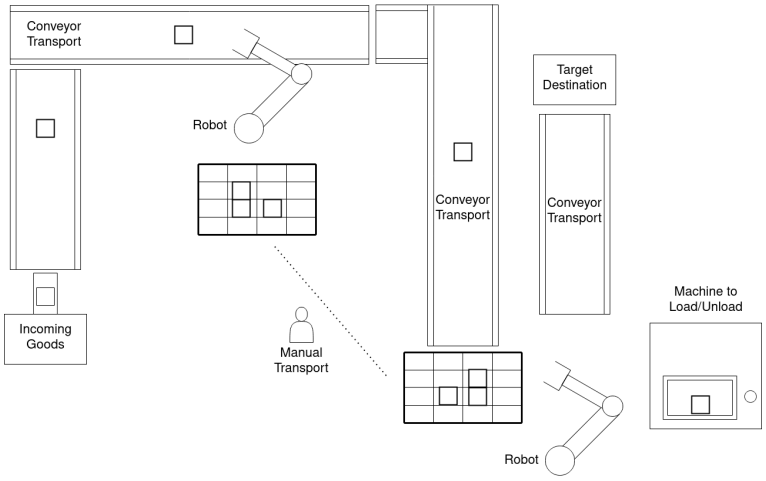
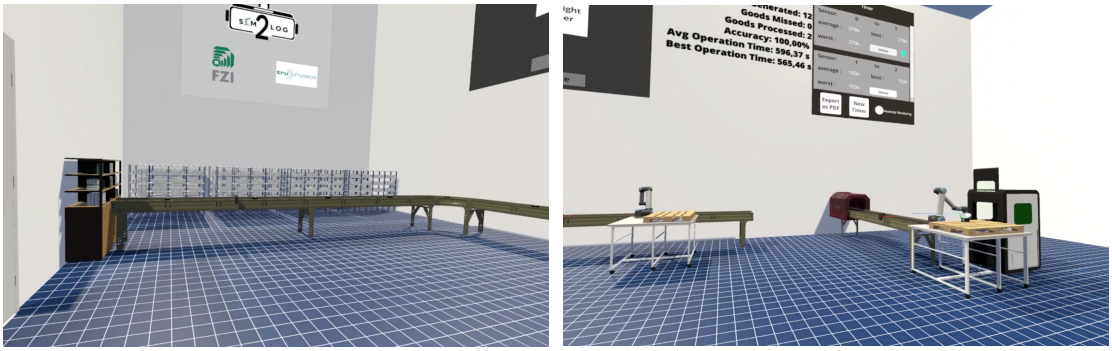


Figure 3.16: Sketch of the target logistic process that needs to be designed and simulated using the proposed VR framework. The different components involved are depicted.¨

The building blocks available in the VR environment have been used to setup the simulation scene. An overview of the resulting logistic line is represented in Figure 3.17.

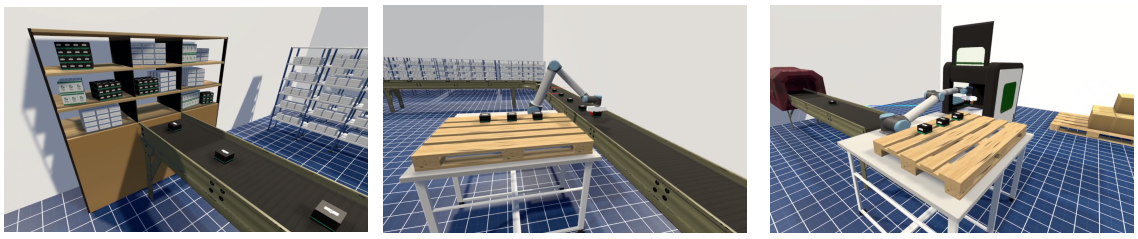


(a) Overview of incoming goods and conveyor belt transportation.

(b) Picking robotic stations and machine tending.

Figure 3.17: Overview of the resulting logistic simulation designed using the VR module and the implemented building blocks.

Figure 3.18 shows some of the components used in the scenario, in particular the simulation of the incoming goods fed to the line which are spawned on a conveyor belt, the robot picking the pieces to place them on a pallet and the manipulator unloading a pallet to load a machine which is simulating a machine tending application.



(a) Simulation of incoming goods.

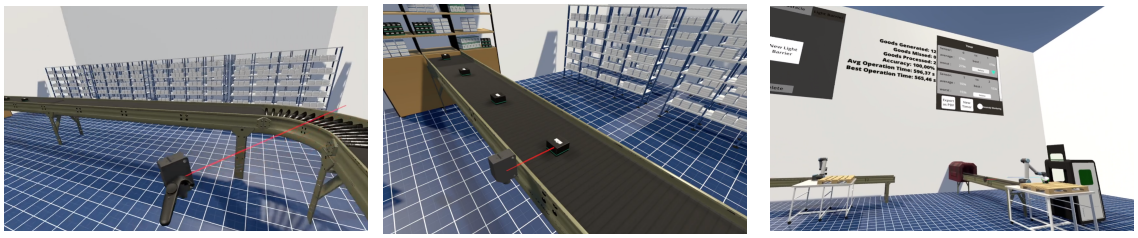
(b) First picking robotic station.

(c) Machine tending robotic station.

Figure 3.18: Simulation of the incoming goods, robotic stations and machine tending in the designed logistic scenario used for evaluation of the VR environment.

The framework provides the computation and visualization of the relevant KPIs of the line, including overall processing time of the goods and accuracy, measured counting the good lost in the line due to a wrong parameterization. The available parameters that can be modified in the VR UI include the number of goods fed per minute as well as the robot and conveyor speed. Acting on these parameters, it has been possible to define the correct values to enable a 100% accuracy of the processed pieces without losses in the process. To enable more flexibility in the computation of the performance metrics, virtual sensors components have been added to allow custom measurements, enabling in this way the user to specify in which part of the line the performance needs to be computed.

In Figure 3.19 is depicted how the sensor can be placed at any point in the line and the relative measurements computed and shown in the VR UI. The average, best and worst time are collected and displayed.



(a) Light barrier placing. (b) Sensor measurement. (c) Metrics visualization.

Figure 3.19: Placement of light barriers sensors to measure the performance in the line. A sensor can be spawned and moved around using the VR controllers (a) in order to be placed at the desired point in the line (b). For each couple of sensor defined, the relative measurements are computed and shown to the user (c).

AR Simulation of Components Interacting with the Real Hardware

The planning and evaluation of robotics solutions in logistic environments is a time consuming process. It usually requires to plan on paper or with 2D CAD software the design and flow of the parts in a line, with the need to buy and setup expensive equipment in order to test and validate the planned solution. Through the use of AR, virtual parts and the required equipment can be simulated in order to evaluate the system beforehand and the needed performance information can be computed automatically. The simulated hardware can be superimposed at the desired position in the real scenario, giving the user a better feeling on how the components in the setup will behave. In this section it is described the use of the framework AR module to plan and configure a picking and place robotic line consisting of a manipulator and a conveyor belt. The performance of the picking task are computed by the system and visualized to the user, as well as the simulated objects. A GUI provides to the worker a way to change and adjust the speed of the components and the position and flow of the parts. An additional robot can be placed and visualized in order to evaluate the improvements in the line performance deploying additional equipment.

The simulation and visualization of the parts flow and motion of other components, such as robots and conveyor belts, can ease the planning of a logistic line. The current configuration of the line could be adapted and optimized without the need to time consuming modification in the hardware configuration. The worker usually has to plan a line and build it in order to validate it. This is necessary because he does not have a tool to get a good estimate of the time needed by a robot to pick and place a part and the synchronization between all the components in the line. Many parameters, such as robot speed and how many parts can be

positioned on a conveyor belt in a minute, can be hard to determine without making tests on the real setup.

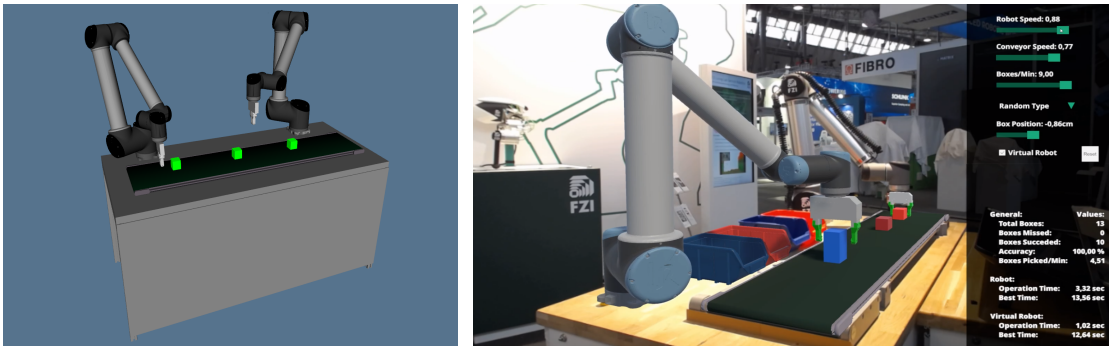
The simulation of virtual parts allows the user to check the current behavior of the planned line. The use of a GUI to make changes in the robot and conveyor speed enables the worker to apply modifications in the line without the need to manually reconfigure the entire workflow to evaluate the system. The position of parts and their number can be easily changed, providing the user useful information such as number of parts successfully handled or if any of them could be missed because of a bad planning. The picking time needed by a robot can also be estimated, as well as the throughput and cycle time. Computing and showing this information to the user, enables him to immediately understand the impact of the changes applied, in order to easily find the best configuration and set of parameters for the needed application.

In a robotic solution, it is also important to apply the changes to the real robot motion, since the user might be interested in testing the behavior of the real hardware in relation to the parameters established in simulation. For this reason, it is important that the real robot is able to communicate with the simulation system. In this way the changes can be applied directly to the real hardware in an intuitive way without the need to a reprogramming phase and programming experts. The user can also add another virtual robot to the setup in order to check the advantages introduced by the use of an additional manipulator in the line.

The implemented scenario in the AR module allows to plan and evaluate robotic solutions in a logistic line. A GUI is provided to the user, who can apply changes in the setup parameters, such as speed of robot's motion and conveyor belt. The selection of how many parts should be placed on the conveyor belt and their position on it, allows the user to explore different configurations. The performance information of the setup deploying the selected parameters is displayed and updated immediately once a change is applied. To give a feedback about the simulated configuration of the line, the virtual parts position and motion is represented through the AR visualization. The direct communication with the software components controlling the robot, allows the user to easily change and apply the modification to the real manipulator as well as to the conveyor belt. The planning of the line can be furthermore evaluated adding a virtual robot, in order to study the improvements in performances deploying additional hardware in specific positions.

The experimental setup includes a table, a conveyor belt and a Universal Robots UR10 with a Weiss Robotics WSG-25 gripper. In Figure 3.20 is depicted the virtual representation of the setup and the AR view in the real scenario, where a real robot is operating and interacting with the simulated components.

Figure 3.21 shows the AR scene, with the visualization of the virtual parts on the conveyor belt. The real robot receives commands to reach the current part to pick, in order to grasp it and proceed with the placement of it. It has also been implemented the possibility to include a virtual robot to the scene, to further test



(a) Virtual representation. (b) Simulated components shown in AR.

Figure 3.20: The virtual representation of the planned line to be shown in AR is depicted in (a). The user can inspect the robot behavior in AR, through the visualization of the virtual parts in the real setup and modifying the parameters of the simulation through a UI (b).

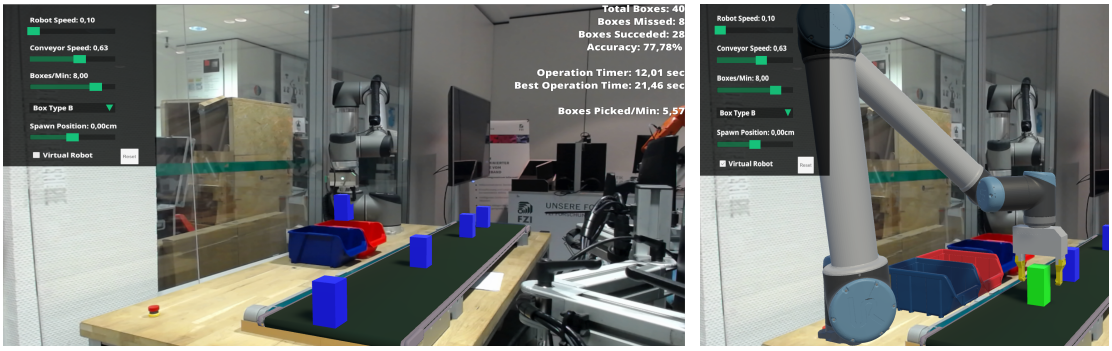
the system and make the user able to evaluate the performances of the line by adding another manipulator to support the picking and place task with the aim to increase the throughput of the process. Figure 3.21 shows the visualization of the additional virtual robot, which is positioned at the end of the conveyor belt.

In order to make the robot behavior and its interaction with the virtual objects more understandable, a change of color for the virtual parts has been added, which is based on their interaction with the robot. In Figure 3.22 this interaction is represented. Once a part is selected as current target of the robot, it is highlighted in green. After the closing of the gripper, the fingers are displayed in red to show the contact with the virtual object and the successful grasp of it.

A button in the GUI allows the user to easily enable or disable the virtual robot in the scene. Once the virtual robot is activated, it will pick the available parts in the area in which it is placed. Of course, the addition of another robot enables a better throughout if the flow of the parts or the speed of the conveyor belt are increased. Anyway, using the system developed, the user can evaluate with precision the performance of the line deploying the simulated additional robot, evaluating the best position for it in terms of reachability and speed requirements.

In the developed testing scenario, two types of parts which differ in size and color have been implemented. The communication of this information from the Unity simulation to ROS, allows the robots to react accordingly to that deploying different picking positions and joint based trajectories for placing the objects in the correct bin. In Figure 3.23 is represented the placing of a part in the bin. The use of the physics simulation allows the user to check if the selected trajectory enables a correct placing of the object in the desired bin.

In Figure 3.24 (a) are represented the input parameters available in the GUI. A set of sliders and a checkbox enable the user to change the desired parameters, as well as adding or removing the additional virtual robot from the simulation.



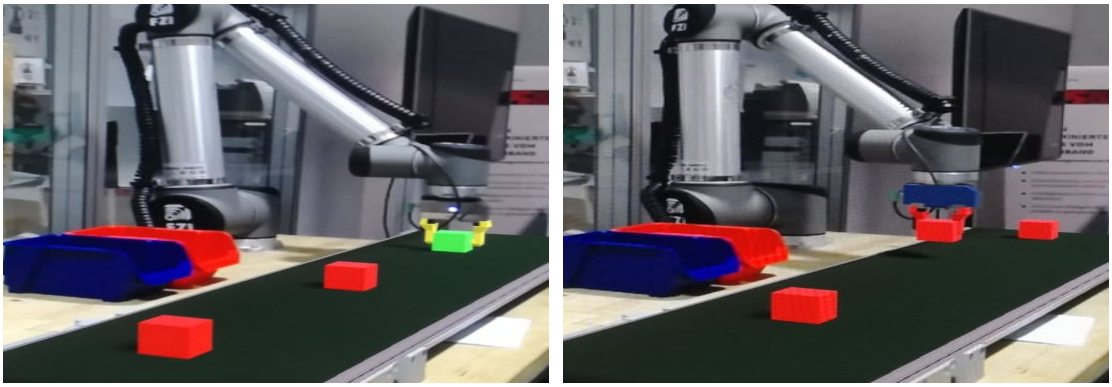
(a) Real robot interacting with virtual objects. (b) Virtual robot.

Figure 3.21: The virtual representation of the parts and conveyor belt is represented in AR. In this way, the current setup consisting of a robot and a table can be augmented and better evaluated in order to check its performances (a). The simulation of an additional virtual robot enables the user to evaluate the improvement in the line performances deploying an additional manipulator (b).

Furthermore, relevant performance metrics related to the simulated line has been included such as the total number of parts positioned on the conveyor belt and information about how many or them have been successfully picked and missed, providing also a percentage related to the current accuracy of the system. As further information has been added a timer, which measures the time taken by the robot to complete the current pick and place operation. The best operation time achieved is also displayed, in order to provide the user with an useful information to understand if the operation cycle time is improving acting on the input parameters. The throughput of the entire line is also visualized to give an additional information on how many parts per minute the system is currently able to handle correctly. Figure 3.24 (b) shows the performance information displayed by the system and related to the current state of the simulation.

A checkbox enables the user to evaluate the performances enabling and disabling the additional virtual robot. A reset button can be used to clear the actual run of simulation, restoring the performance values to zero.

In order to evaluate the intuitiveness and ease of use of the system, the experimental setup has been presented at the Motek 2019 fair in Stuttgart, where many people had the opportunity to interact with the setup in order to change the provided parameters and get to know the consequent performance results. In Figure 3.25 is represented the hardware setup used for the test demonstration, which inclused a conveyor belt, a 6-DOF robotic manipulator equipped with a 2-finger gripper. A touch screen interface has been used to show the augmented simulation scenario, including the GUI to modify the process parameters and the visualization of the achieved line performance.



(a) Current target. (b) Object picked.

Figure 3.22: In (a) the current target of the robot is highlighted in green. As depicted in (b), once the object is successfully picked, the fingers of the gripper are represented in red to show that the object has been correctly grasped.

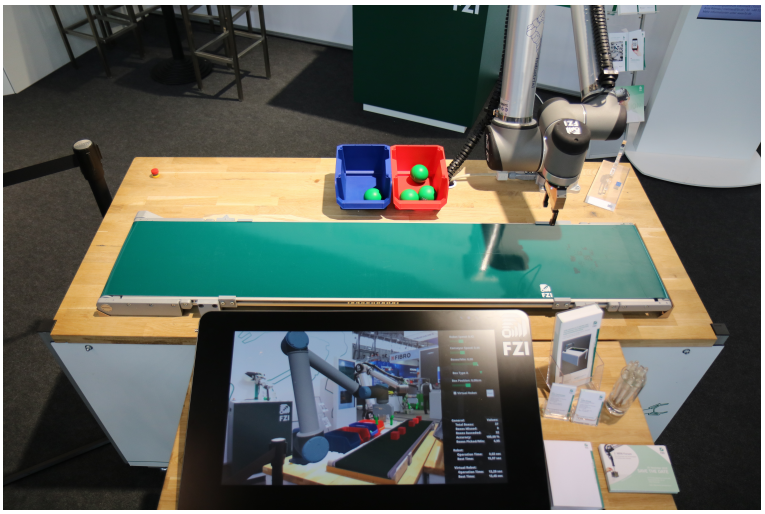
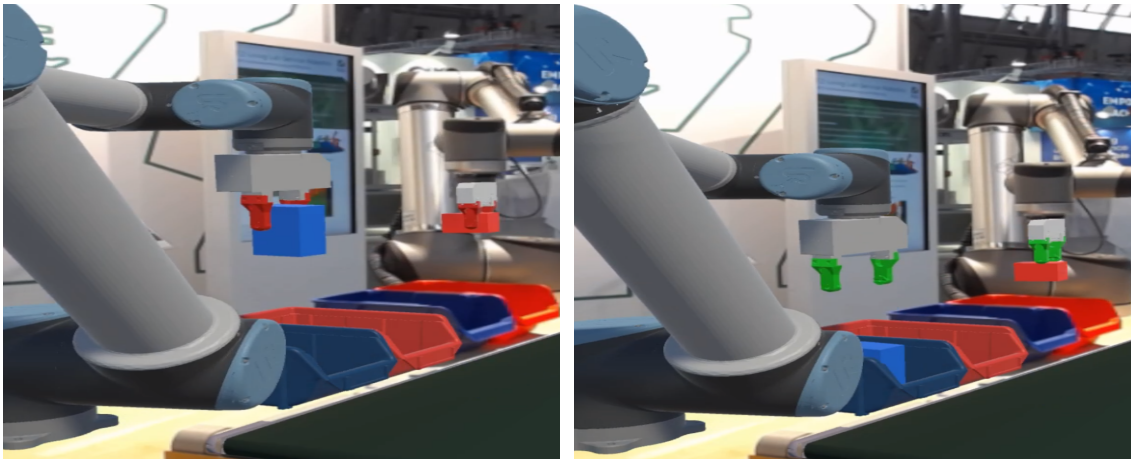


Figure 3.25: The setup used for demonstration at the Motek 2019 fair. The line includes a conveyor belt and a 6-DOF robot equipped with a 2-finger gripper. The users could use a touch screen interface to easily change the simulation parameters, getting immediately the computed performance of the line.

3.1.4 Conclusions

The proposed framework enables an intuitive and effective way to plan and validate complex robotics system, focusing on intuitive interfaces in order to avoid the need of experts and programming skills. The proposed three modules allows to bridge the gap between simulation and real world, from pure offline simulation in VR, test and validation in AR to then the final execution on the real hardware



(a) Robot moves the object to the placing area.

(b) Object placed successfully in the correct bin.

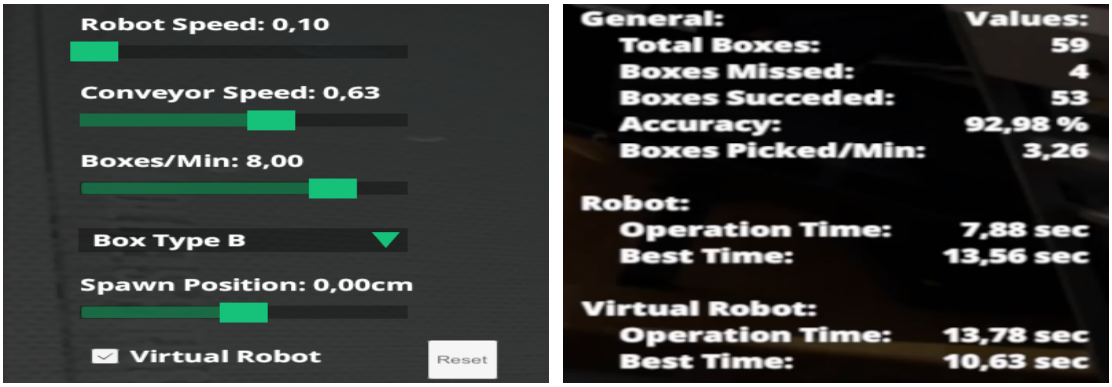
Figure 3.23: In (a) the virtual robot executes the trajectory to place the picked part in the correct bin. As represented in (b), after the release of the object, the physics simulation allows to check if it is correctly positioned inside the bin.

through ROS. The framework architecture enable the exchange of robot programs and simulation data between the different models, making possible to switch between them without any loss of information.

The building blocks implemented in the VR environment enable the definition of complex scenarios involving multiple robots and other components, by simple selecting the desired component from the catalog and moving that to the target position using the VR controllers. In the same way, the robot can be programmed intuitively by just grabbing and dragging the TCP position to the desired configuration. The UI shown in the simulation provides simple buttons to select the desired operations and specify the execution order.

Furthermore, the performance of the line are automatically computed and measurement sensors can be placed in the simulation to compute the relevant metrics in specific position of the process.

The implemented experiments validated the concept, proving that with the proposed framework is possible to simulate arbitrary complex robotic systems and the programs defined in the simulation can be immediately executed on the real hardware without the need of further programming. Moreover, the pick and place VR application proved that people with no experience with the system have been able to program a robot equipped with a gripper to perform the task of taking and positioning objects in a box, requiring a short time to program the manipulator and make it execute the needed motions. The developed evaluation application, provides a base for collecting further users' feedback to further improve the usability of the system.



(a) GUI input parameters. (b) Computed system performance.

Figure 3.24: In (a) is represented the GUI provided by the system to easily and intuitively change the relevant parameters of the line. The speed of robot and conveyor belt can be modified, as well as the position of the parts and the rate of their placement on the belt. Using buttons an additional robot can be integrated in the simulation and the performance information can be reset. The output computed by the system for the current configuration is depicted in (b). The user can check the accuracy of the systems in terms of successful operations and throughput of the line. A timer gives information about the current picking time needed by the robot to correctly pick and place an object. The best time achieved is stored as a reference that the user can take into account while further changing the input parameters.

Further work could aim at expanding the library of available models, enabling the easy integration of custom models from CAD files. Furthermore, the advance in AR technology could be used for providing a more immersive AR visualization using head-mounted display.

3.2 Gesture-Based Control

Human–Robot Interaction (HRI) plays an important and growing role, both in industrial applications and in game development. Over recent years, robots can be controlled by gestures via special devices, but these methods are not intuitive and require usually a learning phase. In this section, an intuitive method for controlling a robot end-effector using human gestures is proposed. Vision based techniques are used to track the position of the user's hand, which is directly translated in control signals. The use of a 3D camera sensor allows to easily control the robot tool position in all dimensions. The system includes a GUI to ease the control through interactive visual feedback. This interface, including 3D markers, text messages and the visualization of the user's point cloud and robot model, enables a control mechanism which does not require a teaching phase to perform before the actual control of the robot.

The approach has been tested and evaluated using realistic experiments to prove that the method works reliably and is extremely intuitive. Two different types of 3D cameras has been used to control the end-effector position of a 6 degree-of-freedom (DOF) robotic arm. The method has been tested implementing an application based on the popular claw crane, in which a robot needs to be controlled to catch prizes. This was used and tested by hundreds of people with and without robotic experience during a three days fair. To further test the system in a standardised, quantifiable manner, a pick and place application has been designed, in which the robot is equipped with a two fingers gripper. A group of users with no experience with the system had to control the robot to perform the task and the time required to place multiple objects in a bin using the control method developed has been measured.

The results presented in this section have been published in [8].

3.2.1 Method

The method described in this section focuses on the design and implementation of an intuitive gesture-based method to control the robot end-effector motion in 3D space. The user, standing in front of a 3D camera, uses the position of his hand related to a reference point in order to control the position of the robot end-effector. The control method concept is shown in Figure 3.26.

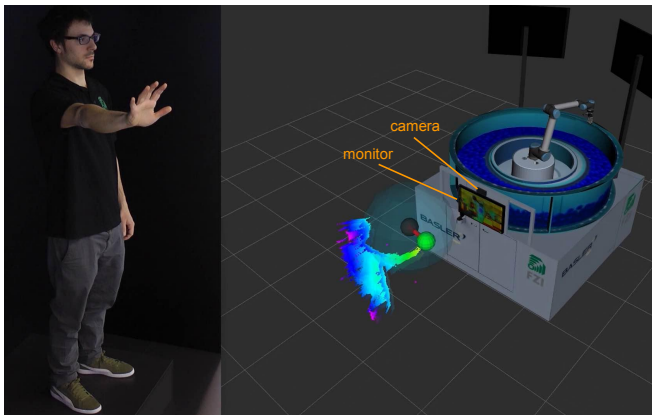


Figure 3.26: Robot control through gestures. The left part of the image shows a user controlling a robot through the movement of his hand. In the right part the visualization of an example set up is visualized in RViz. Primarily this concerns the point cloud of the user and the model of the robot.

One of the key aspect of the proposed method is the use of an intuitive graphical interface in order to allow the user to understand how to interact with the robot during the control. The use of 3D markers, camera data and text messages helps the human operator to have a better understanding of the control area, preventing

undesired commands. In this way the user has an immediate feedback and is able to control the robot without the need of a previous training phase. The aim of this approach is to enable a flexible and intuitive robot control. In particular the method should enable people with no experience to interact with a complex robot without the need of any particular knowledge. By means of the GUI the user has immediate feedback about the commands sent, in order to understand how to interact in the proper and intended way. The intuitiveness of the approach is a really important aspect, since robots are becoming more and more common in everyday life. Therefore, the users need a way to control them without complex methods and complicated interfaces.

A graphical overview of the approach, including the relevant components and their interaction is provided in Figure 3.27. To provide an intuitive way to command the robot motion, a 3D camera to track the position of the user's hand has been used. The user can move his hand in front of the camera in order to send commands to the robot. A segmentation method based on the closest point detected by the camera is used to track the hand position.

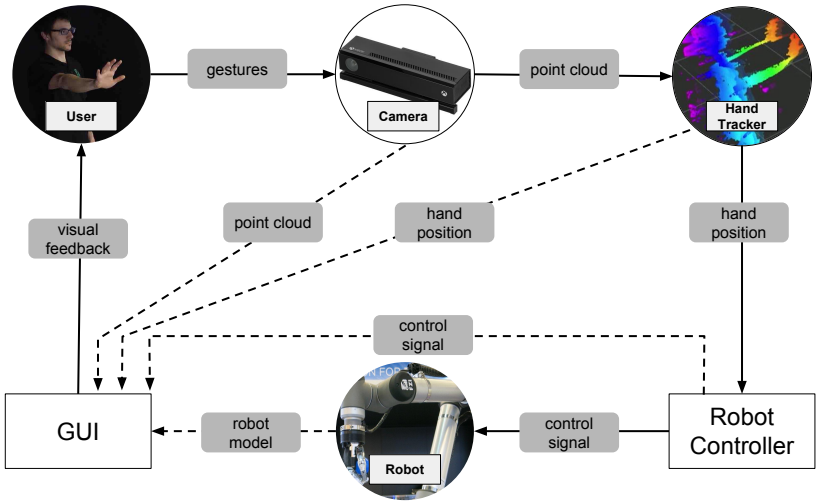


Figure 3.27: Overview of the closed-loop pipeline, representing the components and their relationships. A 3D camera captures the user's body and provides this information as a point cloud to the hand tracking module. The position of the hand is then used to generate the robot control signals. The GUI gives feedback to the user representing the body point cloud, the robot model, the position of the user's hand and the current control signal as 3D markers.

In order to start the control of the robot end-effector, the user has to place its hand steady in front of the camera to initialize the system with a reference point. This point is used to compute the relative position of the hand during the control phase. The relative position of the user's hand is used to send a velocity command to the robot end-effector. This linear velocity has the same direction and it is

proportional to the distance vector of the user hand. In Figure 3.28 and 3.29 a graphical depiction of this coherence is provided.

To test the system, a Universal Robots UR10 robot equipped with a Schunk SVH 5-finger hand has been used. In this way, using the system developed, the user is able to control the 3D position of the robotic hand. An additional command to enable the grasping of objects has been added. When the user puts both of his hands in front of the camera the robot arm stops its motion and only the robotic hand is enabled to grasp an object. The two hands are detected by the vision system when, at a similar distance to the camera, two clusters are recognized inside a certain window. The window and clusters size have been tuned considering the average size of human's hands. In this work this command has been used to switch between two fixed configuration of the tool: considering the anthropomorphic hand, the complete opening and closing of the fingers grip.

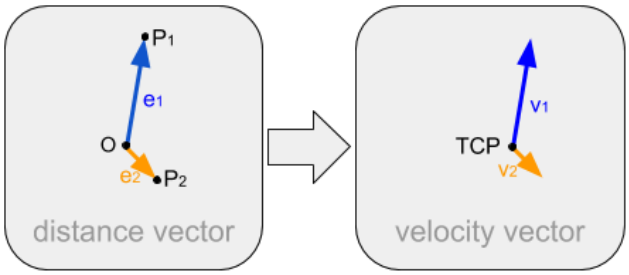


Figure 3.28: A technical drawing of the interrelationship between the distance vector of the user hand (e_i) and the velocity vector command sent to the robot end-effector (v_i).

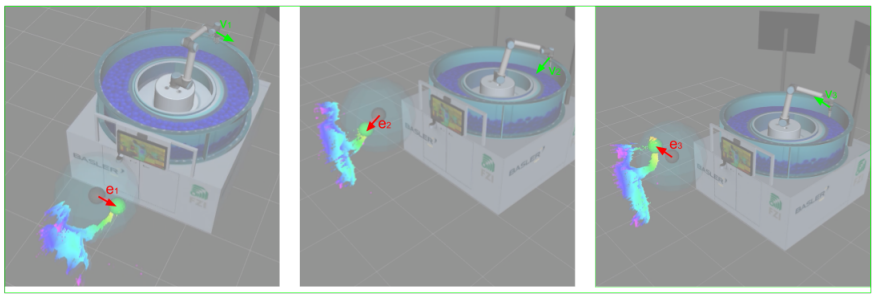
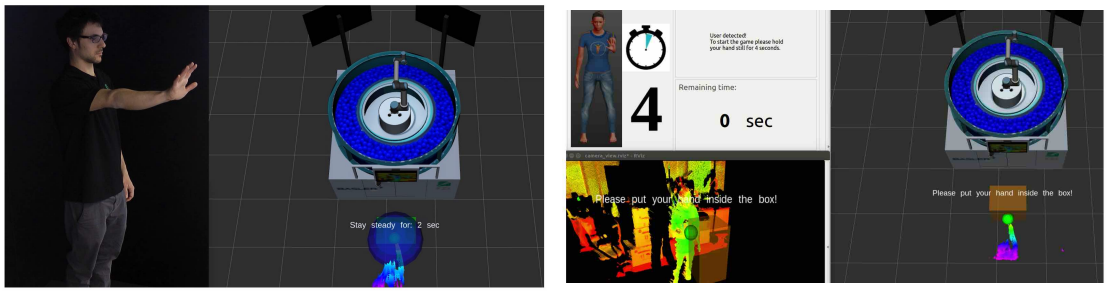


Figure 3.29: The interrelationship between the distance vector of the user hand (e_i) and the velocity vector command sent to the robot end-effector (v_i), visualized by use of the robot model and the user's point cloud.

Since the method focuses on the intuitiveness of the control, the possibility to limit the allowed workspace of the robot is an important aspect. In fact, in this way, it is possible to avoid the robot to reach undesired and dangerous configurations. This allows non-expert users to control it without worrying about damages caused by collisions with the robot itself or the surrounding environment. Before the control phase, the user can input the desired limitations using sliders provided by

a GUI. These parameters are related to the position of the robot end-effector. It is possible to set the minimum and maximum distance from the robot base link and the minimum and maximum height allowed. Further parameters allow to restrict the movements of the robot limiting the joints range. The user can also specify an height threshold to define areas in which the robot should have a different speed. This is useful when there is a need to approach objects with limited speed in order to achieve a better precision in manipulation tasks.

An intuitive GUI assists the user during the interaction, by helping him to initialize the position of the reference point and to visualize the relative position of the tracked hand.



(a) User starting the initialization procedure.

(b) GUI providing support and instruction to the user.

Figure 3.30: Initialization of the hand reference pose. The user positions his hand steady for three seconds within the virtual box marker (a). Text messages and 3D markers assist him during the procedure, providing information about the boundaries and camera field of view (b).

Figure 3.30 shows the initialization phase in which the user has to put his hand in front of the camera. In order to initialize the system at the desired point, the hand has to be positioned steady for three seconds. The GUI helps the user to check the position of the detected hand together with the resulting boundaries. Since the distance from the camera and its field of view restrict the control area, a box marker helps him to stay in the right position. It is also possible to see how text messages invite the human operator to initialize the reference point in the suitable area. To give an immediate feedback about the correctness of the initialization, the box marker turns green when the hand is within this area. After that, a text message informs the user to hold the position for the remaining seconds needed.

The use of 3D markers allows the user to understand the intensity and direction of the velocity command sent to the robot, having an immediate feedback of the control signal transmitted to the manipulator in order to enable quick adjustments. Figure 3.29 shows the user hand position represented with a green sphere and the marker representing its direction related to the reference point. Through the visualization of the control area limits, the user can understand immediately when his hand is outside of the control boundaries. The control of the robot stops immediately when the user hand is outside of the control area for more than a time threshold. Text messages inform him about the current status of the control.

As mentioned previously, in order to evaluate the control system, an additional command to enable/disable the tool mounted on the robot has been implemented. When the user puts both his hands in front of the camera, the tool command is sent. For example, using the robotic hand, this command enables the opening and closing of the fingers to catch an object. The same is done for different types of grippers. A user performing the command is shown in Figure 3.31.

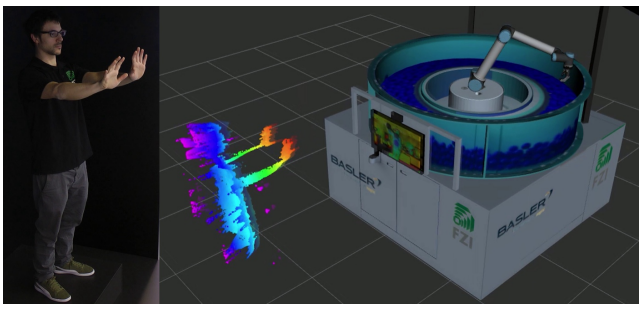


Figure 3.31: The user puts both hands in front of the camera to initialize the grasping of an object with the robotic hand.

3.2.2 Experiments

In order to test the developed system, the proposed method has been integrated in an application based on the popular Claw Crane game, known from carnivals and funfairs. In the considered application, users need to control the position of a robotic gripper in order to grasp prizes. An anthropomorphic hand has been mounted on the robot arm in order to make it possible to grasp small balls. The different balls colouring represented different prizes. In order to collect the desired prize, the user had to catch a specific ball. Once this latter was reached, the grasping command could be used to close the robotic hand and finish the game. This use case was tested for three days during the Embedded World 2018 fair in Nuremberg. Figure 3.32 and 3.33 were taken during this event and show different parts of the system in use. Users with no experience in robotics were able to control the robot and catch the targets without the need of further instructions, by just following the information and visualization provided by the GUI. Furthermore, the implementation of the approach has been tested with two different cameras, a time-of-flight camera (ToF camera) from Basler and a Kinect 2. The system works with both cameras without any further adaptations, demonstrating the modularity and hereby implied reusability of the proposed modules.

To further evaluate the system, a simple pick and place scenario has been designed, in order to collect the time needed to perform such task and how the confidence with the control system improves over multiple attempts. The user had to catch objects positioned in different places within the robot's workspace and position them in a bin. The robotic hand has been replaced replaced by a two fingers gripper in order to have a more precise grip. The time needed to position each

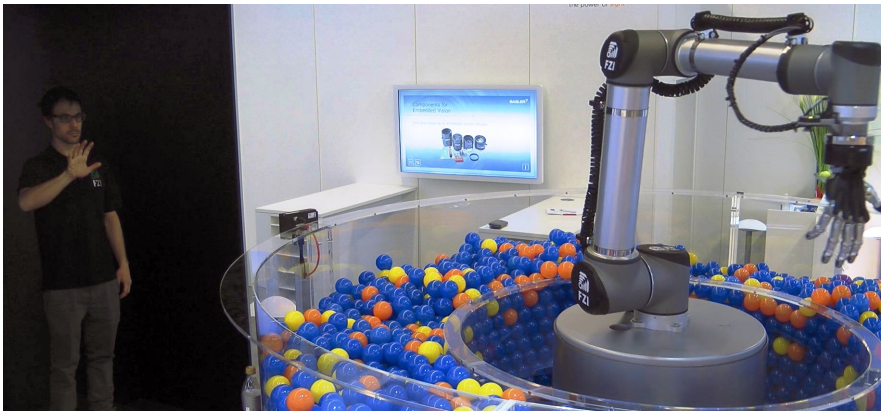
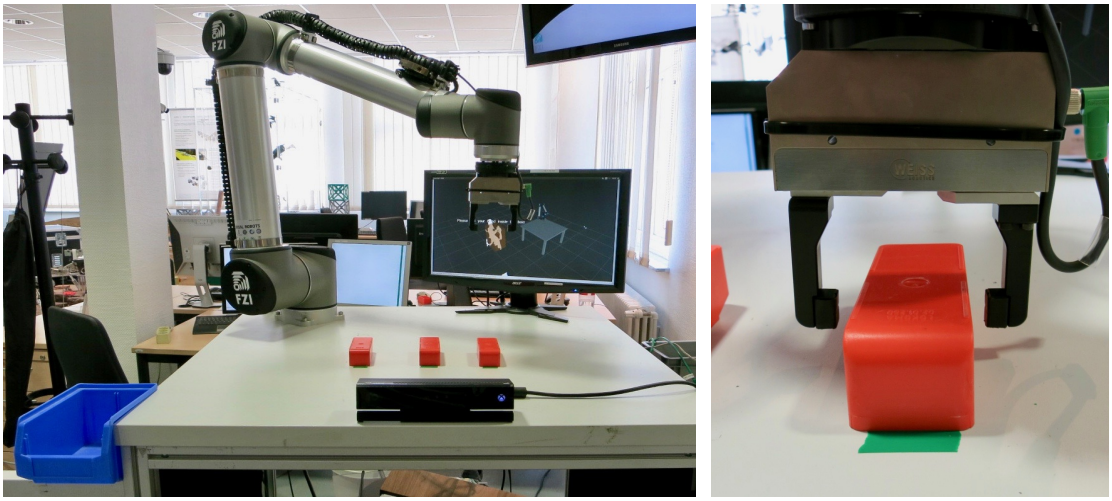


Figure 3.32: Claw crane game application. The user controls a 6 DOF robot equipped with an anthropomorphic hand, in order to grasp the desired prize.

part and the total task completion time hs been measured. All the users tested the system with the same parameters values for speed and distance from the camera. In Figure 3.34 is shown the evaluation setup, with three objects positioned on the worktable, the Kinect 2 used to detect the user commands, the bin for the deposition and the gripper mounted on the robot. As represented in Figure 3.34b, the position of the gripper had to be very precise in order to enable a successful uptake.



(a) Pick and place scenario. (b) Picking of object.

Figure 3.34: Pick and place scenario. A 6 DOF robotic arm and a two fingers gripper are used to pick three objects and position them in a bin (a). The gripper position has to be controlled very accurately, in order to achieve a successful uptake (b).

The pick and place task has been tested with several object locations to prove that, with the proposed system, a robot's end-effector position is controllable accurately

(a) The anthropomorphic hand catches a specifically coloured ball.

(b) The robot's hand holding a ball, after a successful grasp.

Figure 3.33: Grasping of ball after the control of the robotic hand position. The user controls the robot hand position in order to reach the desired target. Once the grasping command is sent, the closing of the robotic hand enables the grasping of the object.

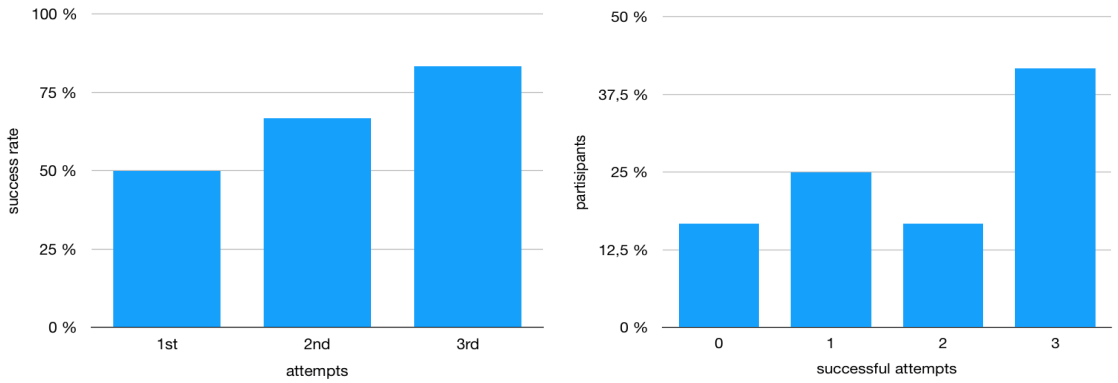
in the entire 3D space. A lower speed limit was set for the gripper positions within a few centimeters above the surface of the table. In this way the user could achieve a more precise control in order to pick the objects. Table 3.1 provides the mean values of the time needed to lift and position each part separately. Additionally, the time taken to complete the entire task is also given as mean. It is conspicuous that the mean value of the third trial-run is the highest out of all three. However the conclusion that people became slower over time is false. The table's values only include the needed amount of time for users which successfully completed the task. Therefore, more people were taken into account for the last trial, also users which failed in the previous attempts.

Part	Part pick and place time	Task completion time
1	25s	69s
2	16s	
3	28s	

Table 3.1: Time as mean values for the picking and placing of the single objects and for the completion of the entire task.

On top of that, the intuitiveness of the system has been evaluated using the same pick and place setup. Twelve participants tested the proposed control method performing the task to pick the objects on the table. After a brief instruction, they had three attempts to complete the quest. Each trial had a time limit of 30 seconds. With this test, the intuitiveness of the method has been demonstrated, since even people without experience managed to control the robot's position precisely. In Figure 3.35 the results of the intuitiveness evaluation are reported. From the first chart (Figure 3.35a) it is possible to see that half of the participants have

successfully picked up an object within 30 seconds at the first try. It shows also that most of the users learned really fast how to correctly interact with the robot, achieving the goal after the second or third attempt. The second chart (Figure 3.35b) shows the percentage of participants related to their successful attempts. It emphasizes that most of the users have managed to successfully pick an object in all three attempts. Only a small percentage of participants were not able to pick up an object within the first three tries.



(a) Percentage of participants for the three attempts.

(b) Percentage of participants for successful attempts.

Figure 3.35: Chart representing the percentage of participants that have successfully picked an object for each attempt (a) and chart reporting the percentage of participants related to their successful attempts (b).

The results proved that using the proposed gesture-based control, unexperienced users were able to control the position of the robot tool in the entire 3D space. The accuracy of the system has been highlighted. Furthermore, the time results showed that using the system is possible to achieve a reasonable task completion time for standard robotic applications. The intuitiveness evaluation demonstrated that people with no prior experience were able to control a complex robot almost immediately.

3.2.3 Conclusions

The vision-based control system presented has proven to be a very intuitive way to control a robot and create new HRI applications. The focus of the method was to enable the easy control of complex robotic systems, such as 6 DOF robotic arm equipped with a humanoid hand, for people with no experience by using 3D vision algorithms and an intuitive interactive feedback system.

In contrast to other methods, the proposed approach went beyond the execution of some predefined tasks or motion control in a finite number of directions, enabling the user to control the robot in a flexible and immediate way.

The implemented system allowed people with no experience, to control the end-effector position of a complex robotic system without the need of any particular knowledge. With the use of the intuitive graphical interface the user received immediately a feedback about the commands sent and noticed when his hand was outside of the control boundaries.

Further developments could extend the control to include also rotations, as presented in the next section, and to enable the control of each finger of the robotic hand, in order to allow more complex manipulation tasks. Additional developments could focus on the addition of AR tools in order to overlay the feedback interface information into the real world. In this way the 3D markers and text information can be immediately be seen by the user at the needed position. For example, the detected direction and module of the user's velocity command could be visualized directly next to operator's hand, without the need to look at external devices.

3.3 Gesture-Based Control Using Constrained Multi-Modal Interactive Strategies

Service robots are becoming able to perform a variety of tasks and they are currently used for many different applications. For this reason people with different backgrounds and also without robotic experience need to interact with them. As highlighted in the previous sections, in order to make the inexperienced user able to control the motion of the robot end-effector, it is important to provide him with an easy and intuitive interface. In this section, an intuitive method for the control of a robot TCP position and orientation is proposed. This is done taking into account the robot kinematics in order to avoid dangerous configurations and defining rotational constraints. The user is enabled to interact with the robot and control its end-effector using a set of objects tracked by a motion capture system. During the interaction, the robot provides different level of autonomy depending on the different phases of the task, in order to achieve a better efficiency and ensure safety. An intuitive GUI has been developed to ease the interaction and help the user to achieve a better precision in the control. This is possible also through the scaling of the tracked motion, which is represented as visual feedback. The system has been tested through multiple experiments that took into account how people with no experience interacted with the robot using the proposed control approach and the precision of the method.

The results presented in this section have been published in [1].

3.3.1 Method

The proposed method is based on the design and development of a system to control a robot in an intuitive way, with the use of different autonomy levels. In order to achieve this, the proposed system focuses on controlling a robot on different ways depending on the task. For example, in order to control a service robot to perform different tasks, various tools might be needed and the user might want to make the robot pick the desired one, without worrying about the specific trajectories to achieve that. On the other hand, once the needed tool is selected, a flexibility in the control is required to enable the execution of arbitrary tasks and actions. This flexibility should include the possibility to control the 3D position of the tool as well as enabling the rotation of it, without any risks caused by unsafe configurations due to the kinematics of the robot.

With the proposed method, the user starts by taking a tool from a set of control objects tracked by a motion capture system. This triggers the robot to autonomously get the corresponding tool and move it to the control area. At this point, the user can teleoperate the 3D position of the robot TCP by moving the tracked tool. The user also controls the rotation of the TCP through the orientation of the tracked object in a safe manner. In the proposed approach, the inclination angle of the tracked tool is mapped into the rotation of the last joint of the robotic manipulator. In this way, dangerous reconfiguration of the robot kinematic due to arbitrary rotations can be avoided, allowing a flexible and safe control of the manipulator TCP.

Figure 3.36: The user controls the robot TCP through the use of a tracked object. A graphical user interface guides the user in the interaction, enabling also a better precision in the control.

The method focuses on the ease of use for people with no robotic experience, enabling them to interact with a robot through different control levels to achieve complex teleoperation tasks. A GUI has been designed to support the user during

the interaction, providing instruction based on various the phases of the control. In particular, a visual feedback for the teleoperation has been added to enable an easy and precise control of the robot tool.

The needed components and the communication between them have been implemented using the open-source framework ROS [12]. In order to test the system with unexperienced users, the approach has been developed for an application in which a service robot had to decorate cookies through the interaction with the user. However, the approach is modular and generic and can be deployed for general applications and with arbitrary robotic arms.

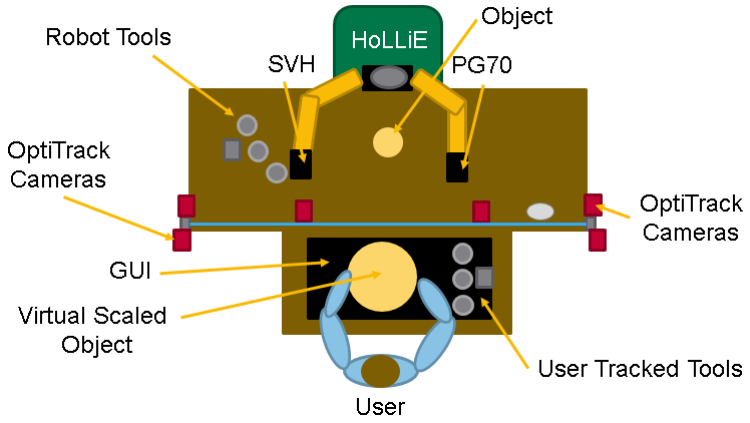


Figure 3.37: The overall hardware setup used for the interaction between HoLLiE and the user.

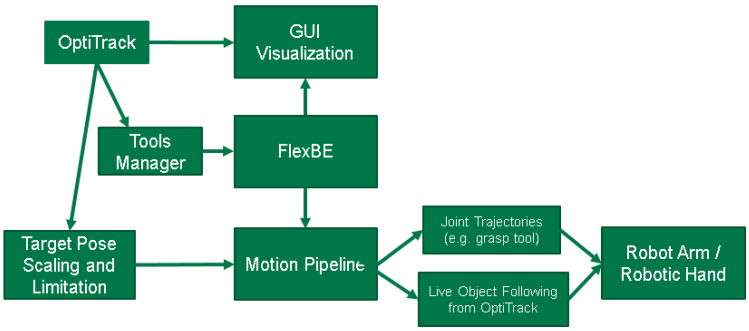


Figure 3.38: Overview of the system software architecture and communication between components.

For the proposed method implementation, the service robot HoLLiE has been used, which has two PILZ PRBT robotic arms with 6-DOF. In particular, the focus has been placed on the control of one arm, which is equipped with a Schunk SVH 5-finger hand. The setup includes the marker based tracking system OptiTrack and a monitor to display the user interface. The overall hardware setup is depicted in Figure 3.36, while Figure 3.37 shows in details the different hardware components involved. Figure 3.38 represents the system software architecture

and the communication between the main components. The behavior framework FlexBE [98] has been used to define the high level logic of the interaction and the handling of the events coming from the other modules. The control of the robot arm is done by alternating between a joint-position controller and the Cartesian controller described in [96], implemented with ROS control [29]. The joint-position controller is used to autonomously reach, grasp and dispose tools with fixed trajectories defined with the FZI Motion Pipeline. The Cartesian controller is used to teleoperate the robot in a constrained manner.

To enable the control of different tools, the user has to possibility to choose between a set of control objects, which positions and orientations are tracked all the time. The use of Optitrack allows to track the 6-DOF pose of the needed objects with sub-millimeter accuracy and low latency. A set of six cameras has been used to monitor the user's workspace and ensure the tracking of the tools for arbitrary positions of those within this area. Each of these objects has a set of reflective markers that are used by the tracking system. Once the user grasps or releases one of them, the robot is triggered to execute the same action on the real one, which is positioned in a fixed position in its workspace. In order to identify the different tools, different patterns have been used for each one. Their home positions are marked on the GUI and once the user takes one of them, the execution of the corresponding robot's trajectory is triggered. Figure 3.39 shows a set of tools, each one with a different pattern of markers. Both flat and 3D markers are used, depending on the shape of the object. This is done to ensure the tracking of the objects while they are grasped and manipulated by the user. The Tools Manager component keeps track of the state of the tools, which is updated accordingly to their distance from their home positions. A specific tool is activated when its distance to its home position is above a threshold value. In the proposed system only one tool can be active at a time and the feedback on the display confirms to the user the current active one. The Tools Manager also communicates with the Target Pose Scaling component, in order to provide the current active tool id to compute the robot target position. As a feedback to the user, the GUI highlights which object has been taken and reports the information that the robot is executing the trajectory to reach and grasp it.

Figure 3.39: A set of control tools which can be used by the user to interact with the robot. Each one of them is identified and tracked by the Optitrack system through the use of different patterns of markers.

After the grasp of the desired tool, the robot moves to the control area and the GUI notifies the user when he can start to control the robot TCP. In Figure 3.40, it is represented the situation in which the user has selected one tool and the GUI notifies him to wait for the robot to get the corresponding one. As represented in Figure 3.41, it also displays as feedback the current active tool and its placing position which is used to finish the interaction or switch to a different one. The GUI displays also a virtual representation of the control area in order to give the user a better understanding of the robot workspace limits. During the teleoperation mode, if the position of the tracked tool is outside of the control boundaries, the closest valid position is sent as target goal. This allows the user to commit errors during the control, avoiding unwanted robot configurations and possible collisions with the surrounding environment. To control the robot with the Cartesian poses computed from the tracking system, the Cartesian controller proposed in [96] has been used. This controller is based on forward dynamics trades-off precision to avoid singularity and sudden reconfigurations. The parameters of this controller allow to set a more reactive or smooth control depending on the application. The position of the tool captured by the tracking system is converted into the reference frame of the virtual object displayed on the GUI. Figure 3.42 shows the reference frames that are used to transform the tracked pose into the desired robot target. In this way the user can move the tool above the virtual reference and its pose is then transformed into the real object reference frame. This position is the one used to send the target goal to the robot controller. Before that, the scaling is applied in a coherent way with the virtual representation on the GUI. The scaling of the tracked movements allows the user to have a better precision in the robot control on a small volume and this can be set up for the needed axis.

Figure 3.40: The GUI notifies the user that the robot is reaching the tool that has been selected. In this way he can understand the robot behavior and detect when the teleoperation mode is enabled to directly control the manipulator.

Figure 3.41: During the teleoperation mode, a scaled virtual representation of the object to be worked is displayed. This is useful to understand the control boundaries and to allow a better precision with the scaling of the motion. The GUI highlights also the current active tool and its placing position which is used to end the interaction or switch to another one.

In order to allow the flipping of the robot tool, an approach to control the rotation of the robot TCP based on the orientation of the tracked object has been designed and implemented. The relevant angle to make it flip, it is the one between its axis and the vertical axis. To avoid the robot to reconfigure completely its kinematics and to make it able to reach all the desired positions without dangerous reconfigurations, this angle is mapped into the rotation of the last joint of the robot arm. In this way, the kinematic configuration of the robot remains unchanged and only the rotation of the tool is controlled. In Figure 3.43 is represented the approach that allows to map the desired angle to the last robot joint rotation. The resulting rotation is then coupled with the desired position, providing a 6-DOF Cartesian target pose

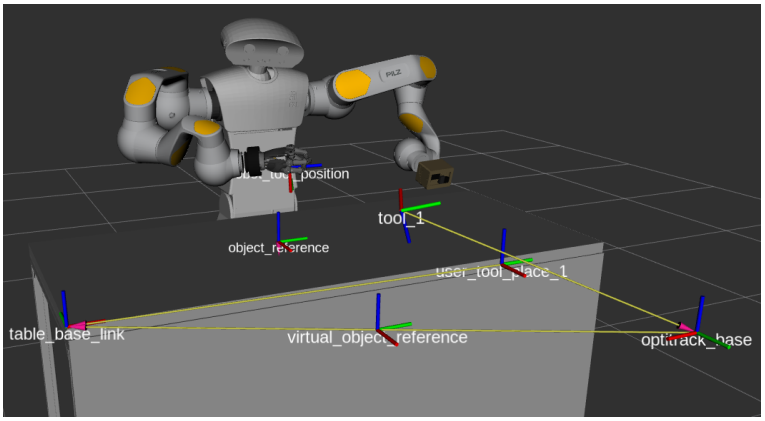


Figure 3.42: The reference frames used to transform the position of the tracked tool into the desired target of the robot. The information from the tracking system is converted in the reference frame of the displayed virtual object, which is then scaled and transformed into the reference frame of the real object in the robot workspace.

to the controller. The use of fixed trajectories to reach the control area has the big advantage to guarantee a safe joint configuration at each start of the Cartesian control. This prevents sudden configuration changes.

To handle the different phases of the interaction, a state machine implemented in FlexBE [98] has been deployed, which is an open source high-level behavior engine. In this way, the various events from the tracking system trigger the state machine to execute the corresponding predefined joint trajectory. Once this execution is finished, the state machine is responsible for the switch to the Cartesian controller in order to enable the teleoperation mode. The placing of the tool back to its home position triggers then the switch to the joint trajectory controller in order to make the robot place the tool back.

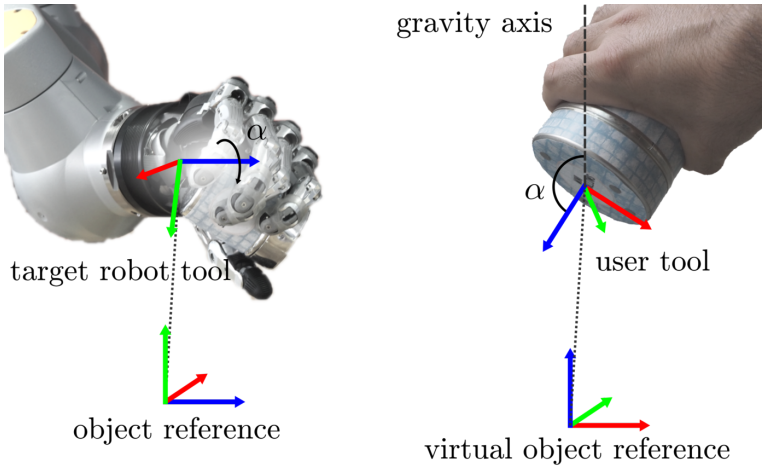


Figure 3.43: The relation between the robot tool pose and the tracked user's tool pose. The translation of the target robot tool is the same as the user's tool in their respective reference frame, as depicted with the dotted lines. However, the target robot tool only rotates with respect to a single predefined axis corresponding to the last joint of the PILZ robotic arm. This facilitates the task of the Cartesian controller which only needs to control the last joint to decrease IK errors. The angle of this rotation α is set to be the angle formed by the user tool and gravity axis.

3.3.2 Experiments

In the developed experiment application, the system has been used on the service robot HoLLiE. The user task was to control the robot arms to decorate cookies with four different toppings. The robot had two different types of tools to use: three canisters containing sugar decorations as sprinkles, stars and marshmallows and a sugar dispenser. The canisters needed to be flipped in order to make the toppings coming out and the sugar dispenser was enabled through the use of an actuated valve. This latter was activated or deactivated based on a height threshold of the tool from the table surface. The user had a corresponding set of control tools and he could decide which one to use just taking one from its position marked on the GUI. This was displayed on a monitor mounted on top of a table. After waiting for the robot to get the selected tool, the user was able to control the TCP with the proposed method. A virtual representation of the different toppings was displayed in order to give a feedback to the user about the activation of the tool during the control. For example, the sugar dispenser was enabled based on the distance of the tracked object from the monitor displaying the GUI. In Figure 3.44 is represented the simulation of the sugar flow, that was used to give the user a feedback about the activation of the tool and to help him to define the desired path with more precision. The right arm of HoLLie has been used to grasp the tools and decorate the cookies, while the left arm was equipped with a Schunk

PG70 gripper in order to place the cookies in the control area and deliver them to the user through the use of a spatula. The control of the two arms was completely decoupled with the use of the FZI Motion Pipeline and FlexBE. In this way the cookies could be handed to the users with the left arm, while the right one was placing back the current tool, with a consequent decrease of waiting time.

Figure 3.44: The GUI shows a simulation of the sugar flow that is dispensed through the tool. This is useful to give the user a feedback about the activation of the tool and help him in the definition of the desired path.

Real-world user evaluation

The proposed method was tested at the Stallwächter Party 2019 in Berlin, where more than two hundreds people with no experience with the system and with no technical background have tested the developed application. All the users understood quickly how to use the system without the need of additional instructions. They interacted properly with the robot in order to get the desired tool. They were also able to control the position of the robot TCP to decorate the cookies using all the tools available, drawing letters and arbitrary shapes on the cookie surface. Figure 3.45 shows a user decorating a cookie with the sugar dispenser tool and using the GUI to have a visual feedback about the tool activation. In this application, the rotation control was also important for some of the tools, in order to make the the toppings coming out. The event participants understood easily how to get the desired rotation of the TCP acting on the tracked tool orientation.

Performance evaluation

To further evaluate the precision and motion latency of the system developed, a test scenario in which an additional robot is used to simulate the user's movements has been designed and implemented. A Universal Robots UR5 has been mounted on a

Figure 3.45: A user without previous experience with the system uses the sugar dispenser to decorate the cookie and the GUI to have a feedback about the tool activation.

table and coupled with a tracked tool attached to its end-effector. This additional robot was used to repeatedly execute a predefined motion between two points in the workspace which are tracked by the cameras. At the lower position a button has been used to record the time in which the position was reached. A similar button was placed in the HoLLiE workspace in order to record the time in which the teleoperated robot reached the same position. In this way the motion latency between the two events has been measured. In Figure 3.46 the setup used for the experiment is reported. The evaluation has been done using different velocity settings for the UR5 motion.

The motion latency between the movement of the tracked tool and the reaching of the goal position for the teleoperated robot is caused by many factors including the delay introduced by tracking system, the computation of transformations, the Cartesian controller and the communication between the components. The charts in Figure 3.47 and 3.48 report the motion latency of the system expressed in milliseconds. The results are plotted for different velocities settings of the UR5 and for 50 evaluation iterations. Using the maximum speed, which is 1500 mm/s, the motion latency caused the teleoperated robot to not be able to reach the target position in order to press the button. The same issue was faced also for some attempts with the speed set to 75%. The chart in Figure 3.49 represents the numbers of failed attempts for each robot speed.

The results of the experiments showed that the system presents a motion latency, which is due to the various components and their communication. The Cartesian controller adapts to this by not reaching all the positions reached by the tracked objects. However this showed to be quite negligible and not proving issue in the reach of the target position, except for high speeds.

Figure 3.46: The setup used to evaluate the precision and motion latency of the system. A UR5 was used to move the tracked tool between different positions. The time needed to reach the different poses was measured with two buttons placed in the robots workspace.

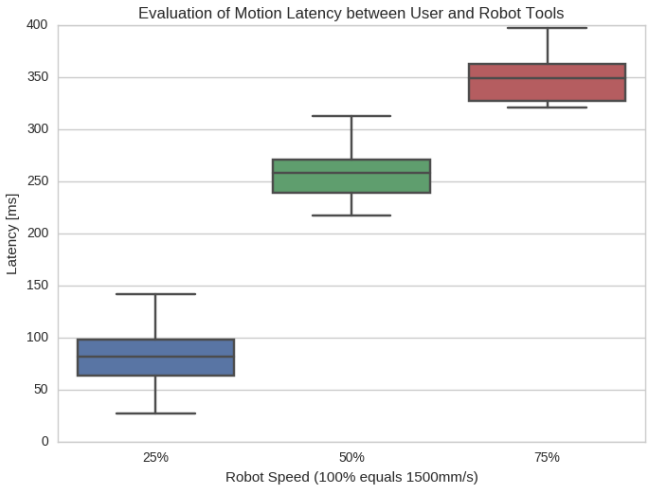


Figure 3.48: Motion latency between the movement of the tracked tool and the reaching of the target position with the teleoperated robot. The values are expressed in milliseconds and are reported for different UR5 speeds.

3.3.3 Conclusions

The system developed proved to be an intuitive and flexible way to interact with a service robot with flexibility and exploiting different control levels. The developed

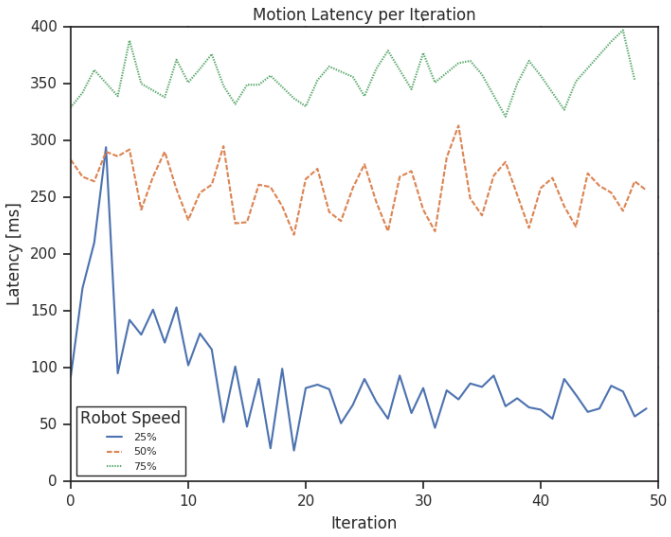


Figure 3.47: Motion latency between the movement of the tracked tool and the reaching of the target position with the teleoperated robot. The values are expressed in milliseconds and are reported for each iteration.

application showed that the proposed control system is easy to use for people with no robotics experience. The use of different autonomy levels allowed the human to control different tools with the robot in a flexible way and without the need to worry about the motion to grasp and switch them. The users that tested the system were able to precisely control a complex robot TCP position and rotation without the need of a teaching phase. The definition of constraints enabled the robot to avoid collisions with nearby object or cause any hazard. The mapping of the tracked tool orientation to the rotation of the last joint of the robot arm, allowed a flexible use of the tools, avoiding dangerous reconfiguration that could occur allowing general 3D rotations. The use of an intuitive GUI helped the users to understand how to interact with the robot and its behavior. In this way the interaction was more comfortable and the users had a better understanding of what the robot was doing. The GUI allowed also them to have a better precision in the control, representing a virtual and scaled version of the object to work on. In this way bigger movement were mapped to smaller ones in the robot workspace.

The evaluation of the system precision and motion latency conducted with an additional robot showed that the system does not introduce a big latency in order to reach the target pose. High speeds caused the robot to not reach the goal position with precision and this is because of the different components involved and their communication. For example, the reduction of latency of the tracking system could reduce this problem.

The overall setup could be improved using a marker-free tracking system in order to avoid the use of external markers to have a good tracking of the needed objects. The control of the TCP rotation could be enhanced in order to allow a safe 6 DOF control which allows the user to have a more flexible control.

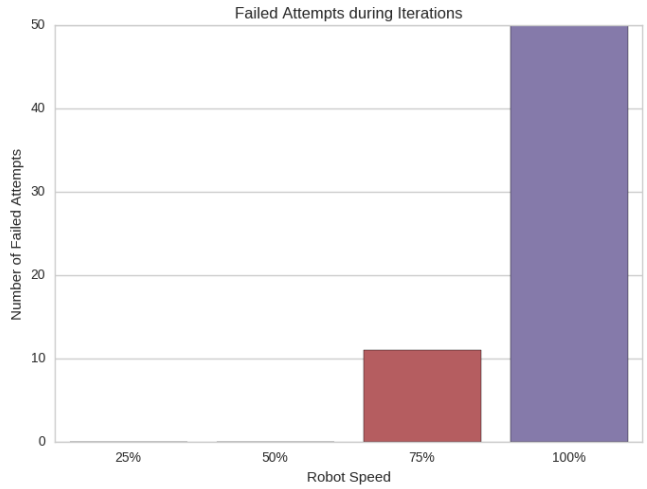


Figure 3.49: Failed attempts to reach the target position tracked from the UR5 motion. The results are reported for different UR5 speeds.

3.4 Summary

In HRI, intuitive interfaces to program the robot are of crucial importance, as well as simulation tools to design and validate the robotic application.

In this chapter, a system to allow the programming and simulation of complex robotics systems has been proposed. The use of building blocks which represent specific hardware components, enables the definition and simulation of complex robotic scenario without the need of complex programming. The use of different levels of simulation combining VR/AR interfaces and an immediate deployment on the real hardware allows to bridge the gap between simulation and real world, enabling a realistic evaluation of the process before transferring the defined programs on the real hardware.

Furthermore, gesture-based interfaces to enable the control of complex robots intuitively have been proposed. In particular, the use of visual feedback proved to ease the interaction, as well as the use of different levels of control which can be switched through the use of a GUI.

In the next chapter, the safety aspects related to the design of application in which human and robot have to share the same workspace are investigated, in particular focusing on a GPU-based external collision avoidance system and metrics on how to evaluate the benefits introduced by such methods in both safety and efficiency of the application.

4 Safety and Efficient Collision Avoidance

4.1 Safety Using External Controllers for 3D Collision Avoidance

This section introduce the motivation and proposed approach for achieving safety while using external control authorities. In particular, the safety effects introduced by using an external 3D collision avoidance system are evaluated, proposing metrics that can be generalized for similar systems. The GPU-based method used for collision detection enables real time collision prediction capturing the live environment with 3D cameras. This section investigates the safety aspects that needs to be taken into consideration using external control authorities, such as ROS, in safety critical applications, proposing a set of guidelines to integrate external controllers ensuring the safety of the application. Furthermore, the focus of this section is to provide realistic and reproducible test routines to evaluate the safety effects introduced by such systems. The tests have been performed on the 3D collision system considered and the collected measurements and results are presented. The results presented in this section are part of the work done in the research European project COVR.

4.1.1 Method

Safety is a critical aspect in HRC. In research we can find many applications that use external control authorities that are not intrinsically safe, making difficult to certify them. For example, monitoring systems which rely on cameras cannot be considered safe but can still provide some safety functions that could help to keep a safe distance between the robot and the human and make the user have a better feeling of safety while interacting with the robot, with a consequent better ergonomics in the interaction.

ROS is becoming more and more used in different robotic applications, particularly in the research field. Regrettably, this is not the case in industrial scenarios, and one of the main reasons is that it has not been developed to be robust and it does not provide safety-critical features. As it is known, the software for safety-critical application requires to fulfill certain standards and regulations to guarantee

an appropriate behavior. Nevertheless, ROS does not satisfied some of these requirements. For instance:

1. Misses real-time capabilities, which are critical for real-time systems and control loops.
2. Limited embedded support.
3. The framework is a collection of Open Source Software packages, which are tied together for quick software prototyping and do not comply with the safety standards.
4. There is a lack of any systematic safety evaluation, functional hazard assessment and evidence of any functional safety integrity level (SIL).

Therefore, it seems essential to start designing general safety guidelines to preserve safety-critical features.

This section proposes a set of guidelines when using ROS as an external control for safety-critical applications (Human-Robot Collaborative workspaces, for example). The external controller should always rely on the automatic protection mechanism of the robot and be considered as an extension of the robot-safety controller. This first guideline should help in a future design of a general validation of any application using external controllers, particularly in use cases implemented in workspaces designed for co-existence and collaboration between human operators and robots.

In this section a methodology to use external control authorities, such as ROS, in safety critical applications is proposed. In particular the focus has been posed on the use of a GPU-based 3D collision avoidance system, proposing metrics and test routines to evaluate the safety effects introduced by such components.

The use case in consideration is a screw assembly scenario, in which a human and a robot have to share the same workspace. The operator is responsible to place the workpieces on a table, while the robot is responsible to tighten the screws positioned by the human. Possible collisions might occur, since robot and human can work in parallel and the human might also need to inspect the workpieces for defects detected by the robot during the tightening task.

The GPU-Voxels library [48] has been used to develop the 3D collision avoidance system that allows the computation of possible collision in real-time. The live environment captured by depth cameras is mapped into high resolution voxel maps that are used for the collision detection. The robot collision model is also voxelized and used to compute the swept volume of robot planned trajectory, which represent the volume that the robot will occupy in the execution of a specific motion. By intersection of the voxel maps representing the live environment and the robot swept volume, it is possible to detect possible collision in real-time with short reaction time to make the robot able to stop and also replan its motion. In this way, the robot can detect and prevent collision before getting close or in contact

with the human, enabling a better motion planning in the workspace and a better interaction with the operator.

The implemented controller is ROS-based and also the collision avoidance system, which relies of 3D camera, cannot be safety certified. In order to make the system safe, a risk assessment of the application has been performed and the needed risks functions have been implemented at the hardware level, in particular on the screwdriver and robot controller.

A set of guidelines for integrating unsafe control authorities,such as ROS, in safe application is proposed:

- **Risk Assessment:** the first step is to do a risk assessment for the application, where all possible hazards from the application are identified. This assessment will also help to recognize which are the safety-critical components. A component is considered as "safety-critical" if the hardware is in direct contact with the human body or if it controls mechanical parts which could become a hazard. These components are also considered inherently unsafe.
- **Update safety-critical hardware into intrinsically safe:** the safety-critical hardware needs to be safety certified, or connected to safety-validated software, this guarantees a minimal inherent mechanical hazard. This might require an update or exchange of hardware in use by safety approved robots or tools. Overall, it is a good practice to select either several specialized safety systems (such as limit switches, valves and so on); or a single one, capable of reducing the required points of configuration to be evaluated. This will guarantee the system as intrinsically safe. For the considered screw assembly application,the hardware has been updated and all the safety-critical components connected through the safety certified robot controller. By using a screwdriver specifically targeted for HRC use, possible hazards (such as sharp edges or crushing) are reduced or prevented.
- **Validation of safety-critical components:** the safety-critical components will form the robot-safety controller, and they will become the application fail safe. For this purpose, the different functionalities considered relevant for the specific robotic application should be tested. Any possible hazard should be sufficiently mitigated, to guarantee that the application is intrinsically safe.
- **Identification of unsafe components**: a clear identification of the external control unsafe components (here ROS-based) from the validated safe modules is necessary. It is also necessary to understand their effects on the safe components. Figure 4.1 shows the separation between unsafe (signals coming from ROS or any external controller) and safe components for the considered application.
- **Interfacing unsafe components with safe components**: the safe components provide the safety functions needed to limit the inputs coming from the unsafe modules. All the unsafe components (in the considered use case,

the ROS-based) need to be interfaced with the robot and any other safe hardware passing through the robot-safety controller, or to a component directly controlled by it (via a safe signal), as shown in Figure 4.2. The signals coming from the ROS controller are not allowed to be connected directly to the robot without passing first through the robot-safety controller, as this will guarantee that in case of any failure, the application is still safe. As an example, for the considered application, safety relays have been used to separate the unsafe screwdriver controller from the screwdriver, as can be guaranteed that the relays will cut the signal (and/or the power) to the screwdriver if the tool safety system is triggered. This ensures the screwdriver safe operation. An overview for the screw assembly application is given in detail in Figure 4.1.

- **Single control authority per interface:** by enacting the previous point, clear interfaces are established, which can be monitored and tested for. It is a good practice to have a single control authority per interface on the ROS side. For example, there should only be one ROS-node sending joint angles to the hardware. If angles arise from multiple locations, they should be collected in a single node, which could then enforce the safety limits and failsafe check them before sending the command. This step will provide multiple benefits:
  - a single point can be used to verify the given commands, lowering the efforts for safeguards and safe values.
  - by catching illegal values, or problems, before sending them to the controller, unwanted states such as emergency stops due to speed violations can be prevented.
  - by handling faults inside the ROS system, the software can be configured to react to faults or failure states. If this is not done, a monitoring of the safe hardware is required to ensure safe re-entry behavior of applications.
- **Worst case scenario validation:** a necessary validation that in the worst-case scenario (for example, if the ROS controller fails), the robot-safety controller is still under control of the application. For instance, if the maximum force measured in a transient collision during the robot-safety controller validation is 180 N, it is not allowed to exceed this value during the external controller tests. This can be validated adding unwanted control inputs or generating random control values. In the considered use case, for example, the impact forces in a transient collision have been measured setting high speeds in the ROS controller. The tests showed that the maximum force was not exceeded.

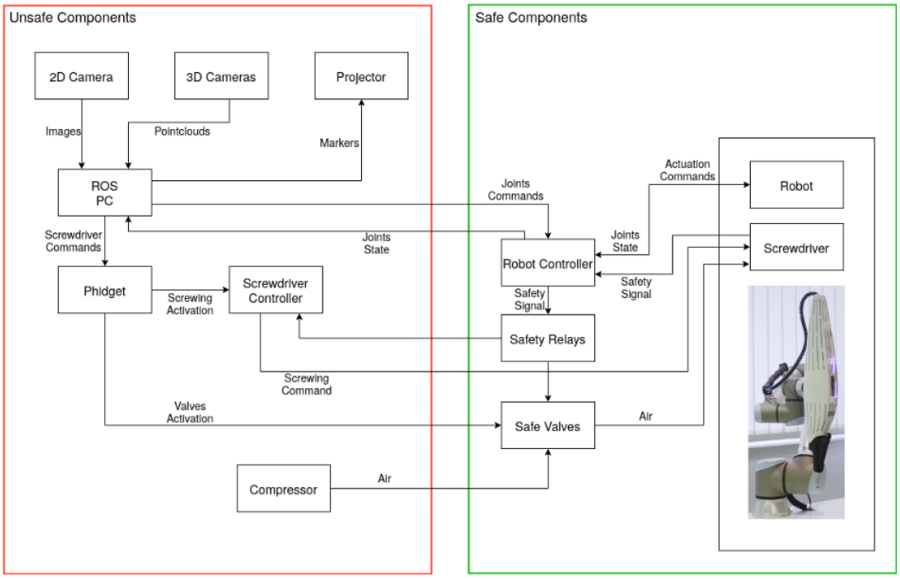


Figure 4.1: Separation of safe and unsafe components for the ROS-based screw assembly application. The safety-critical hardware is controlled by safety certified software for HRC, while the unsafe components are interfaced with the robot safety-critical components.

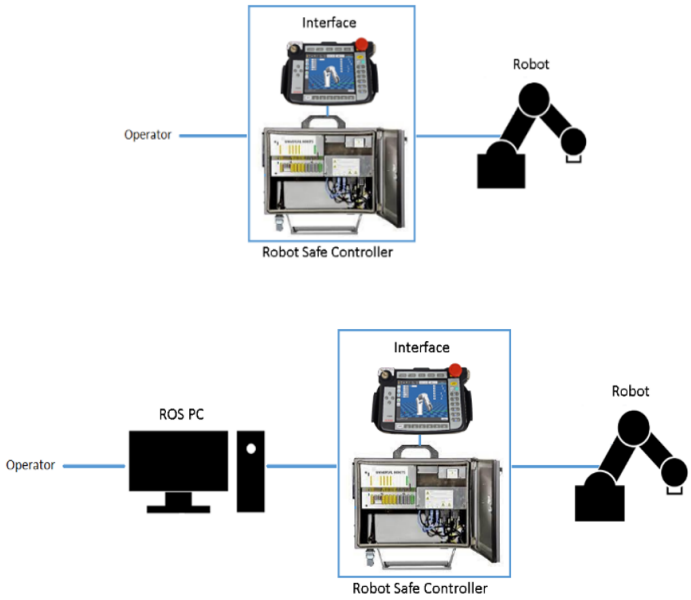


Figure 4.2: Top: safety-critical hardware robot setup without any external controller. Bottom: robot setup including ROS controller in the application, if the ROS controller fails, the robot is still controlled by the robot-safety controller.

An upgrade on the hardware has been done done after a thorough risk analysis of the application. The updated setup consisted on a Universal Robot UR10e and a Stöger Automation CSX automatic screwdriver, both certified for human-robot

collaboration. The hardware is connected through the robot controller, which behaves as the safe controller due to its ability to handle the safety signals from the screwdriver and disable the operation if necessary. Despite the hardware updates, safety trainings for the operator are necessary to reduce any possible hazard due to collision. The operator should not premeditatedly get into an unavoidable collision or crashing situation, and no-go areas in the proximity of the robot need to be avoided in the workspace (such as the head).

The general guidelines on the specifics of using an external control framework (as ROS) have been proposed, which would need to be considered in safety critical applications. In the following, the 3D collision avoidance component integrated in the screw assembly application has been considered to evaluate and find metrics to measure the introduced additional safety behaviors, which could alter the application intrinsic safety validation.

The collaborative screw assembly demonstrator consists of two main components:

1. A screw assembly application in a collaborative workspace, using safety certified hardware: a human-safe collaborative screwdriver and cobot. The robot task consists in approaching a work piece and tightening screws, previously placed in the correct position by a human operator.
2. A ROS controller, which provides collision prediction using a 3D collision avoidance component based on the GPU-Voxels libraty. The 3D collision avoidance is fed by RGBD cameras placed in each corner of the worktable (four cameras) and constantly monitoring the workspace and creating a real-time point cloud of the live environment, allowing to detect static and dynamic obstacles in order to prevent possible collisions.

The proposed metrics to evaluate the safety benefits introduces by the external 3D collision avoidance system are:

- False positives/negatives obstacle detection and avoided collision: in this way it is possible to evaluate how many collision are correctly detected and evaluate the missed detections that could lead to possible collisions with the obstacle.
- Minimum intrusion distance: this metric give information about the minimum distance from the robot in which an obstacle should appear in order to be correctly detected in time to make the manipulator stop and avoid a contact.
- Protective stop and impact forces in case of collision: this measurements are done to collect information about impact forces in case of a collision. Without a collision avoidance system the robot will drive into the obstacle with its normal operation speed. In this way it is possible to evaluate if, in case of an impact, the collision avoidance system is anyway able to slow down the robot leading to lower impact forces.

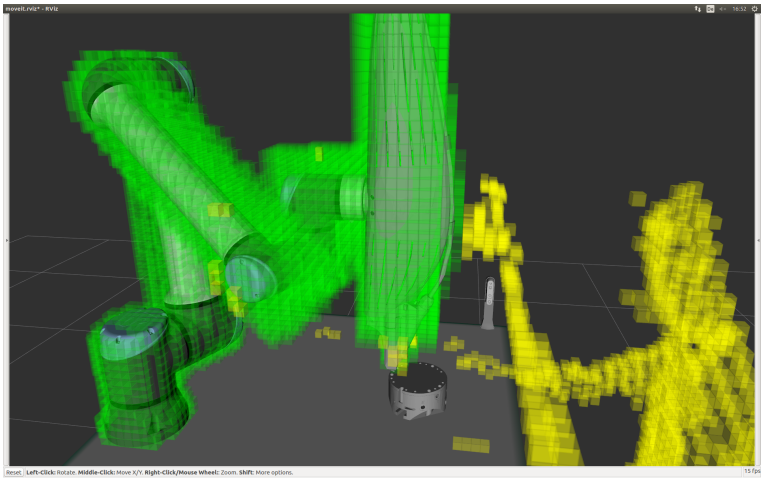


Figure 4.3: Voxels representation of the collision avoidance testing setup.

4.1.2 Experiments

In this section are described the experiments to evaluate the test routines and metrics presented previously . The screw assembly application has been used to perform the tests and collect the relevant data.

False Positives/Negatives Obstacle Detection

Using the considered collision avoidance system, the collision model of the robot influences how good possible contacts can be detected. This could also lead to false negative detections when the target is too close to the robot. The live environment voxels falling into the collision model of the robot are ignored in order to avoid the detection of self collisions and therefore are not considered in the collision check. To test the effect of the collision model on the collision detection, a scenario in which the robot is positioned 2 cm away from the target has been considered and evaluated. Using an exact collision model the collision is correctly detected without false negatives, as shown in Figure 4.3. However, the use of an exact model could introduce false positives: a considerable number of collisions have been detected while the robot had to perform the screw assembly task, which provoked the robot to stop for a short amount of time and making the component less reliable and efficient. The 10% inflated model which has been used for the rest of the presented tests is a trade-off which has been tuned and chosen after an evaluation of different inflation percentages (measuring the amount of false positives detected). The pitfall of having a bigger collision model is that the model covers also the objects in the close proximity of the robot (for example, the 2 cm away target obstacle is not detected as a possible collision).

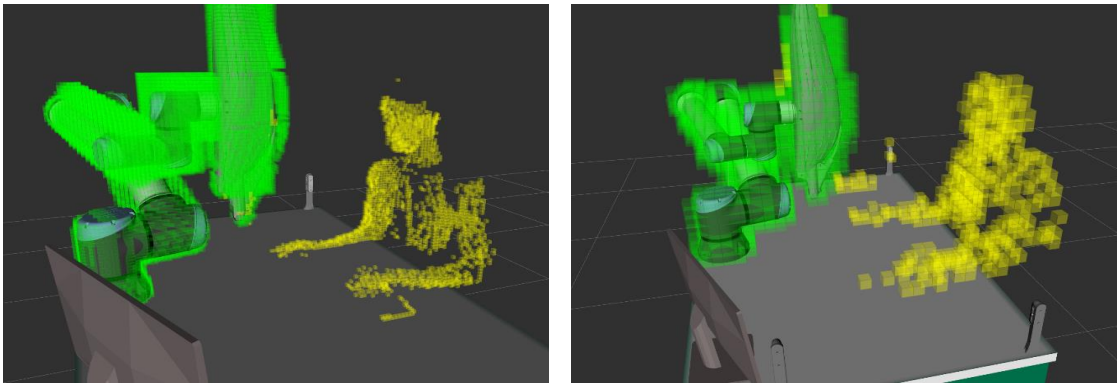
Reaction Time

Using collision detection systems, it is important to provide metrics and tests to evaluate how reactive the component is in detecting possible obstacles, in order to make the robot stop in time and avoid an impact. The evaluation of the different steps involved in the collision detection and their respective duration times, gives a way to evaluate where the major sources of delay are, providing a way to identify how it is possible to improve these computations if necessary. However, the results of these tests are hardly dependent on the specific hardware configuration. For the analyzed use case and tests the following hardware has been used:

- GPU: NVIDIA TITAN Xp.
- CPU: Intel Core i7-6850K @ 3.60GHz.
- Cameras: Intel RealSense D435 which provide environment point-clouds at 30 Hz.

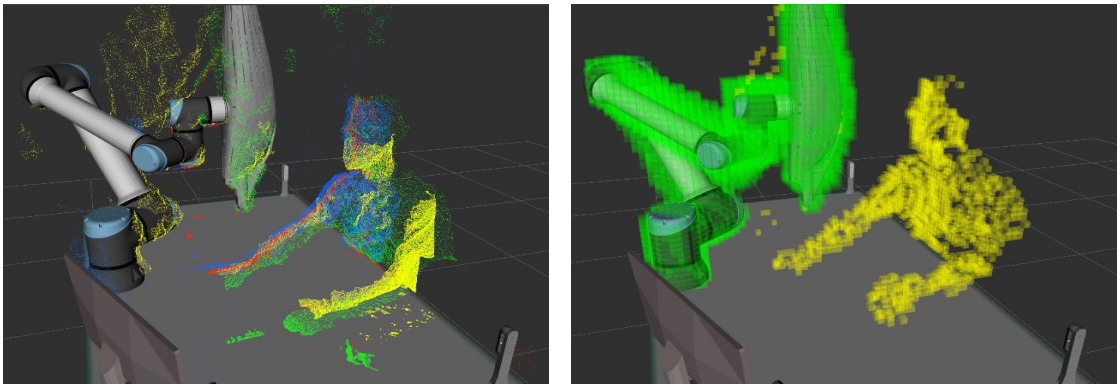
Voxel Resolution The voxel resolution determines how detailed is the voxel representation of the robot collision model and the live environment captured by the cameras. This could also affect the performances of the collision detection and the voxels computations. In Figure 4.4 are represented the robot collision model and live environment voxel maps using respectively 1 cm and 4 cm voxel size. In the previous sections, a voxel size equal to 2 cm has been considered. For the analyzed use case, with an increase of the voxel resolution to 1 cm, the update cycle of the live environment voxel map (used for the collision detection) decreased from $\sim$15 Hz to $\sim$10 Hz. However, this increase also implies that longer computation times are needed for the computation of the collision check. The quantified results are discussed in the next section, but the tests showed that by decreasing the resolution of the voxel maps from voxel size 2 cm, there were not relevant improvements in the reaction times and live environment update loop.

Point-Cloud Processing The cameras used in the evaluation tests acquire point-clouds at $\sim$30 Hz. However, the synchronization of the point-clouds from the four cameras and the merging of this information into a combined live environment voxel map using GPU-Voxels, lower the live environment voxel map update to a rate of $\sim$15 Hz. These two steps can be observed in Figure 4.5.



(a) Voxel size 1cm. (b) Voxel size 4cm.

Figure 4.4: Graphical representation of the voxel resolution. The different voxel size can affect the collision detection performances. (a): 1cm voxels. (b): 4cm voxels.



(a) Cameras point-clouds. (b) Merged voxel map.

Figure 4.5: Graphical representation of the four 3D cameras point-clouds (a) and how they are synchronized and merged into a single live environment voxel-map (b).

The time necessary for the point-clouds processing and the update rate of the live environment voxel-map have been measured (for 67 events). The mean values (and standard deviation) are provided in Table 4.1.

The time needed by GPU-Voxels to project the live environment voxel map into the robot swept volume, which allows the robot to detect real-time collisions, was 0.0007 seconds with a standard deviation of 0.0015 seconds. This is the mean value for an evaluation with 1000 samples. The high standard deviation is due to few outliers, in which the collision detection took longer. This is probably because of the computation load on the pc running the collision checks.

Test description	Time	
	Mean [s]	σ[s]
Merging the four cameras point-clouds	0.0639	0.0066
Merging the four cameras information into a live environment voxel-map	0.0040	0.0038

Table 4.1: Measurements on the necessary point-cloud processing time, and the update rate of the voxel-map for 67 events, using four cameras.

Achieving Protective Stop An important metric to evaluate is the time needed by the robot to achieve a protective stop when a possible impact is detected. For this, it is necessary to measure the time needed by the system to detect a possible collision, and time to stop the robot as soon as the obstacle is captured by the cameras. These values, however, are hardly dependent on the hardware. Here are provided only the results for the considered hardware setup (previously described). Table 4.2 reports the results on three different tests which include: time needed by the robot to stop as soon as the possible collision is detected, time needed by the robot to stop as soon as the camera call-back collects the point-cloud containing the obstacle and the time needed by GPU-Voxels to detect the collision from the moment in which the camera call-back collects the point-cloud containing the obstacle. In this way it is possible to evaluate how long it takes for the robot to stop when an obstacle appears, having clear measurements on the robot controller and hardware related reaction times, as well as the ones depending on the specific collision detection system. Each test was repeated and measured 50 times. The results showed that the system is really fast in detecting possible collision from the 3D camera point-clouds, making the considered robot able to stop in around half a second as soon as the cameras capture the obstacle. It is possible to notice that most of the stopping reaction time is needed by the robot controller to stop the robot as soon as the stopping signal is triggered. Therefore, the overall stopping time could be improved by simply using a robot providing a controller and hardware able to guarantee a smaller reaction time to a stop signal. Based on these observed reaction times, it is possible to infer that an improvement in the computation time in the GPU-Voxels detection would also affect the minimal intrusion distance. As these results are extremely dependent on the hardware setup, they could be improved by repeating these evaluations deploying faster GPUs.

Test description	Time	
	Mean [s]	σ[s]
Time to stop since the collision is detected by GPU-Voxels	0.4436	0.0197
Time to stop since the camera call-back received the point-cloud containing the obstacle	0.5380	0.0292
Time to detect the collision in GPU-Voxels since the camera call-back received the point-cloud containing the obstacle	0.0943	0.0225

Table 4.2: Measurements of the reaction times related to the different process steps involved in the detection of a possible collision. From the moment in which an obstacle is captured by the cameras, through the analysis of the point-clouds and GPU Voxels, until the robot stops. Each test was repeated and measured 50 times.

Minimum Intrusion Distance

Following the suggested metrics previously described, it has been tested the scenario in which the robot has an obstacle suddenly appearing while it is executing a trajectory to a target, observing the reaction behavior: either the robot observed the obstacle and successfully discarded the trajectory, or it engaged the trajectory execution and ended up in an impact. Each test was repeated and measured 20 times (the obstacle was fixed at the same distance, for replicability), and the results are shown in Table 4.3.

From the results, it is possible to notice that if the obstacle was too close to the robot, the inflated collision model used for the collision detection produced false negatives, thus the robot did not detect the collision. Using the defined setup, this was observed only when the distance from the obstacle to the robot was less than 5 cm.

Protective Stop and Impact Forces in Case of Collision

Keeping the setup described previously, the scenario in which the robot is starting the execution of a collision free trajectory with a maximum tool speed equal to 0.2 m/s has been considered. The precise point in which the robot engaged a protective stop, due to an obstacle suddenly appearing at a fixed distance from the tool, has been measured. In case of the robot not having enough room and time for stopping, the impact force (transient force) has been measured using a force sensor attached to the obstacle. In order to simulate an obstacles appearing at a fixed distance, it has been added an interface to enable/disable the collision

Robot Tool distance [m]	Observations
0.1	20/20 events the robot did not start execution of the trajectory
0.05	16/20 events the robot did not start the execution of the trajectory. The other attempts the robot managed anyway to stop before hitting the obstacle
0.02	20/20 events the robot hit the obstacle

Table 4.3: Test to evaluate the minimum intrusion distance to make the robot able to detect a collision.

avoidance based on a signal. A component that monitored the distance of the robot from the obstacle has been implemented, allowing the possibility to enable the collision avoidance based on a specific threshold distance from the obstacle. Each test was repeated 20 times and the mean values (and respective deviations) of the distance and impact force are reported in Table 4.4.

Threshold [m]	Impact avoided	Impact force		Dist. tool-object at stop	
		Mean [s]	σ[s]	Mean [m]	σ[m]
0.2	20/20	NA	NA	0.1352	0.0079
0.1	20/20	NA	NA	0.0365	0.0097
0.075	18/20	165.7	21.21	0.0211	0.0040
0.06	0/20	133.9	38.73	NA	NA
0.05	0/20	180.5	0.59	NA	NA

Table 4.4: Measurements of the distance needed by the robot to perform a protective stop with a fixed threshold intrusion distance. NA in the impact force means that the robot achieved the protective stop. If the robot went through a collision, the impact force (transient) was also measured.

The results show that under the conditions defined in the considered setup and with the collision detection activated, the robot was always able to stop before being in contact with the obstacle when this latter appeared at a distance of at least 10 cm from the manipulator. With a smaller intrusion distance threshold, collisions with the obstacle could be observed due to a limited amount of space to detect the object and make the robot stop in time. Figure 4.6 shows the measured

forces in case of an impact without the use of the collision avoidance component, this has been reported as comparison for the values measured using the collision avoidance system. In Figure 4.7 and Figure 4.8 are reported an example of the force components measured on the obstacle for different intrusion distances using the collision avoidance component. It has also been observed that when the robot did not manage to detect the obstacle, the safe proof behavior has been kept. For example, when the obstacle appeared 5 cm away from the manipulator, the system was not able to detect it, but there were no changes in the measured impact compared to the case without collision avoidance. However, in summary, it has been observed that with a relatively small intrusion distance, the impact forces measured were smaller than the case without any collision avoidance, as the robot managed to reduce its speed before hitting the obstacle.

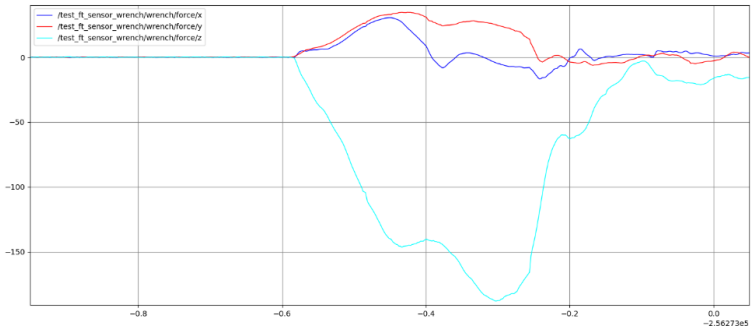


Figure 4.6: Measured force components during a transient collision without the collision avoidance component. The absolute measured force is 189N.

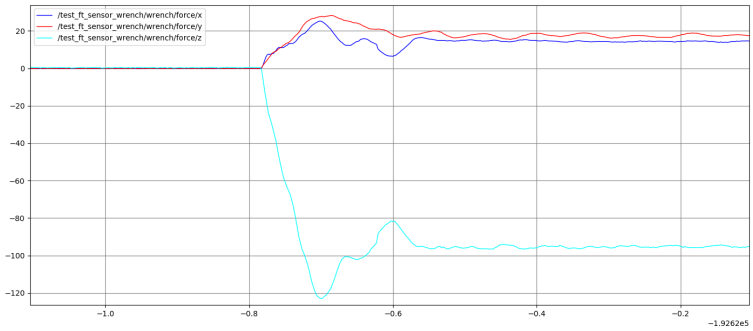


Figure 4.7: Force components measured with an intrusion distance equal to 0.06 m. The protective stop was not achieved, but the transient force was smaller than the case without any collision detection. Impact force module F_R=128.63N.

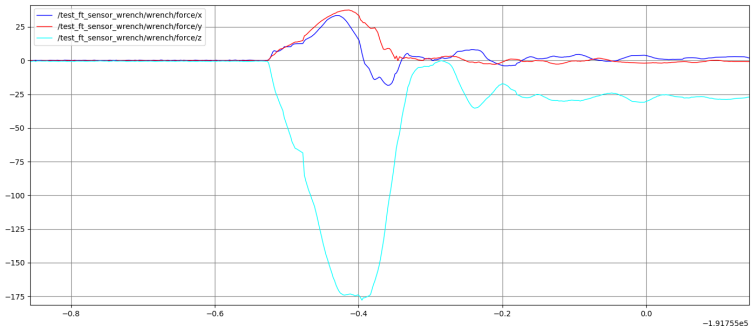


Figure 4.8: Force components measured with an intrusion distance equal to 0.05 m. The protective stop was not achieved, but the transient force was smaller than the case without any collision detection. Impact force module F_R=180.71 N.

4.1.3 Conclusions

In conclusion, the main approach to enable a safety-critical application with an external control authority is the clear separation of component operation into safe and unsafe. By ensuring the safe operation of the safe part (regardless of the inputs), the user is free in the use and design of the external control authority. The main benefit of this approach is that it is usually possible to accomplish: either the robot hardware itself offers the required safety functionality, or it can be added in most cases. This also allows the retroactive extension of existing processes with a reasonable effort. Nevertheless, the downside of this approach is that the maximum performance of the application is restricted to the "baseline" values that enable a safe operation. For example, the top speed of the robot needs to be limited in order to prevent excessive force, even though it might be possible to reach higher speeds if the safe detection of collisions by using ROS could be ensured. Furthermore, metrics and test routines to evaluate the safety benefits introduced by the use of an external collision avoidance system have been proposed and evaluated. For the considered GPU-based 3D collision avoidance, it has been proved that the robot was able to avoid impacts for obstacles with at least a 0.1 m intrusion distance. In case of impacts due to smaller intrusion distances, the detection produced also lower impact forces due to the stop signal issued to the robot which caused the manipulator to slow down before the collision.

4.2 Dynamic Real-Time Task Replanning

An external control authority, such as the presented collision avoidance system, could be used to introduce additional benefits to the application not only related to safety. The real-time collision avoidance system presented in the previous section, can be used to improve the efficiency in the task completion, enabling the robot

to predict collisions and select goals in a more efficient way. The common used approach of safety systems, such as safety skins, is to detect a collision when the robot is in proximity of an obstacle, stopping its motion and waiting for clearance in order to resume its task. The presented system, based on detecting collisions of the robot swept volume with the live environment, allows to predict a possible collision before getting in proximity of an obstacle, avoiding the execution of part of the planned trajectory which will lead to a stop and trying to keep a larger distance from the user. In addition, the fast computations of possible impacts, enable the detection of other collisions free targets in real-time, reducing idle stops and making the robot able to switch efficiently between goals. This improves also the ergonomics of the collaboration with the human operator, encouraging the execution of tasks in collision free areas and away from the worker.

Anyway, such efficiency benefits introduced by a collision avoidance system are not easy to evaluate and general metrics and test routines are missing. In this section, methods to evaluate the efficiency benefits introduced by an external collision avoidance system are evaluated, in particular considering the proposed dynamic task scheduling which is based on live environment information collected by depth cameras. The efficiency of the screw assembly application is analyzed with and without the camera-based 3D collision avoidance. The improvements in using such systems could justify the benefit of adding unsafe components (external controllers) to an already validated safety/critical application.

4.2.1 Method

The GPU-based 3D collision avoidance system has been used to enable a dynamic scheduling of the robot tasks, allowing the robot to predict impacts and replan in real-time a new collision free path towards another target when the current trajectory is blocked by an obstacle (for example, by an operator). This component impacts the efficiency of the application, as the dynamic replanner allows the robot to perform tasks even if some of the targets are not reachable at in a specific moment. For example, an operator could be checking the status of one of the work-pieces while the robot is screwing another work-piece. The considered setup is the one presented in the previous section and consists in four RGBD cameras positioned in each corner of the work-table, to avoid possible sources of occlusion (for example the situation in which the robot is blocking a region of the table while it is reaching for an object) and ease the identification of an obstacle in the trajectory of the robot. The dynamic task scheduling is built on a ROS-based task manager, which keeps a list of the robot tasks and subdivides them as done or not, as well as information about their reachability based on the detected obstacles. As soon as the current path is blocked, the robot can ask for other possible targets to work on, computing the swept volume to reach them and executing the planned trajectory if a collision free path to the new target is found.

In order to perform efficiency evaluations on such collision avoidance systems, a

set of metrics and test routines are proposed. The proposed metrics are based on measurable quantities, which will confirm efficiency gains on the system when compared with the application without the 3D collision avoidance component. In particular, to validate the gains the following metrics are proposed:

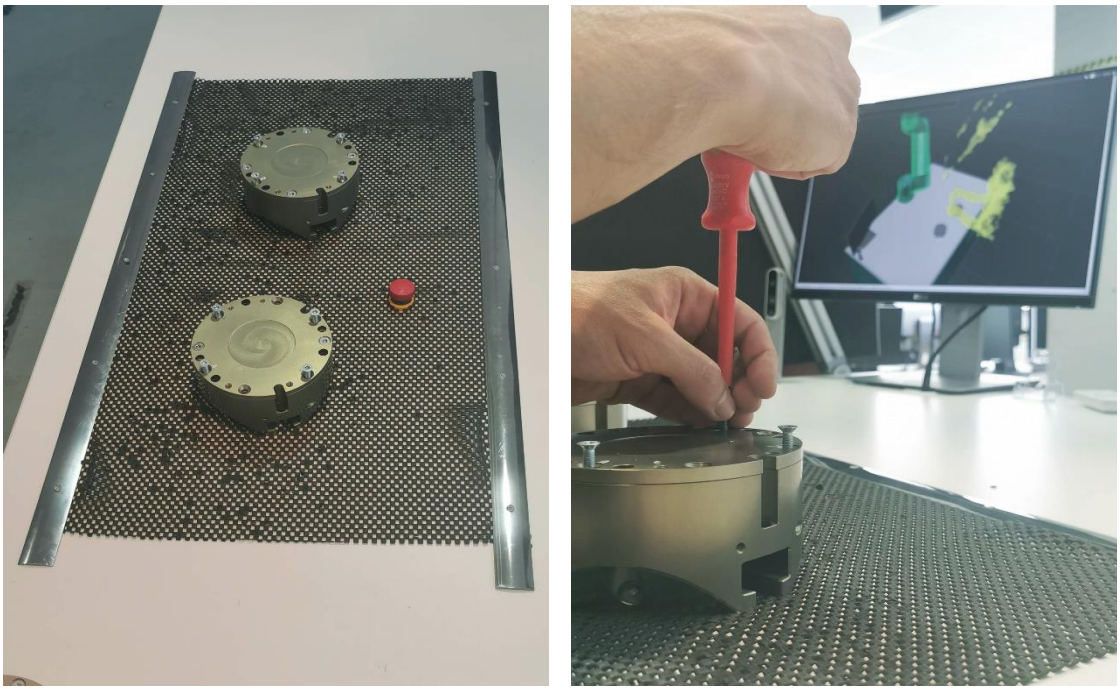
- Task completion time: time to successfully process all the work-pieces (with and without the collision avoidance component).
- Robot idle time: time in which the robot is not moving or is in an idle state (with and without the collision avoidance component).
- Replanning time: time needed by the robot to find another collision free target as soon as it stops due to an obstruction of the current path.

The task completion time gives a clear idea about the time needed by the robot to complete the task, which is directly related to the efficiency. The robot idle time provides a measurement of the amount of the time in which the robot is not performing any action, which of course needs to be limited to the minimum otherwise the execution of the task is highly inefficient. The measurement of the replanning time needed by the robot to find a new collision free target and path as soon as the current one is blocked, provides an evaluation on how fast is the dynamic scheduling system and how fast the robot is able to find possible collisions in order to improve the efficiency provided by the new task selection. These measurements can vary depending on the specific hardware and workspace setup.

4.2.2 Experiments

Similar to the safety effects evaluation scenario analyzed in the previous sections, in the reported experiments the 3D cameras are constantly scanning the workspace. When a possible collision is detected, it is reported to the ROS controller, which slows down the robot speed until a protective stop is achieved. Meanwhile, a new available collision-free target is searched in the available tasks list. If none is found, the robot goes back to the home position, other way it moves to a new target executing a new planned obstacle-free trajectory. For this evaluation, the workspace setup has been defined with two work-pieces available in the same fixed position on the table. The work-pieces described in the rest of this section have four screws each, which are the targets of the manipulator. The robot task is to go on top of each screw head and wait for a couple of seconds as soon as the target is reached. In the following evaluations the action of tightening the screws has not been considered, as this is not part of the 3D collision avoidance component and the evaluation focuses on the efficiency of the application having the manipulator able to switch between targets. Therefore, the robot needs to reach eight screw heads to complete its task, and only after reaching all of them the task can be deemed as completed. The fixed position of the work-pieces guarantees the replicability of the experiment for each test.

In order to include a fixed obstacle into the tests, the camera point-clouds, captured when a human is working on a work-piece to inspect the screws on top of it, have been recorded. In this way the recorded live environment information can be replicated and synchronized at the start of each test run. In Figure 4.9, it is possible to see the workpieces and screws setup used for the test, as well as the human obstacle recorded to include the blocking of the path to one of the work-pieces. Figure 4.10b shows in yellow the voxel map of the recorded live environment, in which it is possible to see the human covering the work-piece on the right.

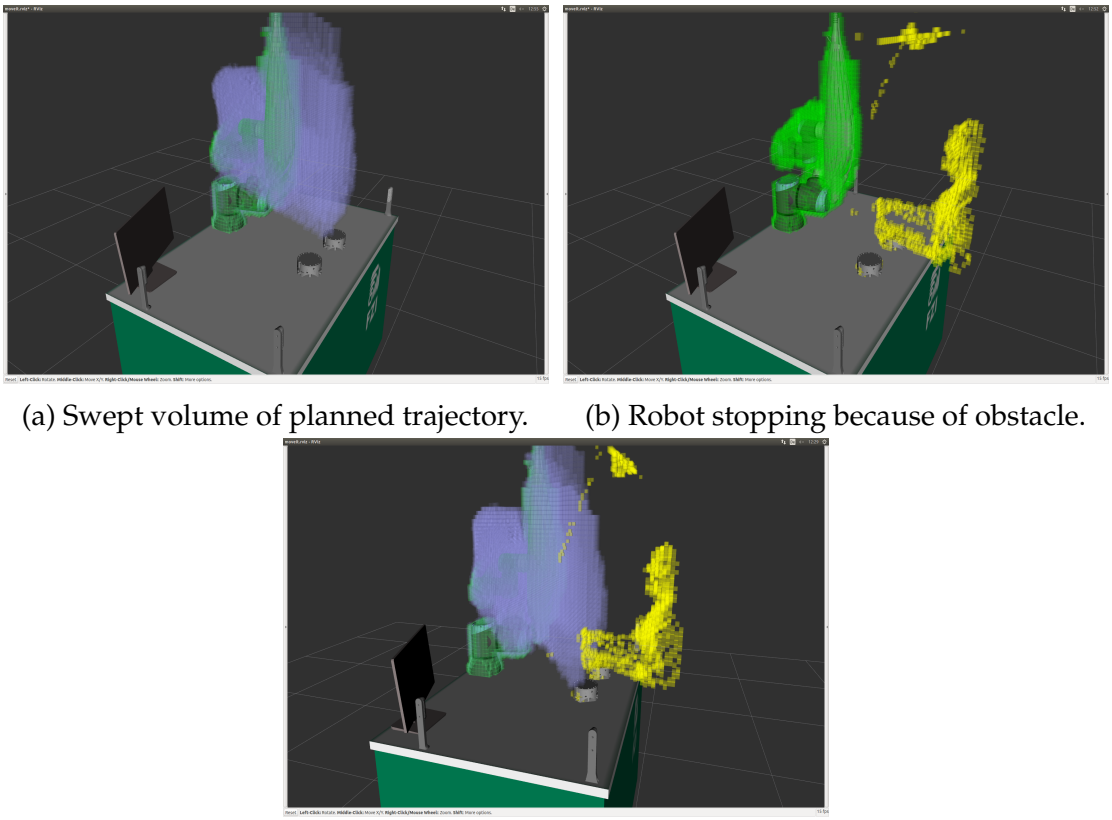


(a) Target workpieces. (b) Human obstacle.

Figure 4.9: Setup used for the efficiency tests. In (a) are represented the two work-pieces with the eight targets (screw heads). In (b) is depicted the human obstacle blocking the path to one of the work-pieces. The camera point-clouds have been recorded to ensure repeatable tests under the same workspace conditions.

In this evaluation, the ability and reactivity of the controller to redirect the robot to the new target are measured, as well as the metrics which reflects how this can improve the efficiency of the application. The task completion time is measured from the moment in which the robot is enabled to start its task until all the work-pieces are completed. In other words, when the robot has successfully reached each one of the eight screws. For the robot idle time measurement, a component which observes the changes in the joints state of the robot has been implemented. This module measures the time in which the robot joints configuration does not change, providing the overall amount of time in which the robot does not move, including the time interval in which the manipulator is waiting for clearance. The

replanning time is measured as the time between the stop of the robot and the notification that a collision free path to another target is found. This provides a measurement on how fast the system is able to replan towards new collision free work-pieces, when the path to the current one is obstructed. A short replanning time ensures that the robot does not need to stop for a long time and enables a more efficient task execution. The various tests have been performed keeping the obstacle in the workspace for different amount of times and reporting the efficiency metrics with and without the use of the real-time replanning component.



(a) Swept volume of planned trajectory.

(b) Robot stopping because of obstacle.

(c) Swept volume of new planned trajectory.

Figure 4.10: Voxel representation of swept volumes and live environment during the efficiency tests. In (a) is represented the swept volume of the robot planned trajectory. In (b) the robot stops because of an obstacle positioned above the planned target. In (c) is represented the swept volume of the new planned robot trajectory towards a collision free target.

Replanning Time

The time needed by the replanning system to find another collision free target as soon as the robot stopped because of an obstacle blocking the current path has been measured under differnet conditions. The test was repeated multiple

times (68 samples) and performed also using different voxel sizes. The results are given in Table 4.5. As mentioned in the previous section, it is possible to observe that a decrease of the voxels resolution, starting from 2cm, does not add any improvement in the replanning time. Therefore, for the rest of the experiments, the 2cm voxels size has been used. From the measurements, it is possible to see that decreasing the resolution the system does not introduce any further improvements on the computation time. In the reported tests, the mean is slightly increasing, but the results are in the same range and depending on the specific samples. This is due to the high parallelization of the voxels computations. Anyway, a deterioration in performances is clearly visible using a voxel size of 1 cm. Summarizing, the results showed that with a voxel resolution of 2cm the robot is able to replan its motion and switch to new collision free target in less then half a second.

Voxel size [cm]	Time[s]	σ[s]
1	1.0695	0.4444
2	0.4029	0.1328
4	0.4740	0.1023
8	0.4304	0.1180

Table 4.5: Mean value (from 68 samples) for the replanning time for different voxel sizes and the respective deviations.

Completion Time

To evaluate the completion time, the time needed by the robot to reach all the target has been measured. To validate the gains in completion time, the evaluations have been done using two different setups. The first setup consisted of two work-pieces, with a total of 8 fixed targets, and one of the work-pieces blocked by an obstacle. The tests have been performed varying the time in which the work-piece was blocked and the time needed by the robot to complete the task with and without the replanning system has been measured. The second setup consisted instead of three work-pieces, with a total of 12 fixed targets. The same behavior was repeated: one of the work-pieces was blocked and the total time for the robot to complete the task has been measured. In the scenario without any collision detection, the robot would just crash against the obstacle and the tasks would fail. For this reason it has only been considered the scenario in which a collision detection component is active (for example, lasers in certain parts of the table which could trigger the robot to stop), focusing on the real-time replanning capability of the system. The same collision detection setup has been kept for both cases: one with the replanning component active and the other case having the robot going into a stop and wait until the path to the target has been cleared.

The results (reported in Table 4.6) show that the online real-time replanning improved the task completion time and reduced the idle time of the robot while

performing the task. As it is possible to see from the replanning time measurements, as soon as the robot stops because of the obstacle blocking the current path, the manipulator is able to find and replan towards a new collision free target in less than one second. In other words, the robot can work on another work-piece instead of waiting for clearance in an idle position. As expected, if the obstacle appears only for a short time, the efficiency improvements are not noticeable. Nevertheless, the benefits area clearly seen when the obstacle remains blocking the target for a longer time.

Targets	Obstacle t [s]	Task completion t [s]		Idle t [s]		Replan t [s]
		With	Without	With	Without	
8	NA	89.257		59.312		NA
	2	89.396	92.294	59.919	62.118	0.722
	5	88.148	91.906	58.471	61.757	0.908
	10	86.655	94.485	56.925	64.098	0.499
	20	85.618	104.852	56.024	74.633	0.549
	30	87.006	115.989	57.186	85.995	0.681
	60	108.132	147.521	67.581	117.062	0.709

Table 4.6: Completion time for the setup consisting of two work-pieces (eight screws). The table shows the time in which the obstacle was blocking the path, the total time to complete the task, the idle time of the robot and the time it took to the replanning system to find a new trajectory and send the command to the robot to move towards the new target.

In fact, the efficiency increases relatively to the number of work-pieces available to the robot. This can be observed in the test with three work-pieces, as observed in Figure 4.11. By including another work-piece, the efficiency improves proportionally to the time in which the obstacle is blocking the path, as it can be observed in the results reported in Table 4.7, since the robot is able to reach more targets instead of waiting for clearance when the obstacle is blocking the current path.

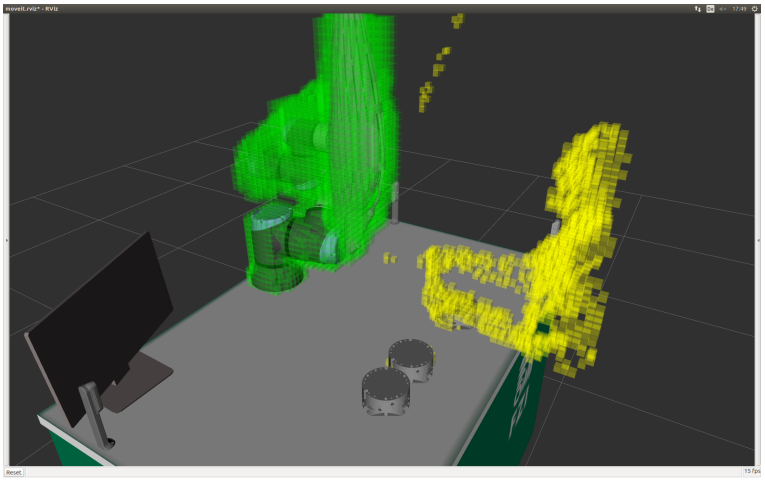


Figure 4.11: Graphical representation of the GPU-Voxels live environment and collision maps for a setup with three work-pieces. As described before, an obstacle is blocking one of the targets (in yellow).

Targets	Obstacle t [s]	Task completion t [s]		Idle t [s]		Replan t [s]
		With	Without	With	Without	
12	NA	100.364		62.334		NA
	2	107.118	114.520	69.041	75.969	0.688
	5	114.101	121.181	76.233	82.636	0.474
	10	100.673	112.903	62.530	74.400	0.477
	20	107.610	131.896	69.920	93.189	0.579
	30	110.226	164.203	72.273	125.689	0.886
	60	127.745	192.760	89.916	154.341	0.988

Table 4.7: Completion time for the setup consisting of three work-pieces (twelve screws). The table shows the time in which the obstacle was blocking the path, the total time to complete the task, the idle time of the robot and the time it took to the replanning system to find a new trajectory and send the command to the robot to move towards the new target.

4.2.3 Conclusions

In this section, metrics and test routines to evaluate the efficiency benefits introduced by external control authorities have been proposed. In particular, it has been analyzed the case of a collision avoidance component, which affects the efficiency gains proving the possibility to dynamically switch the robot target in real-time based on the live environment information captured by depth cameras.

In contrast to standard safety approaches, such as safety skins that allow the robot to stop when a contact occur, the considered approach has the advantage of making the robot able to predict possible collisions, avoiding the execution of parts of motion that will lead to an impact with an obstacle. In this way it is possible to

select beforehand other possible targets in order to enforce the execution of tasks that can be performed far away from the human operator or other obstacles in general.

The efficiency tests proposed, focused on the real-time online replanning capabilities of the considered screw assembly application and have been performed with and without the component enabled. The tests included two and three workpieces, each one of them with four screws placed on top. For each of these targets, the robot task was to move towards the screw head and wait few seconds once the target was reached. The results showed that the dynamic target replanning feature provided the demonstrator with efficiency gains in both task completion time and overall idle time.

4.3 Summary

Safety is a crucial aspect in HRI and collision avoidance systems are really important to make the robot avoid impacts as well as improving the overall efficiency of the robot in the task execution.

In this chapter, the safety aspects that need to be taken into consideration using external control authorities have been introduced and guidelines to enable a safety certification of such systems have been proposed.

Furthermore, test routines and metrics to evaluate the safety benefits introduced by external collision avoidance systems are proposed. In particular these have been evaluated on a GPU-based collision avoidance system, which enables the prediction of possible collision in real-time.

In addition, the safety benefits that can be introduced by such real-time replanning systems, through the use of dynamic task scheduling, have been investigated. Test routines and metrics have been proposed to create a benchmark to evaluate the efficiency benefits introduced by such components. In the evaluated scenario, the GPU-based collision prediction has been used to enable a real-time replanning of the robot task, which is based on the workspace state and the detection of possible collisions. In this way the robot could avoid idle stops and replan its motion towards collision free targets.

Allowing the robot to change dynamically its motion and goal, it is of crucial importance to make the humans operating in the same workspace able to understand the robot intentions and eventual changes in its plan. In the next chapter are proposed methods to represent the robot intentions and motion information to the user, in particular using virtual representations of the manipulator swept volume and combining visual and acoustic information.

5 Robot Intents Communication

5.1 Visual and Acoustic Feedback

In this section, the motivation and proposed approach to communicate the robot intent using visual and acoustic feedback is presented. The use of acoustic signals to notify the user when a possible collision is predicted and showing a virtual representation of the robot target and planned motion could improve the ergonomics of the collaboration and the trust in the robot, in order to make the worker more comfortable while working close to the manipulator.

Parts of the results presented in this section have been published in [5].

5.1.1 Motivation

Nowadays robots are able to work safely close to humans. They are light-weight, intrinsically safe and capable of avoiding obstacles as well as able to understand and predict the human motion. In this collaborative scenario, the communication between humans and robots is a fundamental aspect to achieve good efficiency and ergonomics in the task execution. Many research studies have investigated how to make the robot understand and predict the human behavior, allowing the robot to replan its motion and trajectories. In this section, the focus is posed on the communication of the robot's intentions to the human to make its goals and planned trajectories easily understandable. Visual and acoustic information has been implemented and combined to give the human an intuitive feedback to immediately understand the robot's plan. This allows a better interaction and makes the human feel more comfortable, without any feeling of anxiety related to the unpredictability of the robot motion.´ The use of robot feedback allows the user to work closely to a replan capable robot without feeling anxiety. Through this additional communication he can understand when his motion is in the way of the robot and make intelligent decisions considering the new plan of the manipulator.

Experiments have been conducted in a collaborative assembly scenario. The intuitive feedback system developed has been tested by robot experts and non-experts. The final results have been evaluated through a questionnaire in which the users could express the differences and improvements in the HRI experience with and without the acoustic and visual feedback.

5.1.2 Method

The goal of this section is to present a method to improve the interaction between robot and human, giving the latter the opportunity to understand robot intentions using visual and acoustic feedback. The system proposed is based on the collaborative screw assembly application presented in the previous chapter and reported in Figure 5.1, in which human and robot share the same workspace and this latter is able to predict collision, avoid the human worker and replan its task.

Figure 5.1: Shared workspace used for evaluation of the visual and acoustic feedback system.

In order to notify the user about the detection of collisions, an acoustic feedback that alerts the worker when the robot has predicted a possible impact has been added. In this way the worker can understand when the robot is changing its planned motion and can see this change through a visual feedback on screen. Here the new goal can be visualized alongside a model of the workspace to let the human understand the new goal of the robot. Another information added is the swept volume generated by the planned robot trajectory, that is helpful to understand the area in which the robot is planning to be in the near future. The area of the workspace where the current robot's goal is located is also communicated through the use of synthesized speech. This allows the robot to communicate to which side of the worktable it is heading for, for example the right or left side. This information is less detailed compared to the actual goal position visualized on screen, but it is helpful when the user cannot look away from a workpiece while he is working on it.

In Figure 5.2 is depicted the overall architecture of the feedback system and its communication with the other components responsible to trigger the robot motion and make it react to possible collisions.

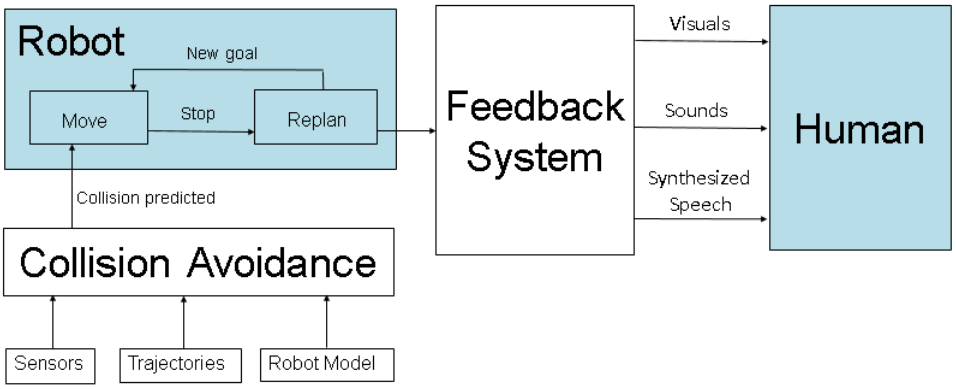
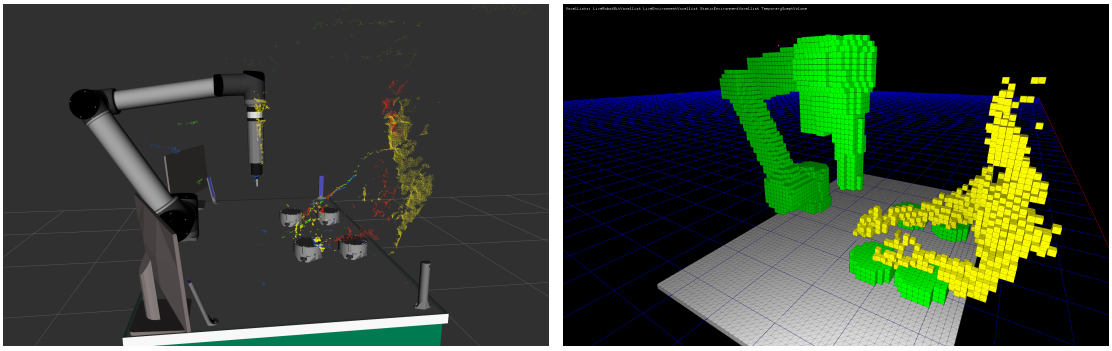


Figure 5.2: System architecture of the robot feedback communication system integrated into the collaborative screw assembly application.

In order to implement and test the visual and acoustic feedback information from a robot, it has been used the shared workspace environment monitored by 3D cameras presented in the previous chapter. The point clouds related to the live environment are used to predict possible robot collisions using the GPU-Voxels library [47], [48], [53]. The use of GPU-Voxels allows the fast computation of the robot swept volume related to the execution of a trajectory and enable the check for possible collision with the live environment. Thanks to the high parallelization of the calculations through the use of the GPU, it is possible to compute the volume that the robot will occupy in the future executing a particular trajectory and check any possible collision with the information related to the workspace occupancy state captured by depth cameras. Figure 5.3a shows the point cloud data representing the live environment captured within the workspace of the robot. Figure 5.3b shows the representation of the workspace using the GPU-Voxels library created voxel maps. The green voxels represent the robot collision model, while the yellow voxels represent the processed point-cloud data related to the live environment, that are used to check possible future collisions with the robot trajectories. Using this approach the robot is capable to dynamically change its motion, changing its goal when an obstacle is in the way.

Although this system enables the robot to avoid collisions, the worker can still have a feeling of anxiety when the robot is moving close to him because he has no information about the next or current planned motion of the robot. In addition, without knowledge about the robot motion, the human worker cannot plan in an efficient way on which part it is best to work without inducing the robot to continuously replan its motion. In fact, in order to work efficiently in a team, each partner should know at any time what the other partner is doing.

The feedback system implemented, enables the robot to provide alert sounds to draw the attention of the human worker towards the manipulator motion to indicate that the robot has detected his presence and that it will start to replan its motion towards another goal. Another notification sound is used to inform



(a) Virtual model. (b) GPU-Voxels representation.

Figure 5.3: In (a) is depicted the model of the shared workspace with cameras point-clouds and (b) shows GPU-Voxels representation of the shared workspace, including robot collision model and live environment voxel maps.

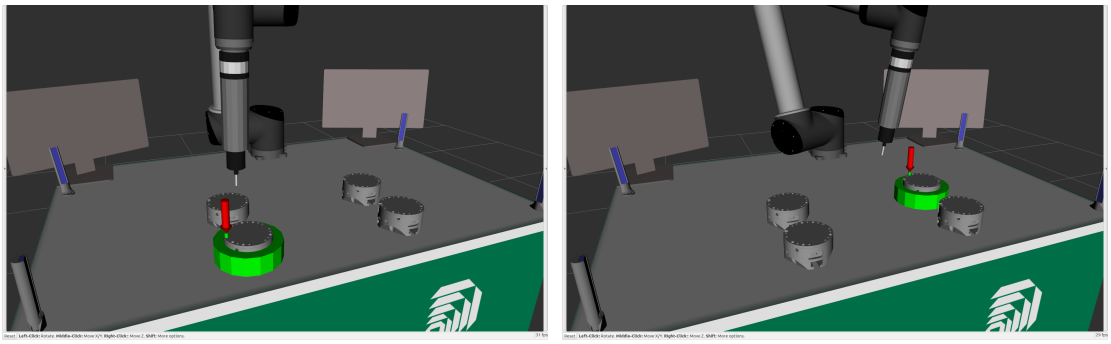
about the execution of a new trajectory by the robot and synthesized speech is used to communicate to the human where the actual goal of the manipulator is. In this use case with four large workspace sections in which the robot can work, the implemented speech sentences are:

- *New goal: on top left part of table*
- *New goal: on top right part of table*
- *New goal: on bottom left part of table*
- *New goal: on bottom right part of table*
- *Cannot find collision free trajectories: please restore clearance*
- *Work done: waiting for new parts*

These phrases allow the human user to know that he can work without any interference in the remaining areas of the workspace. The speech is also useful to understand when the robot motion stops because its current tasks have been executed and it waits for the worker's input or because every planned trajectory is obstructed by the human.

Acoustic feedback has additional impact if combined with visual information. To achieve that, after the robot has drawn the attention of the worker, the feedback system displays on screens the exact position of the goal that the robot is targeting by using a model of the shared workspace. The visual feedback has been implemented through the use of RViz along with a detailed model of the workspace and robot. In this visualization it is easy to represent the actual goal of the robot, that is updated every time after a replan towards a new and different workpiece. Figure 5.4 reports the representation of the workspace and the robot together with the goal markers. In the reported images, the green marker represents the area

around the current robot goal. The red arrow marker points to the exact position of the workpiece that the robot is to planning to reach.



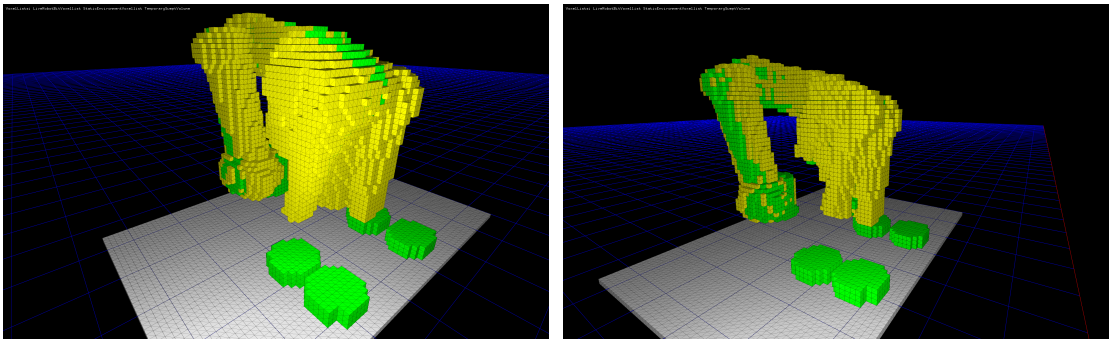
(a) Robot goal marker. (b) Change of robot plan.

Figure 5.4: The target of the robot is highlighted with markers using the virtual representation of the workspace (a). As soon as the robot replan its motion due to a detected collision, the updated plan and goal is shown (b).

Another way to intuitively and precisely understand the next planned robot motions is proposed and implemented by the use of the visualization of the robot swept volume voxel representation, which is also used for the collision detection and replanning functionalities. The GPU-Voxels library computes the swept volume of each trajectory that the robot plans to execute. The communication of this plan to the human worker is an important additional information because it allows him to have an exact spatial understanding of the volume that the robot will occupy in the execution of the next task. Using this information, he can detect the areas of the workspace in which the robot will not be working when executing its current task. In Figure 5.5 the swept volume representation of the planned robot trajectories during execution is visualized. From the figures it is possible to see that the robot, while executing its current task, will occupy only a certain area of the workspace. This allows the user to detect the areas and workpieces on which he can work without interfering with the actual robot motion.

Every time a new trajectory is replanned, the acoustic feedback draws the attention of the human worker to check the swept volume of the robot trajectory in order to make him aware of the updated robot plan and motion. In this way the user has a clear understanding when the robot is changing its motion and he can immediately see the details that allow him to understand the robot's intentions and planned trajectories, increasing the ergonomics in the teamwork. This can increase efficiency, since enabling the human to know the robot movement and goals, he can decide in a more intelligent way on which part to work next, avoiding conflicts with the manipulator and minimizing unnecessary replanning phases.

The use of a fast replanning system coupled with the described intuitive feedback system allows to improve the coexistence of robots and human workers in a shared workspace.



(a) Swept volume of the planned robot trajectory.

(b) Swept volume of the remaining planned robot trajectory.

Figure 5.5: The robot voxel representation of the robot swept volume used for the collision detection (a) provides a feedback on the volume that the robot will occupy executing a trajectory. The already executed sections are removed and the visualization is updated accordingly.

5.1.3 Experiments

The proposed system should improve the comfort and acceptance of the human worker while working close to the robot. In this way he can work without worrying about the robot until he receives notification of its new goal. After that he can acquire knowledge of the new robot's plan, without worrying about unpredictable motions. The transparent robot behavior should also lead to an improvement of the efficiency in the task completion. The human, receiving information about the motion of the robot, can select in a more intelligent way the parts on which he plans to work next. This allows the robot to avoid the continuous replanning of its motion without reaching any goal.

To evaluate the developed feedback system, the collaborative screw assembly application has been considered. The robot task consists in tightening the screws positioned on four parts placed on the worktable and detecting missing and defected ones. The workspace is shared with the human worker, that in the meantime has to replace the worked parts with new ones and check the defected screws detected by the robot. The robot is obviously capable of avoiding collisions and replanning its motion in order to not interfere with any detected obstacles in the workspace.

To evaluate the system, it has been set up a user study with 5 people, including robot experts and non-experts. They had to perform some screws placement and inspection tasks in the shared workspace. Each person had to perform the task first without the acoustic and visual feedback and then with the developed communication component. At the end of each experiment a questionnaire related to the experience in the interaction with the robot has been filled. The main points of interest focused on the user's feeling when working closely to the robot, the understanding of the robot's goals and motions, the understanding of the volume

that the robot will occupy in the near future and the evaluation of the effectiveness of the visual and acoustic feedback. The results of the interaction experiments regarding robot and users are presented.

The two charts reported in Figure 5.6 and Figure 5.7 are scaled from "Strongly Disagree" to "Strongly Agree". The following statement were analyzed:

(A) I was worried that the robot moved towards me

(B) I had a feeling of anxiety working close to the robot

(C) I foresaw the robot's goals

(D) I perceived when the robot had to replan to avoid a collision

(E) I felt the need of additional information related to the robot motion

The first chart in Figure 5.6 reports the results related to the human-robot interaction experiment conducted without the use of acoustic and visual feedback. From the chart it is possible to see that some users had a feeling of anxiety working close to the robot and they were worried that the robot moved towards them. The majority of them had also the problem of not understanding the robot's goals and its motion replanning phases. Most of them felt a need for additional information related to the robot motion.

The chart in Figure 5.7 reports the answers to the same statements but related to the human-robot interaction experiment conducted with the use of the acoustic and visual feedback. The results show that using the transparent robot behavior system, all the users had no feeling of anxiety and they were not worried about the robot motion. All of them have also understood clearly when the robot had to replan and its planned goals.

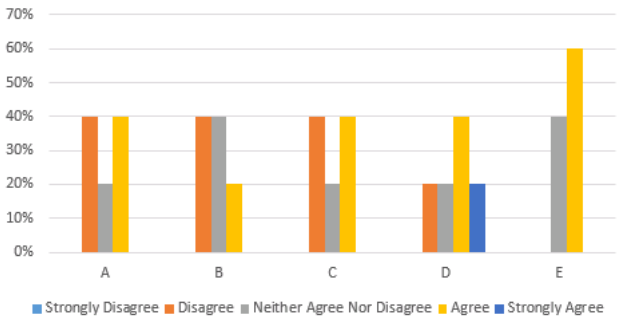


Figure 5.6: Results of the questionnaire for the experiments conducted without the use of visual and acoustic feedback.

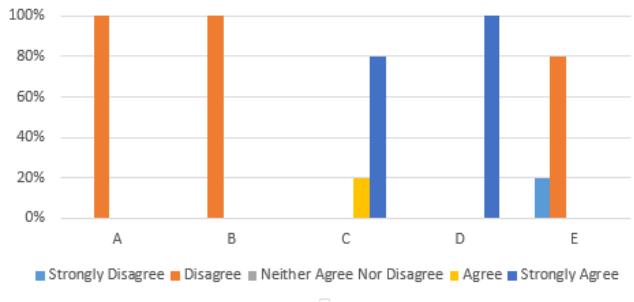


Figure 5.7: Results of the questionnaire for the experiments conducted with the use of the visual and acoustic feedback.

Comparing the two charts in Figure 5.6 and Figure 5.7, it is possible to notice that the acoustic and visual feedback helped the users to understand when the robot replanned its motion to avoid a collision and changed its goal. Furthermore, the results show that the users have experienced a better feeling of comfort in the close interaction with the robot.

The chart in Figure 5.8 reports the answers of the users regarding the following statements, which aim at evaluating the understanding of their impressions related to the addition of the acoustic and visual feedback provided by the robot:

(A) The visual information about the robot's intentions was helpful

(B) The acoustic feedback helped to understand when the robot was replanning

(C) I had the feeling that more robot feedback (for example augmented reality) would help

(D) I have clearly understood the robot's goals

(E) The visual and acoustic feedback helped me to feel more comfortable in the interaction with the robot

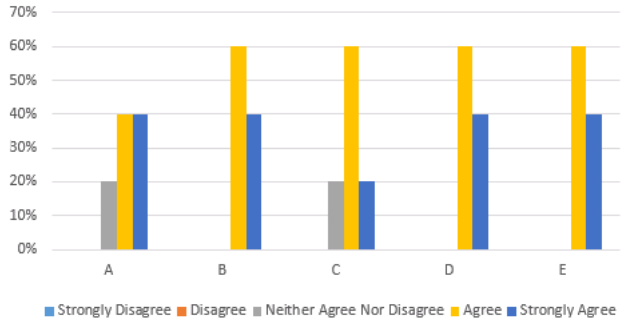


Figure 5.8: Results of the questionnaire related to the evaluation of the usefulness of the transparent robot behavior system.

This last chart shows that the users have found that the combined use of the acoustic and visual feedback was helpful and useful in order to understand the robot's motions and goals. All of them have also agreed that the addition of this information helped them to achieve a better feeling of comfort in the human-robot interaction. Most of them have also agreed that adding more feedback channels, by for example using augmented reality, could improve the interaction even more. During the experiments it has been found that the experience background of the users didn't influence the interaction much. As expected, the people with less robotics experience were more worried about the unpredictability of robot motion in the experiments conducted without any feedback system.

Analyzing the results of the experiments it is possible to say that the users felt the lack of information related to the robot motion that has motivated the proposed method. The presented transparent robot behavior system based on visual and acoustic feedback helped them to better understand the robot motions, making them feel more comfortable in the interaction.

5.1.4 Conclusions

The system presented in this section showed that the use of visual and acoustic feedback to provide information related to the robot execution plan helped the human users while they were working close to robot. The use of the proposed system showed that the users had a better understanding of the robot motion. Furthermore they were able to understand when the robot had to replan its motion in order to avoid them. This helped to avoid the discomfort caused by not understanding the intentions of the robot, even if programmed with intelligent behaviors and collision avoidance features.

Using this system the human workers feel more comfortable and have a better understanding of the robot motion and goals. This allows the users to make better decisions in order to accomplish an improved team work with robots in a shared workspace.

Further improvements in the robot-human communication system could be done using for example augmented and virtual reality, as presented in the next sections. In this way the user has the possibility to directly see the visual information on the work-table. This allows also to represent the robot's swept volumes in a more intuitive and effective way.

5.2 Projector-Based AR Overlay

In this section is presented the motivation and followed approach to communicate the robot intent using a projector-based AR overlay. The representation of the information related to the robot intent and motion, as well as the workspace state,

directly on the worktable can improve the intuitiveness of information provided by the robot and improve the interaction.

Parts of the results presented in this section have been published in [4].

5.2.1 Motivation

With no information about the robot targets and intentions, the user feels uncomfortable even with a safe robot. In close human-robot collaboration, it is very important to make the user able to understand the robot intentions in a quick and intuitive way.

As highlighted in the previous section, in HRI it is really important that the user has awareness and a clear understanding of the robot intentions. Even if the robot is able to dynamically change its motion in order to avoid collisions with the worker, this latter can still have a feeling of anxiety and discomfort. Without any information about the planned robot motion, he is also not able to decide in an intelligent way how to avoid repeated disruption of the robot's current plan. Because of this, the system could be forced to replan repeatedly, inhibiting the progress of the robot's tasks. The feedback system analyzed in the previous section showed that the representation of the current robot's plan helps the user to achieve a better comfort in the interaction. This was done through the use of acoustic and visual feedback, provided to the human by audio speakers and monitors. However, this solution has some drawbacks and it is still not ideal. The use of screens to display the robot's intentions causes the human worker to diverge the attention from his task to these external devices. In addition, the visual information was shown by using a virtual representation of the environment, requiring the user to link the objected displayed with the real world. This is at the expense of the intuitiveness of the system. Indeed, in close collaboration it is important that the user is able to immediately understand the robot intentions without being distracted from his task. Due to the possibility of representing virtual information in the real world, augmented reality systems are becoming popular in many applications and also in robotics. Using a video projector to display internal information directly on the worktable, the user is able to visualize and understand immediately the state of the task and the robot's intentions, without the need to look at external devices such as monitors. The use of AR can help the human worker to quickly understand if the robot has to change its motion, making him aware of its current goal. This is useful in order to reduce the risk of blocking the new robot plan. It can also help the user to detect the status of the parts by highlighting the ones that need human intervention.

In this section it is proposed and described a feedback communication system which makes use of projector-based augmented reality to cast directly into the workspace useful robot information. The robot intuitively shows its planned motion and task state. The AR module interacts with a vision system in order to display the changes in the workspace in a dynamic way. The representation of

information about possible collisions and changes of plan allows the human to have a more comfortable and efficient interaction with the robot. The system is evaluated in different setups.

5.2.2 Method

In this section, the collaborative screw assembly scenario descried in the previous section is considered as a HRC application to apply the proposed robot feedback method. The human and the robot have to share the same worktable, working close to each other. The use of the GPU-Voxels library enables the robot to predict collisions and replan its goal in real time, based on live sensor data from the environment.

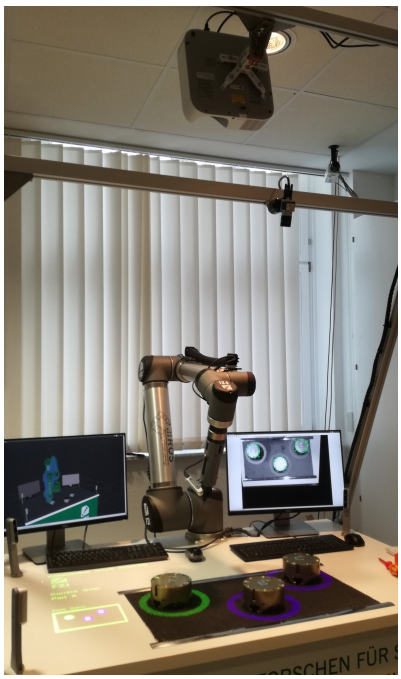


Figure 5.9: The hardware setup used for the projector-based AR feedback system. A projector mounted on top of the worktable is used to dynamically cast information about the parts and robot planned motion.

The aim of proposed system is to enable a better understanding of the robot's behavior, making the user able to understand its planned trajectory and goal. An important information to show is the collision detection information. Sometimes changes of direction in the robot motion can happen, but the human worker could interpret them wrongly as a reaction to his movement. Without this information, the human could try to move in order to step away from the robot motion, making himself instead moving into the robot trajectory and causing the robot stop uselessly. Providing the operators with the possibility to see and perceive potential collisions, it possible to make them understand why the robot has stopped and

they could change their motion in order to not interfere with the robot's planned path.

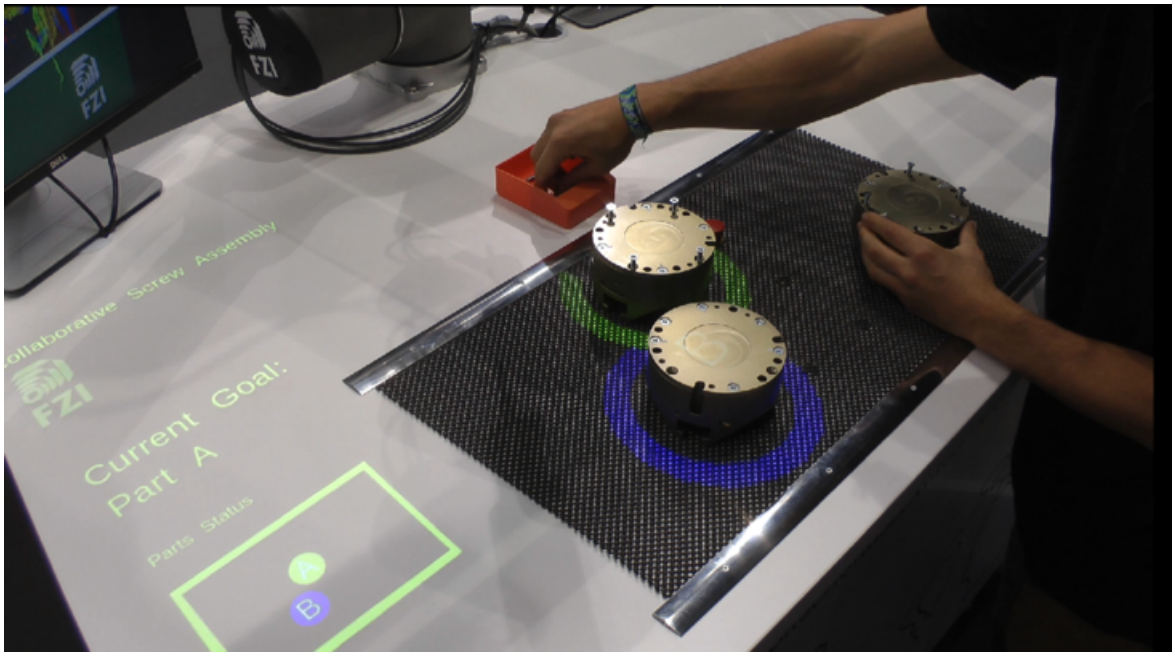


Figure 5.10: Augmented information related to the robot behavior and detected parts visualized with light projection.

In order to display the additional information in the workspace, a projector has been mounted on top of the shared workspace worktable. The overall setup used is reported in Figure 5.9. The information that is shown includes the status of the parts and the current target of the robot, as shown in Figure 5.10. The user is able to see immediately this information, because it is displayed at the position occupied by the actual objects in the real world. In this way the worker can easily understand the robot's intentions and the available parts on the worktable without the need to divert his attention to external devices.

In the scenario considered in this section, the workpieces position and rotation is detected automatically by a camera system positioned on top of the worktable, which can be seen in 5.9. In Figure 5.11 the system architecture diagram is reported.

The parts detected by the camera system are highlighted with colored circle markers around them. They can be placed arbitrarily within the specified area and the projected markers are immediately updated accordingly to their detected positions. In this way the user can see the workpieces that the robot has recognized in the workspace and understand immediately if some of them are not correctly detected or if their position is not updated after a manual change. This feedback is important to make the human aware of the actual status of the vision system. In this way he can detect instantly if the robot has issues in the localization of the parts placed into the shared workspace. Once a part is available to be worked, a

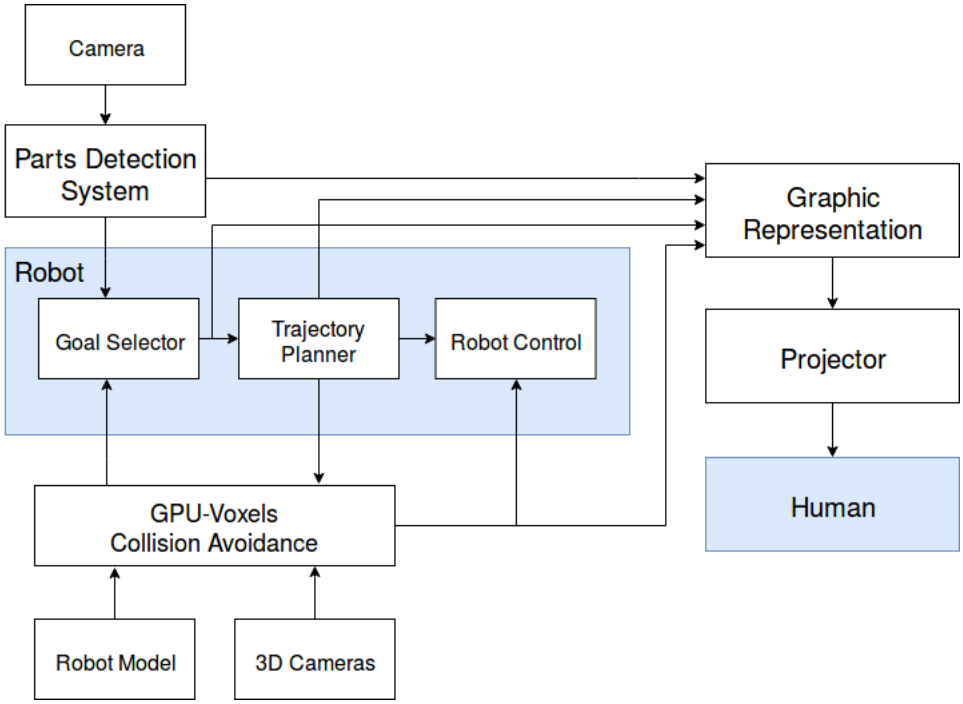


Figure 5.11: Diagram of the projector-based feedback system architecture. The communication between all the components used is reported.

circular purple marker is displayed around it. The current status of them is also represented in one small area on the side of the table. In this way the user can see the needed information even if the workspace is completely occupied by the robot and in case of possible projector occlusions. Text information is also displayed to make the human understand the robot actual goal and how this is represented. Figure 5.12 shows this information, reporting the case of three detected workpieces on the worktable.

Furthermore, the parts are identified with an identification letter that is assigned dynamically and projected on their center. The robot is also able to detect defects in the parts, which need then to be inspected by the human operator. To visualize this request of intervention, the ones involved are then highlighted with a blinking marker. The current target of the robot is represented with a green circle around the related part. The target position of the robot tool is also marked in green as shown in Figure 5.13.

Figure 5.12: Text information on the side of the table allows the human to understand the status of the workpieces and the robot goal even if there are obstructions in the robot workspace.

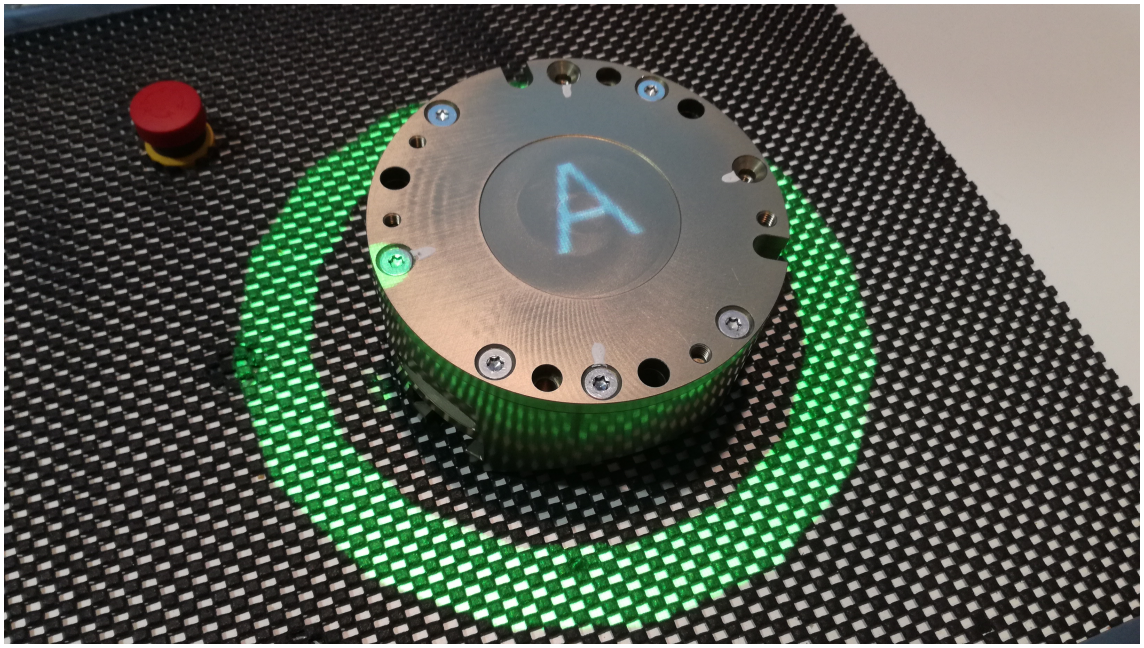


Figure 5.13: The current goal of the robot is highlighted with a green marker around the part. The specific target position of the robot tool is also pointed with a green circle.

Additional information has been added to enable the human to detect easily if the robot stops because of a detected possible collision. In fact, when the human or

another obstacle obstruct the planned robot motion, the AR system casts markers to represent this situation. The previous goal marker is then colored with red and additional text on the side explains the current state. Figure 5.14 shows the situation in which the robot stops in order to avoid a possible collision with the human operator. In this way, the user can have a clear idea when the robot is changing its motion and he can intuitively see the information that allows him to understand the robot intentions and movements, with a consequential improvement of the interaction ergonomics. This leads also to an improvement in efficiency. When the human workers know the robot motion and goal, they can decide which part to work on in a more intelligent way, avoiding a useless continuous replanning of the robot's trajectory.

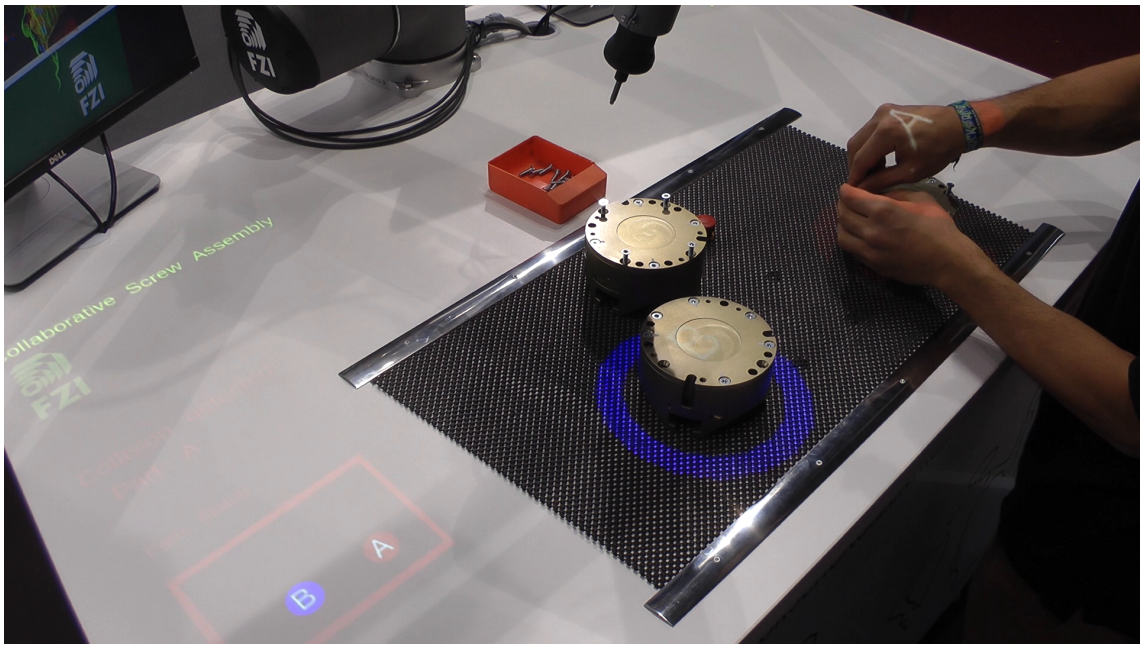


Figure 5.14: A red marker cast around the current obstructed robot's goal and text information on the side of the table inform the user about the prediction of a possible collision with the highlighted part.

For example, Figure 5.15 shows a situation in which the user can work on two workpieces while the robot works on a third one. Once the human starts to work on one of them, he can easily check that this is not selected by the robot. Even if the robot moves suddenly towards his direction, the use of the AR information helps him to understand that the robot's goal it is still located on the same workpieces that the robot was processing. The figure shows this situation in which the robot moves towards the human. The user is able to easily see the AR information, which tells him that the robot is still working on the same part.

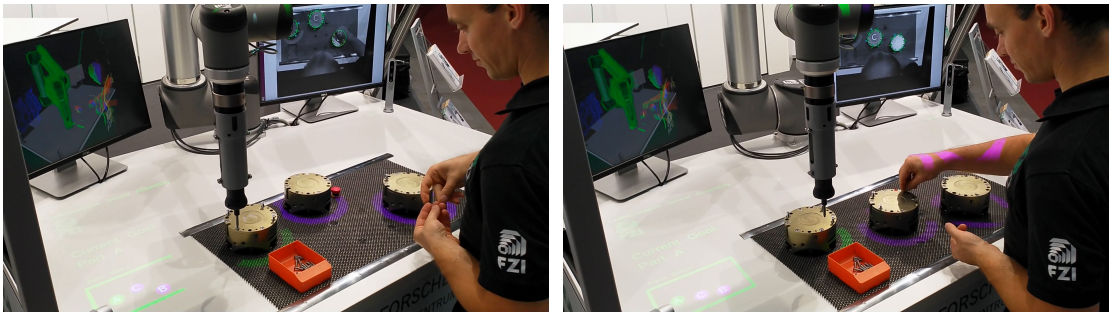
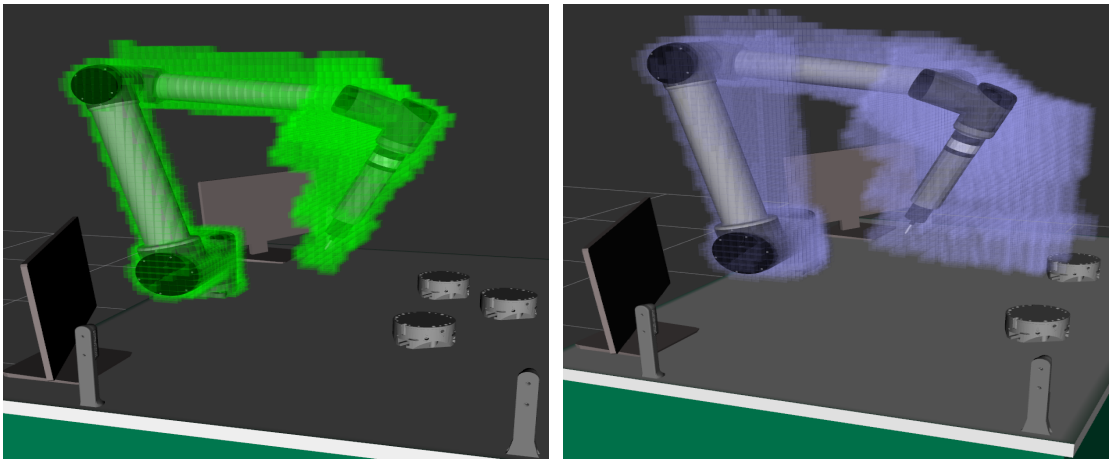


Figure 5.15: The user inspects the workspace and clearly understand the current target of the robot (a). The robot moves towards the user, but he is able to see that the target of the robot is still on the same workpiece (b). In this way he can continue to work without worrying about the robot motion.

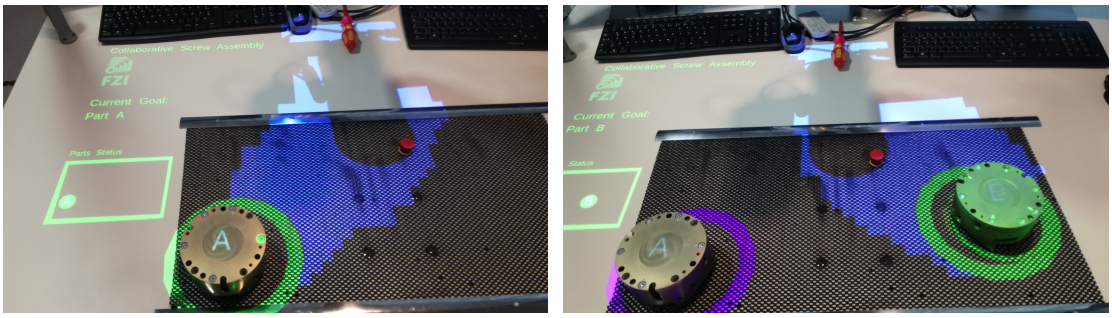
Another important information which can enable the user to understand the robot's plan is the swept volume of the current planned trajectory. This is the volume that the entire robot will occupy during the execution of the motion towards the current target and it is the one used by the collision checking system. The visualization of this information makes it clear where the human could cause the robot to stop because of a possible collision. This data is provided by the GPU-Voxels library, which is used to compute the swept volume for each new planned trajectory. This is computed by adding the voxel representation of the robot collision model for each way-point of a trajectory. Figure 5.16 shows in green the voxel representation of the robot collision model used for the computation of the swept volume of the robot's trajectories. The blue voxels represent the volume that the robot will occupy in the execution of the current planned path. During the execution of the current trajectory, the rendered swept volume of the remaining way-points is checked against the live environment camera data in order to detect possible collisions.

However, this volume information is based on 3D voxels which cannot be directly represented with the projector-based system proposed in this section. For this reason, a 2D representation of this volume has been implemented. The 2D projection of the voxels is rendered onto the table. This allows the user to understand which area of the worktable will be collision free in the next future. This is a useful information if combined with the goal markers. The human can understand the target of the robot and have a clear information about the manipulator planned motion to reach its goal. In Figure 5.17 it is possible to see in blue the projection of the swept volumes for two different robot targets. Looking directly on the worktable the users can understand in which area of the worktable they can move freely without obstructing the robot's motion.



(a) Robot collision model. (b) Trajectory swept volume.

Figure 5.16: In (a) the discretized robot collision model is represented in green. This is used to render the trajectories swept volume (b) which is adopted for the collision checks against the live environment. The voxel representation is overlapped with the 3D mesh of the robot.



(a) Projection of the trajectory swept volume. (b) Representation of updated trajectory swept volume.

Figure 5.17: Projection of the swept volume of the planned robot trajectory (a). This information enables the user to understand ib which area of the worktable the robot is planning to move and the collision free zones. The robot changes its target and the swept volume visualization is updated accordingly (b). In this way the user is aware of the new updated collision free areas.

Another information that has been added to the AR system is the end-effector path. This represents the future positions of the robot TCP in the execution of the remaining trajectory. The projection of these points can be easily rendered on the worktable, giving the user a clear idea of the end-effector path that the robot will execute. It also provides useful information to recognize which part the robot end-effector is aiming for.

The waypoints of the robot trajectory are interpolated in order to have a smooth representation of the motion as the one executed by the robot. These positions,

represented in joint space, are then converted into Cartesian space through the computation of the inverse kinematics for each point. Figure 5.18 shows the 3D representation of the TCP trajectory in the virtual visualization of the workspace. While the robot is moving, the segments of end-effector path already executed are updated and removed from the visualization. In this way the user can visualize only the remaining trajectory that the robot still has to execute.

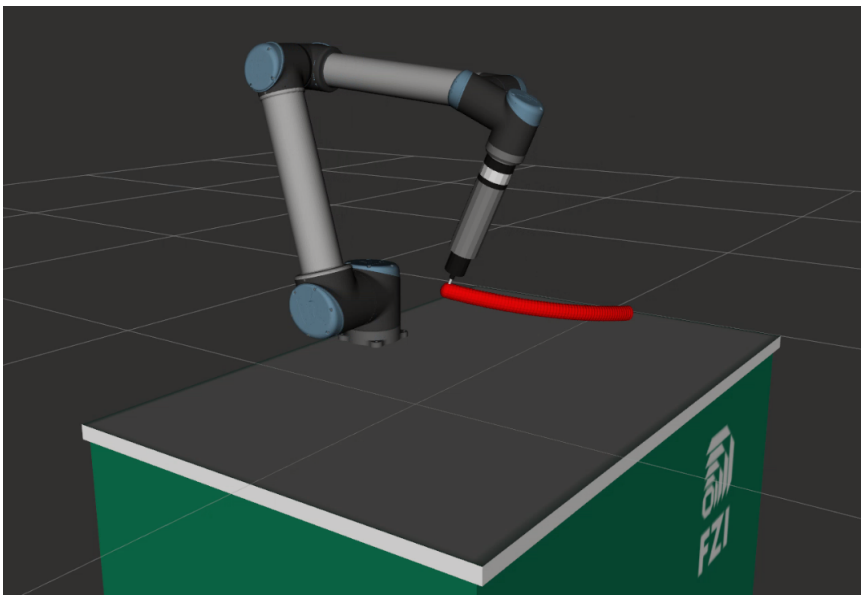


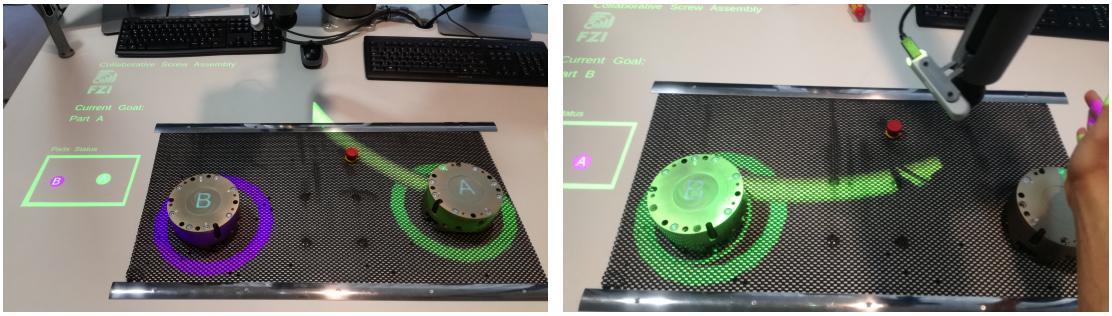
Figure 5.18: Representation of the robot end-effector trajectory in the virtual environment. This information helps the user to understand the path of the robot and its goal.

In Figure 5.19a it is possible to see the projected 2D representation of the end-effector path on the worktable. Figure 5.19b shows the change of trajectory after the detection of a possible collision with a human. Because of that the robot changes its target and the path to new goal is updated accordingly.

5.2.3 Experiments

The proposed projector-based feedback system was presented at the Motek 2018 fair in Stuttgart, Germany. Over four days many visitors tested the AR feedback system. The feedback collected showed that the additional information related to the robot target and motion were intuitive and easily understandable. The use of a video projector to dynamically display the information related to the entire system made them able to figure out the meaning of the projected markers by simply interacting with the workpieces and the robot.

The representation of the robot goal was very helpful in recognizing the part that the robot was aiming for. This enabled the users to have more comfort and confidence in the interaction and decreased the feeling of anxiety caused by



(a) Projection of the planned end-effector trajectory.

(b) A possible collision is detected and the robot replans its motion.

Figure 5.19: Projection of the planned end-effector trajectory (a). The user can thus recognize the path of the TCP towards the current robot goal. A possible collision is detected and the robot replans its motion to another goal (b). The end-effector trajectory is updated immediately to show to the user the new robot motion.

the lack of information from the robot. The information related to the collision avoidance system allowed the users to perceive when the robot had to change its motion in order to avoid a possible collision. This helped in making them aware of their influence on the robot motion and plan.

The visualization of the robot swept volume on the worktable turned out to be useful to make the users quickly understand the area in which the robot was not planning to move during the execution of its current motion. Thus the user was able to easily detect collision free zones in the workspace. The TCP trajectory representation turned out to be more suitable for the interpretation of the end-effector path, which is very relevant for some specific types of tools or applications.

5.2.4 Conclusions

The AR system has improved the understanding of robot intentions in human-robot interaction in a shared workspace deploying a fast replanning capable robot. The use of a projector to display the robot motion information directly on the worktable helped the users to understand the robot's plan in a faster and more intuitive way. Using this system, the human workers feel more comfortable and the acceptance is improved.

The next section will introduce and explore the use of wearable devices for augmented reality, as for example the HoloLens googles. In this way the 3D robot information could be represented in the 3D space without the need of projections onto surfaces. The swept volume representation in particular could be displayed to give exact information on where the user would be colliding with the current robot trajectory. Furthermore, the next chapter will investigate how the human

can react to this information in an immediate and intuitive way. Using gestures and speech, the user could interactively agree with the new planned motion or force the robot to find another path to the desired goal.

5.2.5 Head Mounted Display-Based AR

In this section, the motivation and proposed approach to communicate the robot intent using head mounted display-based AR is presented. In particular, the use of AR goggles such as the Microsoft HoloLens allows to represent 3D virtual models, enabling a more realistic representation of the robot motion and plan as well as workspace information.

Parts of the results presented in this section have been published in [2].

5.2.6 Motivation

As highlighted in the previous sections, sharing the same workspace, both human and robot should clearly understand the intentions and motions of each other, in order to enable an efficient and effective interaction. In this section is proposed an AR-based system to show the robot planned motion and target to the worker, making use of 3D virtual markers displayed through an head mounted display. The focus is on representing this information in an intuitive way for inexperienced users.

Thanks to the visualization of the swept volume of the planned robot trajectory, the user can understand exactly the volume occupation of the manipulator path, with a consequent reduction of anxiety, which is usually due to a lack of information about the robot intentions. Furthermore, detected impact points can be directly displayed and highlighted on the obstacle surface. In this way, the ergonomics in the collaboration is improved and the worker can also take better decisions on how to move in the workspace without making the robot change its motion or target. The AR visualization is also used to display information about the state of the parts in the workspace, highlighting the robot target and the workpieces that still need to be worked by the robot.

The effectiveness of the system has been evaluated with test cases performed by a group of 12 people with no robotic experience. The results showed that the system helped the users to better understand the robot intentions and planned motion, improving the ergonomics and trust in the interaction.

5.2.7 Method

In this section is described the proposed and implemented system to enable the communication of robot intentions, displaying the manipulator motion to the user using 3D virtual objects displayed through AR. For a more complete information about the workspace and the robot target, it has been also added the representation of the available workpieces and information about their state. The robot feedback system has been implemented and evaluated considering the screw assembly scenario described in the previous sections. Therefore, the robot is able to dynamically replan its motion, using the GPU-Voxels library. In the analyzed application scenario, an additional camera is positioned on top of the workspace in order to detect the available parts, providing the target positions needed by the robot to perform the tightening task.

In Figure 5.20 is represented the overall system architecture. The focus of this section is the implementation of the feedback system which is responsible to represent to the user information related to the robot motion and target. The input management module, which provides a command interface to collect the input feedback from the user on the current robot plan, will be introduced in the next chapter.

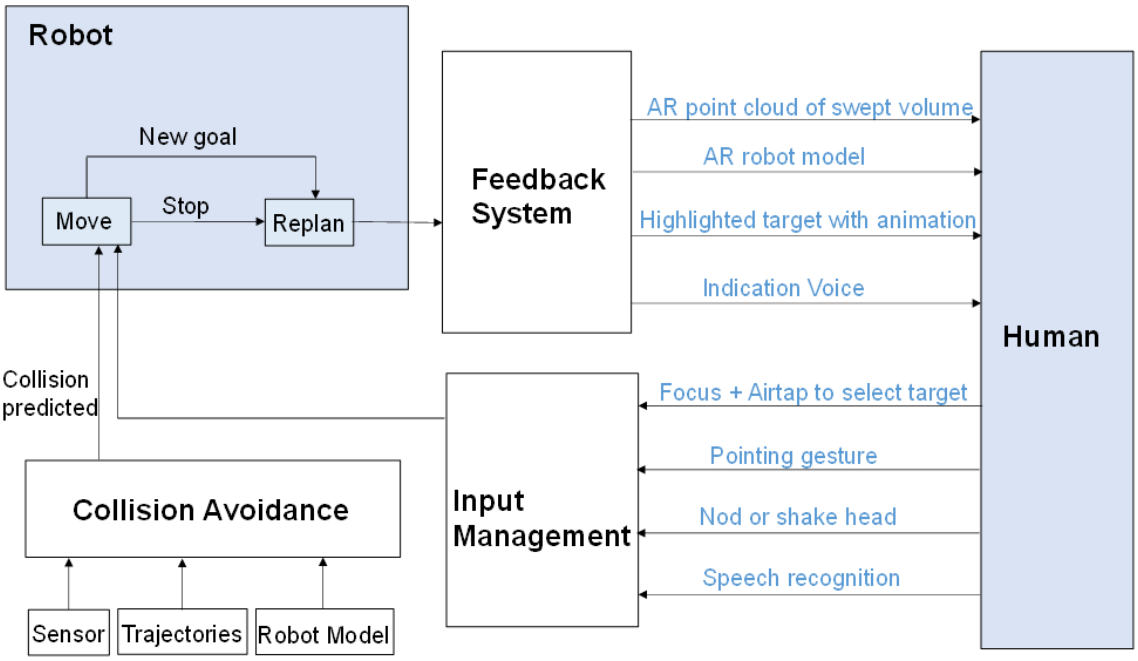


Figure 5.20: The overall architecture of the AR-based system. The feedback system is responsible to provide the user with information about the robot motion and target, which includes the point cloud representing the robot swept volume and animated markers.

In order to provide a feedback about the target configuration of the robot, a virtual model can be adopted to represent this information to the user. As depicted in Figure 5.21, the manipulator virtual model is superimposed on the real robot to show its planned target configuration.

Figure 5.21: A virtual representation of the robot can be used to represent its final target configuration in AR. The current state of the workspace is visualized as well.

However, this information does not give a clear and spatial information about the volume of the workspace which will be occupied by the robot in the execution of its next motion. This is particularly relevant because it is the volume used for the detection of collisions with the live environment, to avoid dynamic obstacles such as humans. The 3D swept volume is an important information to make the human understand the volume that the robot will occupy in the next future. Indeed it represents the volume that the worker should avoid to occupy if not necessary. The representation of this information to the user, enables him to avoid useless changes in the robot motion. Furthermore, understanding the intent of the robot, the human operator is then able to feel more comfortable in the interaction and take better decisions on how to move and work in the workspace, with a consequent better ergonomics. For the AR overlay representing the robot information, the Microsoft HoloLens headset has been used. The programming of the virtual information has been implemented in Unity [112]. The communication of the HoloLens and ROS has been developed using the open source software library ROS# [22]. The voxel maps representing the swept volume of the robot trajectory are converted into ROS point cloud messages, which can be imported

and represented in Unity. In order to be displayed through the goggles without latency, the high resolution point clouds need to be downsampled. This has been done without compromising the swept volume information, by removing the internal points and keeping the one on the volume surface which are the one needed for a correct visualization. In Figure 5.22 the robot swept volume related to the current planned trajectory is represented in blue.

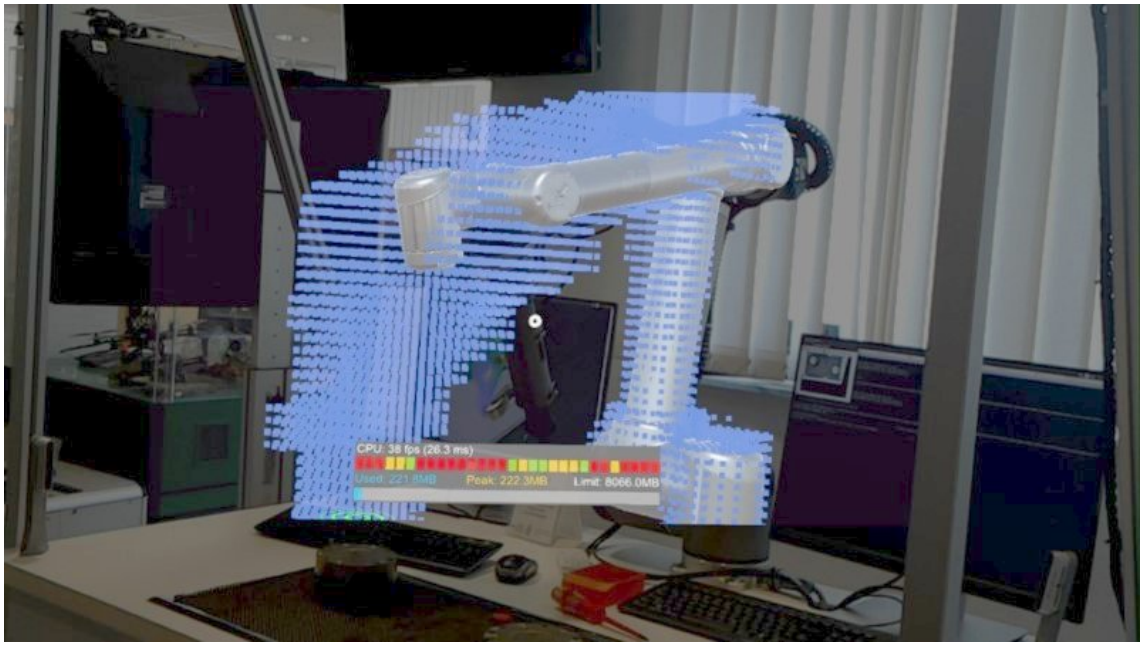


Figure 5.22: The point cloud representing the swept volume of the planned robot motion. This is used to predict collisions with the live environment and it is a useful information to understand the volume that the robot will occupy in the near future.

The information related to the workpieces state has been implemented using 3D markers, which are displayed at the positions detected by the camera system. Every localized part is assigned with a unique id, which is displayed to the user in order to make him able to easily refer to it when deploying the speech input interface developed. Different colors and animations are used to represent the current state of the workpieces and to highlight the one that is selected as current target by the robot. Figure 5.23 shows the representation of this information for two parts available on the worktable. This includes colored markers to highlight the available parts and the current one selected as target by the robot. Text information enables the user to identify the parts in order to allows him to reference them using speech-based commands as described in the next chapter. With this visual information displayed in AR, the user can understand in an intuitive way the current goal of the robot and the volume that it will occupy while executing the trajectory to reach it. The collision free space in the working area can be easily detected, making the worker able to avoid obstructing the robot path. Furthermore,

this enables a better comfort in the interaction, due to the clear understanding of the complete robot motion.

Figure 5.23: The current state of the workspace is displayed in AR. The information related to the workpieces state is represented with markers and text. Synthesized speech is used as well, in order to provide information about the robot target even if the user is not able to visually check the virtual overlay.

Another crucial information that the system is able to display is the specific area that is detected as a possible obstacle for the current robot path. The user, working close to robot, might be more interested in understanding which part of its body it is exactly obstructing the robot motion, in order to be able to give clearance to the robot without moving away from his task. To achieve this, once a collision is detected, the collision area is highlighted through the representation of the colliding voxels in red. These are obtained as result of the intersection between the voxel map representing the live environment and the one related to the trajectory swept volume. Figure 5.24 shows the situation in which the robot detects a possible collision with the user, showing in red the relevant impact area.

Synthesized speech is also used to provide further robot information, deploying a different channel of communication. This has been included because, if the user has to focus on a part and cannot draw his attention to the robot motion, he can still understand the current plan of the robot and which target it is aiming for. The robot communicates its intention using simple sentences that are played using the built-in speakers in the HoloLens. “Moving to target one/two” is used to communicate that the robot has found a new executable trajectory. “Moving to target one/two, executing” it means that the execution of the planned trajectory has started. In case a possible collision is detected, the robot warns the user with “Warning, possible collision detected”, in order to alert him even if he is not looking at the workspace area.

5.2.8 Experiments

In order to evaluate the system, a test scenario with a group of 12 users with no experience in robotics and AR has been considered. Two test cases have

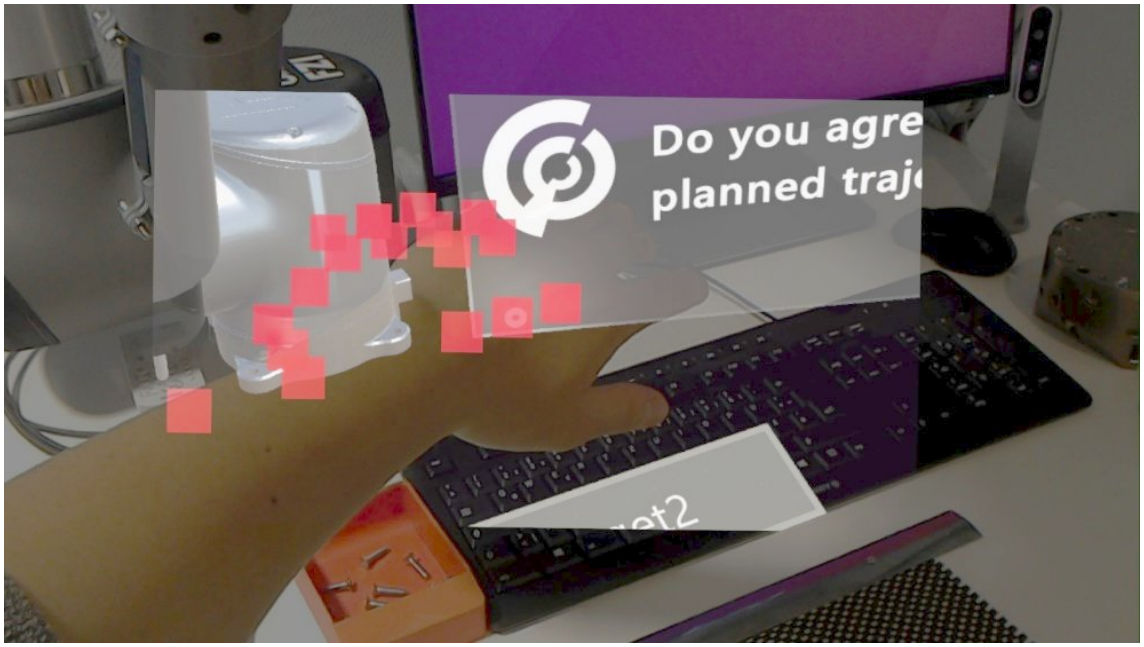
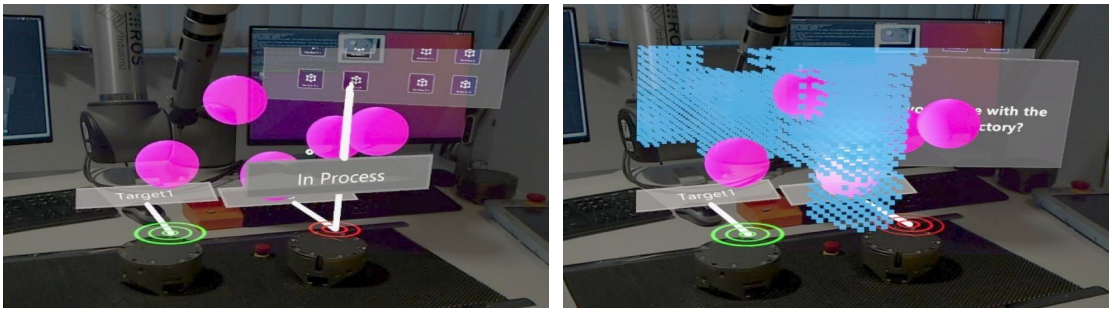


Figure 5.24: The collision points are visualized in AR with a red colored point cloud. These points are the voxels of the trajectory swept volume in collision with the live environment captured by the depth cameras.

been created to examine the effectiveness of the system and collect a feedback about the interaction modalities. The first test had the aim to evaluate the spatial understanding of the robot motion displayed to the user in AR. Virtual obstacles have been added to the displayed information, represented as spheres in the AR overlay. The users were asked to state if these objects would cause a collision with the robot planned motion. The test has been performed as first without any information about the robot motion, apart from highlighting the manipulator target. After that, the 12 users had to perform the same test with the addition of the trajectory swept volume visualization.

In Figure 5.25 is represented the virtual information displayed to the users for this test and Figure 5.26 shows the results of the evaluation. As it is possible to observe from the chart, the error rate in the object classification was lower deploying the visualization of the trajectory swept volume. Without the visual information related to the robot motion and its volume occupation, the users had clearly more difficulties in judging the possible collisions with the virtual obstacles.

In the second test, the overall system and the user perception during the interaction have been evaluated. The users had to fill out a questionnaire related to the general experience in the interaction with the system. The users had to test the interaction in the workspace with the robot performing the placement of the workpieces and screws on the worktable, first without any feedback information and then using the proposed communication system. At the end of all the tests, the users had to rate 10 sentences using a 5-point Likert scale from “Strongly Disagree” to “Strongly Agree”:



(a) Only target. (b) Swept volume.

Figure 5.25: Test for the evaluation of the spatial understanding of the robot motion. Virtual obstacles are represented to the users as spheres and need to be judged as colliding or not with the planned robot motion. In (a) only the target of the robot is visualized. In (b) is represented the same test including the swept volume information.

1. The system improved your understanding of the spatial occupancy of the planned motion of the robot.
2. The system improved your understanding of the next robot goal.
3. The head movement input is effective and convenient to command the robot.
4. The speech recognition input is effective and convenient to command the robot.
5. The pointing gesture input is effective and convenient to command the robot.
6. The focus and AirTap gesture input is effective and convenient to command the robot.
7. The system improved your working efficiency by changing the robot goal by voice and gestures.
8. It is safe to work in collaboration with the robot without the communication system.
9. It is safe to work in collaboration with the robot with the communication system.
10. The system improved your feeling of safety working along with the robot.

Question 3-7 are referring to additional functionalities that will be introduced and described in the next chapter.

Sentences 1-2 regard the understanding of the robot intentions. From the result reported in Figure 5.27 it possible to notice that the users found that the proposed system improved the understanding of the spatial occupancy of the planned robot motion and the manipulator goals.

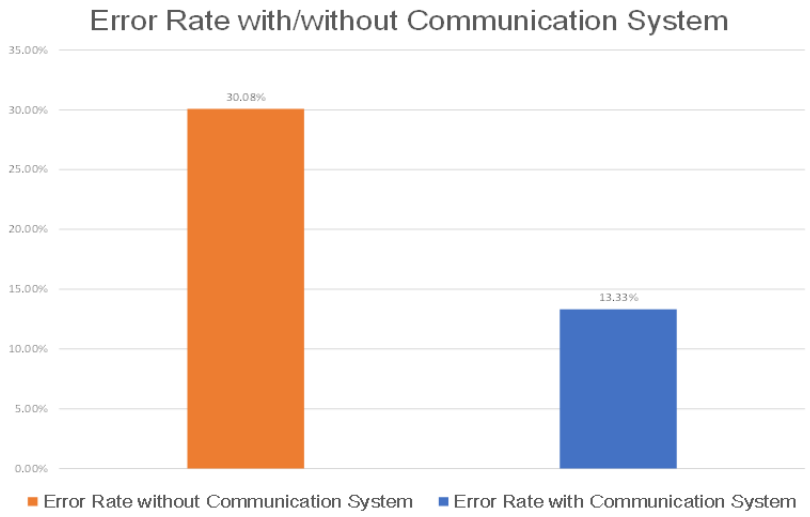


Figure 5.26: The results of the test for the evaluation of the spatial understanding of the robot motion. The chart shows that, deploying the swept volume information, the user had a better understanding of which area and obstacles would cause a collision with the planned robot trajectory.

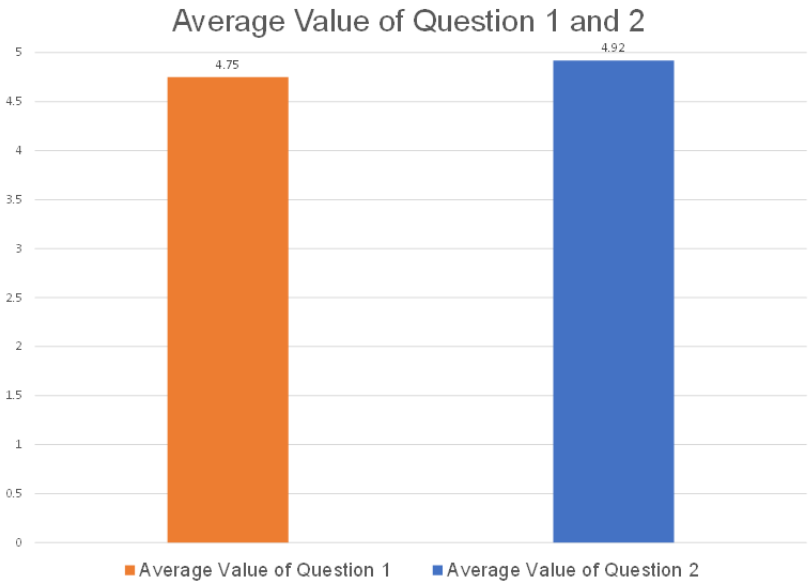


Figure 5.27: The results of the questionnaire regarding the understanding of the robot intentions.

Sentences 8-10 deal with the trust of the users towards the robot. The chart in Figure 5.28 shows that the users had an improved feeling of safety and comfort in the interaction with the robot using the communication system developed.

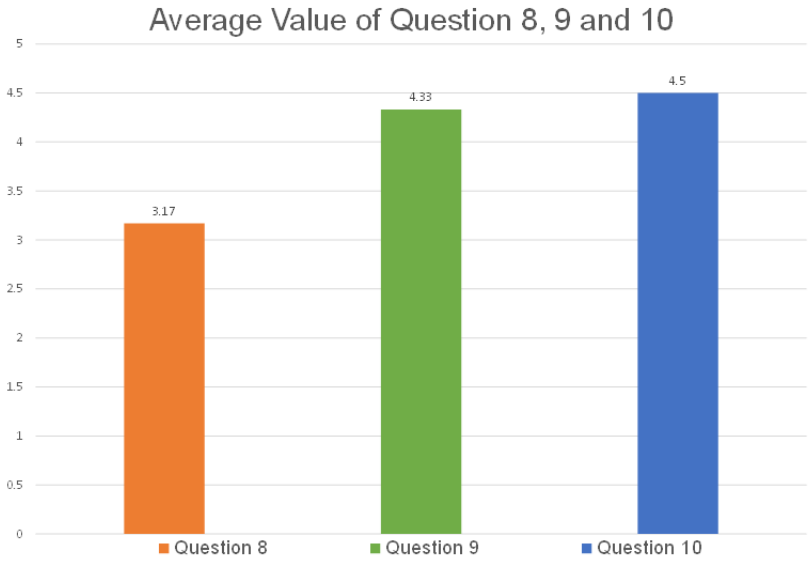


Figure 5.28: The results of the questionnaire regarding the trust towards the robot.

5.2.9 Conclusions

The system proposed in this section aims at improving the interaction with a robot in close HRC, providing robot plan and motion information displayed through the use of 3D virtual markers. The evaluation tests performed with unexperienced users showed that the system improved the understanding of the robot motion in order to make the human more comfortable in the interaction. The use of AR to show the robot planned motion and swept volume, proved to be an intuitive and effective way to represent the robot plan to the user.

The ease of use and the effectiveness of the system developed could be improved with the future enhancements in the AR technology, in order to provide a better field of view for the visualization of the virtual overlay. The communication of the robot plan could be further developed including multi-modal communication through the use of haptics or EMG based interfaces. Input modalities to agree or modify the plan communicated by the robot can enable a more efficient way to exchange information and agree on the robot plan. Multi-modal input modalities that can be used for this purpose are presented in the next chapter.

5.3 Summary

In a team, it is really important that all the members understand the intents and motion of each other. In the same way, in HRC is of crucial importance that both the human worker and the robot understand what the other is planning to do.

In this chapter, methods to display the robot intents and motion information to the user in an intuitive way have been proposed. This has been done using virtual projections and AR representations of the robot trajectory swept volume and workspace state. A multi-modal robot feedback has been proposed combining visual and acoustic information, in particular to make the user understand what the robot is planning to do and possible collision with the manipulator. Acoustic signals have been used to draw the user attention to the more detailed visual information displayed in AR, which allows him to clearly understand the spatial occupancy of the planned robot motion.

Enabling the human worker to understand the robot intentions and planned motions, it is important to provide him with the possibility to deny or change the robot plan if this is not optimal or if the worker wants to operate in a specific area of the workspace in order to check an unexpected failure. For this is important to provide intuitive and effective interfaces to adapt the robot motion at runtime. In the next chapter are proposed multi-modal interfaces to modify the robot plan during operation, using speech and AR/VR based inputs modalities. In particular, the proposed methods focus on making possible to adapt the robot trajectory and workspace constraints at runtime, without the need to stop the application and reducing the reprogramming phases needed to modify the robot behavior.

6 Robot Plan and Motion Modification

6.1 Projector-Based UI

This section introduce the motivation and proposed approach to enable the modification of the robot motion execution in a intuitive way using interfaces made available through the use a projector-based UI. Casting virtual buttons and sliders directly on the worktable, the user can deploy simple gestures to stop/start the robot and change its speed based on his needs and without stopping the entire application.

Parts of the results presented in this section have been presented in [11].

6.1.1 Motivation

The main motivation of the system proposed in this section is the design and development of a user interface to intuitively change the robot trajectory execution. The method available in literature are usually lacking of robustness and only provides ways to stop/start the robot. The proposed method builds on top of the projector-based AR overlay presented in the previous chapter. Virtual buttons and sliders are added to the visualization and user's gestures are used to detect the inputs. A combination of depth camera-based hands segmentation and skeleton tracking, can enable a robust detection of the worker's inputs. The system has been integrated in the collaborative assembly application and test have been performed to evaluate the robustness and limits of the method.

6.1.2 Method

The proposed method is based on casting virtual buttons and sliders on the worktable and tracking the user's hand position in order to detect his inputs. The proposed interface enable to stop/start the robot and change its speed used for the trajectories execution.

In order to enable a more robust input detection, two approaches are combined in order to detected the operator's commands. The skeleton detection library

PoseNet has been used to track the user's pose and hands positions. At the same time, a 3D camera is deployed to collect the point-cloud of the workspace, which is used to perform the hand segmentation in the input areas using the PCL library. Figure 6.1 shows the architecture of the proposed system and the communication between the different components.

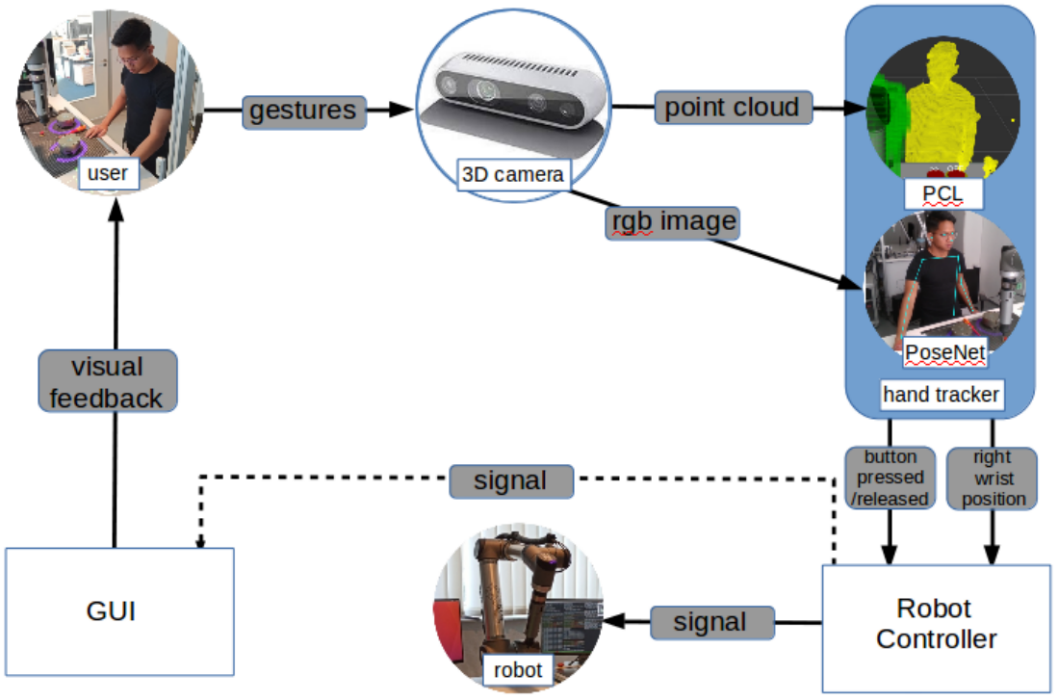


Figure 6.1: Overview of the closed-loop cycle of the proposed projector-based gesture interface.

In the following, a short introduction and description of PoseNet and PCL is presented.

Pose estimation is a technique of computer vision that can detect the human presence tracking the position of his limbs from an image or video. The use of the skeleton tracking provided by the PoseNet library have been selected for the proposed UI for the following reasons:

- PoseNet is very practical to use because it only requires a webcam and a web browser.
- PoseNet can be used to detect single or multiple human's poses. The recognition of single poses is relatively easier and faster compared to the recognition of multiple poses, but requires that only one human is present in an image or video, otherwise the estimated position can be wrong when the limb positions of different persons are combined in a single image.

In Figure 6.2 is depicted how the main components used by PoseNet work to estimate single human pose along with 17 key-points. Figure 6.3 shows the names and locations of the key-points related to an image capturing an entire human body. At the highest level the process can be described in 3 stages:

- Processing an input RGB image through convolutional neural network using MobileNet or ResNet.
- Producing output models in the form of heatmaps and offset vectors.
- Estimating poses from the output models.

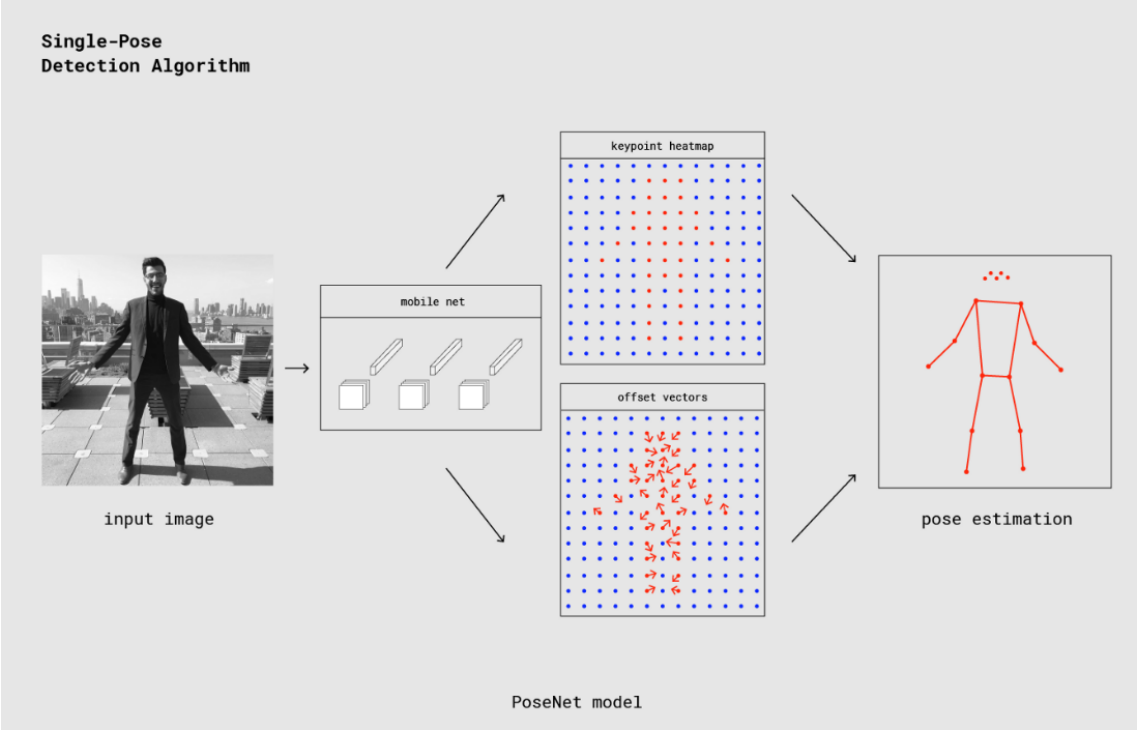


Figure 6.2: Single pose detection algorithm used by PoseNet for skeleton tracking.

The point-cloud is a data structure that represents a collection of points in a multidimensional space. For example, a 3D point cloud is vector of 3D points, which have a defined position on the x, y and z axes.The PCL or Point Cloud Library is a C ++ based library for 3D point-cloud processing. The framework implements state-of-the-art algorithms that can be useful for various types of point cloud processing such as filters, features, key points, registration, kdtree, octree, segmentation, sample consensus, surface, recognition and visualization. In this section, the PCL is used to filter, crop and count the number of points coming from the 3D camera, in order to detect the hand position when placed on the UI.

The proposed system essentially consists of 3 parts:

- Hand position recognition, using both PCL-based detection and PoseNet skeleton tracking.

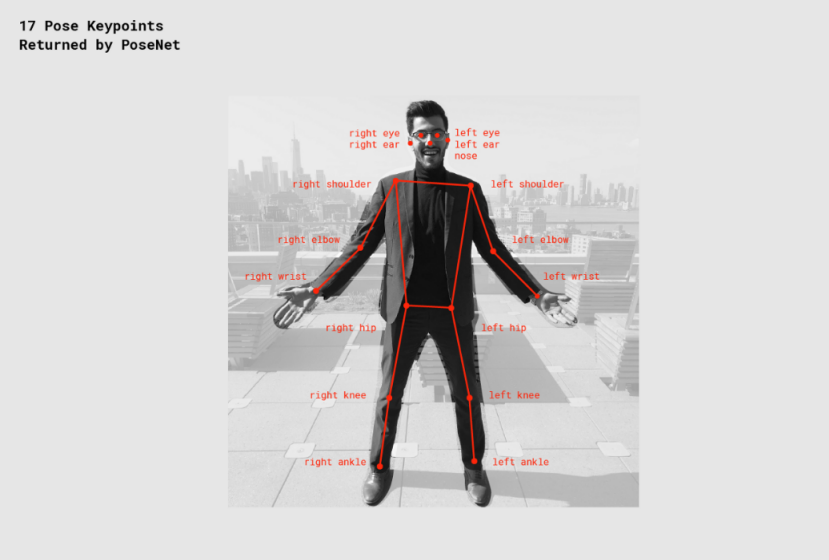


Figure 6.3: Keypoints used by PoseNet skeleton tracking, names and reference poses are shown using an entire human body captured by a camera.

- Interactive marker creation to provide a feedback about the detected commands and show visual information about the updated state.
- Interface integration into the robot controller in order to make the robot start/stop or change speed accordingly to the detected input.

Figure 6.4 shows the workflow of these stages and how they are interconnected.

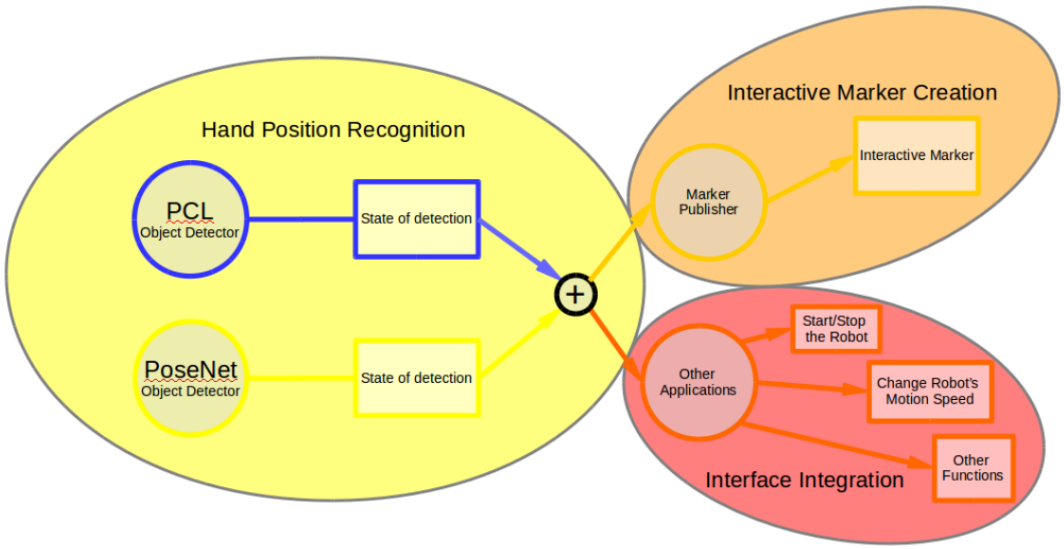
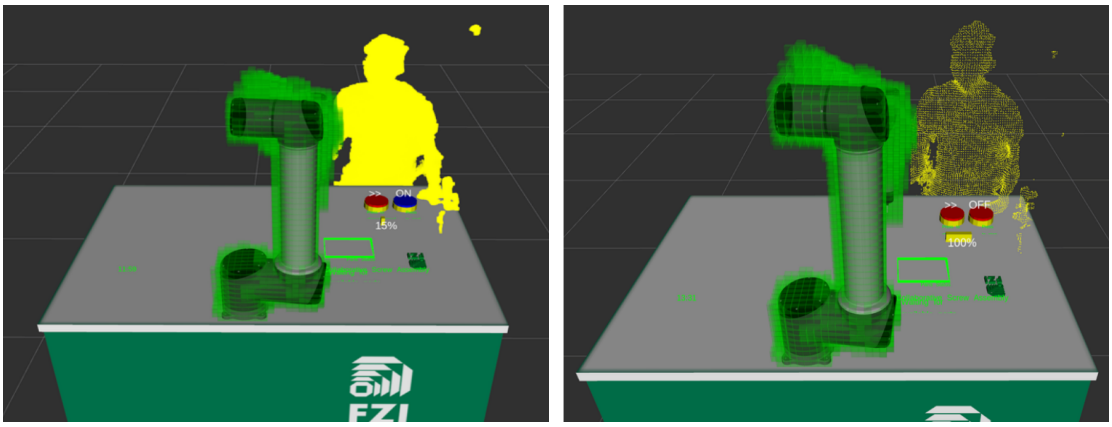


Figure 6.4: Projected UI system architecture. The PoseNet and PCL-based hand detections are combined to have a robust tracking of the user's inputs, that are then use for the creation of the visualization feedback and robot control signals.

In order to detect the user's hand from 3D images using PCL, the input point-cloud is filtered and cropped. A 3D camera is used to collect the depth information of the workspace in the form of point-clouds. These latter are the raw data used for further processing. However, at this point, the raw data contains noise and consequent unexpected points caused by the light sensitivity of the 3D cameras, as depicted in Figure 6.5a. Therefore, the raw data must be downsampled to filter the noise in order to get an exact point-cloud representing the workspace. The other reason for downsampling is to save computation time by reducing the number of points but keeping the relevant data information, so that this data can be processed optimally in the next step.However, the more the point cloud is downsampled, the fewer points come out as output. This is important to take into consideration, because too much filtering can remove relevant points, such as hand related points, which are needed in the next step. Therefore, a downsampling level must be estimated at this point to remove as much as possible points representing noise and still get the data representing the user's hand, which is essential for the method. The result of the downsampling is shown in the Figure 6.5b. The extracted data from this step is then delivered to the next step for further processing.



(a) Raw camera point-cloud. (b) Filtered point-cloud.

Figure 6.5: The raw point-cloud collected by the 3D camera (a), which is filtered to remove noise and improve the overall computation time (b).

After the filtering step, the processed point-cloud is cropped at the desired position, e.g. above the button markers position. In order to achieve an optimal result, the area of the registered points, when the button markers are covered, is estimated, e.g. a box-shaped area with the same size as the button markers. For this purpose it is possible to translate, rotate, transform and resize the 3D basic cube that represents the standard shape. After cropping the point-clouds, the number of points at a certain position, for example above a marker, is counted. Measuring the number of points appearing when an object is and is not located on the marker, a threshold representing the number of points that must be exceeded to assume that an object is located above the marker, have been set. Now the system is able to detect the existence of an object above the marker. At this point, the interface is not robust enough, as it is not possible to determine whether a hand or a general

object is covering the area. Therefore an additional hand detection system which relies on PoseNet has been implemented and added.

The same 3D camera providing the point-cloud processed with PCL, is used to capture RGB images. These RGB images are then fed into Posenet to obtain key points of the user's body parts, such as the left wrist, as shown in the Figure 6.6.

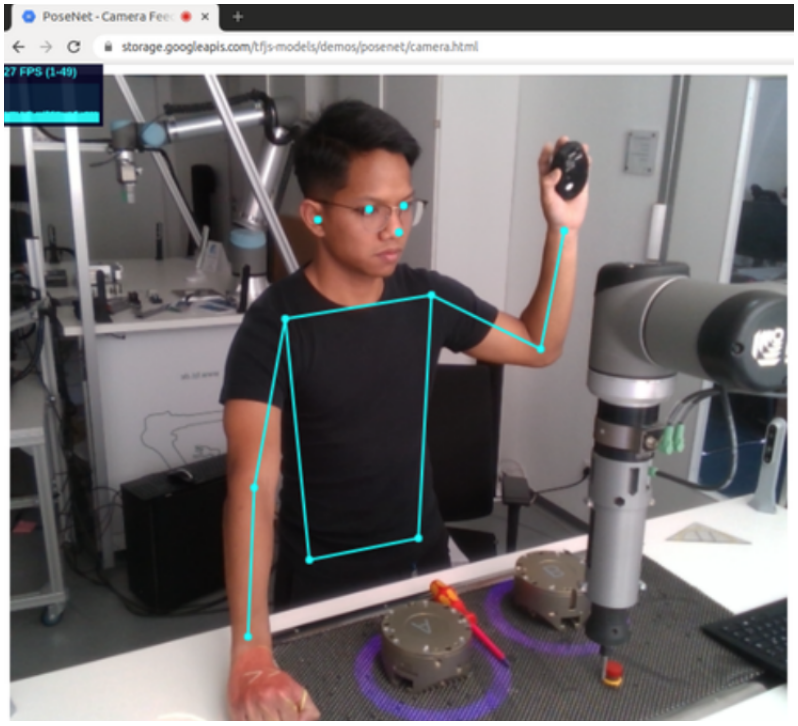


Figure 6.6: Posenet detection: the skeleton keypoints detected are highlighted on top of the input image.

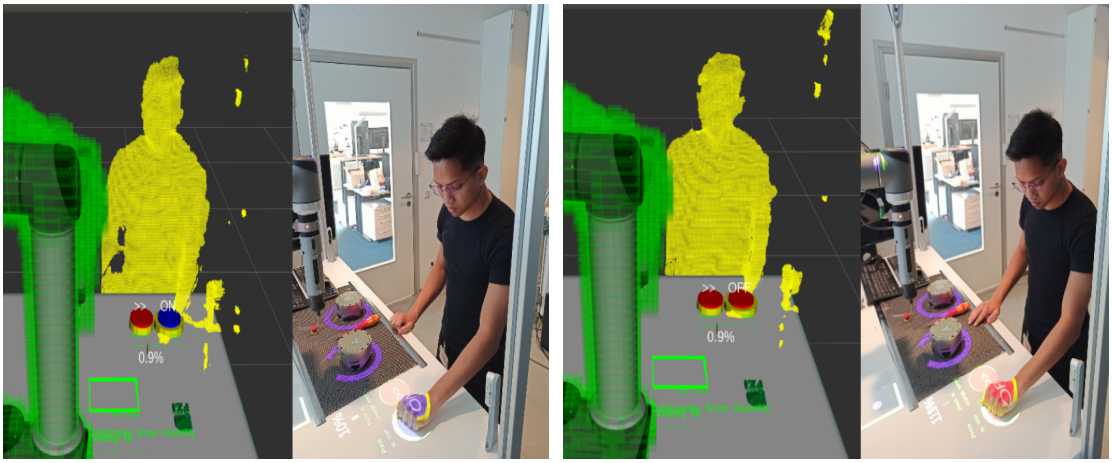
PoseNet generates in real-time a list of keypoints, including their confidence scores and positions as output, as shown in Figure 6.7. To import the PoseNet data into the developed system, a ROS wrapper and a keypoint extractor have been implemented to extract this information and process it in further steps. After the desired information, e.g. the position of the left wrist, has been extracted and obtained, experiments have been performed to determine the distance threshold regarding the position of the user's body parts when it is at the desired location of the work area. The smaller this threshold is, the more difficult the wrist can reach the position considering that the value generated by PoseNet is affected by noise. On the other hand, if the threshold value is too large, the information that is passed on to the next step will not be accurate. Therefore, the threshold value must be defined to obtain the most accurate information regarding the presence of the user's wrist at the exact position, e.g. on the button marker. This information is then fed to the next step for further processing.

```
{
        "score": 0.32371445304906,
        "keypoints": [
        {
                "position": {
                        "y": 76.291801452637,
                        "x": 253.36747741699
                },
                "part": "nose",
                "score": 0.99539834260941
                },
                {
                "position": {
                        "y": 71.10383605957,
                        "x": 253.54365539551
                },
                "part": "leftEye",
                "score": 0.98781454563141
                },
                {
                "position": {
                        "y": 71.839515686035,
                        "x": 246.00454711914
                },
```


Figure 6.7: List of PoseNet keypoints and example of detected values.

Next, the information obtained from the two object detectors, namely PCL and PoseNet, is compared. If both simultaneously detect that the user's body part is in the correct position, e.g. the left arm on the same button marker, a feedback is given to the user. The feedback animation depends on which virtual button was pressed. For example, when the on/off button is pressed, the size of the yellow button frame, the button color and the text are updated, as shown in the Figure 6.8.

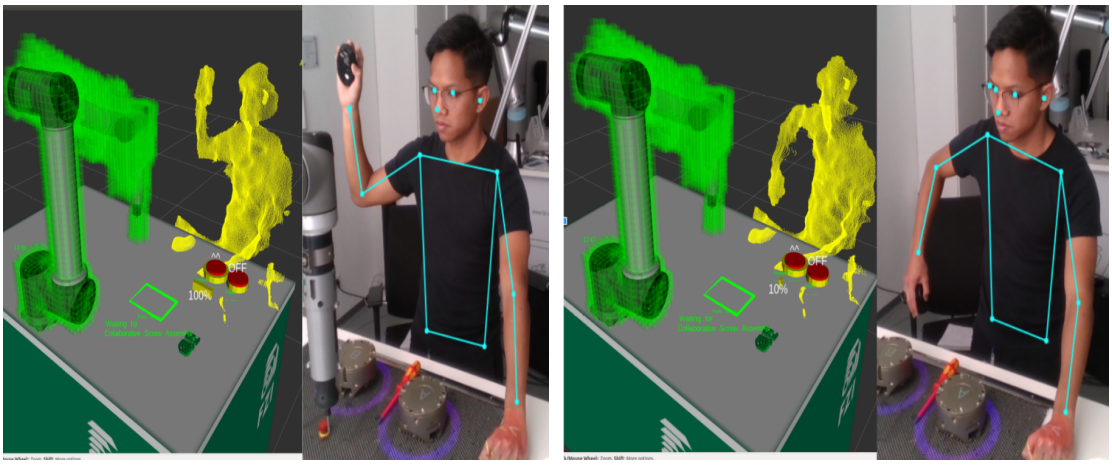
When the speed control button is pressed, only the size of the yellow button frame and the text change. In order to increase or lower the robot speed, a virtual slider has been implemented which can be modified by the user moving his hand. In particular, while the speed control button is still pressed, the user can lift or lower his left arm to adjust the speed of the robot arm, as shown in Figure 6.9. This function is enabled by tracking the key point of the right wrist while the left wrist is positioned on the speed control marker. Two different distance thresholds must be set for this function. Based on the position of the right wrist, there are 3 ranges with corresponding outputs. The first range, which is lower than the lower threshold, is made to reduce speed. The second range, which is higher than the higher threshold, is made to increase the speed. The range between the higher and lower thresholds is used to keep the speed constant. As long as the left wrist is on the speed modification marker, the updated speed value will be displayed by the marker, but will not updated in the robot controller. To send the desired speed to the robot, the user needs then to move his left wrist away from the speed



(a) Stop robot. (b) Start robot.

Figure 6.8: The virtual button can be used to stop and start the robot motion execution. As soon as the input is detected the projected marker is update accordingly to provide a feedback to the user.

control marker.



(a) Increase robot speed. (b) Decrease robot speed.

Figure 6.9: Once the speed button is pressed, the user can use the other arm to increase or decrease the robot speed, by simply raising or lowering his other hand.

6.1.3 Experiments

Robustness

The first experiment is intended to determine the robustness level of the interface to detect the position of the user's hand in a particular location, e.g. above the

markers in the workspace, by using PCL only, PoseNet only, and using both PCL and PoseNet together. Using the PCL library, the detection of the hand position is done by counting the number of points that are above the button. However, the weakness of this approach is that the hands and other objects cannot be distinguished, so that if there are other objects similar in size to a human's hand in the same position, the number of points produced will be enough to pass the threshold, which is set as parameter. Therefore, this approach is not robust if it operates alone, because it allows only to detect the presence of an object in one position in 3D space without knowing whether it is a hand or not. Using PoseNet, the location of the hand is obtained through skeleton tracking on a 2D image. The position o the body keypoints, e.g. the left wrist, is referring to the overall image size. The weakness of this approach is that the keypoints positions resulting from movements along a perpendicular line to the camera are gonna be detected as the same, as seen in the Figure 6.10. So that an unintended button push from a false detection of the user's hand position in the 3D workspace can occur. Therefore this approach is not robust if it operates alone. When using the combination of the PCL and Posenet methods, the detected position of the user's hand at a given location is determined by two conditions that must be fulfilled. These two conditions combine the approaches used in the PCL and Posenet as described above. This approach is far more robust than either of the previous two methods operating by themselves, because objects must be present at specific locations in 3D space and key points must be present at specific locations in 2D images. Therefore this approach is the most robust method compared to the previous two approaches.

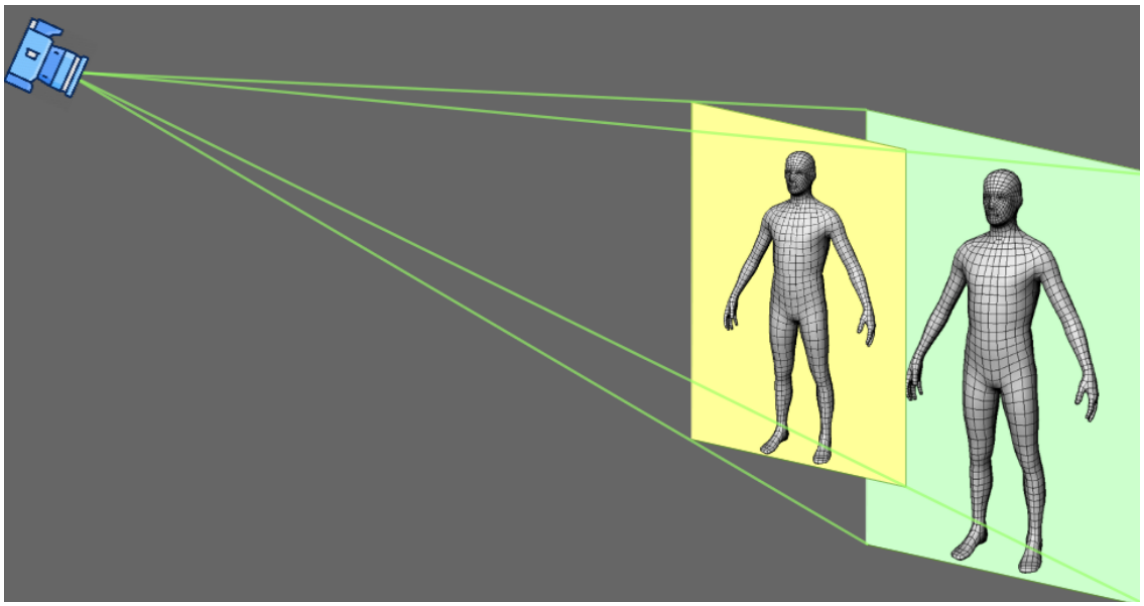
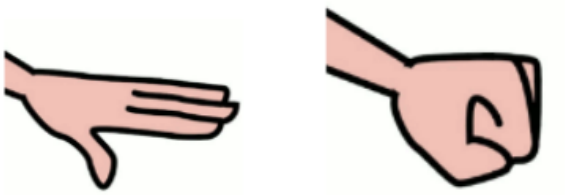


Figure 6.10: Perpendicular line through 2D images, which highlight the limitations of using only PoseNet for the user's hand detection.

Input Gesture

The second experiment was intended to find out which input hand gesture was the most effective in using this projector-based AR interface. For testing that, 2 hand gestures, open and clenched hands, have been selected. With each hand gesture placed directly above each button marker, the experiment has been performed. This experiment has been repeated 10 times for every hand gesture and button marker. The results are reported in Figure 6.11.



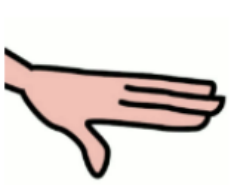
	Open Hand	Clenched Hand
ON/OFF	0/10	10/10
Speed Control	0/10	10/10

Figure 6.11: Evaluation of different hand gestures, placing the hand directly above each button marker on the table surface.

Looking at the results, it can be concluded that the clenched hand is very effective to be used with this interface. Therefore, with a closed hand the threshold that has been set can always be bypassed. Also the thickness of the open hand is very thin and only ± 2 cm above the workspace surface. This causes the point cloud generated by an open hand to be often cut off by the crop function, which is used to eliminate the points that represents the table. Further tests showed that, using the open hand gesture, this latter should be placed ± 5 cm above the button marker. Based on the results of the previous experiments, a similar experiment was carried out, but the hands were placed ± 5 cm above the button marker. The results of the experiment are reported in Figure 6.12. The results show that both hand poses are equally effective when a distance of ± 5 cm from the surface of the table is set. However, since it is difficult to always estimate and place the hand at the correct height when using the interface, users are advised to clench their hands and place them directly on the virtual buttons to use the available input interface.

Interface Reachability

The third experiment was intended to determine the effectiveness of using the interface from various areas in front of the table. For this reason, the length of the table, which is 1.5 m, has been divided into 5 areas of 0.3 m each, as can be seen in Figure 6.13.



	Open Hand	Clenched Hand
ON/OFF	10/10	10/10
Speed Control	10/10	10/10

Figure 6.12: Evaluation of different hand gestures, placing the hand ± 5 cm above each button marker.

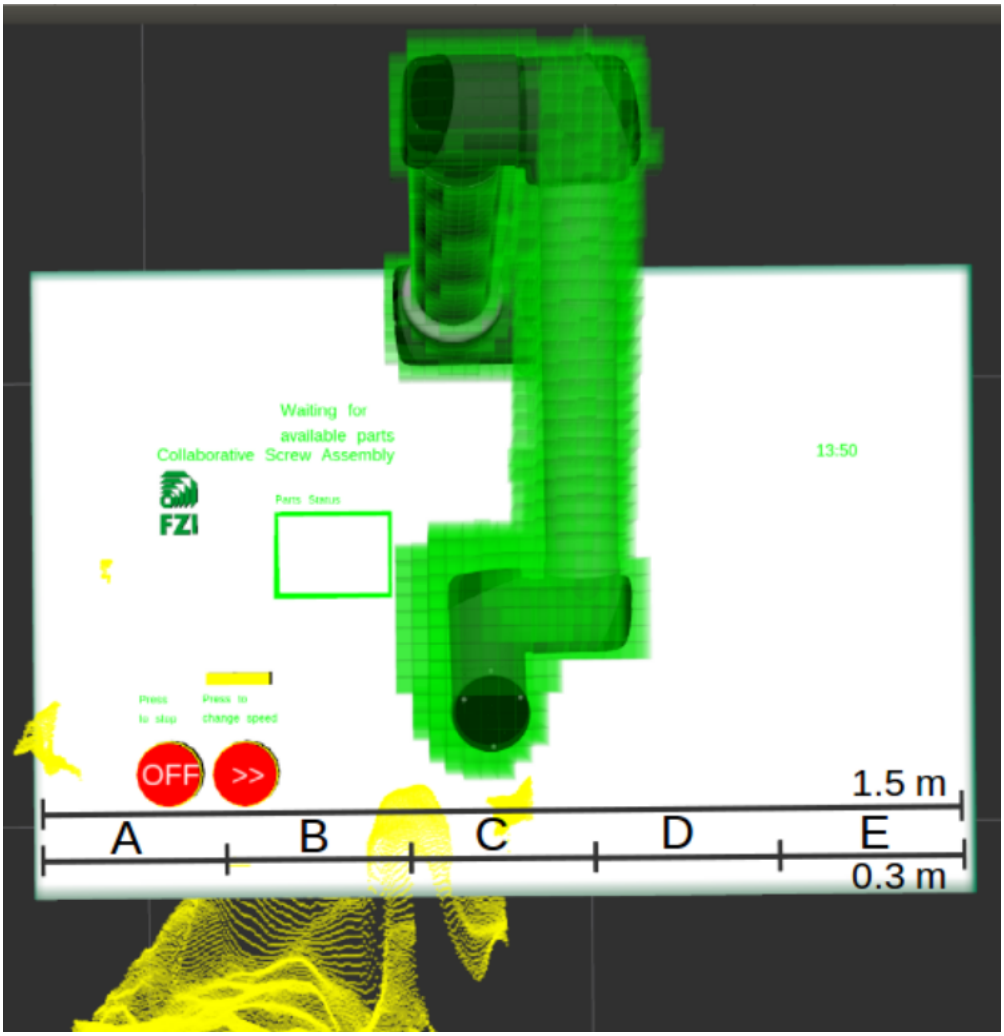


Figure 6.13: Segmentation of different user position in front of the table.

In the test, the user had to try to press both buttons from each area. The results and

the problem encountered are reported in Figure 6.14. From the results, it can be concluded that the most effective interface usage was obtained from areas B and C. In area A there was a problem when pressing the on/off button. This is due to the failure of skeleton tracking from PoseNet. Skeleton tracking failure of PoseNet is caused by an input image that does not meet the requirements for the skeleton tracking, which imply that the image fed must cover the entire upper body. It also happened that trying to press the speed control button from the area A, the other button was accidentally pressed. This occurred because of the specific position of each button. In this case, it happened because the keypoint of the left wrist was in the tolerance zone of the two buttons and the forearm in the point-cloud covered the on/off buttons. The same problem occurred when pressing the on/off button from the area D. The problem that had arisen wwin segment D, when pressing the speed control button, occurred because the keypoint of the left wrist was outside the tolerance zone that has been set for the speed setting button. This resulted in having one of the two conditions not fulfilled, while both conditions must be fulfilled in order to have a successful detection of the virtual button. This may occur because the key point that have been used, and which is given by PoseNet, is the position of the wrist instead of the hand. Therefore, this interface can be used optimally if the forearm and hand are in a straight line or if improvement are made deploying a skeleton tracking system which reliably detect the user's hand pose.

	A	B	C	D	E
ON/OFF	failed	success	success	failed[1]	too far
Speed Control	failed[1]	success	success	failed	too far

Figure 6.14: The results and issues encountered performing the interface reachability experiments. Different user's positions in front on the worktable have been selected in order to test input commands from different position in the workspace.

Anyway the system setup and camera position have been designed having in mind the user standing in front of the robot base, which position has been proven to provide a robust detection of the commands.

6.1.4 Conclusions

In this section a projector-based AR gesture interface was proposed. The developed interface receives sensor data in the form of point-clouds and RGB images. The inout data point-clouds are downsampled and cropped at the desired position, e.g.

above the markers which define the interface buttons. The resulting point-cloud is then analyzed and used as first input of the gesture interface. Moreover, input RGB images are processed by Posenet to estimate the user's hands key points. The desired key point, for example the key point of the left wrist, is then used as the second input of the proposed gesture interface. These two inputs are used to allow the user to interact with the robot arm through the virtual buttons, i.e. to stop and start the manipulator or to slow down or speed up its motion. As feedback to the user, the interface generates markers that are projected onto the workspace through a beamer. Experiments have been conducted to test the effectiveness of using other approaches found in literature such as PCL only and Posenet only. However, the combination of the two approaches has proven to be the most effective for a robust detection of the interface inputs. Other experiments, such as the use of different hand gestures, have also been conducted. This experiment led to the conclusion that the clenched hand is the most effective way to use interact with the virtual button cast on the worktable. The last experiment conducted was related to the evaluation of the effectiveness of the proposed interface for a use from different areas in front of workspace. From this experiment it can be concluded that only 2/5 the area in front of the workspace can be utilized to make the most effective use of the virtual buttons. Anyway the system setup has been designed having in mind the user standing in front of the robot base, which position has been proven to provide a robust detection of the commands.

A 3D skeleton tracking or an open-sourced 3D hand-only tracking could be an alternative to get a more robust and precise input detection. A 3D skeleton tracking can replace the combination of the approaches used in the proposed method and a hand tracking component could be used to finer interact with the interface, e.g. by drag and drop markers or zoom in/out markers representing the virtual buttons. Finally, the use of a beamer to project markers onto a workspace often generates limitations, e.g. the projection of a marker that is blocked by the moving robot arm and therefore not visible to the user. In addition, the width of the work area and the lighting of the environment sometimes become a limit for an optimal projection-based visualization. Based on the above problem, the use of head-mounted augmented reality devices can be considered as a solution, as presented in the next section.

6.2 Head Mounted Display-Based Gesture and Speech Interface

This section introduce the motivation and proposed approach to give to the robot a feedback about the communicated plan, providing interfaces to deny and change the robot plan at runtime. In particular, the HMD-AR system presented in the previous chapter to communicate the robot intents is extended to provide a multi-modal user interface to enable the user to give a feedback to the robot and modify

its plan. The input modalities include gestures as well as speech commands in order to allow a more flexible HRI.

Parts of the results presented in this section have been presented in [2].

6.2.1 Motivation

In the previous chapter, methods to communicate robot intents to the human have been presented. In particular, the HMD-AR system enabled the representation of 3D information directly in the workspace, allowing the user to intuitively understand the robot planned motion and workspace state. In the same way, using HMD-AR it is possible to provide the user with intuitive interfaces to give commands to the robot or feedback about the communicated plan. This is important to provide a flexible HRI, in which the user is able to understand what the robot is planning to do and still have the flexibility to change that based on his needs and without requiring to stop the application or perform complex programming phases.

In this section, multi-modal interfaces implemented through the use of the Microsoft HoloLens are presented and evaluated with a user study. The input modalities include head and hand gestures, as well as speech commands to interact with the robot plan.

6.2.2 Method

In Figure 6.15 is represented the overall system architecture. The focus of this section is the the input management module, which provides a command interface to collect the input feedback from the user on the current robot plan, in order to change or agree with it.

In order to enable a flexible interaction, it is important to provide the user with an input interface which allows him to change or confirm the robot plan displayed in AR. In this section, different input modalities based on speech and gestures are proposed and evaluated. In order to evaluate these methods, as soon as the robot plans a new motion, it shows this latter to the user as described in the previous chapter. Before executing the new trajectory, the robot waits for the user input in order to change or confirm the displayed target and planned path. If the user does not provide any input within 2 seconds from the requested input, the robot will execute the trajectory displayed in through AR overlay. In Figure 6.16 is represented the flow chart of the process and the communication involved between the different components.

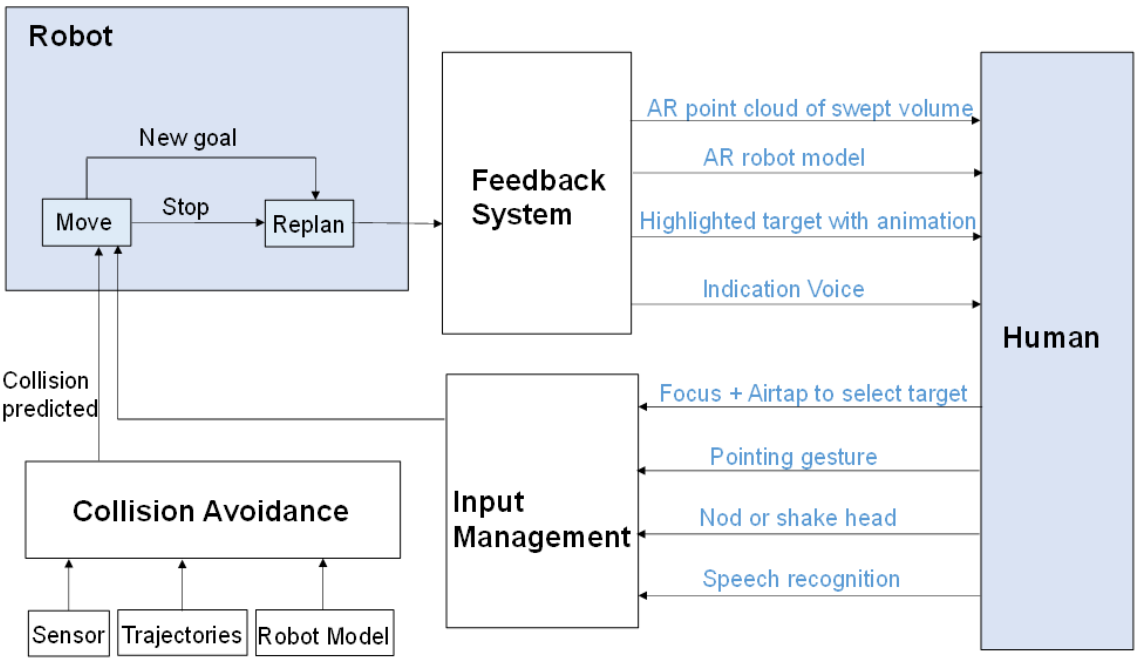


Figure 6.15: The overall architecture of the system. In this section is described the input management module, which is responsible to display the virtual UI to the user and detect the multi-modal inputs. In this way, the user is able to accept or change the current robot plan.

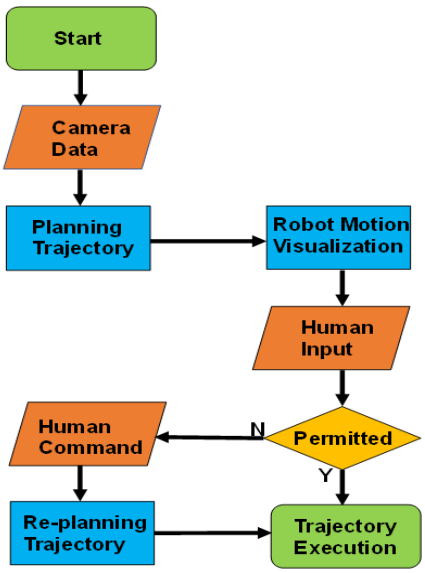


Figure 6.16: Flowchart of the communication involved in the interaction system developed. The robot provides information about its planned motion to the human, who can then deny or confirm the execution of the trajectory. If the user denies the robot planned motion, the manipulator waits for a further human input to select the next target. If the user does not provide any feedback, the robot executes autonomously the planned motion.

Speech

In order to change or confirm the robot plan, speech is a natural way for the user to communicate his need. The HoloLens headset used to display the robot motion is equipped also with a microphone for voice support. Predefined words and sentences can be recognized and mapped into commands. When the robot creates a new plan, it shows its motion to the user and communicates that using synthesized speech as well. The sentence "Moving to Target One/Two" is used to communicate to the human that the robot found an executable trajectory towards a workpiece and waits for the user's confirmation. The user can simply reply with "Yes" or "No" voice commands to confirm or deny the execution of the planned motion. If the planned trajectory is rejected by the human, the robot asks for a target communicating the sentence "Select target". The user can then select the desired goal using the input command "Target one/two". Once the command is received and processed, the robot shows the new trajectory and provides a speech-based feedback related to the execution of the new trajectory.

Gestures

Different gesture-based inputs to change or confirm the current robot motion or to make the robot replan towards a new goal are proposed. The HoloLens headset provides the recognition of predefined gestures such as the Air Tap. A head tracking functionality is also available, which allows the detection of the user's head orientation using IMU and cameras. The following input methods have been developed: focus + AirTap, poiting gestures, nod/shake head movements.

The focus + AirTap method relies on the basic input functionality provided by the HoloLens headset. The human's gaze is tracked in order to get the user's focus of attention. Once the focus is posed on a object, the user can select it using the AirTap, which is a predefined gesture involving the movement of the index finger. However, even if this gesture interface is easy and simple to use, it requires the user to carefully focus his gaze to the desired object and perform the AirTap movement in a way that can be detected by the HoloLens. For this reason a simpler selection input is also proposed, which is the pointing gesture. The direction of user's finger when is pointing is extracted from the point-cloud information coming from the 3D cameras around the workspace. The area in front of the operator is set to be the area of detection. The points lying in this area are seen as the spatial occupancy of the human arm. The direction of this latter is computed using the least squares method and used to detect which part is selected. Figure 6.17 shows a user using the pointing gesture to select a target.

In order to accept or deny the robot plan, a method relying on the use of the head tracking to detect nod/shake gestures is also proposed. The head tracking provided by the HoloLens is used to get the orientation of the user's head. The nodding and shaking gestures are detected by monitoring the changes in orientation of the head pose, as well as its frequency and speed. The parameters to detect

Figure 6.17: The user performs the pointing gesture input in order to select a target. The direction of the pointing is extracted from the data coming from depth cameras and used to detect the workpiece selected by the user. This is used as target for the robot, which then displays the motion to reach it once a collision free trajectory is found.

these movements have been set after experiments on user's performing the two head gestures.

6.2.3 Experiments

In order to evaluate the system, a test scenario with a group of 12 users with no experience in robotics and AR has been considered. The test cases have been designed to examine the effectiveness of the system and collect a feedback about the interaction modalities.

In the test, the overall system and the user perception while interacting with the manipulator have been evaluated. The users had to fill out a questionnaire about the general interaction experience with the system. This includes the main functionality, accuracy and difficulty to learn and use it. In particular, the different input modalities have been evaluated in different runs, in order to collect results about the user's preference and which one is judged as the most intuitive and easy to use. The different input methods were briefly explained and performed by each tester. At the end of all the tests, the users had to rate 10 sentences using a 5-point Likert scale from "Strongly Disagree" to "Strongly Agree":

1. The system improved your understanding of the spatial occupancy of the planned motion of the robot.
2. The system improved your understanding of the next robot goal.
3. The head movement input is effective and convenient to command the robot.
4. The speech recognition input is effective and convenient to command the robot.
5. The pointing gesture input is effective and convenient to command the robot.
6. The focus and AirTap gesture input is effective and convenient to command the robot.
7. The system improved your working efficiency by changing the robot goal by voice and gestures.
8. It is safe to work in collaboration with the robot without the communication system.
9. It is safe to work in collaboration with the robot with the communication system.
10. The system improved your feeling of safety working along with the robot.

In this section the focus in on the sentences 3-7, since the others were related to the robot feedback system and have been already analyzed in the previous chapter.

Sentences 3-7 concern the input interfaces provided to the user in order to deny or agree with the robot motion. As reported in Figure 6.18, the test participants found all the proposed input methods as easy to use and didn't have any issue in sending commands to the robot during the experiments. The preferred methods were the pointing gestures and the speech-based interface, which the users found very intuitive and effective.

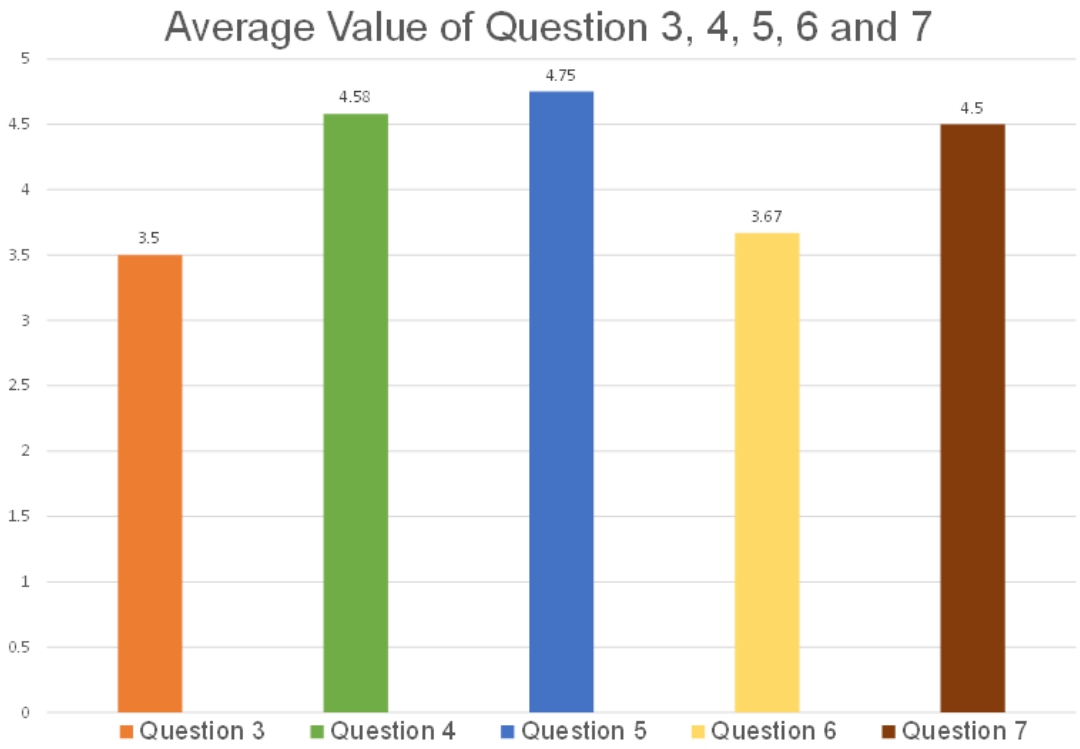


Figure 6.18: The results of the questionnaire regarding the input interfaces to command the robot.

6.2.4 Conlcusions

The system proposed in this section aims at improving the interaction with a robot in close HRC. The evaluation tests involving unexperienced users showed that the designed input modalities were intuitive and easy to use. A survey about the input interfaces provided to the human, such as gestures and speech, has been conducted. Regarding the proposed methods to agree or force a change in the robot plan, the evaluation questionnaire showed that the users found the pointing gestures and the speech-based interfaces as very intuitive and effective methods to give commands to the robot.

The system developed could be improved with the future enhancements in the AR technology, in order to provide a better field of view for the visualization of the virtual overlay. The use of localization of source and source tracking for the speech commands could also improve the usability of the system in manufacturing scenarios with multiple users and noisy environments.

6.3 Multi-Microphones Speech Interface

In this section is introduce the motivation and proposed approach for the development of a multi-microphones speech interface for HRI applications. Speech is a very intuitive channel of communication, but usually very affected by noise in manufacturing scenarios and also quite hard to integrate in different setups. A modular system using array of microphones, can enhance the robustness of the commands detection and also enable an easier integration in existing robotics setup, using web-based configuration tools.

Parts of the results presented in this section have been presented in [3].

6.3.1 Motivation

Speech is a convenient hands-free communication channel where humans are already experienced users. It can implicitly create trustfulness between two operators and lead to a comfortable and natural collaborative environment. As stated in existing literature, speech interaction could increase efficiency and improve certain aspects of HRC. Anyway, speech recognition in industrial scenarios presents different challenges: the typical noisy environment can affect dramatically the interaction performance, leading to an unacceptable inaccuracy in the understanding of the uttered intention. In this section, a modular system for robust and natural speech interaction in challenging acoustical environments is proposed and evaluated. The system has been integrated and tested in a realistic HRC scenario in which the acoustic interaction and efficiency have been evaluated. The developed framework focuses on decreasing the requirements in terms of signal-to- noise ratio, providing a methodology to evaluate the naturalness of the interaction and improvements in efficiency. The solution is designed with a modular approach, providing an easy configuration for ROS-based systems. In this way, it allows a simple integration in existing applications and future research projects, where a dual speech-based interaction can increase the overall performance of the HRC.

Considering the possible benefits of a reliable and natural bidirectional acoustic channel, the proposed system moves toward removing the low reliability of speech interaction. The main aim is to add robustness to multimodal HRC through a framework that can infer the speech content, track the sound source in real-time with a beam-forming technique and produce an acoustic output in the form of speech and audio signals to provide feedback to the user. The presented framework focuses on creating an extendable hardware and software platform that provides completely onboard acoustic processing, with low latency and low Intent Error Rate (IER) in a challenging acoustically environment, as represented in Figure 6.19. Common, inexpensive and readily available hardware in combination with state of art open-source implementations for beam-forming and natural language understanding are the backbone of the built and evaluated ecosystem. The system has been integrated in the screw assembly collaborative application

described in the previous sections, evaluating the efficiency improvements brought by the added communication. Two main aspects have been investigated: the required SNR (Signal to Noise Ratio) based on the array layout and the interaction improvements. For these latter, an experiment in which different participants have been asked to perform a collaborative task has been performed.

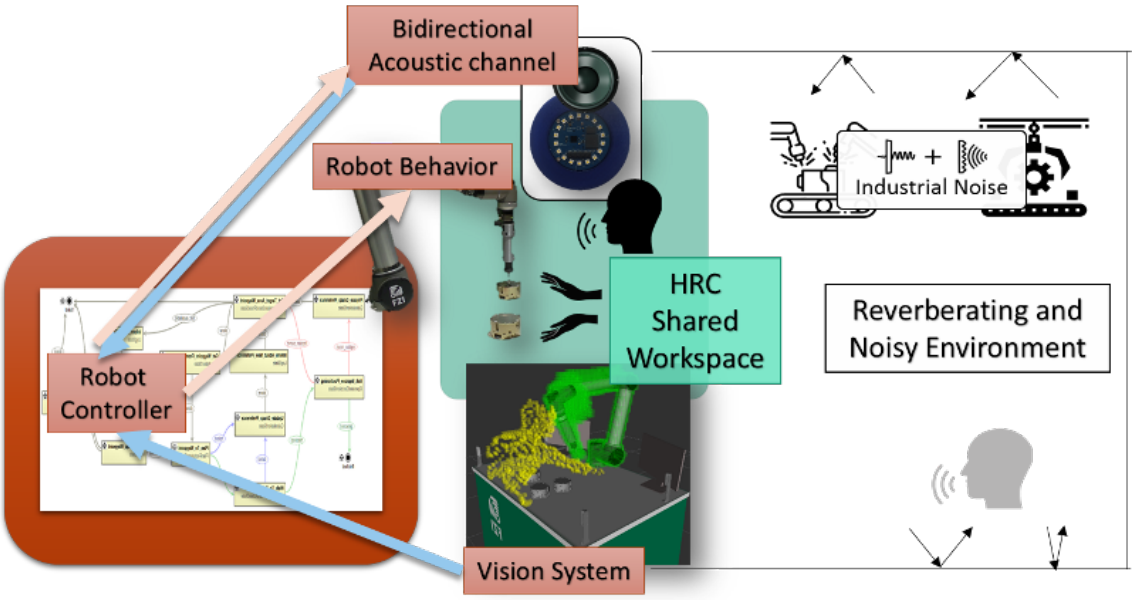


Figure 6.19: The efficiency of HRC applications can increase by adding a reciprocal acoustic channel of communication. The proposed framework focuses on speech interaction in acoustically challenging environments.

6.3.2 Method

In literature, an aspect often under-evaluated is the acoustic channel degradation. This is one of the main reasons why speech interaction is perceived as unreliable by several researchers and in general by end-users [63]. Some works proposed body-worn microphones [14], showing significant improvement in terms of machine comprehension. Anyway, this constraint is considered sub-optimal and a system capable of capture distant speakers is usually needed.

The proposed system is an open and extendable framework that performs real-time and offline sound object tracking and speech analysis from an arbitrary and configurable array of microphones. The framework is agnostic of the underlying hardware. In this work a Raspberry 4 (4x Cortex-A72 1.5 GHz and 4 GB of RAM) and two different microphones array HAT (Hardware Attached on Top) were used to test and validate the concept. The selected mics arrays were a Matrix Voice (8 MEMS microphones, 7 in a circular shape plus a central one) or a Respekaer of 4 MEMS microphones in a squared shape. Figure 6.20 shows the considered hardware.

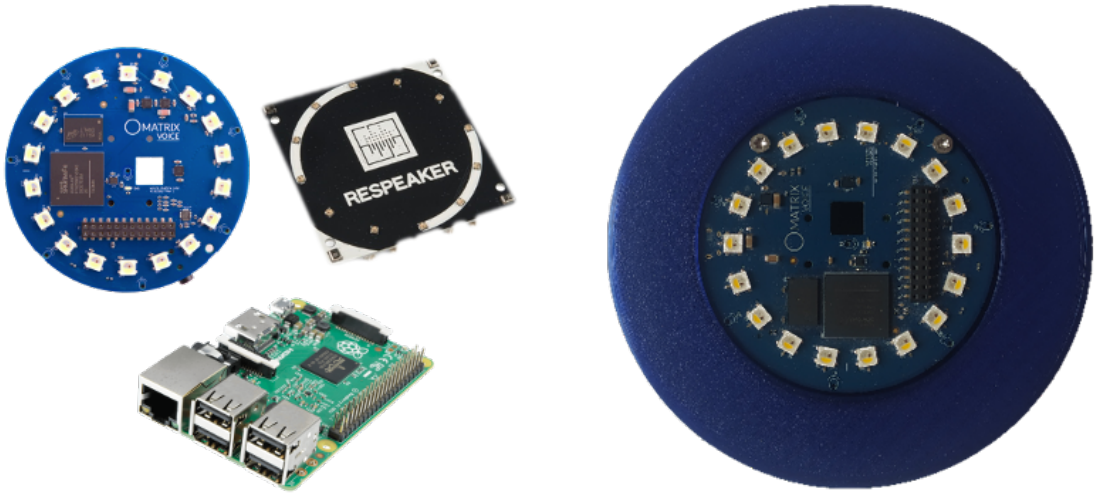


Figure 6.20: Raspberry 4 and microphones HATs integrated and tested with the speech recognition framework. On the right, the prototype design.

However, the input is not limited by particular requirements, the beam forming library integrated in the framework accepts different shapes and number of microphones [43]. Furthermore, USB acquisition systems can be easily integrated if they support a Linux compatible audio USB driver. An external loudspeaker for sound and speech synthesis completes the system.

The selected collaborative application to test the developed system is the shared assembly workspace presented in the previous sections, where the human is in charge of pre-tightening screws and manipulate mechanical parts, while the robot is responsible to tighten the screws with a defined torque. The assembly station has been built as a collaborative environment for investigation of HRC in industrial processes. In particular, the application uses a Universal Robot arm with 6 DOF, which is capable of preventing collisions and re-plan its motion in real-time monitoring the live environment with 3D cameras. The application is implemented in ROS [88] and the task workflow is programmed with a finite state machine behavior defined and executed with FlexBE [98]. The system provided the scenario where a distant speaker needs to interact hands-free with a robot in a noisy and reverberating environment. In the next section, the benefits brought by the bidirectional acoustic channel in the selected application are evaluated.

The developed system is a complex environment of nodes, as represented in Figure 6.21, which use the MQTT[1] and ROS buses for communication, the OS provides the necessary IO (PCM data for audio and typical serial bus for LED control). The audio data are acquired via ALSA (Advanced Linux Sound Architecture) and consumed by an instance of ODAS [43], the algorithm integrated for beam-forming, localization and tracking. The PCM and tracking information are collected by an ad-hoc node that converts the incoming audio stream in chunks suitable for

[1]MQTT is an OASIS standard messaging protocol for IoT.

the Hermes[2] messages (a protocol based on MQTT for personal assistant). The infrastructure backbone is based on Rhasspy[3], a IoT solution for fully offline voice assistant service for many spoken languages implementing the functionalities of the Hermes Protocol.

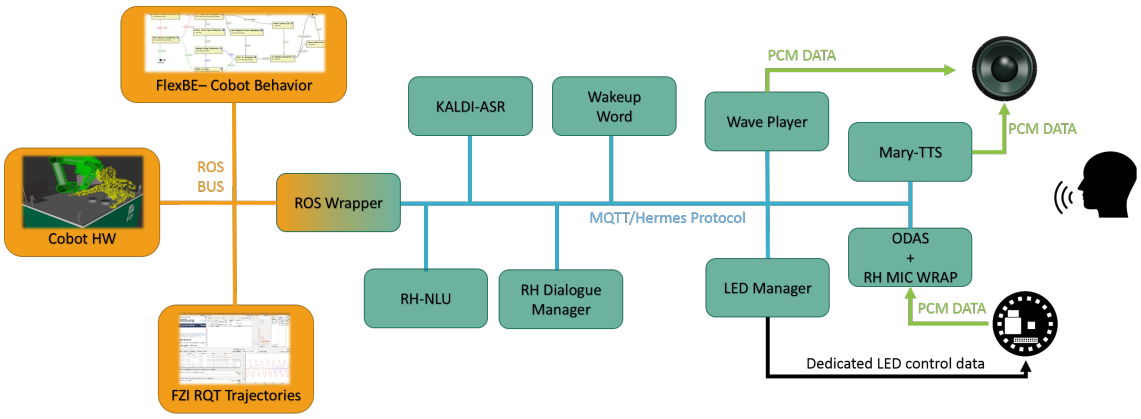


Figure 6.21: Overview of the speech-recognition system architecture and communication between components.

The STT (Speech To Text) engine used is Kaldi [85], a toolkit maintained by the scientific community with a high degree of customization and already available trained model. The TTS (Text To Speech) system is based on MaryTTS[4]. The NLU (Natural Language Understanding) matches the text input to a possible number of intents and extracts an eventual payload. Thanks to the modular nodes architecture provided by Rhasspy, these nodes are interchangeably with other technologies, providing great possibility for flexibility and adaptations. A specific MQTT-ROS bridge allows to store dialogue context information about specific dialogue sessions from ROS clients. This wrapper relies heavily on the Rhasspy dialogue manager node, a state machine that handles the logic of a single basic interaction and sends the appropriate messages to all nodes.

The robotic applications can be interfaced with the proposed framework by using: services (e.g. initiate a conversation), topics (e.g. intent recognized or not) and actions (e.g priority uttering). All these entities are available on the ROS bus. In addition, FlexBE behaviors and states to use these basic interactions have been implemented.

The embedded device is collocated close to the workspace and the board LEDs can be also used to provide an optional visual feedback. The minimal interaction element can be of two types: a bidirectional dialogue (question and answer) or a simple notification from the robot. The interaction can be initiated by the human pronouncing an activation word, or by the robot asking information when in need

[2]https://docs.snips.ai/reference/hermes
[3]https://rhasspy.readthedocs.io
[4]An open source Synthesis platform developed by the DFKI's Language Technology Lab and the Institute of Phonetics at Saarland University.

or simply to notify the user of specific actions (e.g arm movement) or to provide general information (e.g. workpiece completed). Although not investigated in this section, information about the dialogue context is collected and could be used.

To avoid long dialogues, also acoustic and visual feedback complete the HRI. If LEDs are available on the HAT, it is possible to provide information about the state of the dialogue and a basic sound feedback can report, for example, if the intention was understood or not.

Regarding the integration in a HRC application, the screw assembly collaborative task has been analysed, and two main contexts identified: a phase where the robot searches for the next screw within the same or another workpiece and the screwing context where the screw has to be tightened with a specific torque. The design for this specific task is focused in increasing the robot's level of autonomy (e.g., the operator provides information not inferable by the robot) and increasing the level of awareness for the operator during the robot's operations.

In this specific collaborative task, the robot provides information related to the workspace state (e.g., completeness of parts, missing or misplaced screws in one or more workpieces) and arm motion. The robot can also start short interactions, for example: if an is detected, the robot can request a confirmation, asking if the mechanical parts have been finalized correctly and all screws have been properly tightened. If the human operator cannot provide support, the robot will continue and determine by itself the next operation. The framework support also interaction dialogues started by the human, but this situation has not been considered in this work.

At application level the system is configured by defining intents with the interface (web or ini file) provided by Rhasspy. For example, the intent *InsertBeforeTrajectoryPoint*, used for programming trajectories, is decomposed in its lexicon semantic (in most cases verb + adverb/adjective + object) via the Rhasspy slots mechanism. To call this intent, the user could utter the sentence: *add before the item*. The system will emit a ROS *IntentRecognized* message:

- *string context_id*: the id provided by the caller and identifying the active context
- *string original_input*: the original transcribed sentence (in this example:*"add before the item"*)
- *string intent_name*: the identified intent, *InsertBeforeTrajectoryPoint*
- *float32 confidence*: a score between 0 and 1 about transcription confidence determined by the STT
- *string[] pay_load*: a list of strings in the format *["possible_action=add", "position=before", "target=item"]*

With this mechanism, it is possible to define how the application recognizes and parses the intents, making the proposed system agnostic of the end-application as depicted in Figure 6.22 (the FlexBE nodes are generic and not tailored to specific tasks or robots). Another advantage of this approach is to limit the vocabulary size with the benefit of improving the command understanding process. In particular, reducing the number of spurious possibilities and decreasing the computational burden due to the limited dimension of the intent lookup process.

```
[InsertBeforeTrajectoryPoint]
possible_action = (add|insert) {possible_action}
possible_position = (before|before actual|before the actual) {possible_position}
possible_target = (point|trajectory point) {possible_target}
<possible_action> <possible_position> <possible_target>
```

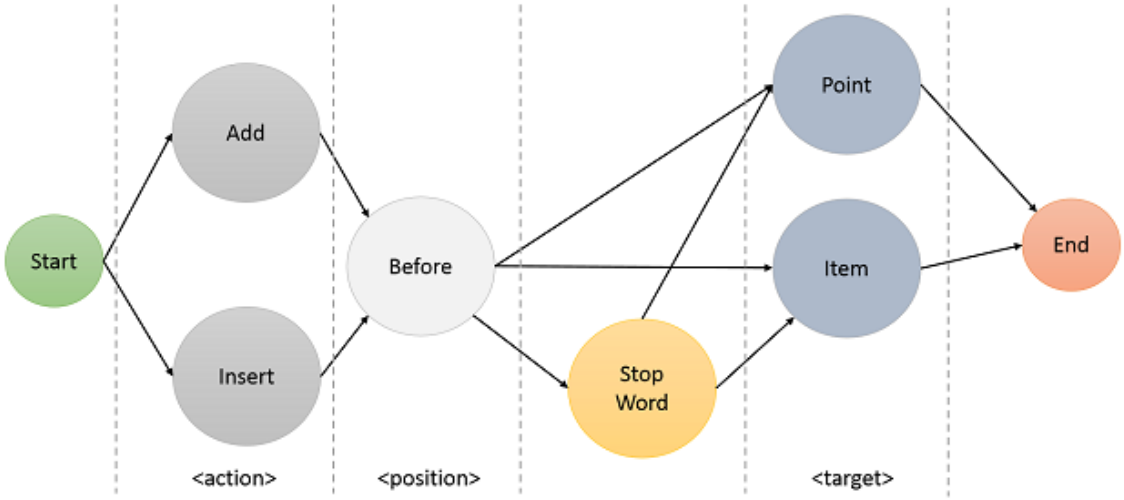


Figure 6.22: Intent definition example and simplified finite state machine used by the intent recognizer.

The realized prototype is an extendable platform to satisfy different user needs through speech recognition, mainly in the field of robotics, but not limited to. In this sense, the system has been developed agnostic of the final application and has focused on an easy way to reconfigure the intent detection. The flexibility in number of microphones and array shapes makes it easy to match the application requirements in terms of noise rejection. Furthermore, the suggested evaluation metrics, presented in the next section, provide an easy method for iterative development of HRC interactions.

6.3.3 Experiments

This section presents the evaluation results of the proposed speech recognition system, with an ad-hoc designed interaction added on the top of the collaborative screw assembly robotic application. Two types of tests have been performed, by replicating the industrial environment conditions, in order to test and validate the following aspects:

- The achievable performance of different hardware (e.g. array layout) by determining the intent recognition rate and the overall dialogue latency. This test has been designed to investigate the relation between a fixed speech source vs different levels of background noise, using various arrays layouts.
- The impact of the proposed solution on HRC metrics, performing user tests with and without deploying the acoustic channel. The experiments have focused on defining interaction metrics to evaluate improvements in the collaborative task.

The results show how speech communication can lead to an increase in productivity and higher acceptance by humans. Furthermore, the performance reachable by different microphones arrays configurations in terms of IER and latency in a noisy and reverberating acoustic environment are reported.

The first round of experiments has focused on determining the dependency between SNR and IER using different configurations. In order to achieve this, a repeatable measurement setup has been designed. Several recorded speech samples have been reproduced by a loudspeaker at 1m distance, the environment noise has been reproduced by another loudspeaker positioned at a distance of 2m with an angle of 90° compared to the active source, as represented in Figure 6.23 on the left.

The level of the source speaker has been fixed manually at 72dB SPL[5] (e.g a loud conversation at 1m in a noisy street)[6] and the experiment repeated with different levels of noise and microphones boards. The speech samples were recorded from 3 females and 4 males native speakers from UK, US and other EU countries, all in the age between 25 and 45 years. The selected commands represented a subset of the final application (7 out of 26 overall commands) and they were uttered three times in different forms. The commands set has been reduced to simplify the processes of voice recording and IER vs SNR test. The subset was chosen to be representative of all possible subcases: single intention, generic intention with semantic payload and intention with numerical payload.

The industrial noise was generated with samples from a BBC archive of recordings[7], where continuous and impulsive sounds were mixed together. The observed IER vs SNR is reported in Figure 6.24 and it is calculated as follow:

[5]Sound Pressure Level, reference $20\mu Pa$
[6]Facts About Speech Intelligibility - DPA Microphone
[7]https://sound-effects.bbcrewind.co.uk/

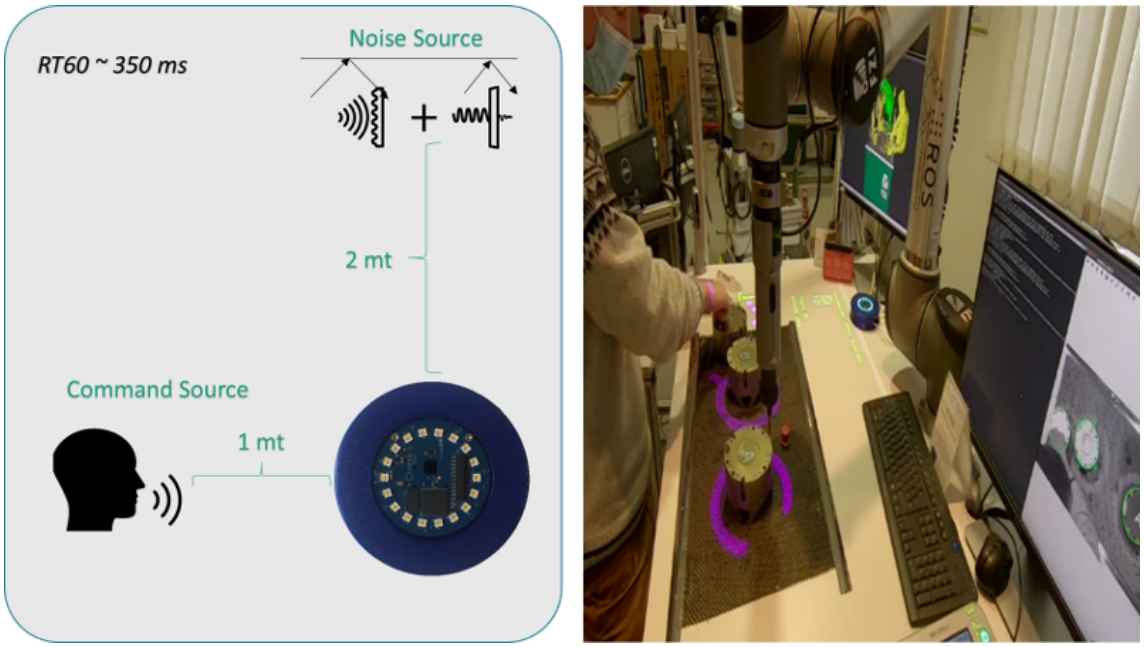


Figure 6.23: Setup used for IER vs SNR test (left). Tester in the collaborative screw assembly environment during the task execution (right).

$$IER = \frac{NR+WR}{N}$$

$$N = NR + WR + R$$

With *WR* intents wrongly recognized, *NR* intents non recognized and *R* intents recognized.

The robustness provided by the beam-forming technique is evident in terms of IER vs SNR. It is possible to observe that, with a larger number of microphones, a consistent stability of the IER against the noise level is achieved. To obtain an IER below 20% using one mic, the minimum SNR is ∼20 dB. The same IER requires about 3 dB of SNR with 8 mics and active beam-forming. An IER of 10% is observed with a SNR of 6 dB or larger.

Another metric investigated was the latency time of the entire acoustic processing. Table 6.1 reports the latency time of each phase of the intent recognition (wake up, transcription after command uttered, intent recognition) for a selected microphones HAT. The noise level can influence the transcription duration by increasing the detection time and introducing a larger variance. This is evident when the noise level get close to the source level. The performed tests showed that the number of microphones don't impact the overall latency.

The second round of tests has focused on the usability benefits. Nine volunteers have been asked to execute the assembly task with and without the use of the acoustic channel. In Figure 6.23 is reported the testing setup on the right.

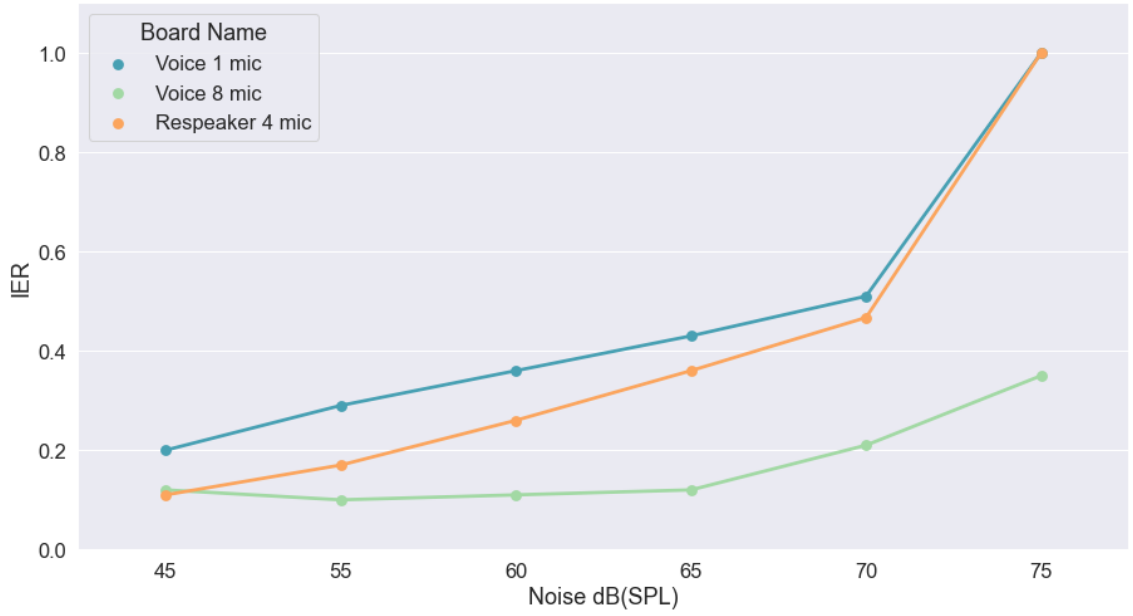


Figure 6.24: SNR vs IER for different microphones boards with sound source fixed at 72 dB SPL.

Talker Level dB(SPL)	Noise Level dB(SPL)	Wakeup Time (sec.)	Speech To Text Time (sec.)	Intent Recognition Time (sec.)
72	45	0.60±0.18	1.44±0.17	0.94±0.05
72	55	0.34±0.13	1.18±0.16	0.88±0.02
72	60	0.33±0.12	1.26±0.20	0.88±0.02
72	65	0.27±0.11	1.18±0.22	0.87±0.02
72	70	0.42±0.13	1.33±0.26	0.87±0.03
72	75	0.28±0.10	1.91±0.22	0.87±0.02

Table 6.1: Latency measurements for matrix voice board vs noise. Reported time values are in the format: median ± standard deviation.

To determine the system effectiveness and efficiency improvements, the metrics proposed by Goodrich et al. [41] have been used. From an operative point of view, *Neglected Time* and *Interaction Time* have been measured to determine the collaboration metrics. Neglected Time NT is defined as the time the robot agent can perform a task before the performance drops under a certain effectiveness threshold. The Interaction Time IT is defined as the time necessary for the human to interact with the robot in order to raise the performance above a defined threshold of effectiveness. The considered collaboration metrics are defined as follow:

- **RAD (Robot Attention Demand)**: the time fraction of the entire task that the operator has to spend with the robot due to a lost of autonomy in executing its task.

$$RAD = \frac{NT}{NT+IT}$$

- **FT (Free Time)**: the time fraction in which the operator can execute other subtasks. In this case it was, as an example, the pre tightening of the screws in a workpiece.

$$FT = 1 - RAD$$

- **FO (Fan Out)**: the maximum number of identical robots that the operator could manage at the same time.

$$FO \leq \frac{1}{RAD}$$

In Figure 6.25 are reported the results without and with the use of the acoustic channel.

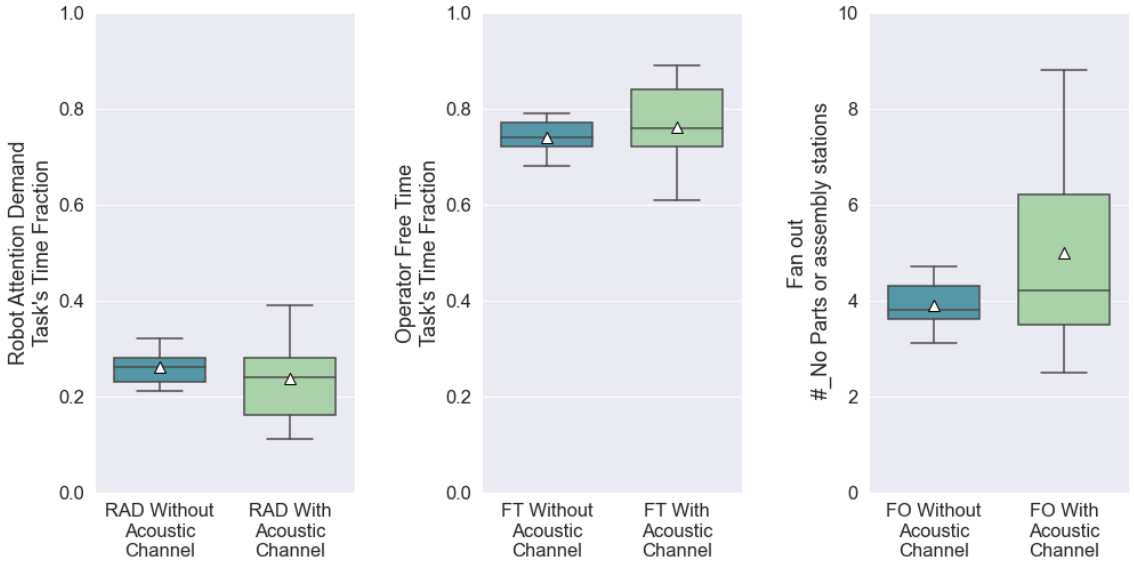


Figure 6.25: Interaction metrics results for the collaborative task execution with and without the use of the acoustic channel.

After the task execution, the testers were asked to fill a questionnaire to determine their point of view in terms of perceived improvements in the interaction with the robot. The survey was divided in two parts: the evaluation of the task with and without the use of the acoustic channel (8 questions) and the general improvements perceived (3 questions). The results are reported in Figure 6.26 and Table 6.2.

In terms of interaction, the determined metrics didn't show a significant increase in the average performance: the RAD decreased of about 3%, the FT median increased by 5% and the FO average increased by one additional robot per user.

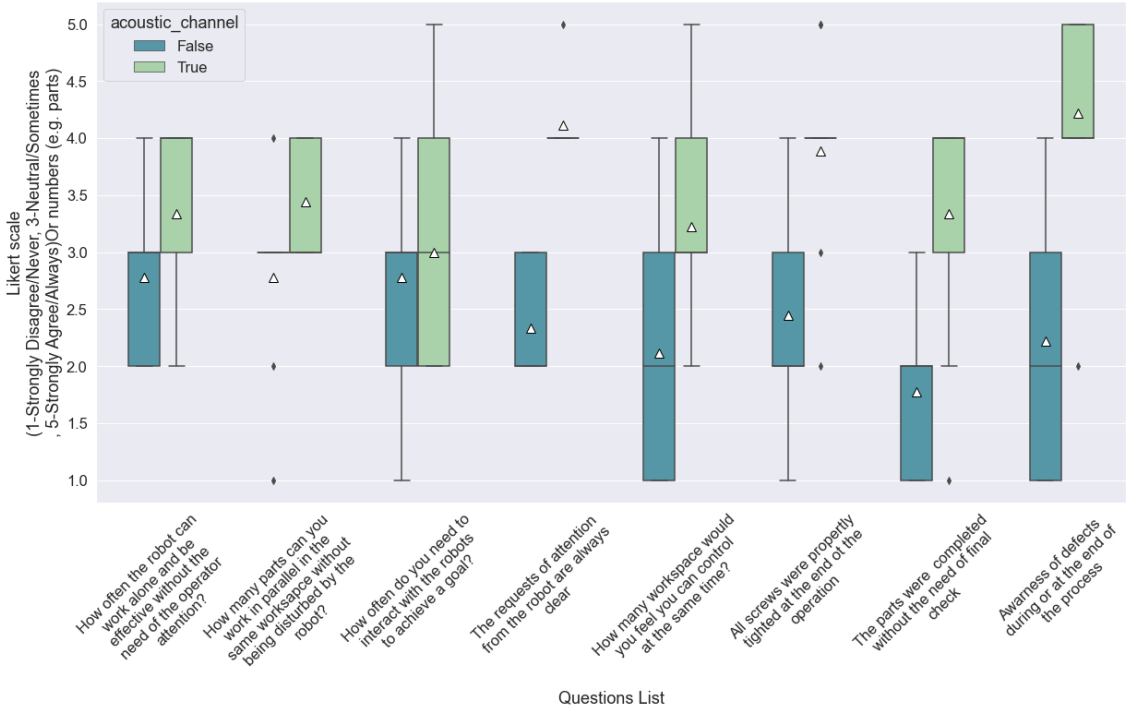


Figure 6.26: Questionnaire results related to the task execution without and with the use of the acoustic interaction.

Anyway, what is evident from Fig. 6.25 is the larger variance in all cases, with a tendency in improving the performances. In all cases at least 50% of the tests presented a performance improvement and 25% a no worse outcome. Furthermore, the best case shows an increase of free time of 10% and a theoretical fan out of three parts in the same assembly station. This large variance is a consequence of some interaction loophole that has been observed during the subjective test. One specific tester exhibited an attitude to answer very quickly (before the robot has finished the sentence) and the system was not able to understand the intent. This has ended in a longer dialogue loop. Such situation can be improved programmatically by avoiding too many repetitions of the same dialogue or in a more elegant manner by using the accumulated information collected by the system. In general, several testers have reported the great benefits of a lengthy spoken explanation during the initial learning phase, but this approach has shown its limits after the first learning and training phase is over.

From a subjective perspective, the testers have shown consistent satisfaction, appreciating the improvement in terms of awareness and perceived safety (on a Likert scale agree (4) to strongly agree (5), as reported in Table 6.2).

The presented framework has to be considered a starting point for better integration of speech-based communication in HRI. It has shown an improved reliability in scenarios affected by noise, with a consistent IER when a mic array beam-former is implemented. The evaluation results showed that a proper system and communication design can improve the overall interaction, making speech recognition

Statement	Min	Q1	Median	Q3	Max
Acoustic interaction had a positive impact on all operations	3	3	4	4	5
Awareness of the robot operation was higher compared to the only visualization of operations	2	3	4	5	5
I felt safer or less concerned in executing the task compared to the case without acoustic interaction	4	4	4	4	5

Table 6.2: Level of testers agreement with the proposed statement.Ratings are: 1.Strongly Disagree, 2.Disagree, 3.Neutral, 4.Agree, 5.Strongly Agree.

reliable in noisy environments. Although some corner cases presented a decrease in performances, it is important to stress out that this result is a consequence of the designed interaction and not of the entire system. The interaction prioritised the learning aspect of the collaborative task, but the approach showed its limits after a couple of interactions, where most of the operators found the repetition of the explanation redundant. This aspect was expected and explicitly ignored for simplicity.

6.3.4 Conclusions

In this section, a bidirectional acoustic interaction system has been designed, built and verified, as well as integrated in an existing collab´orative robotic environment. The proposed framework focuses on a quick deployment of speech interaction on top of existing collaborative robotic applications, using low-cost hardware and open-source software. The onboard processing and limited vocabulary have shown promising results in terms of intent error rate, allowing a reliable and natural voice-based input command system in a noisy environment. Although the system has been tested in an industrial application, it is possible to speculate that fields with similar boundary conditions can benefit from this solution.

Furthermore, it has been proposed an acoustic interaction system to improve multimodal HRI not only from a perceived point of view, but also showing a possibility of increased performances (e.g. number of controllable robots by one operator). The results have shown that a distant speaker (operator) can interact using speech in an acoustically challenging environment and speech interaction can be a reliable element in multi-modal HRC.

The presented system has shown that a proper beam-forming input can dramatically increase the accuracy of the intent recognition under the form of speech. A proper designed interaction can increase productivity by mean of explicit knowledge of the shared workspace state. Finally, the user can benefit from the interaction, feeling safer and more comfortable by having a clear information exchange with the robot.

The approach used in this section was based solely on beam-forming and different array shapes. It has been evaluated the impact of two array layouts and how increasing the number of microphones can dramatically improve the recognition system performances. However, to avoid an increase in hardware costs due to the number of microphones, other possibilities can rely on noise reduction techniques. This approach could also be beneficial in different processing phases (e.g., diminishing latency) by leveraging information extracted from the noise. Lastly, the inclusion of the direction of arrival and the use of accumulated knowledge acquired during the interaction with the user could lead to speech chatbots that learn and adapt the dialogue and other forms of interactions (visual and acoustic) based on the user behavior.

6.4 VR/AR-Based Trajectory Modification

This section introduce the motivation and proposed approach to enable a flexible modification of the robot planned trajectory, through the design of VR/AR-based intuitive interfaces that can be used at runtime by people with no robotic experience.

Parts of the results presented in this section are the outcome of the BMBF research project Sim2log VR.

6.4.1 Motivation

Robot path planners and controllers enable the robot to autonomously create paths and execute them. Anyway, as discussed in the previous chapter, it is usually missing the communication to the user of these motion plans and consequently also the possibility to modify them before or during execution. In particular, when a trajectory is not correctly planned or there is a need to modify it because of changes in the setup, there are no standard approaches to enable the modification of the manipulator path without stopping the robot program and involving complex programming phases. Enabling the visualization and modification of the robot trajectory at runtime it is possible to drastically improve the efficiency in the reprogramming task.

This section presents a method for the modification of the robot trajectory at runtime. The approach focuses on providing an intuitive interface for the inexperienced user, made possible by using interactive virtual objects displayed in VR and AR. The use of a virtual representation of the robotic setup and the communication of the real manipulator joints configurations to the simulated environment, make possible to safely apply modifications to the trajectories during execution and without interfering with the real robot. The possibility to add constraints to the trajectory waypoints modification, allows to avoid unwanted robot configurations.

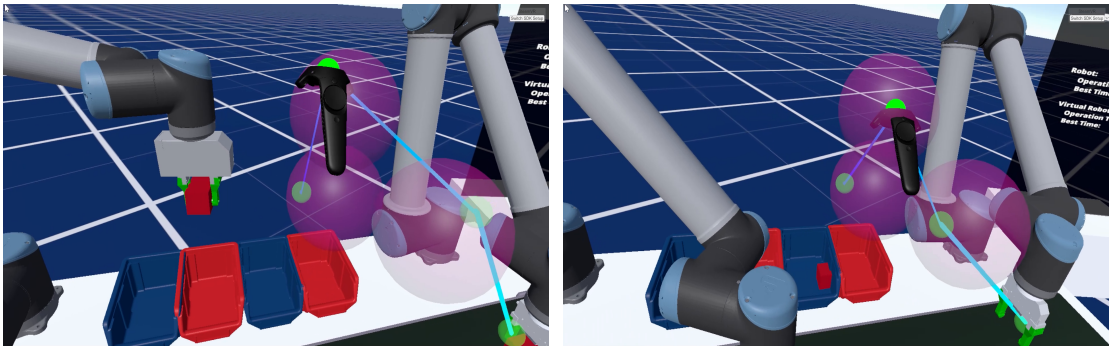
6.4.2 Method

The current trajectory of the robot is displayed in VR through the representation of the TCP waypoints. The information related to the current trajectory executed by the robot is coming from the robot ROS controller, which sends the path message to Unity, which is then used to represent the virtual environment and send back the modification applied to the trajectories. A line between the waypoints is rendered in order to provide the entire TCP path.

Using VR, the waypoints are highlighted using 3D spheres which can be grabbed by the user using the VR controllers. Once the controller is positioned on the interactive marker, it can be taken and moved by just pressing the controller trigger and moving the object to the desired position, as represented in Figure 6.27. Once the trigger is released, the new waypoint pose gets recorded and stored in order to update the TCP path, which is gonna be then executed immediately by the real robot.

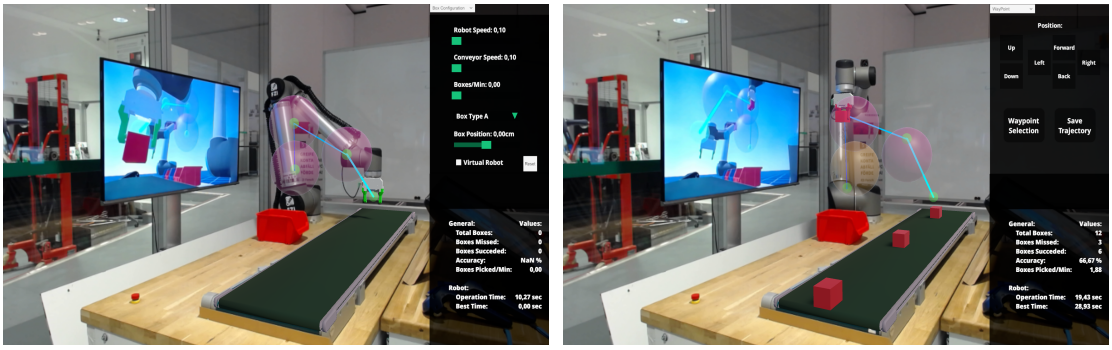
In order to make the user clearly understand the current robot motion for the displayed path, the real robot joints information is sent from ROS to the virtual environment. This allows to display in the simulation environment a virtual representation of the robot which moves accordingly to the real one. In Figure 6.27 it is possible to see the virtual robot which configuration is displayed deploying the real robot joints values.

In a similar way, the virtual interface proposed for the runtime path modification is implemented in AR, giving the user the opportunity to modify, check and adjust the robot trajectory using a tablet or smartphone. For this method, a UI to apply modifications of the path in the x, y and z direction has been developed, as depicted in Figure 6.28.



(a) Visualitation of trajectory waypoints. (b) Modification of trajectory waypoint.

Figure 6.27: Visualization and modification of trajectory waypoints in VR. In (a) the waypoints and the interactive markers to modify them are displayed. In (b) the position of one waypoints get modified using the VR controllers and the path gets immediately updated and displayed.



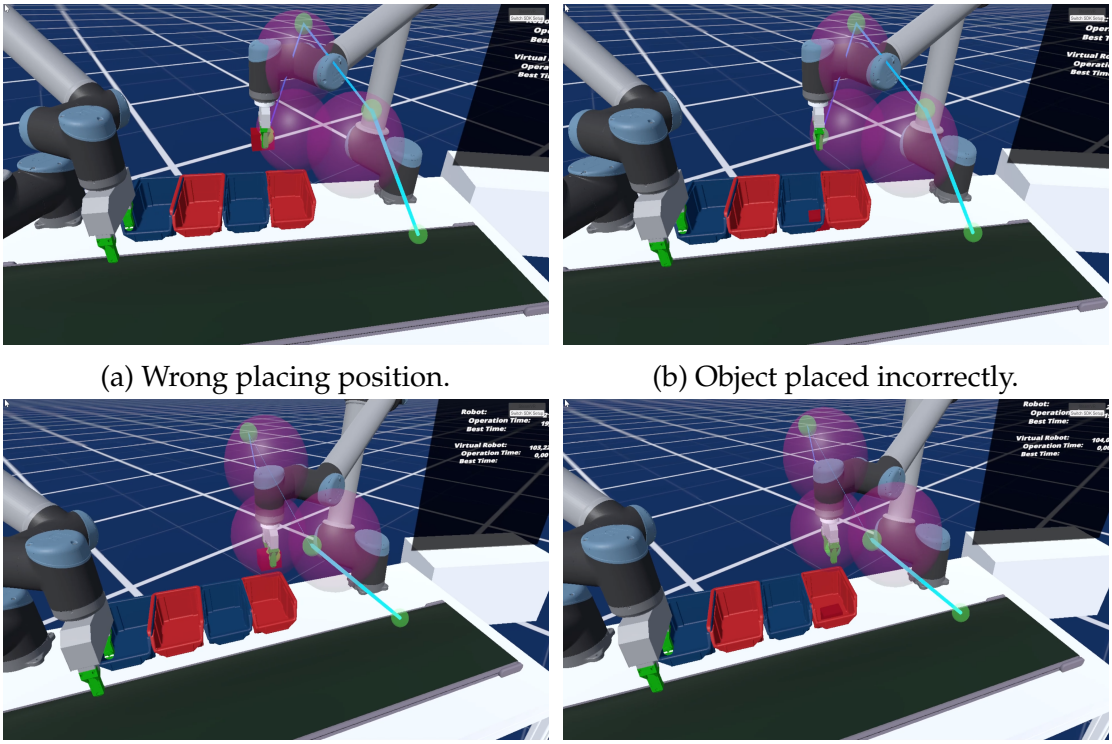
(a) Visualization of trajectory waypoints. (b) Modification of trajectory waypoint.

Figure 6.28: Visualization and modification of trajectory waypoints in AR. In (a) the waypoints and the UI to modify them are displayed. In (b) the position of one waypoints get modified using the UI buttons and the path gets immediately updated and displayed.

In order to avoid unwanted configurations, it has been implemented a way to setup the modification boundaries, in order to prevent unexperienced user to apply undesired changes and unacceptable modifications. The maximum distance allowed for the adjustment of a waypoint can be set and then represented as a transparent sphere with size of the specified radius. When the user tries to move the interactive marker representing the trajectory waypoint outside of its allowed boundary, it will be forced to stay in the bounding sphere, keeping the closest allowed position.

6.4.3 Experiments

To validate the proposed method to adapt the robot motion during runtime, a pick and place scenario in which the robot has to pick parts from a conveyor belt and place them in a bin has been deployed. The trajectory executed by the robot has been set to release the object in a wrong position as shown in Figure 6.29 (a) and (b). The user had to understand the incorrect trajectory executed by the robot and fix it deploying the VR-based interface to adapt the robot motion. As shown in Figure 6.29 (c) and (d), the user could modify easily and quickly the last waypoint of the robot trajectory to enable the robot to place the object in the correct position in the bin, without the need to stop the application or interfering with the real manipulator.



(a) Wrong placing position.

(b) Object placed incorrectly.

(c) Correct placing position.

(d) Object placed correctly.

Figure 6.29: In the experiment test for the runtime modification of the robot trajectory using the proposed VR interface, it has been considered a wrong trajectory definition for which the robot is not able to correctly place an object in a bin (a). Deploying the virtual objects displayed, it is possible to update the path to correctly place the object in no time, having the real robot to react immediately to the modification.

6.4.4 Conclusions

The proposed method showed that it is possible to enable a quick and intuitive way to adapt the robot motion during runtime, without the need of programming phases or experts. The visualization of the robots waypoints and path, enables the understanding of the current planned trajectory and the use of interactive markers allows the modification of the trajectory by just moving the virtual objects using the VR controllers or displayed UI buttons in AR. In this way, it is possible to reprogram the robot at runtime without the need to stop the application, improving the efficiency of the reprogramming phases. The test performed, showed that it is really easy and quick to modify for example the target position of the end-effector for a pick-and-place task. The system could be extended to enable the modification of single robot joints, as well as possible limits, in order to allow a more flexible adaptation of the robot configuration.

6.5 VR-Based No-Go/Safety Areas

This section introduces the motivation and proposed approach to enable the definition of no-go and safety areas for the robot during execution. The proposed method uses the VR representation of the workspace in order to define such zones at runtime, using intuitive interfaces and virtual markers.

Parts of the results presented in this section are the outcome of the BMBF research project Sim2log VR.

6.5.1 Motivation

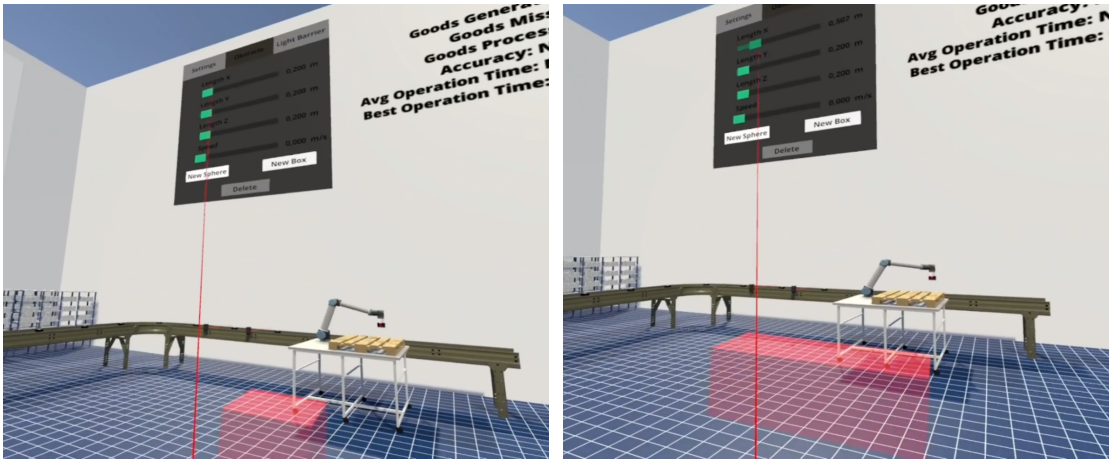
During a close collaboration between human and robot, the user might need to specify areas in which the robot needs to slow down or stop, based on the task progress or in case of unexpected failures. This usually involves expensive tools and also the need of complex programming phases in order to define such zones and the corresponding robot behavior. The possibility to handle this situation in a flexible way at runtime, make it possible to avoid the complete stop of the application or the reprogramming of the robot path execution. In this section, a VR-based method to specify no-go and safety area is presented, which enables the use of an intuitive interface to design no-go and safety areas through the use of virtual markers. The communication of this information to the component responsible to control the robot, allows the real manipulator to react immediately to the new workspace constraints, without the need to interrupt the task execution. The UI buttons in the virtual environment, allows to easily adapt the constraints at runtime and define the desired robot behavior in the specified areas.

6.5.2 Method

In this section is presented a method to define no-go and safety areas in which the robot needs to stop or slow down, enabling the definition of these constraints ar runtime during execution.

VR allows the user to have access to a virtual representation of the real environment, enabling the possibility to define workspace constraints in an interactive way and without interfering with the real robot during the execution of its tasks. The direct communication between VR and ROS, enables the communication of the constraints defined in the virtual environment to the motion executor responsible to handle the execution of the robot trajectories.

A UI represented in VR enables the user to define specific areas in the workspace using basic geometric shapes such as boxes or spheres. The size of the created object can be adjusted easily using sliders to specify the length in the x, y and z directions as depicted in Figure 6.30. In a similar way, the maximum speed that the robots needs to keep when entering in the defined area, can be specified. A value equal to zero means that the robot should stop as soon as it enters the zone. The defined object can be moved to the desired position by just using the VR controllers.



(a) Definition of box-shaped area. (b) Object size adjustment.

Figure 6.30: Definition of a box-shaped area in which the robot needs to reduce its maximum speed. In (a) the zone object is created and displayed in the virtual environment. In (b) the size of the object and the speed limitation value can be adapted using sliders in the virtual UI.

The defined areas are directly communicated to ROS, using a direct communication. In ROS, the environment state is defined and updated using the MoveIt Planning Scene tool. The URDF file containing the environment and robot descriptions is used to keep track of possible collisions with the designed zones. As soon a no-go or safety area is defined, a component is responsible to convert

the message to data compatible with the MoveIt planning scene, updating the environment with the new defined objects. Doing so, the system is able to notify possible collisions between the robots and the created zones.

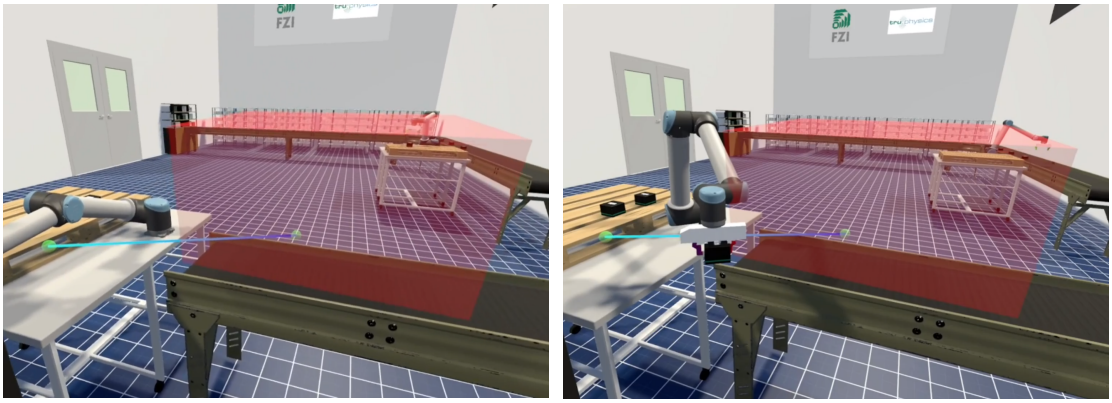
A node is responsible to collect collisions information from the MoveIt Planning Scene in order to handle the robot trajectory execution communicating with the Motion Pipeline which interface directly with the manipulator controller. When a collision with an object is detected, the maximum speed allowed is retrieved and the current trajectory execution stopped to trigger a new execution with the needed speed constraint. In the same way, as soon as the robot exits from the colliding area, a notification is sent to enable the execution with the nominal speed parameters.

6.5.3 Experiments

To validate the proposed method to define no-go and safety areas in which the robot needs to slow down, a pick and place scenario in which the robot needs to place parts on a conveyor belt has been considered. In the designed scenario, the user had to create a safety zone in which the robot needed to slow down to 0.1 m/s when its TCP was reaching the zone above the conveyor belt. As shown in Figure 6.31, the user was able to define a box-shaped area to cover the section of the robot path located on top of the conveyor belt. The speed limitation in the define zone has been set correctly to the allowed maximum value. As depicted in Figure 6.31 (b), as soon as the robot approached the conveyor belt and entered the define safety zones, it updated the execution maximum speed and slowed down.

6.5.4 Conclusions

The proposed method showed that it is possible to enable a quick and intuitive way create safety zones in which the robot needs to slow down during runtime without need of programming or stop the application in order to interact in the workspace. The virtual representation of the real environment and real robot configuration in VR, enables the definition of the needed areas without interfering with the real robot in a safe and efficient way. The test performed, showed that is possible to define really quickly a safety area in which the robot needs to slow down, having the robot reacting to defined virtual objects immediately without the need to stop the application. The system could be further improved to allow more complex zone shapes and to include interfaces to easily include different thresholds based on the distance from objects or humans.



(a) The safety area is positioned where the robot needs to slow down.

(b) As soon as the robot enters the zones, the execution speed is reduced.

Figure 6.31: Test for the evaluation of the definition of a safety area in which the robot needs to slow down. In (a) the defined box is placed above the conveyor belt in the needed position. In (b) is represented the robot entering the defined zone, as soon as the robot gets in contact with the object, the execution speed is updated to slow down the manipulator.

6.6 Summary

In the previous chapter have been presented a way to represent robot intentions to the user in order to enable a better understanding of the robot plan for a better interaction and improved trust in the collaboration. Making the robot intention transparent to the worker, it is important to provide this latter with the possibility to modify the communicated manipulator motion, depending on the user needs in the workspace. Methods to allow this at runtime during execution can improve the efficiency in the task completion and enable a more flexible interaction.

In this chapter, novel interfaces to change or deny the robot plan, as well as modify the robot planned trajectory and workspace constraints at runtime, have been proposed. In particular, multi-modal interfaces based on speech and AR/VR have been designed to enable a quick and intuitive way to interact with the robot plan without the need to stop the application or interfere with the robot motion and reducing the reprogramming time.

In the next chapter, a summary of this thesis is presented, as well as conclusions and an outlook for possible future work.

7 Conclusions and Outlook

This chapter provides the final conclusions for this thesis. A summary of the achieved results and contributions is presented. Finally, it is presented an outlook on the challenges in HRI and possible future research directions.

7.1 Summary of the Approach

In this thesis, different key topics in the field of HRI have been discussed. In particular the proposed approach to enable a more efficient and flexible HRI in different aspects can be summarized as follow:

- Robot programming is the first step needed to enable the robot to autonomously perform a task. The proposed approach focuses on the use of VR to enable the offline definition of programs for complex robotic systems in a virtual environment and without the need of programming knowledge. AR can then be used to validate the programs in the real world before exporting the defined trajectories for the execution on the real manipulator. Furthermore, gesture-based approaches for intuitive and flexible robot have been proposed, which focus on the ease of use through the deployment of visual feedback and different levels of robot autonomy.
- Safety is another critical aspect, since the robot needs to be able to work closely to humans and avoid collisions. In this thesis, methods to evaluate the safety effects of external control authorities, such as collision avoidance components, have been proposed. Furthermore, metrics and test routines to evaluate the additional efficiency benefits introduced by such systems have been proposed and evaluated for a GPU-based collision avoidance and online replanning component.
- Robots should communicate their intents to the human, in order to enable an efficient HRI and increase trust. In this thesis, multiple channels of communication to enable a transparent understanding of the robot intent and motion have been proposed. In particular, the combined use of acoustic signal and speech, together with AR-based visualizations of the swept volume of the robot trajectory, enable the user to easily understand the volume that the robot will occupy in the execution of a path, improving the ergonomics in the interaction.

- Once the user has a clear understanding of the robot plan, this latter should be easily adaptable or change it based on the worker needs. Multi-modal interface to modify the robot plan at runtime have been proposed, in order to enable a flexible adaptation of the robot plan execution without the need to stop the application and without the need of complex programming.

7.2 Summary of the Contributions

In this section, the main contributions of this thesis are summarized. In particular, the key achievements in the following HRI fields are listed:

- **Programming, simulation and control of complex robotic system for non-experts:** The main contribution is the design and development of a VR-based programming framework to enable the definition and simulation of complex robotic systems for users with no programming experience. The intuitive interfaces proposed, allow to quickly and easily define robot programs and test them in VR. Furthermore, thanks to the system architecture, the defined robot motions can be further evaluated in the real setup using AR and then immediately exported on the real hardware for the execution on the real robots without the need of further programming. This work contributed also in the design of gesture-based robot teleoperation interfaces, leveraging the use of visual feedback to enable inexperienced users to perform complex control tasks.
- **Safe and efficient collision avoidance:** The key contribution is the definition of a general system design and guidelines for the use of external control authorities in safety critical applications. Metrics and test routines to evaluate the safety effects introduced by such systems can be used as benchmarks for HRC scenarios. External control authorities, such as the considered GPU-based collision avoidance system, can also be deployed to provide efficiency benefits to the application. This is usually hard to estimate because of missing evaluation standards. The key contribution of this thesis in this field, is the design and definition of metrics and test routines to benchmark the evaluation of efficiency benefits introduced by non-safe external control authorities in a collaborative scenario.
- **Communication of robot intention and motion in a multi-modal manner:** Robotics systems usually lack in the communication of plan and motion information, in particular for robotic manipulator without any anthropomorphic features. The key contribution of this thesis is the design and implementation of multi-modal robot feedback systems to enable the communication of robot intents and motion information to the user. The swept volume information of the robot trajectory, displayed in AR, enables the user to intuitively understand the volume that the robot will occupy while executing a specific motion, improving the ergonomics and trust in the interaction.

- **Multi-modal online robot plan and motion modification:** The lack of communication of robot motion information and interfaces to adapt the trajectory execution at runtime, cannot enable a flexible interaction in which the user can understand what the robot is planning to do, adapting the manipulator plan and trajectories based on his needs. The key contribution of this thesis is the design and implementation of multi-modal interfaces to adapt the robot plan and motion at runtime, without the need to stop the application and additional complex reprogramming phases. The use of speech commands and intuitive VR/AR-based interfaces, enable the user to adapt the trajectory waypoints and workspace constraints immediately and safely using a virtual representation of the environment.

7.3 Outlook

This thesis has introduced key concepts to enable a more efficient and flexible HRI in different aspects. The proposed programming framework, can be used as starting point to further develop and simulate complex systems which can include different sensors and more interactions between various robots.

The improvements in the VR/AR technology could also lead to more a more realistic experience which could bridge even more the gap between simulation and real world.

The proposed multi-modal feedback system could also benefit from such advancements in the VR/AR technology and other communication channels could be integrated for a more immersive and complete feedback experience. For example, haptic interfaces could be added to provide a tactile feedback about the robot proximity or possible events detected such as possible collisions.

List of Figures

List of Tables

Own Publications

This directory lists all publications for which the author of this dissertation is either the first author or has contributed significantly to the publication as coauthor (in the form of problem definition, solution, discussion or experimental evaluation).

[1] Gabriele Bolano, Pascal Becker, Jacques Kaiser, Arne Roennau, and Ruediger Dillmann. Advanced Usability Through Constrained Multi Modal Interactive Strategies: The CookieBot. In *2019 19th International Conference on Advanced Robotics (ICAR)*, pages 213–219, 2019.

[2] Gabriele Bolano, Yuchao Fu, Arne Roennau, and Ruediger Dillmann. Deploying Multi-Modal Communication Using Augmented Reality in a Shared Workspace. In *2021 18th International Conference on Ubiquitous Robots (UR)*, pages 302–307, 2021.

[3] Gabriele Bolano, Lawrence Iviani, Arne Roennau, and Ruediger Dillmann. Design and Evaluation of a Framework for Reciprocal Speech Interaction in Human-Robot Collaboration. In *2021 30th IEEE International Conference on Robot Human Interactive Communication (RO-MAN)*, pages 806–812, 2021.

[4] Gabriele Bolano, Christian Juelg, Arne Roennau, and Ruediger Dillmann. Transparent Robot Behavior Using Augmented Reality in Close Human-Robot Interaction. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1–7, 2019.

[5] Gabriele Bolano, Arne Roennau, and Ruediger Dillmann. Transparent Robot Behavior by Adding Intuitive Visual and Acoustic Feedback to Motion Replanning. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 1075–1080, 2018.

[6] Gabriele Bolano, Arne Roennau, and Ruediger Dillmann. Planning and Evaluation of Robotic Solutions in a Logistic Line Through Augmented Reality. In *2020 Fourth IEEE International Conference on Robotic Computing (IRC)*, pages 422–423, 2020.

[7] Gabriele Bolano, Arne Roennau, Ruediger Dillmann, and Albert Groz. Virtual Reality for Offline Programming of Robotic Applications with Online Teaching Methods. In *2020 17th International Conference on Ubiquitous Robots (UR)*, pages 625–630, 2020.

[8] Gabriele Bolano, Atanas Tanev, Lea Steffen, Arne Roennau, and Ruediger Dillmann. Towards a Vision-Based Concept for Gesture Control of a Robot Providing Visual Feedback. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 386–392, 2018.

Student Works

This directory lists student theses that were advertised and supervised by the author of this dissertation as part of his research. This includes the relevant specification of the problem, discussion of the work, as well as marginal specifications for solution, visualization and experimental evaluation.

[9] Yuchao Fu and Gabriele Bolano. EstablishingaBidirectional Human-RobotCommunicationSystem UsingDifferentCommunication Channels. Masterarbeit, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2019.

[10] Yuchao Fu and Gabriele Bolano. L.I.S.A.: Towards Smooth Robotics Soundscape Acquisition and Acoustic Interaction. Masterarbeit, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2021.

[11] Nikkolas Edi Pujantoro and Gabriele Bolano. Augmented Reality-Based Gesture Interface for Human-Robot Interaction in a Shared Workspace. Bachelorarbeit, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2020.

Bibliography

[12] Syed Mohsin Abbas, Syed Hassan, and Jongwon Yun. Augmented Reality Based Teaching Pendant for Industrial Robot. In *2012 12th International Conference on Control, Automation and Systems*, pages 2210–2213. IEEE, 2012.

[13] Janis Arents, Valters Abolins, Janis Judvaitis, Oskars Vismanis, Aly Oraby, and Kaspars Ozols. Human–robot collaboration trends and safety aspects: A systematic review. *Journal of Sensor and Actuator Networks*, 10(3), 2021.

[14] Tamim Asfour, Rüdiger Dillmann, Nikolaus Vahrenkamp, Martin Do, Mirko Wächter, Christian Mandery, Peter Kaiser, Manfred Kröhnert, and Markus Grotz. *The Karlsruhe ARMAR Humanoid Robot Family*, pages 337–368. Springer Netherlands, Dordrecht, 2019.

[15] Mehrnoosh Askarpour, Dino Mandrioli, Matteo Rossi, and Federico Vicentini. Safer-hrc: Safety analysis through formal verification in human-robot collaboration. In Amund Skavhaug, Jérémie Guiochet, and Friedemann Bitsch, editors, *Computer Safety, Reliability, and Security*, pages 283–295, Cham, 2016. Springer International Publishing.

[16] Ramez Awad, Manuel Fechter, and Jessica van Heerden. Integrated risk assessment and safety consideration during design of hrc workplaces. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–10, 2017.

[17] Michael Baker, Olsen Hall, Holly A Yanco, and Olsen Hall. Autonomy Mode Suggestions for Improving Human- Robot Interaction *. pages 2948–2953, 2004.

[18] Lucian Balan. Real-time 3D collision avoidance algorithm for safe human and robot coexistence. *ProQuest Dissertations and Theses*, NR28251:150, 2006.

[19] Tommaso Lisini Baldi, Giovanni Spagnoletti, Mihai Dragusanu, and Domenico Prattichizzo. Design of a Wearable Interface for Lightweight Robotic Arm for People with Mobility Impairments. In *2017 International Conference on Rehabilitation Robotics (ICORR)*, 2017.

[20] Geoffrey Biggs and Bruce Macdonald. A Survey of Robot Programming Systems.

[21] Wolfgang Birkfellner, Michael Figl, Klaus Huber, Franz Watzinger, Felix Wanschitz, Johann Hummel, Rudolf Hanel, Wolfgang Greimel, Peter Homolka, Rolf Ewers, and Helmar Bergmann. A head-mounted operating binocular for augmented reality visualization in medicine - Design and initial evaluation. In *IEEE Transactions on Medical Imaging*, volume 21, pages 991–997. IEEE, 2002.

[22] Martin Bischoff. ROS#, 2019, https://github.com/siemens/ros-sharp/releases/tag/v1.5.

[23] Sebastian Blankemeyer, Rolf Wiemann, Lukas Posniak, Christoph Pregizer, Annika Raatz, Paul Stief, Jean-yves Dantan, Alain Etienne, and Ali Siadat. ScienceDirect A new methodology to analyze the functional and physical architecture of Robot Programming Using Augmented Reality Nantes , France Reality Intuitive Robot Programming Using Augmented existing products for an assembly oriented product family identification. In *Procedia CIRP*, volume 76, pages 155–160. Elsevier B.V., 2018.

[24] Frank Broz, Alessandro Di Nuovo, Tony Belpaeme, and Angelo Cangelosi. Talking About Task Progress: Towards Integrating Task Planning and Dialog for Assistive Robotic Services. *Paladyn, Journal of Behavioral Robotics*, 6(1):111–118, 2015.

[25] A. Bruce, I. Nourbakhsh, and R. Simmons. The role of expressiveness and attention in human-robot interaction. *Proceedings 2002 IEEE International Conference on Robotics and Automation*, pages 4138–4142, 2002.

[26] Elsa Bunz, Ravi Chadalavada, Henrik Andreasson, Robert Krug, Maike Schindler, and Achim J Lilienthal. Spatial Augmented Reality and Eye Tracking for Evaluating Human Robot Interaction. 2016.

[27] Elizabeth Cha, Maja Matarić, and Terrence Fong. Nonverbal signaling for non-humanoid robots during human-robot collaboration. *ACM/IEEE International Conference on Human-Robot Interaction*, 2016-April:601–602, 2016.

[28] C Chang, M J Chung, and B H Lee. Collision Avoidance of Two Robot Manipulators by Minimum Delay Time. *IEEE Trans. Syst., Man, Cybern.*, 24(3):517–522, 1994.

[29] Sachin Chitta, Eitan Marder-Eppstein, Wim Meeussen, Vijay Pradeep, Adolfo Rodríguez Tsouroukdissian, Jonathan Bohren, David Coleman, Bence Magyar, Gennaro Raiola, Mathias Lüdtke, and Enrique Fernández Perdomo. ros_control: A generic and simple control framework for ROS. *The Journal of Open Source Software*, 2017.

[30] Andrea Corradini and Horst-michael Gross. Recognition for Robot Control. pages 133–138, 2000.

[31] Eva Coupeté, Vincent Weistroffer, Olivier Hugues, Fabien Moutarde, Sotiris Manitsaris, and Philippe Fuchs. New Challenges for Human-Robot Collaboration in an Industrial Context: Acceptability and Natural Collaboration. In *Fifth workshop "towards a Framework for Joint Action", iEEE RO-MAN 2016*, pages 1–4, 2016.

[32] Sebastian Delden, Michael Umrysh, Carlos Rosario, and Gregory Hess. Pick-and-place application development using voice and visual commands. *Industrial Robot: An International Journal*, 39, 10 2012.

[33] Andrzej Drygajlo and Rudolf Haraksim. *Biometric Evidence in Forensic Automatic Speaker Recognition*, pages 221–239. 02 2017.

[34] Scott Durling and Jo Lumsden. Speech recognition use in healthcare applications. In *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia*, MoMM '08, page 473–478, New York, NY, USA, 2008. Association for Computing Machinery.

[35] M Ehrenmann, R Zollner, R Becher, R Dillmann, and Fault Tolerance. Using Gesture and Speech Control for Commanding a Robot Assistant 0. pages 454–459, 2002.

[36] Kerstin Fischer and Franziska Kirstein. Teleoperation for Learning by Demonstration: Data Glove versus Object Manipulation for Intuitive Robot Control. In *6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, number 269959, pages 346–351, 2014.

[37] P Mojiri Forooshani, A Speers, H Wang, and M Jenkin. From ROS to Unity : leveraging robot and virtual environment middleware for immersive teleoperation. *2014 IEEE International Conference on Information and Automation (ICIA)*, (July):932–936, 2014.

[38] Shinya Fujie, Yasushi Ejiri, Kei Nakajima, Yosuke Matsusaka, and Tetsunori Kobayashi. A Conversation Robot Using Head Gesture Recognition as Para-Linguistic Information. pages 159–164, 2004.

[39] Samir Yitzhak Gadre, Eric Rosen, Gary Chien, Elizabeth Phillips, Stefanie Tellex, and George Konidaris. End-User Robot Programming Using Mixed Reality.

[40] Andre Gaschler, Maximilian Springer, Markus Rickert, and Alois Knoll. Intuitive Robot Tasks with Augmented Reality and Virtual Obstacles. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 6026–6031, 2014.

[41] M. A. Goodrich and D. R. Olsen. Seven principles of efficient human robot interaction. In *SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme - System Security and Assurance (Cat. No.03CH37483)*, volume 4, pages 3942–3948 vol.4, 2003.

[42] Scott A Green, J Geoffrey Chase, Xiaoqi Chen, and Mark Billinghurst. Evaluating the Augmented Reality Human-Robot Collaboration System. pages 2–4, 2008.

[43] François Grondin and François Michaud. Lightweight and optimized sound source localization and tracking methods for open and closed microphone array configurations. *Robotics and Autonomous Systems*, 113, 01 2018.

[44] Jan Guhl and Son Tung Nguyen. Concept and Architecture for Programming Industrial Robots using Augmented Reality with Mobile Devices like Microsoft HoloLens. pages 2–5, 2017.

[45] Patrik Gustavsson, Anna Syberfeldt, Rodney Brewster, and Lihui Wang. Human-robot collaboration demonstrator combining speech recognition and haptic control. In *Procedia CIRP*, volume 63, pages 396–401, 05 2017.

[46] M Hasanuzzaman, V Ampornaramveth, Tao Zhang, M A Bhuiyan, Y Shirai, and H Ueno. Real-time Vision-based Gesture Recognition for Human Robot Interaction. *2004 IEEE International Conference on Robotics and Biomimetics*, pages 413–418, 2004.

[47] Andreas Hermann, Sebastian Klemm, Zhixing Xue, Arne Roennau, and Rüdigger Dillmann. GPU-based real-time collision detection for motion execution in mobile manipulation planning. *2013 16th International Conference on Advanced Robotics, ICAR 2013*, 2013.

[48] Andreas Hermann, Felix Mauch, Klaus Fischnaller, Sebastian Klemm, Arne Roennau, and Ruediger Dillmann. Anticipate your surroundings: Predictive collision detection between dynamic obstacles and planned robot trajectories on the GPU. In *2015 European Conference on Mobile Robots, ECMR 2015 - Proceedings*, 2015.

[49] Aaron Holroyd, Charles Rich, Candace L. Sidner, and Brett Ponsler. Generating connection events for human-robot collaboration. *Proceedings - IEEE International Workshop on Robot and Human Interactive Communication*, pages 241–246, 2011.

[50] Dana Hughes, John Lammie, and Nikolaus Correll. A robotic skin for collision avoidance and affective touch recognition. *IEEE Robotics and Automation Letters*, 3(3):1386–1393, 2018.

[51] Soshi Iba, J Michael Vande Weghe, Christiaan J J Paredis, and Pradeep Khosla. An Architecture for Gesture Based Control of Mobile Robots.

[52] Anja Jackowski, Marion Gebhard, and Roland Thietje. Head Motion and Head Gesture-Based Robot Control : A Usability Study. 26(1):161–170, 2018.

[53] Christian Juelg, Andreas Hermann, Arne Roennau, and Rüdiger Dillmann. Fast online collision avoidance for mobile service robots through potential fields on 3D environment data processed on GPUs. In *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 921–928, 2017.

[54] Christian Juelg, Andreas Hermann, Arne Roennau, and Rüdiger Dillmann. Efficient, collaborative screw assembly in a shared workspace. In Marcus Strand, Rüdiger Dillmann, Emanuele Menegatti, and Stefano Ghidoni, editors, *Intelligent Autonomous Systems 15*, pages 837–848, Cham, 2019. Springer International Publishing.

[55] Günther Knoblich, Stephen Butterfill, and Natalie Sebanz. *Psychological Research on Joint Action. Theory and Data*, volume 54. 2011.

[56] Sarah Claudia Krings, Enes Yigitbas, Kai Biermeier, and Gregor Engels. Design and evaluation of ar-assisted end-user robot path planning strategies. In *Companion of the 2022 ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, EICS '22 Companion, page 14–18, New York, NY, USA, 2022. Association for Computing Machinery.

[57] Tin Lun Lam, Hoi Wut Yip, Huihuan Qian, and Yangsheng Xu. Collision avoidance of industrial robot arms using an invisible sensitive skin. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4542–4543, 2012.

[58] Pai-chia Li and Chih-hsing Chu. Augmented Reality based Robot Path Planning for Programming by Demonstration Augmented Reality based Robot Path Planning for Programming by Demonstration. Number December, pages 1–5, 2016.

[59] Songpo Li and Xiaoli Zhang. Implicit Intention Communication in Human – Robot Interaction Through Visual Behavior Studies. *IEEE Transactions on Human-Machine Systems*, 47(4):437–448, 2017.

[60] Ying Siu Liang, Damien Pellier, Humbert Fiorino, and Sylvie Pesty. Evaluation of a Cobot Programming Framework for Non-Experts using Symbolic Planning Representations. pages 1121–1126, 2017.

[61] J I Lipton, A J Fay, and D Rus. Baxter's Homunculus: Virtual Reality Spaces for Teleoperation in Manufacturing. *IEEE Robotics and Automation Letters*, 3(1):179–186, jan 2018.

[62] Lorenzo Lucignano, Francesco Cutugno, Silvia Rossi, and Alberto Finzi. A dialogue system for multimodal human-robot interaction. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*, ICMI '13, page 197–204, New York, NY, USA, 2013. Association for Computing Machinery.

[63] H. W. Löllmann, A. Moore, P. A. Naylor, B. Rafaely, R. Horaud, A. Mazel, and W. Kellermann. Microphone array signal processing for robot audition. In *2017 Hands-free Speech Communications and Microphone Arrays (HSCMA)*, pages 51–55, 2017.

[64] Jim Mainprice and Dmitry Berenson. Motion Planning for Human-Robot Collaborative Manipulation Tasks Using Prediction of Human Motion. In *RSS Workshop*, pages 299–306, 2014.

[65] Eloise Matheson, Riccardo Minto, Emanuele Zampieri, Maurizio Faccio, and Giulio Rosati. Human–robot collaboration in manufacturing applications: A review. *Robotics*, 8:100, 12 2019.

[66] T. Mazzocchi, A. Diodato, G. Ciuti, D. M. De Micheli, and A. Menciassi. Smart sensorized polymeric skin for safe robot collision and environmental interaction. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 837–843, 2015.

[67] George Michalos, Panagiotis Karagiannis, Sotiris Makris, Önder Tokçalar, and George Chryssolouris. Augmented Reality (AR) Applications for Supporting Human-robot Interactive Cooperation. In *Procedia CIRP*, volume 41, pages 370–375. Elsevier B.V., 2016.

[68] S Michas, E Matsas, and G-c Vosniakos. Interactive programming of industrial robots for edge tracing using a virtual reality gaming environment. volume 10, pages 237–259, 2017.

[69] Julie Milovanovic, Guillaume Moreau, Daniel Siret, Francis Miguet, Julie Milovanovic, Guillaume Moreau, Daniel Siret, Francis Miguet Virtual, and Augmented Reality. Virtual and Augmented Reality in Architectural Design and Education To cite this version : HAL Id : hal-01586746 An Immersive Multimodal Platform to Support. 2017.

[70] Mohammad Mehdi Moniri, Fabio Andres Espinosa Valcarcel, Dieter Merkel, and Daniel Sonntag. Human gaze and focus-of-attention in dual reality human-robot collaboration. In *Proceedings - 12th International Conference on Intelligent Environments, IE 2016*, pages 238–241, 2016.

[71] Sebastian Muszynski and Sven Behnke. Adjustable Autonomy for Mobile Teleoperation of Personal Service Robots. 2012.

[72] Bilge Mutlu, Allison Terrell, and Chien-ming Huang. Coordination Mechanisms in Human-Robot Collaboration. In *International Conference on Human-Robot Interaction - Workshop on Collaborative Manipulation*, pages 1–6, 2013.

[73] A Naceri, D Mazzanti, J Bimbo, D Prattichizzo, D G Caldwell, L S Mattos, and N Deshpande. Towards a Virtual Reality Interface for Remote Robotic Teleoperation. In *2019 19th International Conference on Advanced Robotics (ICAR)*, pages 284–289, 2019.

[74] Cherie-Ann O. Nathan, Vinaya Chakradeo, Kavita Malhotra, Horacio D'Agostino, and Ravish Patwardhan. The voice-controlled robotic assist scope holder aesop for the endoscopic approach to the sella. *Skull base : official journal of North American Skull Base Society ... [et al.]*, 16(3):123–131, Aug 2006.

[75] Kai Nickel. 3D-Tracking of Head and Hands for Pointing Gesture Recognition in a Human-Robot Interaction Scenario. 2004.

[76] S K Ong, A W W Yew, N K Thanigaivel, and A Y C Nee. Augmented reality-assisted robot programming system for industrial applications. In *Robotics and Computer Integrated Manufacturing*, volume 61, page 101820. Elsevier Ltd, 2020.

[77] H. Osawa, A. Ema, H. Hattori, N. Akiya, N. Kanzaki, A. Kubo, T. Koyama, and R. Ichise. Analysis of robot hotel: Reconstruction of works with robots. In *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 219–223, 2017.

[78] Zengxi Pan, Joseph Polden, Nathan Larkin, Stephen Van Duin, and John Norrish. Recent Progress on Programming Methods for Industrial Robots. pages 619–626, 2010.

[79] Amit Kumar Pandey and Rodolphe Gelin. A mass-produced sociable humanoid robot: Pepper: The first machine of its kind. *IEEE Robotics and Automation Magazine*, PP:1–1, 07 2018.

[80] Jinesh Patel, Yahu A Choudhary, and Gary M Bone. Fault Tolerant Robot Programming by Demonstration of Sorting Tasks with Industrial Objects. *2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS2017)*, pages 5–7, 2017.

[81] Lorenzo Peppoloni, Filippo Brizzi, Carlo Alberto Avizzano, and Emanuele Ruffaldi. Immersive ROS-integrated Framework for Robot Teleoperation. In *2015 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 177–178. IEEE.

[82] Brian S. Peters, Priscila R. Armijo, Crystal Krause, Songita A. Choudhury, and Dmitry Oleynikov. Review of emerging surgical robotic technology. *Surgical Endoscopy*, 32(4):1636–1655, Apr 2018.

[83] J. Norberto Pires. Robot-by-voice: Experiments on commanding an industrial robot using the human voice. *Industrial Robot: An International Journal*, 32:505–511, 12 2004.

[84] M E Portman and A Natapov. Computers , Environment and Urban Systems To go where no man has gone before : Virtual reality in architecture , landscape architecture and environmental planning. *CEUS*, 54:376–384, 2015.

[85] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011. IEEE Catalog No.: CFP11SRW-USB.

[86] Jürgen Pripfl, Tobias Körtner, Daliah Batko-klein, Denise Hebesberger, Markus Weninger, Christoph Gisinger, Astrid Weiss, Markus Bajones, and Markus Vincze. Results of a Real World Trial with a Mobile Social Service Robot for Older Adults. pages 497–498, 2016.

[87] Kun Qian, Jie Niu, and Hong Yang. Developing a Gesture Based Remote Human-Robot Interaction System Using Kinect. In *International Journal of Smart Home*, volume 7, pages 203–208, 2013.

[88] Morgan Quigley, Ken Conley, Brian P Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. ROS: an open-source Robot Operating System. In *ICRA Workshop on Open Source Software*, 2009.

[89] V S Rao, Communication Engineering, and Der Malsburg. Gesture Based Robot Control. In *2006 Fourth International Conference on Intelligent Sensing and Information Processing*, pages 0–3, 2006.

[90] Rupert Reif and Dennis Walch. Augmented & Virtual Reality applications in the field of logistics. *The Visual Computer*, 24(11):987–994, nov 2008.

[91] Florian Richter, Ryan K Orosco Member, and Michael C Yip Member. Motion Scaling Solutions for Improved Performance in High Delay Surgical Teleoperation.

[92] Chongguanchun Road. Head Gesture Recognition for Hands-free Control of an Intelligent Wheelchair. pages 1–10, 2001.

[93] A Rohacz and S Strassburger. Augmented Reality in Intralogistics Planning of the Automotive Industry : State of the Art and Practical Recommendations for Applications. In *2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)*, pages 203–208, apr 2019.

[94] Nina Rudigkeit, Marion Gebhard, and Axel Gr. An Analytical Approach for Head Gesture Recognition with Motion Sensors. pages 720–725, 2015.

[95] P.E. Rybski, K. Yoon, J. Stolarz, and M.M. Veloso. Interactive Robot Task Training through Dialog and Demonstration. *2nd ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 49–56, 2007.

[96] S Scherzinger, A Roennau, and R Dillmann. Inverse Kinematics with Forward Dynamics Solvers for Sampled Motion Tracking. In *19th International Conference on Advanced Robotics (ICAR)*, page (submitted), 2019.

[97] Philipp Schillinger, Stefan Kohlbrecher, and Oskar Von Stryk. Human-Robot Collaborative High-Level Control with Application to Rescue Robotics. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2796–2802. IEEE, 2016.

[98] Philipp Schillinger, Stefan Kohlbrecher, and Oskar von Stryk. Human-Robot Collaborative High-Level Control with an Application to Rescue Robotics. In *IEEE International Conference on Robotics and Automation*, Stockholm, Sweden, 2016.

[99] Bernard Schmidt and Lihui Wang. Depth camera based collision avoidance via active robot control. *Journal of Manufacturing Systems*, 33(4):711–718, 2014.

[100] Kashyap Shah and Yan Meng. Communication-efficient dynamic task scheduling for heterogeneous multi-robot systems. In *2007 International Symposium on Computational Intelligence in Robotics and Automation*, pages 230–235, 2007.

[101] Pradeep Shenoy, Kai J Miller, Beau Crawford, and Rajesh P N Rao. No Title. Number 3, pages 1128–1135.

[102] Thomas Sheridan. Human-robot interaction: Status and challenges. *Human factors*, 58, 04 2016.

[103] Chathurangi Shyalika, Thushari P. Silva, and Asoka S. Karunananda. Reinforcement learning in dynamic task scheduling: A review. *SN Comput. Sci.*, 1:306, 2020.

[104] Cary R. Spitzer. *The Avionics Handbook*. CRC Press, 2000.

[105] R Stiefelhagen, C Fugen, P Gieselmann, H Holzapfel, K Nickel, and A Waibel. Natural Human-Robot Interaction using Speech , Head Pose and Gestures. pages 2422–2427, 2004.

[106] Leila Takayama, Menlo Park, Doug Dooley, and Wendy Ju. Expressing Thought : Improving Robot Readability with Animation Principles. In *2011 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 69–76. IEEE, 2011.

[107] Jovica Tasevski, Milutin Nikolić, and Dragisa Miskovic. Integration of an industrial robot with the systems for image and voice recognition. *Serbian Journal of Electrical Engeneering*, 10:219–230, 01 2013.

[108] C Tonn, F Petzold, and D Donath. Put on Your Glasses and Press Right Mouse Button: AR-based User Interaction Using Laser Pointer Tracking. pages 201–208, Antwerp, Belgium, 2008. eCAADe, eCAADe.

[109] Nghia X Tran, Hoa Phan, Vince V Dinh, Jeffrey Ellen, Bryan Berg, Jason Lum, Eldridge Alcantara, Mike Bruch, Marion G Ceruti, Charles Kao, Daniel Garcia, Sunny Fugate, and LorRaine Duffy. Wireless Data Glove for Gesture-Based Robotic Control. In Julie A Jacko, editor, *Human-Computer Interaction. Novel Interaction Methods and Techniques*, pages 271–280, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[110] Rudolph Triebel, Kai Arras, Rachid Alami, Lucas Beyer, Stefan Breuers, Raja Chatila, Mohamed Chetouani, Daniel Cremers, Vanessa Evers, Michelangelo Fiore, Hayley Hung, Omar A Islas Ramírez, Michiel Joosse, Harmish Khambhaita, Tomasz Kucner, Bastian Leibe, Achim J Lilienthal, Timm Linder, Manja Lohse, Martin Magnusson, Billy Okal, Luigi Palmieri, Umer Rafi, Marieke van Rooij, and Lu Zhang. *SPENCER: A Socially Aware Service Robot for Passenger Guidance and Help in Busy Airports*, pages 607–622. Springer International Publishing, Cham, 2016.

[111] Annemarie Turnwald, Daniel Althoff, Dirk Wollherr, and Martin Buss. Understanding Human Avoidance Behavior: Interaction-Aware Decision Making Based on Game Theory. In *International Journal of Social Robotics*, volume 8, pages 331–351. Springer Netherlands, 2016.

[112] Unity, 2019, https://unity.com.

[113] S Vasanth, D Gayathiri, and Vinoth Kumar Paramasivam. Control and programming of a robotic arm using gestures and skeleton tracking. 10:33987–33989, 2015.

[114] Camilo Vasquez Tieck, Lea Steffen, Jacques Kaiser, Arne Roennau, and Ruediger Dillmann. Controlling a Robot Arm for Target Reaching without Planning Using Spiking Neurons. *17th IEEE International Conference on Cognitive Informatics & Cognitive Computing*, 2018.

[115] Shamsheer Verma. Hand Gestures Remote Controlled Robotic Arm. volume 3, pages 601–606, 2013.

[116] Federico Vicentini, Mehrnoosh Askarpour, Matteo G. Rossi, and Dino Mandrioli. Safety assessment of collaborative robotics through automated formal verification. *IEEE Transactions on Robotics*, 36(1):42–61, 2020.

[117] Michael Walker, Jennifer Lee, and Daniel Szafir. Communicating Robot Motion Intent with Augmented Reality. pages 316–324, 2018.

[118] W P Wang, General Electric, and K K Wang. Geometric Modeling for Swept Volume of Moving Solids. volume M, 1986.

[119] Atsushi Watanabe, Tetsushi Ikeda, Yoichi Morales, Kazuhiko Shinozawa, Takahiro Miyashita, and Norihiro Hagita. Communicating Robotic Navigational Intentions. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5763–5769. IEEE, 2015.

[120] Tomio Watanabe. InterRobot: A Speech Driven Embodied Interaction Robot. pages 322–327, 2000.

[121] David Whitney, Eric Rosen, Daniel Ullman, Elizabeth Phillips, and Stefanie Tellex. ROS Reality : A Virtual Reality Framework Using Consumer-Grade Hardware for ROS-Enabled Robots. 2018.

[122] Michael T Wolf, Christopher Assad, Matthew T Vernacchia, Joshua Fromm, and Henna L Jethani. Gesture-Based Robot Control with Variable Autonomy from the JPL BioSleeve. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1160–1165, 2013.

[123] P.G. Xavier. Fast swept-volume distance for robust collision detection. *Proceedings of International Conference on Robotics and Automation*, pages 1162–1169, 1997.

[124] Hironao Yamada, Ni Tao, and Zhao DingXuan. Construction Tele-Robot System With Virtual Reality. *2008 IEEE International Conference on Robotics, Automation and Mechatronics, RAM 2008*, 00:36–40, 2008.

[125] Deng Yongda, Li Fang, and Xin Huang. Research on multimodal human-robot interaction based on speech and gesture. *Computers & Electrical Engineering*, 72:443–454, 2018.

[126] Tian Yu, Jing Huang, and Qing Chang. Optimizing task scheduling in human-robot collaboration with deep multi-agent reinforcement learning. *Journal of Manufacturing Systems*, 60:487–499, 2021.

[127] Lingqi Zeng and Gary M. Bone. Mobile robot collision avoidance in human environments. *International Journal of Advanced Robotic Systems*, 10(1):41, 2013.

[128] Yu Zhang and Lynne E. Parker. Multi-robot task scheduling. In *2013 IEEE International Conference on Robotics and Automation*, pages 2992–2998, 2013.

[129] Bing Zhou, Kaige Wu, Pei Lv, Jing Wang, Gang Chen, Bo Ji, and Siying Liu. A new remote health-care system based on moving robot intended for the elderly at home. *Journal of healthcare engineering*, 2018:4949863, Feb 2018.

[130] Kateryna Zinchenko, Chien-Yu Wu, and Kai-Tai Song. A study on speech recognition control for a surgical robot. *IEEE Transactions on Industrial Informatics*, PP:1–1, 11 2016.