POLITECNICO DI TORINO
Repository ISTITUZIONALE

Robots learn to behave: improving human-robot collaboration in flexible manufacturing applications

Original
Robots learn to behave: improving human-robot collaboration in flexible manufacturing applications / Sibona, Fiorella. - (2023 Jun 13), pp. 1-219.

Availability:
This version is available at: 11583/2979885 since: 2023-07-05T07:26:21Z

Publisher:
Politecnico di Torino

Published
DOI:

Terms of use:
Altro tipo di accesso

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

17 October 2023

Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Electrical, Electronics and Communications Engineering
(35^{th}cycle)

Robots learn to behave: improving human-robot collaboration in flexible manufacturing applications

Fiorella Sibona

******

Supervisor:
Prof. Marina Indri

Doctoral Examination Committee:
Prof. Pedro Neto, Referee, University of Coimbra, Portugal
Prof. Maria Letizia Corradini, Referee, Università di Camerino, Italy
Prof. Bruno Siciliano, Università di Napoli Federico II, Italy
Prof. Roberto Oboe, Università di Padova, Italy
Prof. Carlo Novara, Politecnico di Torino, Italy

Politecnico di Torino
2023

Declaration

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

This thesis is licensed under a Creative Commons License, Attribution-Noncommercial-NoDerivative Works 4.0 International: see `www.creativecommons.org`. The text may be reproduced for non-commercial purposes, provided that credit is given to the original author.

Fiorella Sibona

2023

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

To my family and all those who have always believed in me.
To failure, and to perseverance.
To all women who dare to be loud.

Acknowledgements

This is it. My PhD dissertation is now complete and I am left to try and thank all those that made this possible. Where to even start? With my mind, I go back in time and relive these tough but beautiful years. Inevitably going back to when getting a PhD seemed to me like some "mission impossible", simply not for me. But I came to the realization that nothing is *not for you* if you are surrounded by the right people.

I cannot leave Politecnico di Torino without expressing my gratitude to my supervisor, Prof. Marina Indri, for her profound belief in my abilities and work, and for the opportunity she gave me to be part of this university's research community. She played a decisive role in shaping this chapter of my life, supporting my crazy ideas and pushing me to challenge myself, paving the way to what's coming next. Special thanks goes to Dr. Pangcheng David Cen Cheng, or simply David, my study buddy and best colleague and friend. Without you professional and emotional support none of what I have achieved would have been possible. Daily office and lab life by your side has been really fun and it will be hard to find the same somewhere else. Remember I will be always in for a coffee and a chat.

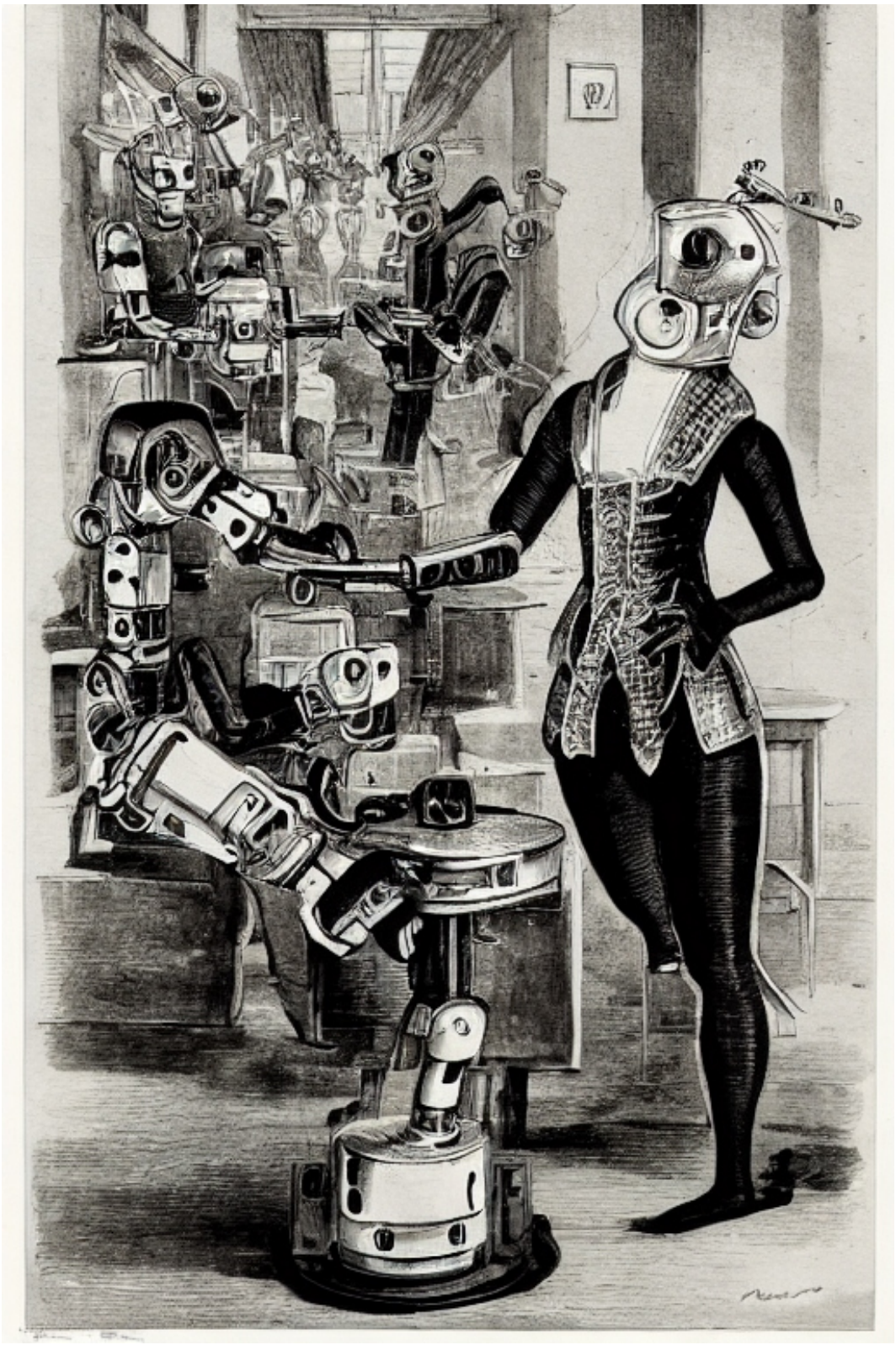
Thanks to (former) Prof. Basilio Bona who first made me get to know robotics, with his passionate teaching, and really represented and inspiration. Many thanks to Luigi Spagnolo and Andrea Galletto for their invaluable support over the years. Thanks to Dr. Stefano Primatesta for helping me, especially when I first approached teaching ROS. I would like to thank the invaluable support and guidance of Dr. Ludovico Orlando Russo, my trusted reference during my MSc thesis development, and mentor and friend in the recent years: thanks to you I have came out of my comfort zone many times. Thanks to Dr. Bushra Anjum inspiring me, and for keeping updated and interested in my career path: thank you for making me feel empowered as a female researcher and as a human.

I want to thank the Politecnico di Torino as an institution, for which I have been a student, a researcher and also a PhD students representative. Thanks to Prof. Eugenio Brusa, Prof. Stefano Grivet-Talocia, and Prof. Arianna Montorsi and all others with whom I have collaborated during my mandate. Deepest thanks to the ADI Polito team and ADI Torino representatives who were there along the way, Francesca Gamna, Mattia Spedicati, Raffaele Cucuzza, Dr. Enrico Virgillito and Dr. Francesco Surano. It was a pleasure founding with you the very first PhD students' association in PoliTO.

I also wish to thank all the research fellows from other institutions I collaborated with. In particular, Prof. Corrado Possieri from Università degli Studi di Roma "Tor Vergata". I'd like to praise his capability of sharing his deep knowledge on very complex topics in a clear and humble way. Thanks also to Prof. Andrea Bonci from Università Politecnica delle Marche. A big thank you to the Open Source community, for the invaluable shared knowledge. I'd also like to extend my deep gratitude to Prof. Laura Ferranti from Delft University of Technology in the Netherlands, who gave me the possibility to be a visiting PhD under her supervision at the R2C lab within the CoR Department, allowing me to challenge myself in a new stimulating research environment. My sincere thanks also goes to the colleagues and friends who made me feel so welcomed and included during this adventure, making me feel at home in Delft, in a new country and university (you know who I am talking about...that's you! You are really unbelievably so many and I don't want to forget anyone, so I will just go with listing no one, fair enough...right?).

I would like to extend my heartfelt *thank you* to my family: my mother Leriza Cudapas and my father Franco Sibona, my sister Angelica Sibona with Giacomo Marietta Goccio, my aunt Maria Grazia Sibona with Paolo Camiz, my partner in life Massimiliano Terenzi with his family, and -of course- my cat Dotty for the constant and unwavering emotional support, constructive advices or criticisms, and insightful suggestions during these PhD years. Thanks for trying to distract me during stressful times. Thanks for trying to understand, sometimes unsuccessfully, my research and what I do for a living but always being enthusiast about it. Thanks for the food that tastes like home and the videocalls that sound like "I miss you". Thanks for being there along the way and, above all, thank you for the love you gave, give and always will give, whereever life will lead me to.

Last but not least, I want to thank *You*, reader, for your interest in my dissertation manuscript, result of research and hard work, which hopefully will be useful to you professionally. If you are a student, let me drop some words of wisdom as a closure to this long-winded speech (I need to work on verbosity, noted down). Nobody will tell you what awaits you in the future, but nobody can tell you what's not. You can be anything, don't let them think you otherwise. Let me be that one person that tells you, now: I believe in you, you can do it, I see you, and you are enough. Never aim for less than what you dream.



Abstract

Over the past decades, industrial revolutions hinted to future fully automated scenarios, where humans would have taken on the role of mere supervisors. However, the need for mass product customization, rather than mass production, has led to a new concept of production line, where robots share workspaces with humans. The solutions envisaged by Industry 5.0 are human-centred and flexible manufacturing setups, where fixed-base and mobile robots collaborate with humans. This interaction has yet to become smooth given the difference in cognitive capabilities. Hence, to teach robots the correct and safe behaviour during collaboration, artificial intelligence can be leveraged to enable operation recognition and allow for learning.

In order to implement a robot able to cooperate and collaborate with human operators, an Autonomous Mobile Robot (AMR) demonstrator has been imagined as a way to support an Automated Guided Vehicle (AGV) network, serving as a meta-sensor, i.e., a distributed sensor. The meta-sensor AMR is made able to detect human obstacles, so as to overcome them conservatively. When needed, the meta-sensor AMR is capable of following a supervised safe path, using a modified version of the A* global planner, triggering an online global-plan re-computation when a human is detected. Then, with the aim of enabling robot learning to improve collaboration, the AMR has been upgraded to be able to monitor the scene where the human is working, to collect data on the operator's 2D motion as map images, used for training a deep neural network to learn how to recognize performed operations. Lastly, the role of fixed-base manipulators have been approached through an interface to supervise the execution of a desired assembly task, along with a proposal for exploiting simulation to improve interactive imitation learning (IIL).

These contributions feature the integration of open source tools, libraries and data, to tackle the challenge of improving collaborative manufacturing tasks. In particular, the provided solutions aim at upgrading existing networks of AGVs with new technologies represented by the meta-sensor AMRs. Also, a data-driven framework, able to recognize human operations from 2D images featuring the human 2D motion, demonstrates that minimal but useful results can be achieved even in case of data scarcity, exploiting data that is independent of the specific features of the operator. Finally, the concept research idea making a fixed-based cobot exploit simulation to regulate interactive demonstrations in IIL, represents an effort towards less demanding learning processes for the user, in terms of time and expertise.

Contents

Preface

It is undeniable that the development of humanity and technology has been closely connected for decades. What was important before is no longer so. This is because, given technological support, some aspects are now entrusted to machines and automation. In particular, in the manufacturing context, mass production is no longer suitable for the market demand, fueled by a new awareness and sensitivity to sustainability. Indeed, overproduction and current industries and production systems have proved to be not sustainable and seen as responsible for environmental degradation [1]. Nevertheless, what has deeply transformed the structure of production lines is today's market demand in developed countries, which shifted its interest from mass production to products on request.

When it comes to product on request, the optimization parameters differ as the focus shifts towards quality optimization by leveraging human expertise (often experience-based) at the cost of production efficiency, particularly in terms of time. In addition, the COVID-19 crisis has resulted in many Small-and Medium-sized Enterprises (SMEs) incurring losses, making them more vulnerable to slow recovery and less resilient to declining demand than big players [2]. The custom product production vision can aid SMEs and support the revival of local businesses. In particular, given the limited resources of SMEs, an ideal solution would prioritize enhancing human manual operations systematically by collaborating with robots and utilizing Artificial Intelligence (AI) algorithms to enhance overall efficiency instead of optimizing machines for custom production.

The concept of products on request goes beyond simple customization, as it involves specific design, construction, and testing procedures for each individual product, leading to higher costs and longer manufacturing times than standard mass-produced products. Mass production offers lower costs but higher risks, while custom manufacturing offers greater sales security but requires higher labor costs due to the need for human operators with unique cognitive abilities. Although certain tasks require human intelligence and cannot be replaced by AI, automation can reduce the need for human involvement in the production process, minimizing human error [3]. However, the challenge of custom manufacturing is to achieve the same level of cost and production time as mass production while meeting the specific needs of each product. To standardize custom manufacturing production, collaborative robotics can be a key enabler, leading to the evolution of a smart custom manufacturing system.

Therefore, being the human foreseen to be a relevant and irreplaceable component of the flexible production line, increasing focus is put on the perception of personal well-being in the workplace, leading to new requirements for applications involving humans, in terms of safety and health. To be a resilient society we need to foresee how the societal changes will undoubtedly affect the production system paradigms to be prepared to comply with the new needs, where for sure a lot of attention will be dedicated to the human comfort, independently of the worker specific features. This implies that research should aim at *unbiased solutions*: inclusive, gender neutral and body type neutral, more in general intersectional solutions are key as, more and more, human operators will see further representation. Research is slowly getting sensitive to the issue, e.g., authors in [4] borrow feminist principles to guide the design of more ethical human-robot interaction.

This manuscripts will walk the reader through the evolution that industrial robots, namely fixed-base and mobile robots, underwent during the industrial revolutions. The research works I contributed to will accompany the reader through a journey from "dumb" automated machines, through cooperation in human-shared workspaces, up to collaboration, and eventually robot learning. Interaction of humans and robots has yet to become smooth given the difference in cognitive capabilities. To achieve proactive human-robot collaboration (HRC), the two must interact as a team. As such, communication, interpretation, safe cooperation are important aspects. To teach robots the correct and safe behaviour during collaboration, AI can be leveraged to enable correct surroundings interpretations and allow for learning.

Robots slowly learn to behave as human teammates,
possibly exploiting unbiased AI-based solutions to
improve HRC in flexible manufacturing applications.

This thesis seeks to provide steps forward to answer the following questions:

- *How can the planning and navigation of an autonomous mobile robot be made safe when working in cooperation or collaboration with human workers?*
- *How can we teach robots how to behave? To this aim, how to overcome the cognitive mismatch to improve collaborative interactions?*
- *Is there a trade-off between complex/data-hungry solutions and sustainable but feasible minimal ones, exploiting unbiased data?*

A content roadmap for the reader

This section guides the reader through the thesis contents, so as to give a high level overview of how the reported contributions are linked, complemented by some motivation on the manuscript structure.

The research focused on the evolution of differential drive mobile robots from human-free to human-shared workspace. In particular, to implement a robot able to cooperate and collaborate with human operators, an Autonomous Mobile Robot (AMR) demonstrator has been made capable of:

- representing a support for Automated Guided Vehicle (AGV) networks [5]
- following a supervised safe path generated exploiting attractive curves and barrier functions, using a modified version of the A* global planner [6]
- detecting human obstacles, through a state-of-the-art neural network-based object detection algorithm, so as to overcome them conservatively [7]
- triggering the online re-computation of global plans to follow the safe curve when a human is detected [8]

Then, with the aim of enabling robot learning to improve collaboration the AMR has been upgraded to be able to:

- monitor the scene where the human is working [9]
- collect data on the operator's 2D motion as map images [10] (as an outcome after investigation of the main Human-Robot Perception methods [11])
- use map images for training a deep neural network to learn how to recognize performed operations [12]

Lastly, also the evolution of *fixed-based manipulator* collaborative robots (cobots) have been investigated. In particular the role of manipulators have been approached through:

- An interface to supervise a cobot to execute a desired sub-task [13]
- A concept proposal for exploiting simulation to improve interactive imitation learning [1].

Some other works, not mentioned above, namely [14], [15], and [16], are contributions I have co-authored, in the development of which I was not on the front lines. Nevertheless, all works have been referenced in this manuscript, although with variable emphasis. In particular, works have been arranged based on their content level of generality, relevance within the thesis topic, degree of involvement and their technology readiness level (TRL), which are used to measure the advancement and maturity of a technology [17]. In fact, according to definitions in [18], the reported research works range from basic principles observations (TRL 1), to proofs of concept (TRL 3), up to experimental validation through laboratory demonstrators, i.e., a maximum of TRL 4, as expected for academic research. For a direct mapping of thesis sections to research papers, refer to Table 1.

[1] submitted as F. Sibona, J. Luijkx, B. van der Heijden, L. Ferranti, and M. Indri, "EValueAction: a proposal for policy evaluation in simulation to support interactive imitation learning," in IEEE International Conference on Industrial Informatics (INDIN23).

Table 1 Published co-authored research works and where to find them.

Research paper	**Ref.**	**Section/s**
Smart sensors applications for a new paradigm of a production line	**[5]**	**1 / 2.1**
Supervised global path planning for mobile robots with obstacle avoidance	**[6]**	**1.2.1**
Sensor data fusion for smart AMRs in human-shared industrial workspaces	**[7]**	**2.1.3**
Online supervised global path planning for AMRs with human-obstacle avoidance	**[8]**	**2.2.1**
Sen3Bot Net: a meta-sensors network to enable smart factories implementation	**[9]**	**2.1.2**
Data-driven framework to improve collaborative human-robot flexible manufacturing applications	**[10]**	**3.1**
Human-Robot Perception in Industrial Environments: A Survey	**[11]**	**1.1 / 2.2.2**
PoinTap system: a human-robot interface to enable remotely controlled tasks	**[13]**	**3.2.1**
A framework for safe and intuitive human-robot interaction for assistant robotics	**[14]**	**3.1.2**
Dynamic Path Planning of a mobile robot adopting a costmap layer approach in ROS2	**[15]**	**1.2.3**
How to improve human-robot collaborative applications through operation recognition based on human 2D motion	**[12]**	**3.1.1**
Path planning in formation and collision avoidance for multi-agent systems	**[16]**	**1.2.2**
A simulation approach to support interactive imitation learning	**In review**	**3.2.2**

About the above contributions, the following is worth mentioning:

- The work developed in [12], in addition to being presented at the IECON 2022 conference – 48th Annual Conference of the IEEE Industrial Electronics Society, in Brussels in October 2022, has been presented in a 3 minutes video (available at [19]), for participation at the IES-SYPA (IES Student and Young Professionals Paper Assistance Program) 3M Video contest 2022. As a winner of the IES-SYPA 3M Video contest, my participation to the conference has been fully funded by IES. Moreover, after my 3-minutes pitch presentation during the conference, the video has been selected as one of the best 5 out of 44 winning videos, for higher financial assistance.
- The research presented in Section 3.2.2, has been investigated during my visiting at TU Delft, from September 2022 to January 2023, at the Reliable Robot Control (R2C) Lab within the Cognitive Robotics (CoR) Department, supervised by Prof. Laura Ferranti. The preliminary research work brought on there led to a concept idea introduced in a paper I co-authored, currently under review, submitted to the INDIN 2023 IEEE conference.

The remainder of the manuscript is organized as follows. Chapter 1 first gives a general overview of the evolution of robots across industrial revolutions, then in Section 1.1 the main approaches for HRP are presented (Section 1.1.1), with focus

on the methods and sensors used in industry (Section 1.1.2). Then, Section 1.2 directs the attention towards industrial mobile robots; in particular, Section 1.2.1 presents a supervised global path planning for mobile robots with obstacle avoidance, Section 1.2.2 briefly reports the results obtained for an approach for path planning in formation and collision avoidance for multi-agent systems, and Section 1.2.3 outlines a dynamic path planning method based on costmap layers in ROS2.

Chapter 2 core is the mobile robot navigation metamorphosis to adapt to human presence. First a proof of concept (POC) for a network of AMRs is provided (Section 2.1), introducing the concept of meta-sensor and Sen3Bot (Section 2.1.1). The working principles of such network are outlined in Section 2.1.2, while a preliminary working implementation of a Sen3Bot is provided in Section 2.1.3. The application of the Sen3Bot are explored in Section 2.2. Namely, the Sen3Bot capabilities are used to update online the previously introduced supervised global path planning algorithm (Section 2.2.1), and to be used as the base for a POC Sen3Cobot, described in Section 2.2.2.

Chapter 3 showcases the steps towards achieving robots that learn to behave in the human presence. Specifically, Section 3.1 introduces a data-driven framework designed to improve collaborative human-robot flexible manufacturing applications, whose implementation -exploiting a Sen3CoBot that monitors and recognizes human operations- is detailed in Section 3.1.1, along with a POC Sen3Cobot mobile manipulator (Section 3.1.2). Furthermore, Section 3.2 briefly describes the evolution of industrial fixed-base manipulator from automated production machines to human collaborators. Section 3.2.1 briefly reports a HRI aiming at implementing a supervised collaborative assembly task, then a concept idea for exploiting simulation for improving interactive imitation learning is presented in Section 3.2.2.

Finally, the **Conclusions** of this manuscript summarize the main results that the research brought on in the last three years led to, along with lessons learned and open issues, with relative research challenges and opportunities. **Methods and Tools** can be found as Appendix A.

Curiosities

- The thesis' and chapters' covers have been generated using AI. In fact, I have fed the title of the thesis and of each chapter to the `Text To Image - AI Image Generator API` available at `https://deepai.org/`, exploiting the `Old Drawing Generator API` for image styling. Following several image generations, the final images have been selected to my taste. It was interesting to note how for the thesis' cover and Chapter 3 cover, I had to force the input text in order to obtain images featuring a woman: yet another demonstration of the data bias in big data AI – the model apparently has been trained over (supposedly online) data that associate a context containing a robot straight off to a male gender human.
- In the last publications I co-authored, when possible, I always have tried to include a female user/operator in the paper's figures, because I strongly believe in the power of representation and love the idea of reinforcing in female students/researchers the feeling that they belong to STEM (Science, technology, engineering, and mathematics) topics. Images featuring a woman working with robots can be found in contributions [7], [9], [10], [11], [12], and [13].

Chapter 1

Robots in industry: from cages to autonomous navigation

INDUSTRIAL development has undergone several revolutions, strictly connected to the technological and social development of humankind. In fact, all of the industrial revolutions are, in a way or another, linked to a certain degree of automation introduced by the use of machines.

The first industrial revolution (1760-1840) saw the transition from manual labor to the use of steam engine, which powered machinery. This led to a transformation in manufacturing and transportation industries, disrupting the way goods were produced and distributed. This cooperation between humans and machines was boosted during the second industrial revolution (1870-1914) with the use of electricity and the introduction of the assembly line, representing a turning point also from the social point of view, as mass production became possible at that time.

Since the start of the third industrial revolution, also known as the *digital revolution*, in the 60s, industries exploited electronics and information technology to automate production, making use of two major inventions: programmable logic controllers (PLCs) and robot manipulators. Furthermore, the widespread use of computers and the Internet changed the way that people communicated and accessed information as well as their perception of automated machines in the industrial and everyday contexts.

The integration of these technologies and of further advanced technologies, such as artificial intelligence, the Internet of Things (IoT), and robotics into various industries and areas of society has enabled new forms of automation and connectivity, and has had a profound impact on the way that goods are now produced, services are provided, and people live and work. The possibility of managing big amounts of data at low cost, together with the almost infinite data processing capacity, is leading to industrial plants free from classical, rigid constraints, which are no more considered as insurmountable. Furthermore, the need to transfer huge quantities of information in a fast and reliable way, demands advanced communication methods, drawing extensive research interest towards forefront wireless networks able to solve performance requirements issues and to meet the increasingly stringent requirements of specific industrial applications [20].

Hence, this fourth revolution can be framed as the result of both (i) *technology push*, thanks to the advances in enabling technologies allowing for new frameworks and devices, and (ii) *market pull* as new technologies and paradigms have been dictated by the need for a flexible setup of production lines. This led to a whole new industry concept known as *Industry 4.0*. The term was first introduced in 2011 in a high-tech strategy by the German government, and popularized in 2015 by Klaus Schwab, executive chairman of the World Economic Forum [21].

New paradigms of production lines are being implemented and are in the process of being integrated, characterized by a strict collaboration between humans and robots, and by a high degree of flexibility. Various examples of ideal systems projected over the next decade can be found, e.g., the *Multi-silhouette production line* by PSA [22], as well as pilot plants, like the one proposed by Kuka in [23], based on the *matrix production paradigm*, which represents a recent effort to break down the conventional linked production line into standardized and categorized manufacturing cells, placed along a grid layout.

Due to a change in market demand, the big challenge has now become the production of custom versions of specific products, in quantities that may change on demand, relying on an enhanced flexibility of the production lines [24], [25]. The increasing demand on customized products requires the combined efforts of intelligent manufacturing systems along with unique human skills, such as creativity, complex reasoning and socio-emotional intelligence [26]. The resulting manufacturing system has high-precision automation and flexible infrastructure able to react to dynamic needs, thanks to the synergy between intelligent machines and humans with flexible problem-solver and decision-maker skills. In fact, even though the 4th industrial revolution is still ongoing, research is gathering more and more awareness of some deep changes that are going to shape the whole perspective of automation, fostered by the quick evolution of AI-based solutions and digital technologies in the recent years. Moreover, the necessity to improve the productivity without penalizing human operators in the manufacturing industry is becoming a challenge. Instead of implementing the Industry 4.0 solutions, where all the tasks are automated while ignoring the human during the process optimization, it is possible to adopt the solutions envisaged in the so-called *Industry 5.0* [27], where the factory is human-centered, meaning that the autonomous robots are perceptive and aware about human intentions. Human activity recognition is widely investigated, leveraging sensors for multiple modalities to enable specific applications [28]. In the Industry 5.0 context, the robot is able to actively observe and learn patterns from human workers using machine learning algorithms, so as to predict the human actions and attempt to help. With these features, it is possible to enable mass customization instead of mass production [29]. A timeline of the industrial revolutions, along with their relative main features, can be see in Figure 1.1.

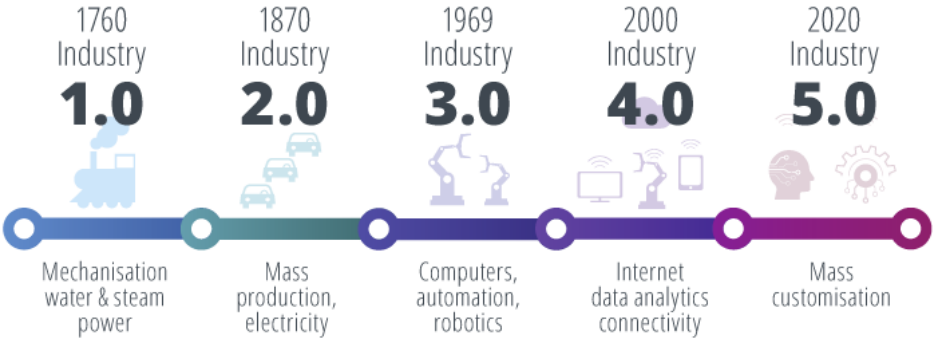


Fig. 1.1 Industrial Revolutions timeline [30].

All the envisaged improvements, leading to the smart *Factories of the Future*, have sensors and their proper usage as fundamental pillars [31]. The desired more adaptive production lines and the whole product life cycle have to be supported by a smart choice of heterogeneous sensors and/or a heterogeneous choice of smart sensors, through which not only the final goals can be achieved, but also a smooth transition can be set up from the current industrial standards toward a flexible and human-centered industrial scenario. In particular, robot perception can be implemented with several levels of detail and understanding, depending on the application and on the environment the robot has to operate in. Hence it is clear the importance of how the robot perceives its surroundings, as this affects how the robot acts and behaves.

1.1 The importance of perception

The perception ability is necessary for robots to perform tasks requiring an interaction with the environment, e.g., motion planning, localization, and object recognition. For example, robots can use visual sensors to inspect products for defects or use force sensors to detect the position and orientation of parts during assembly. In fact, in industrial settings, perception has a major role in several applications, such as quality control, and assembly. By accurately perceiving their surroundings, robots can perform these tasks more efficiently, accurately, and safely.

Perception not only is applicable to manipulators or mobile platforms but also to industrial workstations. Enhancing the perception capabilities of a workstation allows for innovative applications. The original combination of multiple sensors and actuators together with human contribution is another example of innovation and application of the 4.0 methodologies to today's industry. Current trends are aimed at leading the quality and repeatability levels of manual stations closer to automated cell standards. Several solutions are available on the market, possibly given by a unique, complete station or by sub-systems integrated to obtain a station. Sarissa Assistance Systems, produced by Sarissa GmbH [32] is made up by ultrasonic emitters and receivers that allow to locate the position of the tool in a three-dimensional environment. The receiver collects the data and sends it to a Box-PC via an USB connection. The Box-PC elaborates the information from the receiver and sends the xyz-coordinates to a PLC controller or a PC controller via Ethernet TCP/IP connection. Der Assistent, produced by ULIXES Robotersysteme GmbH [33] mainly consists in a projection system that highlights the process steps through graphics, photos and videos, projecting directly into the processing area and into a 3D vision system that controls the correct sequence of operations. Light Guide Systems, produced by OPS Solutions [34] is composed by colour-coded, animated light beams and visual prompts in the form of text, symbols, graphics, blueprints or video projected on any workstation or off-line training area. This solution eliminates the operator reliance on printed work instructions, computer screens or memory for

assembly guidance. The main points of comparison among the different solutions found on the market are summarized in Table 1.1.

Table 1.1 Comparison of relevant manual station digitalization solutions.

Feature	*Sarissa*	*OPS Solution*	*Ulixes*
Lights	False	2D Beamer	2D Beamer
Sensors	3D Ultrasound	False	3D Vision
Tool Tracking	3D Ultrasound	False	False
Gesture/Speech Recognition	False	False	False/True

The SMARTMAN 4.0 workstation

In [5], thanks to a strict academia-industry collaboration between Politecnico di Torino and COMAU S.p.A., we investigated different aspects of the described industrial renewing process, aiming for a new paradigm of a production line. The proposed SMARTMAN 4.0 solution, showcased how digital systems can be used to support a worker during the execution of his/her activities, in order to optimize, monitor and control the quality of the work in a complete way. Objectives of the SMARTMAN 4.0 project were both the streamlining of the personnel training phases and the maintenance of a constant and high quality of the produced objects. Streamlining staff training has a strong impact in the case of small productions, concentrated in time or that change very quickly. In these cases, reducing the time devoted to pre-production stages (e.g., for staff training) can be a significant advantage, with a view to minimize costs for small production series.

With the proposed solution, the quality is guaranteed through a constant and step-by-step supervision of the entire production process. This objective is achieved by a strict control of sequences and localization in space and time of the activities of the operator, e.g., in manual assembly stations, logistic picking stations, or more generally whenever there is a manual station where a worker has to perform a process that needs to be verified. The employed elements can be collected in different clusters: a *System Control Unit*, a *Pointing System* and a *Sensors Remote Unit*.

The *System Control Unit* is basically the PC, on which the necessary algorithms are implemented and which is responsible for supporting the graphical interface to the operator. The *Pointing System*, exploiting a laser pointing system produced by Z-LASER [35], is mainly dedicated to indicate to the operator where it is needed to perform an operation and to return a visual feedback of the execution status, ensuring projections on non-coplanar surfaces without distortions, and the possibility of automatically re-adjusting the reference frame used for the projections, in case of processes or supports where position and orientation may vary. The *Sensors Remote Unit*, a SmartRobots three-dimensional camera [36], is able identify and trace the hands of the operator and return their positions. This camera processes both three-dimensional and colour information and returns the position of the operator's hands,

distinguishing the right from the left. To help the system to correctly identify the right operator, it was decided to let the operator wear colour-coded gloves. Thanks to this simple adjustment, the camera is guaranteed to "capture" the correct worker, even in the case of several people appearing within the camera visual space.

The SMARTMAN 4.0 stations is able to learn from an expert operator the set of operations and the relative sequence, through a specific *recording work-flow* (Figure 1.2a). Then, during a *working phase* (Figure 1.2b), the system guides the operator along the correct sequence of operations, checks the correct positioning and, only in case of congruence, enables the equipment to perform the task, thus generating an interlock between correct sequence and manual activity, ensuring a constant quality of production. In this sequence there are no direct interactions between the operator and the *System Control Unit* if everything is done correctly; in case the operator takes too long time to perform the operation the system signals a time-out. In the event of time-out, the operator can set the repetition of the step or the abortion of the whole sequence, again, via voice interface.

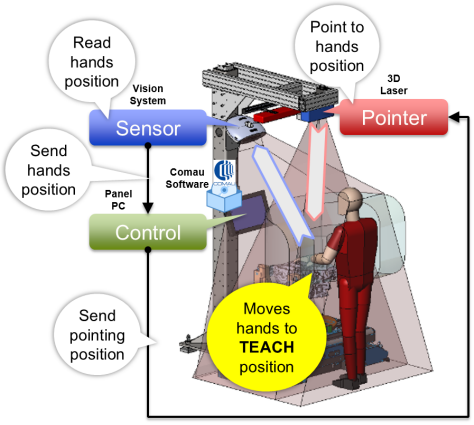
Particular attention has been paid to the HMI (Human-Machine Interface) issues. The use of graphic symbols directly drawn on the worker's hands, as a feedback during teaching and working phases, along with the speech recognition and synthesis, enhance the ergonomics of the station. Thanks to the adopted solutions, the operator is not forced to shift the gaze, each time focusing elsewhere on HMI supports, or to make wearying continuous movements of the neck.

The described system can therefore be positioned, with respect to the state of the art, as indicated in Table 1.2, which puts in evidence the completeness of the SMARTMAN 4.0 solution.

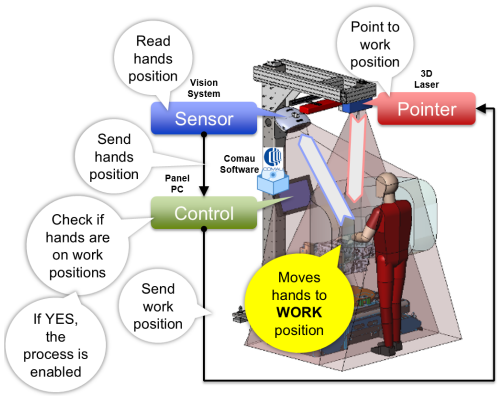
Table 1.2 Comparison of current manual station solutions and SMARTMAN 4.0.

Feature	*Sarissa*	*OPS Solution*	*Ulixes*	*SMARTMAN 4.0*
Lights	False	2D Beamer	2D Beamer	3D Laser
Sensors	3D Ultrasound	False	3D Vision	3D Vision
Tool Tracking	3D Ultrasound	False	False	3D Vision
Gesture/Speech Recognition	False	False	False/True	True

The SMARTMAN 4.0 station is a sensor system, or more briefly over-sensor, born from the intelligent fusion of information and specific capabilities of individual existing objects, which is going to improve, in an alternative way, a complex and structured reality, without the need for a long complex and expensive development. A smooth transition from the current industrial methods toward a fully integrated 4.0 reality is simplified by the design of a modular sub-system to be easily added to existing items. The modularity of the sub-system can be exploited in the future also for new major additions in a production cell, increasingly oriented to cooperation between humans and collaborative machines.



(a) Workflow for *recording phase* of SMARTMAN 4.0.



(b) Workflow for the *working phase* of SMARTMAN 4.0.

Fig. 1.2 SMARTMAN 4.0 phases.

Indeed, perception is significantly important for human–robot interaction, as it enables robots to detect and respond to human presence and actions. This is essential for tasks that require close collaboration between humans and robots, such as in the medical or service industries.

In [11], we analysed various sensors and perception methods utilized in applications involving HRI, robot guidance and collision avoidance for all the main types of robotic systems commonly used in industry, such as fixed-base manipulators, collaborative robots (cobots), mobile robots and mobile manipulators. The analysis investigates how these robotic systems perceive the presence of human operators and how they react for cooperative and collaborative applications. Also, various applications, strongly relying on HRP to achieve HRC, are reviewed with a particular focus on the handled type of data, the need to fuse the information that comes from different sensors to guarantee an efficient and safe HRI, as well as the specific requirements of the perception tasks (e.g., the perception range, the safety issues,

the environmental influences, etc.). Particular attention is devoted to vision and distance sensors, which are the most employed for human perception in various types of robotic systems. Monocular RGB (Red Green Blue), stereo, RGB-D (Red Green Blue-Depth), and more recent event-based cameras stand out among the vision sensors, whereas the most used distance sensors are based on light scan technology, such as the LIDAR.

To meet the growing demands of the market, which are becoming faster and more cost-effective, upcoming industrial settings will necessitate a considerable degree of automation to enable flexibility and adaptability. Autonomous and collaborative robots able to adapt to varying and dynamic conditions of the environment, including the presence of human beings, will have an ever-greater role in this context. However, if the robot is not aware of the human position and intention, a shared workspace between robots and humans may decrease productivity and lead to human safety issues. The survey we conducted provides an outline of various sensors and methods for perception. Various types of robotic systems commonly used in industry, such as fixed-base manipulators, collaborative robots, mobile robots and mobile manipulators, are considered, conducting an analysis of the most useful sensors and perception techniques for identifying and responding to the presence of human operators in cooperative and collaborative industrial scenarios.

The perception capabilities of robots will gain ever-greater importance in the next smart factories. As discussed, the robot has been assuming a progressively important role within factories and warehouses, recently witnessing a boost in its use as a support to human workers, as a team member or a flexible part of manufacturing processes. Autonomous and collaborative robots will be increasingly involved in operations requiring a shared working space with human actors. Most of the activities will have to be done avoiding obstacles, working collaboratively with human beings, autonomously locating and identifying the parts to be worked or moved.

This perspective of collaborative environment between humans and robots in production settings goes beyond the concept of Cyber Physical Production System (CPPS) [37,38]. In CPPSs, a smart production plant is itself a Cyber Physical System (CPS) integrating cyber aspects as computation, communication, control, and networking technologies into the underlying physical system. A CPS can quickly react and adapt to market changes negotiating production resources as in [39], or using some intelligent reasoning tools as suggested in [40], but humans are generally considered to be intruders into the automated tasks. CPSs should be able to reprogram their activities reacting to the presence of humans or other mobile systems, but generally they do not interact or collaborate actively with them. This leads to conceive the next future evolution of the CPSs towards Cyber Physical Human Systems (CPHSs) [41,42], where the control, communication and automation technologies, physical plants and humans must pursue a common goal. This recent development poses some challenges regarding the traditional understanding of Cyber-Physical Systems (CPSs), where humans were very often considered to be independent passive entities that operate, use or consume the CPS resources. This also motivates the research for new solutions for developing trustworthy, safe, and efficient Human-

Robot (HR) perception to achieve an enhanced HR Interaction (HRI) in collaborative work environments, thus allowing the development of CPHSs.

In the context of CPHS, the adoption of HR teaming is still hampered by the lack of clear guidelines for safety, interfaces and design methods [43]; the HR Perception (HRP) step and its requirements are crucial for the successful implementation of the paradigm of CPHSs, as the core of the Factory of the Future (FoF). The digitalization of the whole manufacturing system requires managing plenty of heterogeneous sensors able to share and fuse the information provided by other sensors, as well as enhancing the capabilities beyond detection.

To fully digitalize the manufacturing process, a multitude of heterogeneous sensors must be managed, capable of exchanging and merging data collected from other sensors, while also broadening their scope beyond detection. Indeed, a high technological level is needed by the sensors, which have not only to read the data and reduce the noise effects, but also to process them (edge computing) to enable predictive maintenance operations [44]. In such a scenario, the availability of sensors for the HRP becomes a key issue for managing the operations of HRI in the FoF. The choice of the type of perceptive system to be used is highly related to the task to be fulfilled, to the level of autonomy to be guaranteed, and to the kind of HRI that must be established. It is worth highlighting that there exist three types of HRI in the industrial scenario [45]:

- *HR Coexistence*, where humans and robots share the same working space, but performing tasks with different aims; here, the human is perceived as a generic obstacle to be avoided, and the robot action is limited to collision avoidance only.
- *HR Cooperation*, in which humans and robots carry out distinct activities, but with identical objectives that must be achieved in a synchronized manner in both temporal and spatial dimensions. Under these circumstances, the collision avoidance algorithm includes human detection techniques, so the robot can differentiate the human operator from a generic object.
- *HR Collaboration* (HRC), where a direct interaction is established between the human operator and the robot while executing complex tasks. This can be achieved either by coordinated physical contact or by contactless actions, such as speech, intentions recognition, etc.

The aim of the Sections 1.1.1 and 1.1.2 is to provide useful information to accomplish several robotic tasks in HR collaborative industrial environments, and to offer to the reader a quite complete overview of the different solutions proposed in the literature and of the modalities with which they have been applied to different types of robotic systems. The carried-out analysis provides detailed and aggregated information about the various types of sensors adopted to handle the presence of human operators in several industrial scenarios, as well as about the sensors and algorithms combinations that seem to offer the best performance.

1.1.1 Robotic systems and Human-Robot Perception (HRP)

In this section, several types of robotic systems commonly used in industry (i.e., fixed-base manipulators, cobots, mobile robots and mobile manipulators) are considered with the aim of illustrating how they perceive and react to the presence of human operators or static obstacles in the industrial scenarios, for cooperative and collaborative applications.

Fixed-Base Manipulators and Cobots

HRC in the context of fixed-base robots is a matter of considerable interest in recent research. In the various applications in which humans and robots coexist in the same environment, two scenarios may be considered:

(i) A full awareness of the human presence and the environment is necessary.

(ii) Only the safe management of the shared spaces can be sufficient, guaranteeing that humans cannot be injured during the robot motion.

In (i), different types of sensors are used to track humans or obstacles in the manipulator workspace, obtaining a complete 3D model of the environment. In such a way, it is possible to monitor the distance between the robot and any object in the workspace, whereas a high-level controller can re-plan the robot trajectory to avoid collisions or stop the system, if necessary. In (ii), collisions are generally detected estimating the dynamic properties of the robot, together with the information coming from the proprioceptive sensors, which industrial robots are usually equipped with. The robot's motion is then re-planned and controlled to limit the contact forces, so that potential collisions with humans or obstacles result being no more critical.

This section investigates the enhancements achieved in recent HR applications mainly using exteroceptive sensors, without leaving out the available proprioceptive-based methods to estimate the robot contact forces through the knowledge of the robot dynamics. Without the use of exteroceptive sensors, the robot has not a perception of the 3D external environment: the interaction contact forces are kept limited, leaving the robot unaware of the presence of humans and obstacles. Therefore, by using proprioceptive sensors only, collisions are not avoided, but they become not dangerous for humans. Typically, commercial cobots mainly implement force-restriction approaches, sidestepping the use of vision sensors. Thanks to their smooth surfaces and operating velocities that are suitable for the collaboration with humans, cobots work *together with* the operator, even if they actually do not perceive his/her presence, but only the possible contact.

Some solutions have been proposed to also make the traditional manipulators able to establish some kind of collaboration with humans [46], according to ISO/TS 15066:2016 [47].

The method proposed in [48] limits the force for a traditional industrial manipulator, and detects collisions without the use of external sensors. It adopts time-invariant dynamic models and supervised feed-forward input-delay neural networks on signal processing to estimate the required current signals for a given robot motion. The predicted current signals are then compared with the actual absorbed motor currents, which are persistently gauged by the robot controller; the detection of a collision takes place as the current required by the manipulator is greater than the predicted one. Another approach is proposed in [49], which avoids the use of external force sensors, generally not present in standard manipulators, and exploits the dynamic model of the robot in both dynamic and quasi-static modes to detect the external forces.

To surmount the existing limits in HR applications, the use of exteroceptive sensors, in particular vision sensors, can be a viable solution, even if there are still problems related to the accuracy and repetitiveness. Indeed, vision sensor performance is highly affected by environmental conditions, such as exposure, brightness, reflectiveness, etc. However, such sensors are the most suitable for providing the robot complete awareness of the environment, to avoid obstacles and re-plan trajectories. Consequently, vision sensors are generally used to check the workspace, allow humans safety and detect the presence of objects, notwithstanding their critical issues.

The most used vision systems for HR collaborative applications are stereo cameras, RGB-D (Red Green Blue—Depth) cameras, proximity sensors or laser scanners. Obstacle tracking data can be employed to estimate human intentions, to create models of the 3D environment, to calculate distances between the robot and the obstacle, to integrate data coming from virtual and real world to test an application in simulation, and so on. This section reviews the current state of the art, emphasizing the three essential aspects for the development of HR applications:

- *Type of sensor*: the sensor output depends on the sensor technology. Therefore, the algorithms used to process the data can be quite different.
- *Methodology to detect obstacles in the scene*: the chosen methodology depends on the type of sensor but also on its location. The sensor can be positioned somewhere in the environment to monitor the entire scene or can be mounted on the robot arm. In the first case, it is necessary to distinguish humans and obstacles from the manipulator, otherwise the robotic system can identify itself as an obstacle. In the second case, the sensor position is not fixed and must be estimated to recreate the 3D scene.
- *Anti-collision policy*: once the obstacle is detected on the robot path, and hence the risk of a possible collision, the robot can be stopped providing some warning (e.g., sounding an alarm) or its trajectory can be automatically re-planned to avoid the obstacle.

Numerous studies suggest employing a Kinect RGB-D sensor, which provides RGB and depth space images to reconstruct the 3D environment. In [50], the Kinect

sensor is used to generate 3D point cloud data and to study the collision prediction of a dual-arm robot (Baxter). To detect obstacles in the scene and prevent self-collision avoidance, the authors proposed a self-identification approach that relies on the over-segmentation method using the forward kinematic model of the robot. To improve the processing speed, a region of interest is determined based on the skeleton of the robot, then a collision prediction algorithm estimates the collision parameters in real time for trajectory re-planning. Flacco et al. [51] presented a fast method to calculate the distance between several points and moving obstacles (e.g., between robot joints and a human) in depth space with multiple depth cameras (Kinect). The robot kinematics is used to identify the point cloud data representing the robot itself to eliminate it from the scene. The distance is used to generate repulsive vectors that control the robot while executing a motion task, thus achieving a collision avoidance application. Also, in [52], the Kinect sensor was used to add data coming from real obstacles in a virtual scene, where the robot is modelled. Their method seeks to test re-planning algorithms and HR interaction in safe conditions, simulating possible scenarios where humans and robots must collaborate. However, in all these works the Kinect sensor shows its limits in terms of accuracy and reliability. In [53], a method was proposed to improve the accuracy of the Kinect sensor merging real and virtual world information; in particular, some accuracy problems are overcome using a skeletal tracking approach. A highly detailed avatar is created to represent human behavior in the 3D scene, consisting of thousands of polygons. Then, the Kinect sensor is used as an input device for skeletal tracking and positioning of the user. Nevertheless, there are different types of low-cost RGB-D cameras; valuable insights concerning the choice among the most used in research can be found in [54], where sensor performance is compared in an agriculture application.

The use of a simple RGB camera to detect obstacles was put forth in [55], in a case study in which an industrial manipulator is used. The robotic system is provided with smart sensing capabilities, such as vision and adaptive reasoning, for real-time collision avoidance and online path planning in dynamically changing environments. The machine vision module, composed of low-cost RGB cameras, employs a color detection technique based on the hue saturation value space to make the robot aware of environmental changes. This approach allows the detection and localization of a randomly moving obstacle; the path correction to avoid collision is then determined by exploiting an adaptive path planning module along with a dedicated robot control module. It must be underlined that using only a standard RGB camera, the obstacles detection can be performed in 2D assuming a constant height along the third direction. This solution may be valid for manipulators employed for simple pick-and-place tasks and it can be executed in a fast-working cycle.

Another approach that integrates sensors used for virtual world interaction, is proposed in the field of robotics surgery, where any possible collision between the robot and the medical staff is considered to be critical [56], but some of its characteristics could be exploited in different contexts, such as the manufacturing one, for applications that necessitate a strict HR collaboration. The HTC VIVE PRO controllers are used as an Internet of Things technology to measure the distance between surgeons and the robot. When the distances between humans and the robot

-gauged through the smart controllers- become critical, a virtual force is applied to the manipulator to move the robot elbow in a spare workspace. This avoids the direct hands-on contact of the surgical robot arm by applying the virtual force to move the swivel angle of the KUKA iiwa. Due to the kinematic redundancy of the manipulator, a swivel motion with the robot elbow can be performed without moving the robot tool pose avoiding compromising the surgical intervention. In [57], the same authors previously investigated the cartesian compliance strategy that involves online trajectory planning to avoid violation of some defined constraints.

A novel sensor proposed in [58], which consists of skins with proximity sensors mounted on the robot outer shell, provides an interesting solution to occlusion-free and low-latency perception. The collision avoidance algorithms, which make extensive use of these properties for fast-reacting motions, have not yet been fully investigated in this work. A collision avoidance algorithm for proximity sensing skins is put forward as a first solution, by formulating a quadratic optimization problem. The authors highlight that compared with common repulsive force methods, the algorithm confines the approach velocity to obstacles and keeps motions pointing away from obstacles unrestricted.

It is noteworthy that a good HRC requires good HR interfaces and the possibility for the human operator to easily establish some kind of communication with the collaborative robot [59]. A proper use of adequate sensors is fundamental to this aim. Cameras can be employed, but better results can be achieved integrating also specific sensors such as the Leap Motion, which can be used to recognize coded operator's gestures as input commands for the robotic systems (e.g., as for the teleoperated robotic arm in [60]), but also to enhance the perception capabilities provided by cameras, as in [61]. Here, a multi-source heterogeneous vision perception framework is proposed to acquire information regarding the human workers in various conditions and on the working environment during HRC tasks in manufacturing. The proposed system includes RGB-D cameras (i.e., Kinect sensors), located around the working area to produce 3D point cloud data, and Leap Motion sensors on the workbench to track the worker's hands. Thus, a wide and clear perception is achieved of both the working area and the worker.

In [62], a system composed by five Inertial Measurement Unit (IMU) sensors is used to recognize human gestures. The IMU sensors are distributed in the upper part of the operator's body, along with an ultra-wide band positioning system. The latter activates the collaborative mode when the human operator is in close proximity to the robot. Static and dynamic gestures used to command the robot are processed and classified by an Artificial Neural Network (ANN). A similar work related to gestures in the industrial context is presented in [63], in which IMUs and a stereophotogrammetric system are used to track and analyze the human upper body motions, in particular when he/she picks and places several objects at different heights. The gestures sequences are collected in a database, and can be used to optimize the robot trajectories and guarantee the safety of the human operator.

A sensor data fusion algorithm is proposed in [64] to estimate and predict the human operator occupancy within the robot workspace. The algorithm merges the

information coming from two different depth sensors, a Microsoft Kinect and an ASUS Xtion, defining a set of swept volumes that represents the space occupied by the human. In this way, the motion of the robot can be re-planned to be compliant with the safety constraints, thus avoiding any collision with the human operator.

More insights into hand gestures recognition by means of the Leap Motion and other solutions for HR interaction are provided in the following subsections.

Mobile Robots

The Autonomous Mobile Robot (AMR) is gradually taking over a fundamental role in the new dynamic and productivity-oriented industrial environment, in which flexibility leads the production line development. Spatial and temporal flexibility, when considering production plants, can improve productivity and reduce overall downtime, for example, when the production sequence configuration must be changed. Usually, an AMR is equipped with an heterogeneous set of sensors whose output data streams are processed by complex control systems [65]. It is well known that merging the information coming from several sensors improves the efficiency and robustness of the measurements, yet increasing the complexity of the hardware and software required for merging and processing the information deriving from different sources [66].

It is clear that the optimization of processes has a pivotal role for the overall efficiency (e.g., productivity, energy consumption) of a working setup. A mobile robot's perception of its surroundings thus acquires relevance for achieving optimal integration with other CPS elements, going beyond the basic role in the localization of the platform during navigation. In particular, autonomous navigation of AMRs introduced a further need for effective HRP approaches. Indeed, for what concerns cooperative operations, effectiveness may be undermined by the perception that human operators have of moving autonomous agents. Conversely, the mobile base task execution could be slowed down by ill-managed perception of humans. As a matter of fact, the pre-definition of Automated Guided Vehicle (AGV) motion paths guarantees predictability in opposition to the AMRs motions, which are often hard for a human operator to interpret. The perception systems of traditional AGVs [67] have undergone heavy changes [68, 69] to achieve navigation autonomy and advanced perception of the environment and humans in industrial scenarios [70], favoring the investigation of real-time approaches [71]. Moreover, due to the gradual and now extensive use of fixed-base collaborative robots along the production line, the implementation of safe collaborative operations using IMRs has been attracting a lot of interest. Industrial mobile platforms then need to elevate their perception level from a merely informative approach to a semantic interpretation of the robot surroundings.

Although advanced perception of humans appears to be a still relatively new topic within the industrial context, it has already been widely explored and adopted in other fields, from assistive service robotics to agricultural ones, where robotics plays a significant role in the process chain. Hence, it is relevant and interesting to report

some of the approaches to HRP developed in these fields, since it is highly probable that the FoF will implement similar or comparable approaches on intelligent IMRs. In [72], a non-intrusive solution to robot aware navigation is presented, where the behavior of the robot is determined by the user's preferences in a domestic workspace sectioned in virtual areas. In [73], the human and the mobile robot share a common task, since the robot is teleoperated by an operator, whose visible 360-degree scene is enriched by interactive elements drawing the attention to information-rich areas; a 360-degree camera is exploited, and its frames are processed using the You Only Look Once (YOLO) Convolutional Neural Network (CNN)-based framework [74]. In this case, the goal achievement is common, and the perception of the human operator and the robot somehow enhance each other. Also, in [75] teleoperation is implemented, using a hybrid shared control scheme for HRC. The operator sends commands to a remote mobile robot using an electromyography (EMG) signal sensor to reflect muscle activation; the human collaborator is equipped with a haptic device, which receives a force feedback to inform about the existence of an obstacle.

The work presented in [76] aims at highlighting challenging natural interactions between a mobile robot and a group of human participants sharing a workspace in a controlled laboratory environment, demonstrating that humans follow less jerky and irregular paths when navigating around one autonomous navigation condition than around a teleoperated robot. The experiments are performed on autonomous mobile robots using optimal reciprocal collision avoidance, social momentum and teleoperation as navigation strategies. In [77], an approach named RObot Perceptual Adaptation (ROPA) is proposed. This algorithm learns a dynamical fusion of multi-sensory perception data, capable of adapting to continuous short-term and long-term environment changes; a special focus is set on human detection, based upon different types of features extracted from color and depth sensors placed on the mobile robot, with the aim of achieving long-term human teammate following. A structured light camera is used for color-depth data and a digital luminosity sensor for luminosity data. Similarly, in [78], the authors introduced a representation learning approach that learns a scalable long-term representation model, for scene matching. The features of multiple scene templates are learned and used to select, in an adaptable way, the most characteristic subset of templates to build the representation model for the current surrounding environment. The latter procedure is performed with the aim of implementing long-term delivery of information in collaborative HRP applications, exploiting Augmented Reality (AR). Moreover, what seems clear from works reviewed within the agricultural field, concerning collaborative applications and relative perception between humans and robots, is the focus on safety without leaving out comfort of the interaction [79, 80].

The work presented in [81] proposes a planning model based on RNNs (Recurrent Neural Networks) and image quality assessment, to improve mobile robot motion in the context of crowds. Acquired images are pre-processed exploiting OpenCV (Open Computer Vision) calibration tools and then the background noise is filtered out using the designed RNN-based visual quality evaluation. Additionally, concerning the assistance service robotics context, the bidirectional meaning of perception is particularly evident, since the robot should be perceived by users as naturally

as possible, and the robot itself must have capabilities of intention recognition to be actually beneficial for the human partner, e.g., in Sit-To-Stand assistance [82]. Moreover, the SMOOTH robot project, presented in [83], provides an example of adaptive sensory fusion computed via a single multi-sensory neuron model with learning, to boost perception of human capabilities of a welfare robot. The robot is equipped with a front safety laser scanner and two cameras, one front and one back facing. Finally, the survey presented in [84] emphasizes the significance of data fusion in improving the perception capability of mobile robots. The reviewed works consider data coming from multiple sensors (e.g., LIDAR, stereo/depth and RGB monocular cameras) to obtain the best data for the tasks at hand, which in this case are autonomous navigation tasks such as mapping, obstacle detection and avoidance or localization.

Given these example approaches, it is easy to envision how they could greatly impact the emerging HRP research in the industrial context. Many algorithms are being developed with the aim of being ideally applicable in any context involving humans and robots. The need for a unified framework to enable Social-Aware Navigation (SAN) is stressed in [85], where a new method is presented by the authors for an autonomously sensed interaction context that can compute and execute human-friendly trajectories. The authors examine various contexts and implement an intent recognition feature at the local planning layer.

Regarding the industrial logistics context, the authors of [86] put forth a range finder-based SAN system to implement collaborative assembly lines with a special emphasis on human-to-robot comfort, considering the theory of proxemics. A cost function is assigned both to assembly stations and operators to affect the costmap for the mobile robot navigation. In [87], a human-aware navigation framework is proposed, to work within logistics warehouses. The simulated mobile robot is equipped with a laser scanner and an RGB-D camera to detect a person and estimate the pose to consider it as a special type of obstacle and avoid it accordingly. The presented strategy is made up of 2-steps: (i) the use of the depth information for clustering and identifying 3D boxes that are likely to enclose human obstacles, then (ii) the computation of a confidence index for human presence based on the RGB data. Instead, the approaches proposed in [88] seek to demonstrate the integration of AR as an enabler for enhanced perception-based interactions along assembly Manufacturing Execution Systems (MES). The authors propose an application involving mixed reality smartglasses for AR implementation for collaboration with a cobot, and a path visualization application for humans working with AGVs, using an AR computing platform. Another work proposes a solution to HRI using (i) gesture control and eye tracking technologies for the robot to interpret human intentions, and (ii) a pocket beamer to make robot information interpretable by the human operator [89]. Finally, in [90] the authors propose an HR skill transfer system: instructions are given to a mobile robot to follow a trajectory previously demonstrated by a human teacher wearing a motion capturing device, an IMU in this case. A Kinect sensor is employed for recording the trajectory data, used to model a nonlinear system called a Dynamic Motion Primitive. Then, exploiting multi-modal sensor fusion, the pose and velocity

of the human teacher undergo a correction process and a novel nonlinear model predictive control method is proposed for motion control.

Mobile Manipulators

Manipulators have found widespread use in various applications, leading to improved efficiency in industrial production lines. However, these are usually located in fixed positions along the line, which is a limitation for some applications that need to cover large working spaces, as in the automotive or aerospace industry. To overcome this problem, it is possible to rely on mobile manipulation. For a variety of tasks, the flexibility of the system is improved when a manipulator is attached to a mobile platform, since the redundancy offered by a mobile manipulator allows the planning of human-like motions while avoiding singularity configurations. Given the mobility advantages, it is also used for intralogistics and service robotics applications [91, 92].

The prevalent configuration for mobile manipulators in the market comprises a collaborative lightweight manipulator and a mobile platform combined. The mobile platform in these cases may be collaborative or not. It is worth highlighting that for a long time no safety standards specific to these hybrid systems were available, making it necessary to combine two or more standards, e.g., ISO/TS 15066 [47] and/or ISO 10218-1 [93] for manipulators, and ISO 3691-4 [94] for mobile robots. Nevertheless, new regulations terms are including platforms equipped with manipulators, as defined in ISO 19649:2017 Mobile robots — Vocabulary [95], which was last reviewed and confirmed in 2022 [96].

Since the collaborative manipulator itself was designed for collaborative applications, it can react to the human operator's physical contact without causing any harm. However, to allow the robot to perceive better its environment, and therefore improve the decision-making process for the motion planning needed for a specific task, other sensors may be integrated to the robot. The way the robot may sense its surroundings and the way it reacts to the human actions strongly depend on the application. In fact, there are applications in which vision sensors are widely used to emulate the decision making based on the human vision.

As an example, the mobile manipulator proposed in [97] is designed for HRC tasks, in which object detection and manipulation are considered to be critical skills. As stated by the authors, using an RGB-D camera is more robust than using stereo vision cameras, since the latter ones only rely on image features. The images and videos coming from RGB-D cameras are also useful for either (i) configuring the motion constraints based on the human presence, differentiating human-type obstacles from the generic ones, or (ii) predicting the human activity, so the robot can respond appropriately based on the actions of the operator [98]. A sensor system that provides reliable 2½D data for monitoring the working space is presented in [99]. The system processes information obtained from three sets of grayscale stereo vision cameras and a Time-of-Flight camera that monitors the motion of the human operator collaborating with the manipulator. The area monitored by

the sensor system corresponds to the safety zone, in which specific actions of the robot are enabled when the hand of the human operator is in close proximity of the manipulator tool.

The mobile manipulator proposed in [100] uses two sensors able to perceive the environment: an RFID sensor that lets the robot know where the objects are in the space and an RGB-D camera that identifies tags with unique IDs, which contain semantic information and properties of the world entities. The paper did not specify if the robot is working with a human or not, but the interesting aspect is that the robot has the ability to acquire knowledge through experience, and each time it must perform an action, the motion planning comes from experiential knowledge and the geometric reasoning for doing such task.

To give more information to the robot regarding the human intentions or actions, gestures and speech are commonly used for controlling a robot. Nevertheless, hand gestures are preferred over speech, since the industrial environment is often noisy, and so verbal communication is difficult [101]. The gesture recognition is performed by analyzing two features from an RGB-D camera: a convolutional representation from deep learning and a contour-based hand feature. This permits the robot to recognize the hand gestures of the human and execute specific commands. Moreover, the same authors proposed alternative methods for human tracking [102], such as applying multi-sensor integration (for example, mounting low costs laser range finders and camera systems at specific poses) and using laser readings and train the tracking system according to human body patterns. The authors in [103] suggest that a 3D sensing system is important for human detection and for understanding the behavior. In that regard, a redundant sensory system, such as a combination of 2D laser scanners and sensors that reconstruct the environment in 3D using stereo vision, may ensure safety and be compliant with the ISO 10218-1 and ISO/TS 15066 regulations, which are related to safety for collaborative robots. Nevertheless, those standards involve collaborative manipulators, so the safety concerning the mobile platform should be also considered, as discussed in [104] that analyzes the possible hazards of mobile robotic systems in industry and proposes some countermeasures for those risks. Therefore, the use of sensor fusion or artificial intelligence-based methods are suggested, since they increase the coverage of the information from different sensors and overcome safety problems.

A framework referred to as ConcHRC [105], which represents a more extensive version of the previous FlexHRC framework [106], allows the human operator to interact with several robots simultaneously for carrying out specific tasks. The architecture is composed of three layers: perception, representation and action. In particular, the perception layer elaborates the information related to the human activities and object locations in the robot workspace. The overall scene is measured through motion capture sensors, the objects to be manipulated are detected by employing an RGB-D camera, while the data related to the operator action comes from the inertial sensor of a smart watch.

The teleoperated mobile manipulator presented in [107] is controlled according to the posture for the operator's hand. The tracking of the operator's hand is achieved

by employing a Leap Motion sensor, in which a Kalman filter is used for the position estimation while the orientation is computed by a particle filter. A similar contactless hand gesture recognition system is presented in [108] for safe HRI. This multi-modal sensor interface uses proximity and gesture sensors, and it can identify real-time hand gestures to control the robot platform. An ANN is used for the recognition of hand gestures. Further approaches, such as the one presented in [109], allow to work with a human through an admittance interface, enabling conjoined action. If the human is not in close proximity, the mobile manipulator can perform its routine work autonomously. In particular, the admittance interface is a mechanical connection from the robot hand to the human wrist, and transmit the interaction forces of the human to the robot to perform conjoined movements. When the human needs assistance, the robot can be "called" using the armband that recognizes the human operator's gestures.

1.1.2 An overview of sensors and methods for HRP in industry

Currently, most of the sensors used for robotic systems to perceive the environment and the human operators are of vision type. In particular, in the field of human collaboration with manipulators, the most used sensor is the RGB-D camera. Indeed, the use of new types of sensors may require huge effort to define new algorithms and exploit their characteristics. Moreover, the already developed obstacle detection algorithms would need to be rethought to work with different data types. An interesting new vision sensor is proposed in [110], as Dynamic and Active-pixel VIsion Sensor (DAVIS). This novel sensor seems to have great potential for high-speed robotics and computer vision applications and incorporates a conventional global-shutter camera with an event-based sensors in the same pixel array, enabling the integration of their advantages as well: low latency, high temporal resolution, and very high dynamic range. However, additional algorithms would be needed to fully utilize the sensor properties and cope with its unconventional output, which consists of a stream of asynchronous brightness changes (called "events") and synchronous grayscale frames. In those applications in which the employment of vision sensors is not sufficient or accurate, other kinds of sensors are used instead.

For what concerns IMRs, applications involving human perception mainly exploit laser range finders, to perceive the environment (humans included), usually used in combination with a vision sensor to perform data fusion. The massive use of laser range finders for human-perception goals is expected and justified, as they are present on IMRs both for localization and navigation, and for industrial safety guidelines (safety-rated scanners).

Likewise, mobile manipulators exploit predominantly laser sensors for navigation, while vision sensors are mainly used for the manipulator to perceive the human operator, in particular, to have some visual guidance and be able to replicate the movements of the human. Most of the vision sensors used in mobile manipulators

are of RGB-D type, as they provide accurate information related to the image and depth of the detected object. An alternative way for the robot to perceive the human actions is to make use of an inertial sensor, attached to the human wrist. This lets the robot predict and react based on the detected human movements.

Figure 1.3 gives a visual overview of the most relevant sensors used for HRP depending on the robot type, according to our research.

Fig. 1.3 Relevant sensors for HRP within the industrial context. Vision sensors are highlighted in blue, safety laser scanners rays in red, and wearable sensors in orange.

In order to offer an overview of relevant sensors and methodologies, obtained from the described state of the art analysis, the following material is introduced:

- Figures 1.4 and 1.5 aggregate the sources based on the employed sensors types, also carrying information about the used robot type. It is worth noting that by splitting sensors according to their presence on robots or on human operators, it is clear how (based on what has been analyzed) the sensory equipment for HRP are currently mainly positioned on the robot counterpart.
- Table 1.3 aims to enrich the overview presentation and ease the reader consultation, focusing on algorithms to implement HRP.

Although it is true that the provided material is a useful tool for getting a taste of the trending sensors and algorithms in HRP, it should be considered that it is limited to the survey authors' research and, for this reason, it may not be exhaustive.

In the light of the brought on analysis of sensors and algorithms combinations for HRP in the industrial context, human–robot perception seems to be inevitably interwined to robot-human perception: each information on the human partner behavior is perceived by a robot, interpreted and transformed into action but, at the same time, human reaction to the presence of a robot is affected by the perception the operator has of the robot itself. An optimal reciprocal perception is however not easy to implement, given the lack of a common ground for cognitive skills among humans and robots, which affects the interaction. The robot needs to not only identify the presence of a human but also comprehend the collaboration context, with the aim of effectively assisting human collaborators.

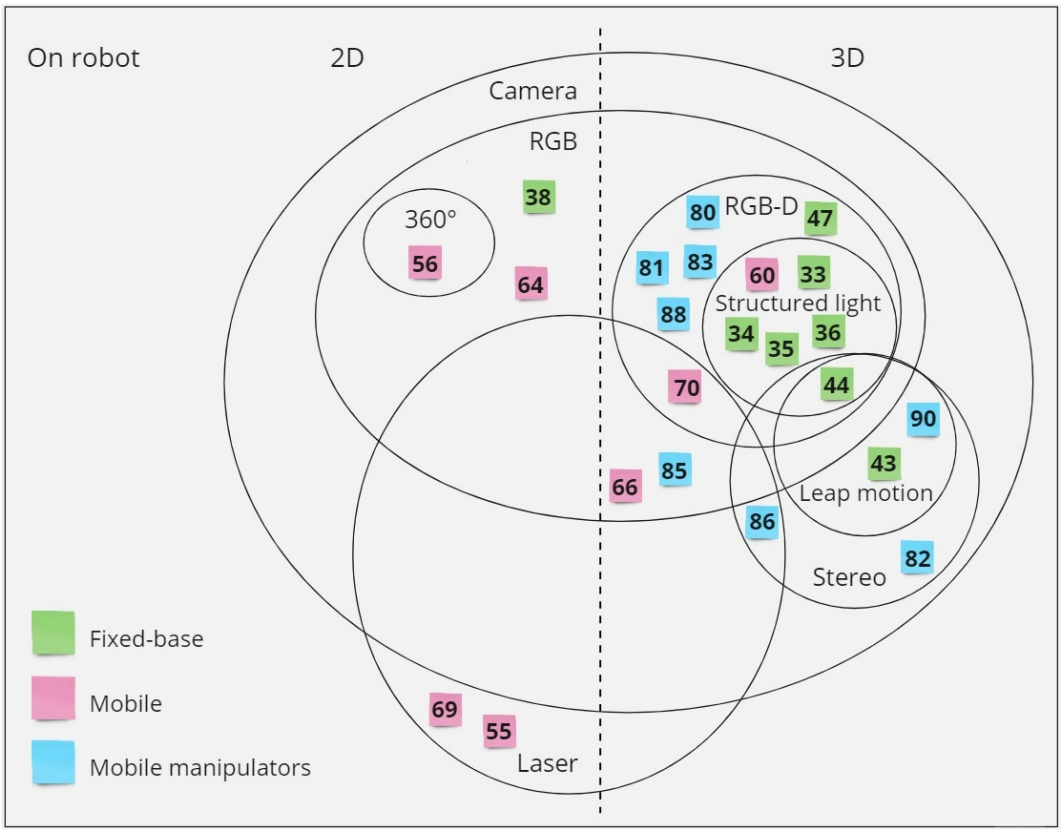


Fig. 1.4 Most relevant sensors for HRP which can be found on the robot.

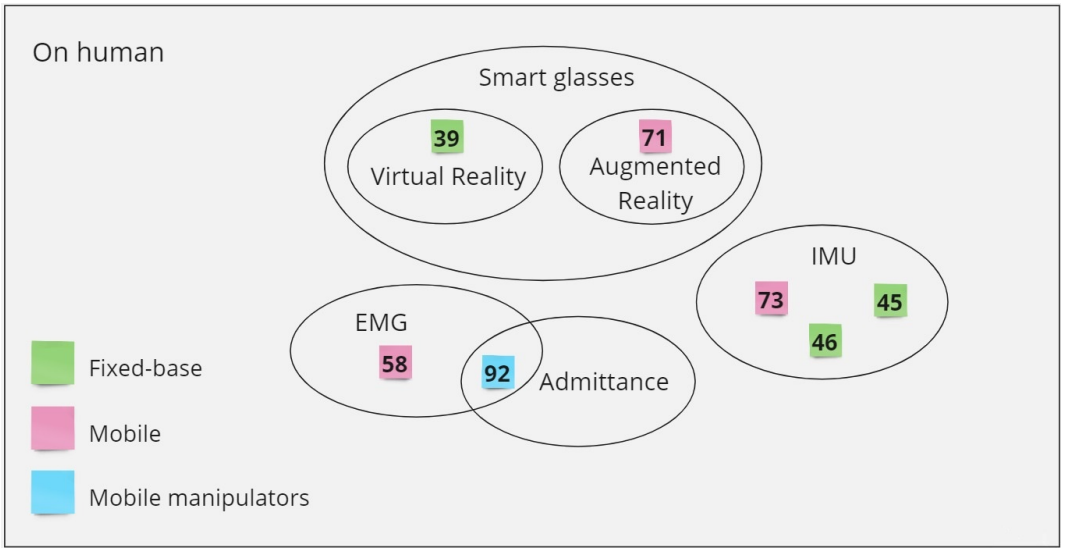


Fig. 1.5 Most relevant sensors for HRP which can be found on the human operator.

Table 1.3 Most relevant algorithms for HRP applications, grouped according to how the perception is implemented.

	Collision Avoidance
[48]	Collision prediction using time-invariant models and neural networks on signal processing.
[50]	Collision prediction based on over-segmentation using forward kinematic model.
[51]	Collision avoidance through generation of repulsive vectors.
[52]	Collision avoidance and re-planning algorithms.
[53]	Collision avoidance exploiting skeletal tracking and positioning of the user.
[55]	Collision avoidance using color detection and allows online path planning.
[56]	Collision avoidance through virtual forces applied on the manipulator.
[58]	The algorithm imposes velocity limitations only when the motion is in proximity of obstacles.
	Aware Navigation
[72]	The robot travels in virtual areas defined a-priori by users.
[76]	Social momentum, teleoperation and optimal reciprocal collision avoidance are used as navigation strategies.
[81]	The planning model is based on RNNs and image quality assessment, to improve mobile robot motion in the context of crowds.
[85]	An autonomously sensed interaction context that can compute and execute human-friendly trajectories.
[86]	Robot navigation takes into consideration the theory of proxemics to assign values to a cost map.
[87]	A confidence index is assigned to each detected human obstacle, enclosed in a 3D box, to avoid it accordingly.
	Environment Representation
[77]	The algorithm adapts to continuous short-term and long-term environment changes with focus on human detection, through feature extraction.
[78]	Scene matching through a representation learning approach that learns a scalable long-term representation model.
[100]	Object localization and tags recognition allow the robot to gather semantic information about the environment.

	Recognition of Objects and Behavior
[83]	The robot assistance is improved using adaptive sensory fusion.
[60]	Teleoperation using coded gestures recognition as input commands.
[61]	Simultaneous perception of the working area and operator's hands.
[89]	Gesture control and eye tracking technologies are used by the robot to interpret human intentions.
[90]	The human motion here is registered and used for skill transfer purposes.
[99]	The motion of the collaborating human operator is monitored to enable specific robot actions.
	Environment Representation
[101]	Gesture recognition is performed considering a convolutional representation from deep learning and a contour-based hand feature.
[102]	Human tracking is implemented and trained according to human body patterns.
[103]	Human detection and behavior recognition is implemented exploiting redundancy of sources to reconstruct the environment.
[105]	The information related to the human activities and object locations in the robot workspace are used for the approach.
[107]	The operator's hand pose is estimated using a Kalman filter and a particle filter.
[108]	Real-time hand gesture recognition is implemented using a ANN.
[62]	The gestures used to command the robot are processed and classified by an ANN.
[63]	The motion of the human operator's upper body is tracked, with a focus on objects manipulation.
[64]	Sensor data fusion algorithm for prediction and estimation of the human occupancy within the robot working area.
	Conjoined Action
[73]	The 360-degree scene is enriched by interactive elements to improve the teleoperated navigation.
[75]	Teleoperated navigation is implemented through a hybrid shared control scheme.
[88]	Enhanced perception-based interactions using AR for collaborative operations.

[109] Interaction forces of the human are transmitted from the admittance interface to the robot to perform conjoined movements.

To achieve a comparable level of cognition among humans and robots, *multi-modal sensor fusion* is the preferred solution for the robot perception, either when considering environment perception and human perception, which are obviously interlinked. Autonomous systems rely on data fusion as an essential module to implement perception. The fusion processing of multi-modal data involves analyzing it at a raw level and interpreting it at a higher level to extract meaningful features. Fusing together data coming from a set of multi-modal sensors can potentially bring out interesting information which, if single-source data only were considered, would have not emerged. This facilitates the implementation of a more informed perception of the environment and the humans within it.

Additionally, what emerged is that safety considerations strongly affect the choice of algorithms and sensors: their combination must aim at being compliant with safety requirements imposed by standards. Moreover, along with safety, an adequate interface design has a crucial impact on the development of HRC tasks.

It is clear that where sensor accuracy is lacking, algorithmic complexity aims for compensation, to achieve an overall reliable interaction. As can be easily inferred, the balance between the accuracy of the sensor and the computational effort of the algorithm is highly dependent on the available resources and on the application requirements.

1.2 Industrial mobile robots: a quest for autonomous navigation

Mobile robots have been widely employed in many different fields, since they can be adapted to a vast range of applications. As mentioned before, service robotics brought in innovative technologies and solutions that are slowly migrating to the industrial sector, where smart robots will be able to learn tasks without formal programming, and to autonomously cooperate with other smart devices and factory workers [111].

Mobile manipulators (i.e., robotic arms on mobile bases) are entering warehouses and factories, making obsolete the idea of an industrial robot strictly associated to a fixed and caged manipulator: AMRs, fixed-base cobots, mobile manipulators, mobile cobots and enhanced manual workstations –fully integrated within the automated lines– are going to characterize the smart factories of the future [112].

Although the concept of autonomous mobile robots is not new (first generic AMR patent is from 1987 [113]), its application to an industrial context has come up in recent developments and is expected to significantly increase in the near future.

In fact, logistic systems, e.g., AGVs, were foreseen to make up 66% of the total forecast of service robots (with different degree of autonomy) from 2019 to 2021 (Figure 1.6) and, Industrial Mobile Robots (IMRs), in general, represent essential elements of the present and future production line and logistics workspaces, as foreseen by a report by the International Federation of Robotics (Figure 1.7).

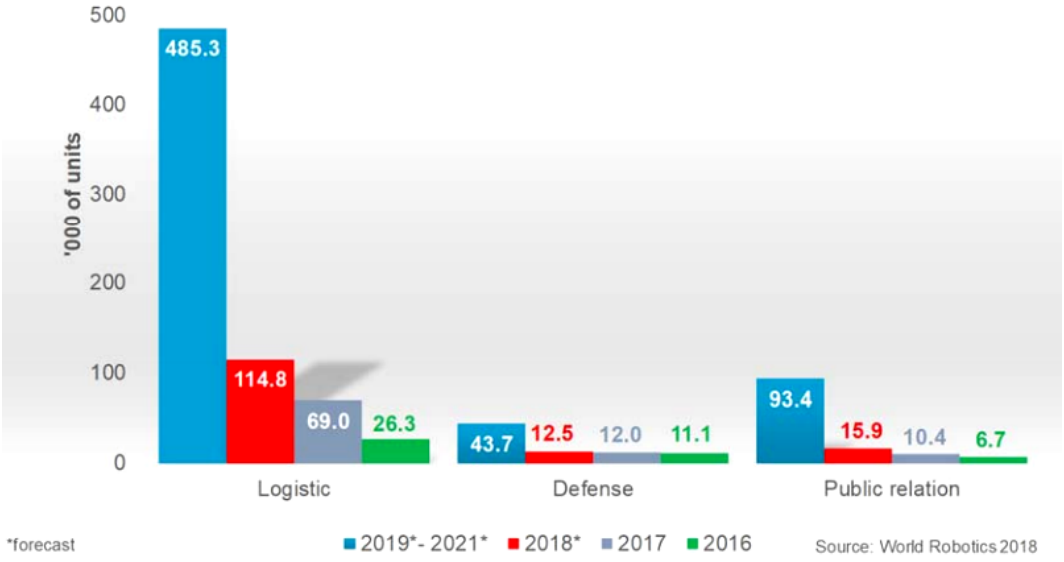


Fig. 1.6 Service robots for professional use in main applications. Unit sales 2016 and 2017, forecast 2018 and 2019-2021 [114].

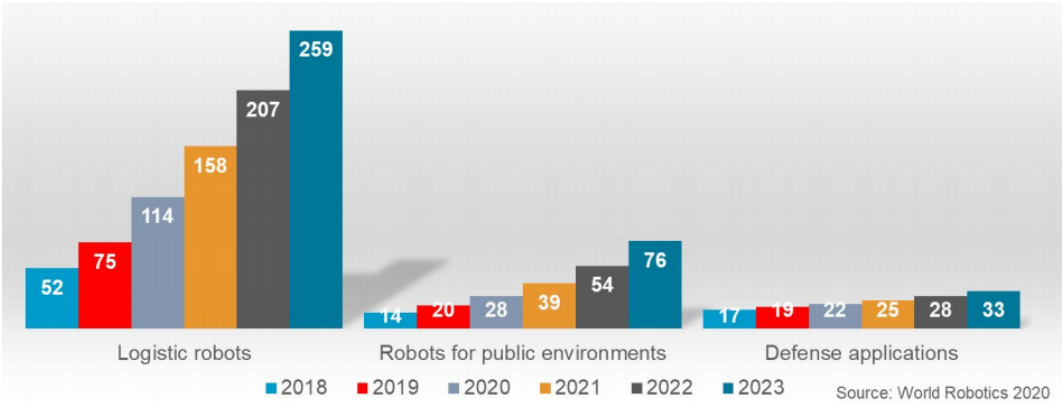


Fig. 1.7 Service robots for professional use. Top 3 applications unit sales 2018 and 2019, potential development 2020–2023 (thousands of units) [115].

In particular, AMRs fall within the so called *new robots*, i.e., robotic systems compliant with collaborative operations, characterized by a higher degree of autonomy and capable of autonomous decision making. According to a new IDTechEx

research [116] about New Robotics and Drones forecast on the 2018-2038 period, we will see a dramatic increase in new robotics deployment, with the support of decreasing hardware and software development costs (Figure 1.8).

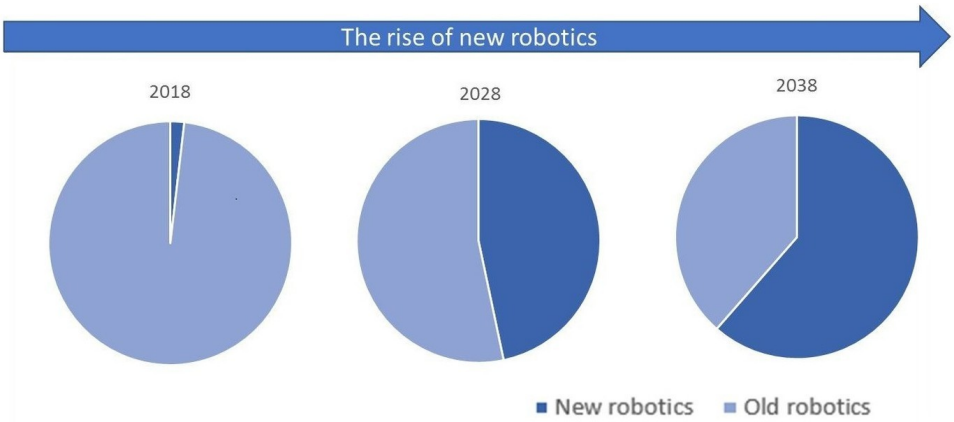


Fig. 1.8 Long-term forecasts of the market evolution for new robotics according to the New Robotics and Drones IDTechEx report.

Fig. 1.9 Service robots and new robotics trends.

According to the "Mobile Robotics in Logistics, Warehousing and Delivery 2022-2042" report by IDTechEx [117], the market for mobile robots (including trucks and drones) in logistics, delivery and warehousing is likely to reach a staggering $83 and $334 billion in 2032 and 2042, respectively. In particular, as AMRs do not require any additional infrastructure change – saving cost and time, while allowing for scalable and flexible fleets, it is assessed that the market for such technology will continuously grow by 2042, backed up by prominent investments and acquisitions on this technology in recent years.

Starting with the forth industrial revolution, the main components of the production line were uprooted from their fixed position and role: industrial manipulators enter the human operator working space in the guise of collaborative robots (or cobots), the traditional Automated Guided Vehicle (AGV) gets smart, autonomously finds its way through the distributed working stations and becomes increasingly intelligent, turning into the present-day Autonomous Mobile Robot (AMR). The AGVs are mobile platforms that typically perform repetitive tasks, such as material transportation, following a pre-defined path within industrial environments [118].

Human-accessible workspaces are traditionally separated from the AGV operational space due to safety reasons, since most AGVs are not capable of intelligent decision-making and require a particular infrastructure setup [70]. On the other hand, an AMR is able to sense its surroundings in order to create a model of the environment and locate itself in it, leading to the capability of operating in an unknown or partially known environment. Specifically, AMRs allow the improvement of flexibility of the working setup, since they get rid of the path constraints of classical

Automated Guided Vehicles (AGVs) [119]. As stressed previously, there are several reasons behind the push towards autonomous IMRs in manufacturing and logistics, including:

Flexibility. Autonomous navigation also allows AMRs to be more flexible in terms of their operation. These machines can be programmed to navigate around obstacles and adjust their routes in real time based on changing conditions. This flexibility allows companies to adapt to changing production requirements and handle unexpected events more effectively.

Efficiency. When needed, mobile robots able to navigate autonomously are able to move across facilities quickly and efficiently, without the need for human intervention. This can help to reduce cycle times and improve overall throughput, relevant in industries with high production volumes.

Safety. By reducing the number of workers needed to perform manual material handling tasks, companies can reduce the risk of accidents and injuries in the workplace.

Scalability. AMRs with autonomous navigation capabilities can also help to enable scalable operations. As companies grow and their production volumes increase, they can easily add more AMRs to their fleet to meet demand. This scalability makes it easier for companies to adapt to changing market conditions, e.g., to handle unexpected spikes in production.

Cost savings. AMRs with autonomous navigation capabilities is that they can help to reduce labor costs. By automating the transportation of goods within a facility, companies can reduce the number of workers needed to perform manual material handling tasks, achieving significant cost savings over time.

Nevertheless, considering the current and future direction unfolded in the previous sections, autonomous navigation has also to take into account human presence, either for cooperation or collaboration in shared workspaces. Hence, autonomous navigation can come in different flavours, depending on the application and the environments constraints.

The following contributions highlight the different degrees of autonomous navigation capabilities, including a planning algorithm to enable a soft transition from the AGV's fixed path towards autonomous but safe AMR navigation, patrolling for multi-agent systems, and a proof of concept of mobile robots networks towards human-shared scenarios.

1.2.1 Supervised global path planning: from AGVs to safe path-constrained AMRs

This section presents the implementation of a three-level path planning procedure, which allows: (i) the imposition of a set of waypoints, tending to a safe path, generated by a supervisory planner on the basis of a static map of the environment (not necessarily fully updated), (ii) the generation of a global path including such waypoints, grounded on a a cost-based algorithm, which also takes into account the obstacles not included in the static map, but detected at the beginning of the global planning phase, and (iii) the avoidance of dynamic obstacles appearing during the robot motion, thanks to the action of a local planner. The procedure has been experimentally tested to plan the motion of a differential mobile robot.

After a brief introduction on the topic and related works, a presentation of the proposed procedure is given, at first providing some theoretical details about the supervisory planner, then describing the algorithm implementation including the obstacle avoidance capability. Finally, the results gathered from some testing are provided.

Path planning for mobile robots

In general, path planning for mobile robots is mainly carried out at two levels: *global planning* and *local planning*. The global planner computes (offline) the path for a well known environment map, so allowing the robot to navigate from a starting location to a destination one, avoiding the known static obstacles, while optimizing specific objectives, e.g., searching the shortest path (so to reduce time traveling and energy consumption). The local planner updates (online) the path of the robot, taking into account the information coming from its sensors. This way, the robot still follows the path generated by the global planner when possible, but has it modified when unexpected obstacles appear, to avoid them.

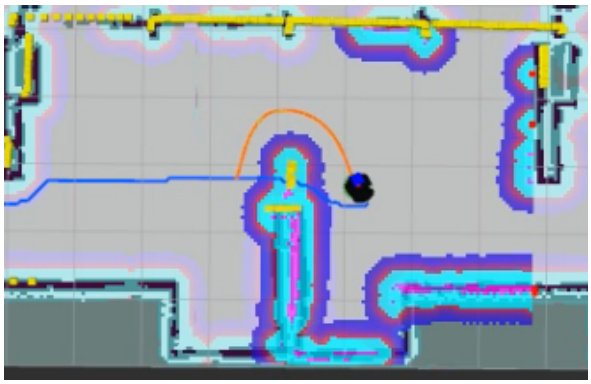


Fig. 1.10 An example path plans visualization (top view). The global plan is coloured in blue and the local plan in orange.

Numerous algorithms have been proposed and developed by researchers for path planning, depending on the tasks to be performed and the environment. Among the heuristic-based algorithms, it is worth reporting the following:

- *Dijkstra algorithm* [120]: computes the shortest path between two nodes in a map; however, its execution needs to evaluated too many nodes, making it an overall low-efficiency process.

- *A* algorithm* [121]: based on Dijkstra, A* is able to solve the issue described above. In this algorithm, a function evaluates the distances between nodes, and their cost to reach the goal point. Nevertheless, this requires relatively long computational time, making A* not suitable for performing sequential tasks in real-time [122].

- *D* algorithm* [123]: allows to carry out such response time constrained tasks. Based on A*, D* is a dynamic re-planning algorithm, which takes into account the direction of the robot position as expressed by a series of states, whose values are updated each time the elements on the map change. Nonetheless, this still requires high computational effort, since the re-planning phase needs to calculate twice each state on the environment [124].

On the other hand, to mitigate the computational burden introduced by the algorithms presented above, probabilistic methods come in handy. Two relevant examples are:

- *Probability Roadmap* (PRM) [125]: free spaces on the map are randomly sampled, and the planner tries to connect the generated points into a roadmap feasible motion. This method is commonly exployed in case of large maps, where the use of other algorithms may increase the overall computational cost. Although, this method does not guarantee the shortest path [126].

- *Rapidly-exploring Random Tree* (RRT) [127]: feasible trajectories are obtained by growing expanding *trees* starting from the starting point, across random points called *seeds*, until the tree is connected to the goal point.

Since the release of the original RRT algorithm, researchers have proposed several other improved versions. For example, the method presented in [128] consists in "planting" a tree in both the starting and the goal points, and then expanding both trees in the whole world map until an intersection point is found; the algorithm presented in [129] controls the tree edges growth direction and density of the RRT* variant (an RRT version converging to the shortest path [130]). The main drawback of sampling based stochastic searches is that, in general, path cost is not taken into account, leading to a solution solution that is not necessarily the optimal one [131].

Other planning algorithms are inspired by natural phenomena. One of the most popular is the Genetic Algorithm (GA) [132], based on the natural selection theory

and applied as a path searching algorithm in the field of robotics. The cost function for computing the best path is structured similarly to a chromosome, where each location is considered as a gene [133].

Another frequently employed method consists in computing off-line the considered safe path and controlling the robot in such a way that it goes to the destination point following a set of waypoints. In this case, it is possible to use the feedback linearization technique, in order to design a control system for path following, like in the fuzzy logic based control architecture proposed in [134]. Likewise, some researchers proposed navigation functions based on artificial potential fields to solve the robot motion planning [135], and a feedback control law that guarantees the robot reaches the destination point while avoiding obstacles [136].

The proposed supervisory algorithm

The developed supervised planning takes advantage of the algorithm developed in [137]. In particular, it integrates a *supervisory planner* on top of the common two-level planning hierarchy, seeking to make the robot reach a goal position traversing a virtual track previously identified as safe (background details in Appendix A). Consider an industrial scenario, e.g., a warehouse or factory plant, where wheeled mobile robots move along predefined routes that are considered safe for a mobile platform to move (no unexpected obstacle is assumed): the a-priori desired path computed by the algorithm, proposed in [137], tends by construction to a curve representing a safe route. The computed path takes on the role of the guide tapes in the most traditional industrial mobile navigation set-up of AGVs.

The path planning algorithm we propose has a hierarchical structure based on three levels: the Supervisory Global Planner (SGP), the Global Planner (GP) and the Local Planner (LP). Going down across these levels, the robot awareness of its surroundings increases, transitioning from a “blind" planning (only based on the static map) to a dynamic-obstacles aware system (Figure 1.11).

We integrated the planned waypoints at the GP level, allowing the robot to follow the supervisory planned trajectory (if possible) thanks to a cost-based algorithm. The avoidance of unexpected obstacles that may appear during the robot motion is left to the standard, lowest level LP, which further modifies on-line the path computed by the GP, when necessary. The outcome of the algorithm designed in [137] is computed upon a known environment and is *collision free* (i.e., it does not intersect the curves describing the boundaries of the obstacles), and tends to a preassigned algebraic curve, which is considered *safe* for motion.

The aim of the supervisory plan is to obtain a path that tends to a planar curve $\mathscr{V}$, where

$$\mathscr{V} := \{x \in \mathbb{R}^2 : p(x) = 0\}. \tag{1.1}$$

Here, $p(x)$ is a given polynomial function. By using the algorithm given in [138] (omitted here for brevity), it is possible to compute two vector fields $\phi(x)$ and $\psi(x)$, whose entries are polynomials in x, such that the curve $\mathscr{V}$ given in (1.1) is *attractive*

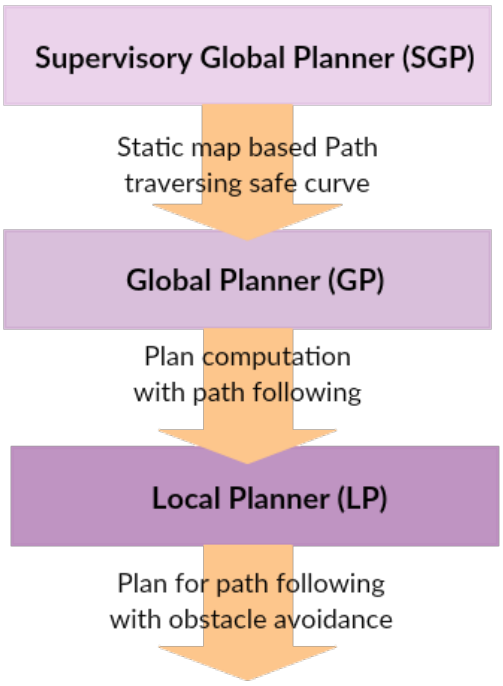


Fig. 1.11 Planning hierarchy schematics.

and *invariant* with respect to the dynamical system

$$\dot{\xi} = \phi(\xi) + \psi(\xi)\alpha(\xi), \tag{1.2}$$

where $\alpha(\xi)$ is an arbitrary function. Although the trajectories of system (1.2) converge to the set $\mathscr{V}$, they are not necessarily collision free. Thus, in order to ensure collision avoidance, the results given in [138] are coupled with classical navigation functions [135].

Let $b(x)$ be a function describing the boundaries of the obstacles (i.e., $b(\xi) = 0$ implies that the point ξ belongs to the boundary of one of the obstacles) and let

$$\mathscr{W} := \{x \in \mathbb{R}^2 : b(x) > 0\}$$

be the workspace of the mobile robot, which is assumed to be a connected set. Furthermore, we assume that

$$\mathscr{V} \cap \{x \in \mathbb{R}^2 : b(x) = 0\} = \emptyset,$$

i.e., that $\mathscr{V}$ is actually safe for motion. Thus, define

$$r(\xi) := \frac{p^2(\xi)}{b(\xi)}, \qquad \eta(\xi) := \left(\frac{\partial r(\xi)}{\partial \xi}\right)^{\top}.$$

By construction, the function $r(\xi)$ is nonnegative $\forall \xi \in \mathscr{W}$, it is zero if $\xi \in \mathscr{V}$, and it tends to $+\infty$ if ξ tends to $\mathscr{W}$. This implies that, letting $\beta(\xi)$ be a nonnegative function such that $b(\xi) = 0 \implies \beta(\xi) \neq 0$, the trajectories of

$$\dot{\xi} = -\eta(\xi)\beta(\xi) \tag{1.3}$$

are collision free (see [137, Prop. 2]).

Thereby, the supervisory algorithm is obtained coupling systems (1.2) and (1.3). Namely, by letting $\zeta(\xi)$ be an arbitrary function (which is amenable for further optimization, see [136]) and k be a positive constant, it can be guaranteed that the trajectories of the dynamical system

$$\dot{\xi} = b^2(\xi)\phi(\xi) + b^2(\xi)\psi(\xi)\zeta(\xi) - k\,p^2(\xi)\eta(\xi) \tag{1.4}$$

tend to $\mathscr{V}$, while avoiding collisions with the obstacles.

Integration with ROS planners

The algorithm has been validated in [137] only in a pure theoretical context, here we integrated it in a three-levels planning procedure and tested on a differential robot. In order to adapt the theoretical algorithm to a physical implementation, the supervisory algorithm has been executed using a real environment map (the laboratory we have mapped) and taking into account the physical dimensions of the performing robot. Note that, at this level of description, we will omit map details and parameters tuning/adjustment specific to the chosen robot (more details about the physical set-up are provided later on) to highlight the general validity of the approach.

The SGP algorithm has been written in MATLAB, while the navigation driving the robot takes advantage of ROS (Robot Operating System) [139] tools. Specifically, the ROS *Navigation Stack* [140] (NavStack) provides the user with off-the-shelf packages for mapping, localization, navigation and reference frames tracking. In order to guarantee an accurate interpretation of the supervisory algorithm output (i.e., a set of 2D desired points converging to the preassigned safe curve to be traversed on the static map), the ROS `/world` frame origin (reference frame for the whole ROS coordinate frames transform tree, managed by the `tf` package) has been aligned to the frame origin used in MATLAB. Then, the output plan has been conveniently packed in an array of desired positions stored on the ROS Parameter Server, exploiting the MATLAB Robotics System Toolbox™ [141]. The ROS framework provides the `actionlib` library stack [142], that allows the user to interact through a standardized interface with preemptable tasks, which in our case correspond to the desired poses for the mobile platform to reach. A custom Python ROS node is in charge of sending a request, through a ROS Action Client, to the Action Server, via a message containing the information about the next goal position to be reached. ROS actions represent the ideal Client-Server-based mechanism for goal achieving, since while the whole operation is brought on, a feedback message about its status can be sent to the client node. A sketch of the software set-up is presented in Figure 1.12.

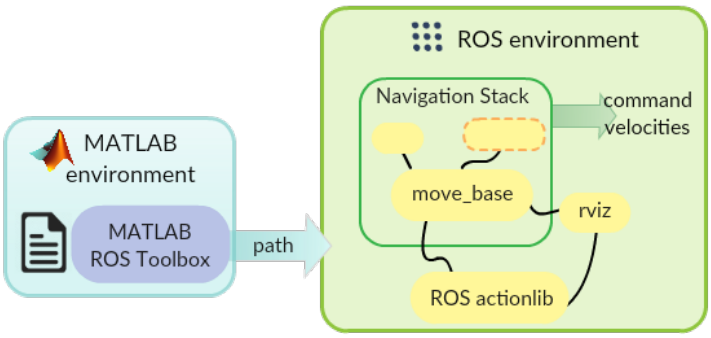


Fig. 1.12 Software setup for implementation of the proposed algorithm.

In our case, the GP main role is that of generating a plan that must be as close as possible to the path computed by the SGP. Instead, the LP algorithm computes a short-term plan considering a *local costmap* and a predefined portion of the GP-generated long-term plan, while exploiting sensors' data to guarantee obstacle avoidance of unexpected objects (either moving or fixed). Moreover, the whole planning performance relies on a series of tunable *costmap parameters*, whose values depend on the final planning objective and on the physical characteristics of the robot. ROS package `move_base`, as part of the ROS NavStack, provides several ready-to-use global planners (e.g., planner implementing Dijkstra and A* algorithms. To deepen what was briefly introduced before, A* is a combination between Dijkstra's algorithm and Best First Search algorithm, meaning it is a weight-based process, or informed algorithm, where the graph is explored and expanded only in nodes resulting convenient, based on an assigned cost with heuristic content. Indeed, A* inherits the benefits of a uniform-cost search from Dijkstra's, adding heuristics to efficiently find an optimal solution in less time [143].

Supervisory planner costs-mechanism

Our GP algorithm is built upon the A* cost-based structure (background details in Appendix A): with the aim of forcing the robot to follow the set of waypoints computed by the SGP within the GP, a low cost is assigned to the preferred points to be traversed. To clarify how this works, the cost mechanism of the A* search algorithm is outlined hereafter. Consider a pathfinding problem, where an optimized path must be found between two points on a 2D costmap: at each main loop, starting from the source cell the algorithm expands the most suitable, and for construction most "convenient", cell depending on a function $f(c)$, defined as

$$f(c) = g(c) + h(c), \tag{1.5}$$

where c is the currently expanded cell, $g(c)$ denotes the cost from the source position to the current one, and $h(c)$ represents the heuristic estimated cost from the current cell to the goal (destination) cell. At each iteration the expanded cell, i.e., the cell

whose neighbour cells are visited and their costs computed (Figure 1.13), is the one having the lowest $f(c)$.

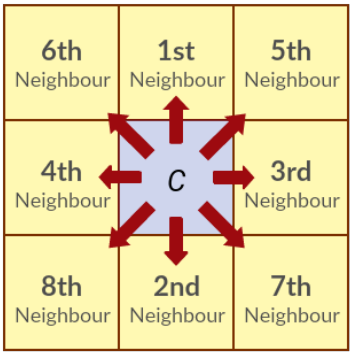


Fig. 1.13 Expansion of the most convenient cell.

Therefore, the $f(c)$ function encloses an estimated path cost from the source cell to the destination cell, traversing cell c. While $g(c)$ can be easily calculated, $h(c)$ computation usually involves an approximation method, since the exact heuristics computation, i.e., computing the actual value attained by the cost-to-go function at cell c, would be unfeasible especially in terms of time. Among the approximated heuristic functions at disposal, the Euclidean distance has been chosen, obtaining an $h(c)$ function defined as:

$$h(c) = \sqrt{(c.x - dest.x)^2 + (c.y - dest.y)^2}, \tag{1.6}$$

where *dest* is the destination cell and the expressions $.x$ and $.y$ extract the fields corresponding to the x and y coordinates of the considered cell, respectively.

To ensure path following, a low cost is associated to the positions produced by the SGP, so to guarantee their presence in the output plan. In particular, the passage on these positions is favoured by the introduction of a new cost h_{WP}, defined as

$$h_{WP}(c) = \sqrt{(c.x - WP.x)^2 + (c.y - WP.y)^2}, \tag{1.7}$$

where WP is the next expected waypoint and the expressions $.x$ and $.y$ extract the fields corresponding to the x and y coordinates of the considered cell, respectively. The quantity in (1.7) is the Euclidean distance from the next expected waypoint, and assigns costs which penalize positions far from the desired path, based on four possible cases: (i) the currently visited neighbour cell n is a waypoint, (ii) the currently expanded cell c is a waypoint, (iii) all waypoints have been already traversed, and (iv) none of the previous cases is valid.

This way, the planner will acts as A* when far from the SGP-computed path. Note that we consider all discrete points making up the supervisory algorithm output path as waypoints that we would like the robot to traverse when in their proximity.

Further details on how the costs have been assigned to achieve an overall equilibrated and suitable cost system are given in Algorithm 1 at page 45, reporting the pseudocode for the proposed algorithm. Handling of the above enumerated possible cases can be found at lines 10, 12, 14 and 16, respectively. Note that highlighted lines point out edits with respect to the original A*, to have a direct comparison. The inputs to the algorithm are the environment map and the start and destination cells, while the output is the computed path. The proposed GP boasts the capability of reaching a goal position not necessarily in the shortest way, but as safely as possible.

Hardware and Software setup and testing

In order to test the supervisory algorithm with path following and obstacle avoidance we have decided to employ a Pioneer 3DX mobile robot [144] equipped with a SICK LMS200 laser range finder [145] with 10-meter range and scanning angle of 180 degrees, and a Raspberry Pi [146] 3 Model B mounting an ARM Cortex-A53 (x4 core) CPU (1.2 GHz) and 1-GB RAM as a processing unit

Note that the physical specifications of the robot (Figure 1.14) are crucial for modelling the kinematic equations necessary for executing the supervisory algorithm, to suitably adapt ROS visualization and map inflation parameters, and to generate proper commands taking into account the maximum allowed speeds and accelerations.

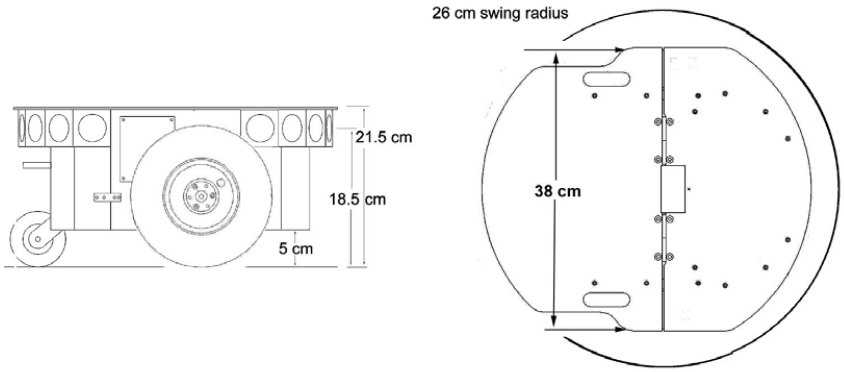


Fig. 1.14 The Pioneer3Dx mobile base dimensions [144].

Figure 1.15 shows the interplay between hardware components: the Raspberry Pi receives environment states via laser scan data and sends proper commands to the robot. Simultaneously, the robot states and environment information are exchanged between the Raspberry Pi board and a computer: on both processing units are running specific processes (nodes) of the ROS framework, respectively the ROS master node and sensor drivers nodes on the former, and navigation and visualization nodes on the latter, constituting a typical distributed ROS system.

As already said, the first step towards a practical implementation of the SGP algorithm has been that of feeding the algorithm implemented in MATLAB with

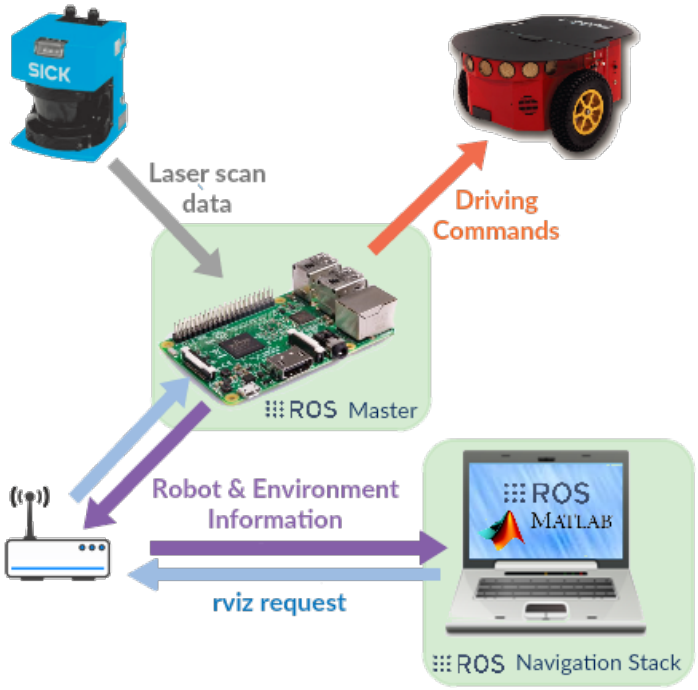


Fig. 1.15 General schema mapping hardware to the software distributed system.

the real map of the environment in which the robot can move, where fixed obstacles have been conservatively isolated, by describing their boundaries. The map has been generated by ROS through the `gmapping` [147] package, which provides laser-based SLAM (Simultaneous Localization and Mapping) built upon the OpenSlam's Gmapping algorithm [148]. The output map (saved as a *.pgm* file) describes the environment using the binary occupancy grid format, in which a black-coloured cell represents an occupied area (obstacles), while a white one indicates free space. Such described map is interpreted by the ROS visualization tool, *rviz* [149], as a base upon which building the costmap. A costmap inflates costs based on the occupancy grid information and physical features of the performing robot. In our case the original map of the whole research laboratory has been restricted to an area where unexpected obstacles are less probable, to fall within the assumed conditions, leading to a 160x190 cells grid with a resolution of 0.05 m/cell.

To integrate our algorithm in the ROS planners structure, the default global planner of the `navfn` package has been replaced exploiting the ROS plugin mechanism, in order to have access to a simpler standalone piece of code, to be customized for the integration of the required path following capabilities. We have used the ROS *cpp* library to adhere to the `nav_core::BaseGlobalPlanner` C++ interface provided by the `nav_core` package (by overriding some specific methods), and employed by the `move_base` package to drive the mobile platform. The employed local planner exploits the `TrajectoryPlannerROS` wrapper, which adheres to the

`nav_core::BaseLocalPlanner` interface. This local planner version implements the so called *trajectory rollout* algorithm: the robot's control space is discretely sampled and, for each sampled velocity a forward simulation is executed starting from the current robot state, to predict the robot motion if the considered velocity were applied for a restricted period of time. Each simulated trajectory is assessed based on some evaluation parameters, e.g., proximity to the global path, to obstacles, to the goal, and the speed. Among the evaluated trajectories, those which would potentially lead to a collision are discarded, while the trajectory obtaining the highest score is chosen and the relative velocities are sent to the mobile robot. This procedure is looped at each motion step [150]. In order to make the resulting path track as precisely as possible the plan provided by the upper level, the knowledge of the LP about the GP long term path is reduced, influencing the short-term plan generation, since a wider overview on the tracked plan would result in local path optimization.

Experimental results

The aim of the executed tests is to demonstrate the capability of the robot of reaching the desired goal position traversing the safest path as soon as possible. Note that the algorithm execution time is affected by: (i) the chosen GP and LP, (ii) the computational capability of the computer executing the navigation and planning instructions, and (iii) the goal to be reached. Hence, the given execution times (relative to each specific test case) are to be considered for our specific choice of planners, for reaching a specific goal position, and running the heavier algorithm nodes on a high-performing computer.

The chosen values for the local costmap dimensions represent a tradeoff for ensuring (i) a faithful tracking of the GP plan and (ii) obstacle avoidance. For all the tests, the starting and destination points (expressed in meters) with the format (x,y) have been set to $(4,7)$ and $(2.5,2)$, respectively (Figure 1.16); as can be seen, the SGP plan tends to the the desired safe curve, avoiding any intersection with the curves surrounding the obstacles present in the static map. The execution of the algorithms in all test cases has been collected in the video that can be found in [151].

Test Case 1: absence of unexpected obstacles

Here we consider no unexpected obstacles, i.e., not present in the static map (Figure 1.17a). As shown, the GP (green line) faithfully tracks the SGP generated path, and the LP also follows precisely the GP, thanks to the ad-hoc tuning of local costmap parameters. Note that, if the A* algorithm were used, it would generate a straight plan towards the goal position. However, we search for an optimal solution (in terms of safety) that leads the robot to reach a safe virtual track as soon as possible, making it part of the plan. The time needed to reach the goal position is 43.05 s.

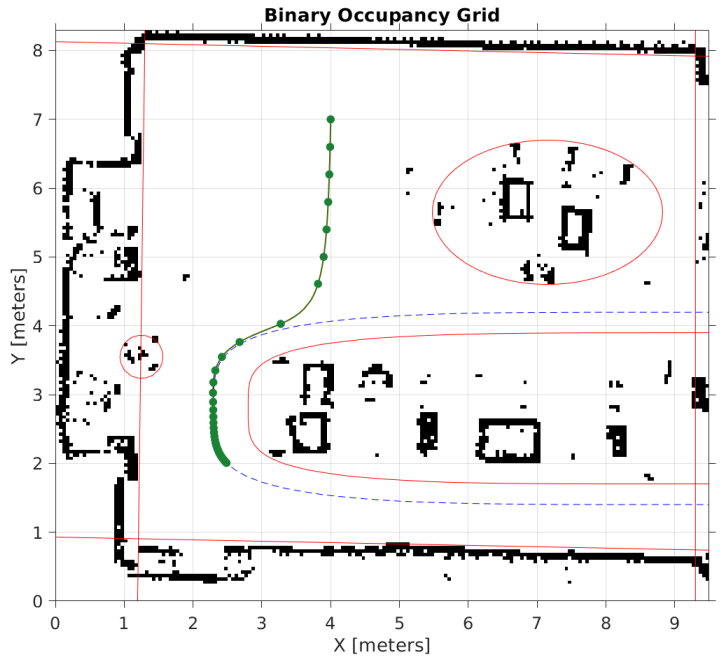


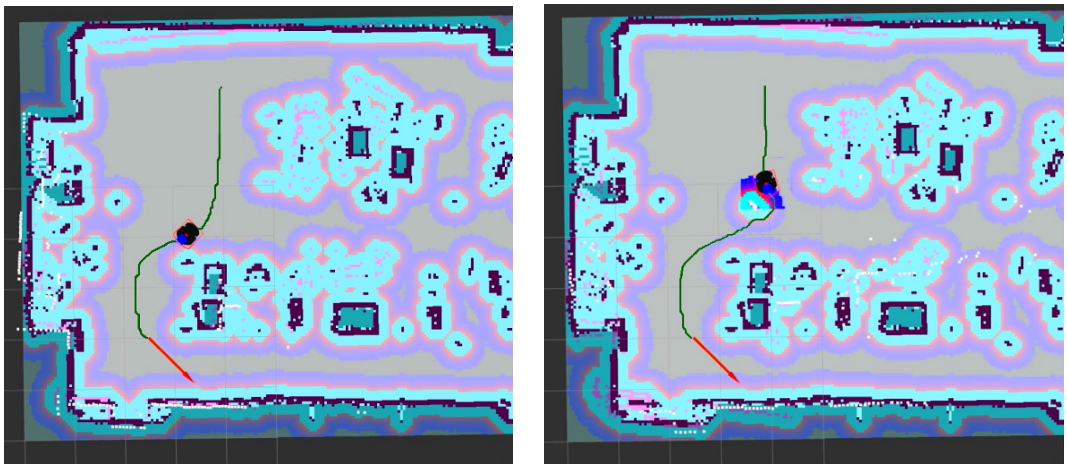
Fig. 1.16 MATLAB simulation plot for the SGP algorithm. Red lines: obstacles' boundary curves. Blue dashed line: desired safe curve. Green solid line: SGP plan; a subset of the computed waypoints is highlighted with green dots.

Test Case 2: behaviour with obstacles not in static map

Here we consider the presence of an unknown object, i.e., not included in the static map, but detected before the GP path computation (Figure 1.17b). The reported screenshot shows that the GP-generated path deviates from the desired path since the sensor data detected an unexpected obstacle; being the global costmap updated with this new information, the traversed path going backwards, i.e., from the destination point to the starting point, will be the same. The execution time is 57.34 *s*.

Test Case 3: behaviour with dynamic obstacles

Here we deal with the appearance of an obstacle during the robot motion, i.e., after the GP has computed the path (Figure 1.18). As can be seen, initially the GP algorithm calculates a path consistent with the desired one as no new obstacles are present from the beginning. Instead, when an unexpected obstacle is detected , the LP re-plans the path in order to avoid it, trying to go back to the desired curve, when possible. The present test case has been considered in order to show that, while the robot moves towards the safety curve, if something or someone enters the scene, a safe behaviour is preserved, as shown in the "Test Case 3" section of the video, in which a human operator appears at 05:55 and the reaction of the LP lets the robot avoid him. The execution time is 52.39 *s*.



(a) *rviz* view during Test Case 1 execution.

(b) *rviz* view during Test Case 2 execution.

Fig. 1.17 Test Cases 1 and 2.

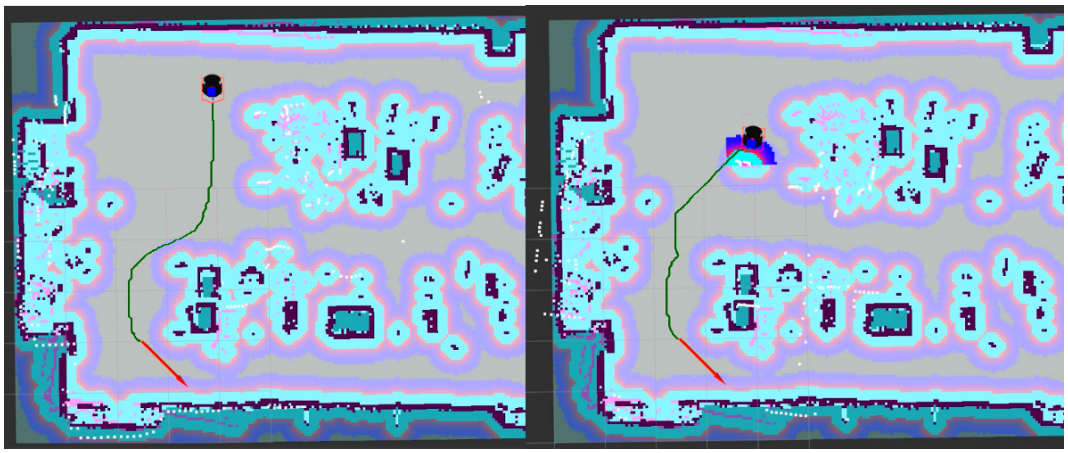


Fig. 1.18 *rviz* view during Test Case 3 execution. The left figure reports the plan before the obstacle appears, while the right one shows the re-planning action.

Therefore, through the presented supervisory planner algorithm, the physical safe paths that AGVs traditionally followed are upgraded to virtual safe paths that autonomous AMRs can be induced to follow in those areas where safety has greater relevance than time efficiency, e.g., in human-shared spaces where mobile robots are expected to follow their lane. In fact, the SGP's plan induces the AMR towards a path considered to be safe. Then, the GP extrapolates waypoints from the SGP path and follow it using a modified version of the well-known A* algorithm. Moreover, given the custom cost-based mechanism of our algorithm, the original A* path search (including obstacle avoidance) is executed when the mobile robot is located significantly far away from the waypoints. In such a way, whenever the platform is sufficiently close to one of the desired waypoints, it is "attracted" by the path computed by the SGP.

Algorithm 1 Proposed algorithm with path following.

Input: map, start position, goal position
Output: path

```
/* Set-up cell expansion lists */
1  openList                                                    // Declare the open list
2  closedList                                                  // Declare the closed list
   /* Insert starting cell in the open list */
3  openList.insert(start)
   while openList is not empty do
       /* Pop cell c with lowest f(c) off the openList */
4      c = openList.pop()
          /* Push cell c into the closedList */
5      closedList.push(c)
       foreach neighbor n of cell c do
6          if neighbor n is the destination cell then
7              n.parent = c                                    // Assign parent node
8              Stop searching
9          else if neighbor n ∉ closedList and is not blocked then
10             if n is a waypoint then
11                 Assign minimum values to n.g, n.h and n.h_WP
12             else if c is a waypoint then
13                 Assign minimum values to n.h and n.h_WP
14             else if waypoints have been traversed all then
15                 Assign minimum values to n.h_WP
16             else
17                 if ( n ∈ openList) and ( new f(n) > old f(n) ) then
18                     Ignore n
19                 else
20                     n.g = n.g + distance from c to n
                           n.h = distance from n to destination cell
                           n.h_WP = distance from n to next waypoint
                           /* Compute f(n) */
21                     n.f = n.g + n.h + n.h_WP
                           /* Assign parent cell */
22                     n.parent = c
                           /* Insert cell n in the openList */
23                     openList.insert(n)
   /* Trace the output path from the destination cell */
24 p = dest
   while p.parent is not null do
25     Path.push(p)
       p = p.parent
26 Path.push(p)
```

1.2.2 Path planning in formation and collision avoidance for multi-agent systems

The attraction towards predefined curves can also be exploited to achieve multi-agent collision-free patrolling and formation control. To this aim, a research work I contributed to and presented in [16], showcases a centralized algorithm grounded on (i) stability analysis of hybrid systems, (ii) algebraic geometry, and (iii) classical potential functions. Note that the co-author researchers from Università Tor Vergata (Rome, Italy) mainly conceived the presented idea and developed the theory behind it, while all authors participated to the design of experiments, discussion of results and final manuscript writing.

The objective is achieved by designing a Lyapunov-based hybrid strategy that autonomously selects the navigation parameters. Tools borrowed from algebraic geometry are adopted to construct Lyapunov functions that guarantee the convergence to the desired formation and path, while classical potential functions are exploited to avoid collisions among agents and the fixed obstacles within the environment (background details in Appendix A).

The use of a team of mobile robots, cooperatively acting toward a common objective, has seen a notable increase in various applications over the past few years, including but not limited to automation and logistics scenarios for monitoring and target searching, as well as in surveillance and rescue missions. The advantages of employing a group of robots instead of a single mobile agent lie in the possibility of obtaining a robust, efficient collaboration among the members of the team, and on their ability of suitably moving to carry out the assigned task, avoiding any collision with other elements in the environment and between themselves.

Multi-robot path planning techniques are aimed at finding the optimal path for the members of the robotic team according to some criteria. The problem has been tackled by using graphs and minimizing over specific objectives [152], or trying to provide constant factor makespan (i.e., the last arrival time) optimal solutions on average over all problem instances on grids and grid-like environments [153]. Such path planning algorithms, however, do not allow to impose a desired arrangement to the robotic agents while they are moving.

Nevertheless, a desired arrangement of robotics agents can be provided by formation control approaches, which are often based on the definition of suitable navigation functions, originally introduced in [154] using artificial potential functions for a single robot, and subsequently extended to the multi-robot case (e.g., in [155], [156]). These approaches exploit the knowledge of the environment topology and of the obstacles present in it to build control policies that guarantee the convergence of the team of robotic agents to the assigned formation scheme, while avoiding collisions. A quite complete survey of multi-agent formation control approaches can be found in [157], where the types of sensed variables used in the different solutions are investigated, classifying the existing schemes into position-displacement and distance-based control approaches.

Several formation control approaches are based on leader-follower strategies, mainly with the aim of achieving a simpler and easily scalable architecture, with reduced communication requirements; the downside of this kind of solutions is a potential weakness in practice, if the substitution of the predefined leader is not envisaged or it is even impossible in case of some fault.

Also, centralization and decentralization of the control strategy is a key issue: a centralized architecture, including a single control law, can be more complex from the computational point of view, but it can generally guarantee the completeness of the solution. On the other hand, decentralized solutions allow for scalability.

The centralized approach proposed in this work solves the patrolling and formation control problems for a group of *unicycle-like* mobile robots, guaranteeing that the trajectories of the multi-robot system are collision-free. In particular, being centralized, this approach allows for several desirable features that are difficult to achieve in a decentralized setting

- The motion planning architecture does not rely on a leader-follower hierarchy and, hence, it is intrinsically robust with respect to faults of one of the robots.
- The hybrid design proposed to tune online the navigation parameters can be used to optimize the trajectories followed by the team of mobile robots. Such an optimization cannot be easily carried out using a decentralized approach.
- This centralized design strategy allows also to guarantee robustness of the navigation scheme with respect to measurement and implementation errors.

The originality of the contribution is given by the use of a hybrid controller to ensure the convergence to a prescribed set, and the combination of algebraic techniques and methods inspired by the classical navigation functions to achieve the convergence to the desired path in formation, avoiding collisions among the agents and with static obstacles in the environment. More in detail, the guarantee of convergence provided by the hybrid system approach ensures patrolling in formation, while the adopted barrier function prevents collisions during the transient time. Preliminary results were developed in [158] for the single-robot case and in [137] for the multi-robot case.

Algorithm overview

We want to design paths of motion for *unicycle-like* mobile robots moving on the Euclidean plane (see Fig. 1.19), described by equations of the form [159]

$$\dot{X} = \cos(\Theta)v, \tag{1.8a}$$
$$\dot{Y} = \sin(\Theta)v, \tag{1.8b}$$
$$\dot{\Theta} = \omega, \tag{1.8c}$$

where $(X(t),Y(t)) \in \mathbb{R}^2$ denotes the Cartesian position of the center of mass, $\Theta(t) \in \mathbb{R}$ denotes the orientation with respect to the horizontal axis, $v(t) \in \mathbb{R}$ and $\omega(t) \in \mathbb{R}$ represent the linear and the angular velocity inputs, respectively.

By hinging upon well-known techniques for instance illustrated in [160, 161] and recalled in [158], the relative dynamics of any point on the robot that does not belong to the segment connecting the two wheels (referred to as P in Fig. 1.19) can be feedback linearized and consequently arbitrarily assigned.

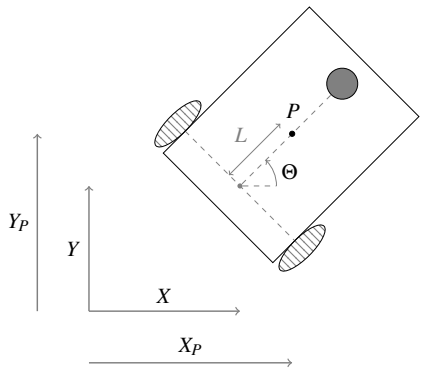


Fig. 1.19 Schematic representation of a unicycle-like mobile robot. The hatched areas represent the actuated wheels of the mobile robot, the filled gray area represents its passive wheel, the point P is the center of mass of the robot, and the length L is the distance of such a point to the line connecting the two wheels. In this two-wheels configuration, the input v represent the mean velocity of the two wheels whereas the input ω represents their differential velocity.

Namely, the dynamics of the Cartesian position $(X_P,Y_P) \in \mathbb{R}^2$ of the point P are given by

$$\begin{aligned} \dot{X}_P &= v\cos(\Theta) - L\omega\sin(\Theta), \\ \dot{Y}_P &= v\sin(\Theta) + L\omega\cos(\Theta). \end{aligned}$$

In the following, we select P as the center of mass of the mobile robot.

Thus, assuming that $L \neq 0$, i.e., that the center of mass of the mobile robot does not belong to the line connecting the two wheels, and letting

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} \cos(\Theta) & \sin(\Theta) \\ -\dfrac{\sin(\Theta)}{L} & \dfrac{\cos(\Theta)}{L} \end{bmatrix} \begin{bmatrix} u_X \\ u_Y \end{bmatrix}, \tag{1.9}$$

where $u = \begin{bmatrix} u_X & u_Y \end{bmatrix}^\top$ is an auxiliary input, the dynamics of the point P are described by *virtual* single integrators of the form

$$\dot{X}_P = u_X, \tag{1.10a}$$

$$\dot{Y}_P = u_Y. \tag{1.10b}$$

Therefore, letting $x_i = [\ X_{P,i} \quad Y_{P,i}\ ]^\top$, $i = 1,\dots,N$, denote the position of the center of mass of the i–th mobile robot and letting its inputs v_i and ω_i be

$$\begin{bmatrix} v_i \\ \omega_i \end{bmatrix} = \begin{bmatrix} \cos(\Theta_i) & \sin(\Theta_i) \\ -\dfrac{\sin(\Theta_i)}{L_i} & \dfrac{\cos(\Theta_i)}{L_i} \end{bmatrix} u_i, \tag{1.11}$$

the main objective of the approach briefly described in this section is to design the input u_i so to guarantee that the corresponding solution constitutes a collision free path for the i–th agent, with $i = 1,\dots,N$, and the set of all solutions steers the mobile robots to patrol a preassigned path in controlled formation. This problem is addressed by combining:

(i) A hybrid controller that autonomously tunes the weights of a parametric continuous-time system to ensure convergence to a prescribed set.

(ii) An algebraic technique that allows us to design parametric vector fields such that the solution to the associated dynamical system achieves patrolling, formation control and obstacle avoidance.

(i) **Hybrid stabilization of parametric continuous-time systems.** The developed novel hybrid strategy autonomously tunes the parameters of a parametric vector field to ensure that a given set is asymptotically stable for the corresponding dynamical system. Such a strategy is then coupled with an algebraic geometry technique, which allows to jointly design parametric vector fields having a given variety as attractive and invariant set and the corresponding Lyapunov function to solve the motion planning in formation with collision avoidance problem.

Consider the parametric nonlinear system

$$\dot{x} = f(x,k), \tag{1.12}$$

where $x(t) \in \mathbb{R}^n$ denotes the state, $k \in \mathscr{A}$ is a vector of parameters, $\mathscr{A} \subset \mathbb{R}^m$ is the compact set of admissible values for the parameters, and the mapping $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is assumed to be C^z for some sufficiently large $z \in \mathbb{N}$.

Definition 1. A compact set $\mathscr{V} \subset \mathbb{R}^n$ is *locally asymptotically stabilizable for* (1.12) if there exist $k^\circ \in \mathscr{A}$, an open set $\mathscr{W} \subset \mathbb{R}^n$ containing $\mathscr{V}$, a function $V \in \mathrm{C}^1$, functions $\alpha_1, \alpha_2 \in \mathscr{K}_\infty$, and a function $\rho \in \mathscr{PD}(\mathscr{V})$ such that, for all $x \in \mathscr{W}$,

$$\alpha_1(\|x\|_{\mathscr{V}}) \leqslant V(x) \leqslant \alpha_2(\|x\|_{\mathscr{V}}), \tag{1.13a}$$

$$\langle \nabla V(x), f(x,k^\circ) \rangle \leqslant -\rho(x). \tag{1.13b}$$

By classical Lyapunov arguments [162], if (1.13) holds, then the set $\mathscr{V}$ is locally asymptotically stable for the system $\dot{x} = f(x,k^\circ)$. Therefore, if the auxiliary inputs $u_1,\dots,u_N$ appearing in (1.11) are designed as $[\ u_1 \quad \cdots \quad u_N\ ] =$

$f(x,k^\circ)$, where $x = [\ X_{P,1} \quad Y_{P,1} \quad \cdots \quad X_{P,N} \quad Y_{P,N}\ ]^\top$, and the set $\mathscr{V}$ describes the target patrolling curve in formation, then the inputs $v_1, \omega_1, \ldots, v_N, \omega_N$ constitute a local *centralized* solution to the patrolling in formation problem for the considered team of mobile robots.

Following constructions similar to those given in [163], a control design technique that guarantees global asymptotic stability of the set $\mathscr{V}$ for a hybrid implementation of system (1.12), under the following Assumptions 1 and 2 have been obtained.

Assumption 1. Set $\mathscr{V}$ is locally asymptotically stabilizable for (1.12) and functions V and ρ such that (1.13) holds are given.

Assumption 2. For all $x \in \mathscr{U} \subset \mathbb{R}^n$ there is $\kappa \in \mathscr{A}$ such that

$$\ell(x,\kappa) \leqslant -\rho(x). \tag{1.14}$$

Note that under Assumption 1, it results that $\mathscr{W} \subset \mathscr{U}$. A technique to design functions V and ρ such that (1.13) holds, hence ensuring that the set $\mathscr{V}$ is locally asymptotically stabilizable for (1.12), is proposed hereafter.

(ii) **Design of a vector field to obtain patrolling, formation control and obstacle avoidance.** The main goal is the design of an algorithmic procedure to compute a parametric vector field $f(x,k)$, together with functions V and ρ satisfying Assumptions 1 and 2, such that the solutions the system (1.12) constitute a path for mobile robots that is collision-free and that steers them to patrol a preassigned path in controlled formation.

This goal is pursued by combining algorithms that use tools borrowed from algebraic geometry with methods inspired by classical navigation functions. The former allow to automatically construct Lyapunov functions certifying the convergence to the desired path in formation, in the absence of obstacles, while the latter allow to avoid collisions among agents and with fixed obstacles.

Hence, two control methods have been proposed, one for solving the patrolling and formation control goals alone, whereas the second to only guarantee that the trajectories of the multi-agent system are collision-free. In particular, firstly an ideal obstacle-free scenario is considered and patrolling of a desired path is ensured, having the robots maintaining a specific formation among them. Then, the obtained vector fields are modified in order to avoid collisions. These resulting control methods have been suitably combined together to solve the patrolling in formation and collision avoidance problem.

For the details and mathematical step of how this was achieved, refer to the complete work presentend in [16].

Experimental validation: simulation to reality

The approach has been tested for groups of three or four mobile agents, both in simulation and in experimental tests, carried out using a remotely accessible robotic testbed (namely, `Robotarium` [164]). The results achieved in all the tests confirm the effectiveness of the proposed solution, which is expected to be applicable up to ten robots, or even more for simple agents like the ones of the remote testbed. Indeed, the scalability of the approach is also affected by the computational and communication capabilities of the available hardware/software architecture.

Please note that alle experiments results are here reported at a high level, omitting details that can be found in [16].

Simulation results

In this section, the two control techniques described before are tested via numerical simulations. In particular, Example 1 shows how the proposed hybrid mechanism can be used to steer the agents to desired target positions, Example 2 shows how it can be used to let the agents patrol a selected algebraic curve, and Example 3 shows how it can be used to let the leader patrol a selected curve while the other agents keep a prescribed formation.

Example 1. Let $N = 4$ and suppose that the size of the 4 agents is $r_i = 0.3$, $i = 1, \ldots, 4$. The objective of this example is to steer the four robots to the target points

$$x_{t,1} = \begin{bmatrix} -1 \\ -0.5 \end{bmatrix}, \qquad x_{t,2} = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix},$$
$$x_{t,3} = \begin{bmatrix} -1 \\ 0.5 \end{bmatrix}, \qquad x_{t,4} = \begin{bmatrix} 1 \\ -0.5 \end{bmatrix},$$

Also, assume that in the workspace of the multi-agent system, there are four obstacles with boundaries that make the the actual workspace of the multi-agent system the interior of the rectangle $[-1.4, 1.4] \times [-0.8, 0.8] \subset \mathbb{R}^2$.

As reported in Figure 1.20, the hybrid implementation generates collision-free path of motions for the mobile robots that avoid collision and that steers the agents toward the desired target points. Note that the hybrid implementation triggers a change in the parameters k at time $t = 9.4818 \cdot 10^{-4}$ so to guarantee convergence of the path of motion to the desired target points. This emphasizes the fact that the proposed navigation algorithm is capable of autonomously changing the initial values of the parameters k so to ensure convergence. △

Example 2. Let $N = 3$ and suppose that the size of the 3 agents is $r_i = 0.3$, $i = 1, 2, 3$. The objective of this example is to let the three robots patrol an affine variety. Note that the workspace is as defined in Example 1. The resulting trajectories of the hybrid implementation of system (1.12) are collision-free path of motions for the multi-agent system.

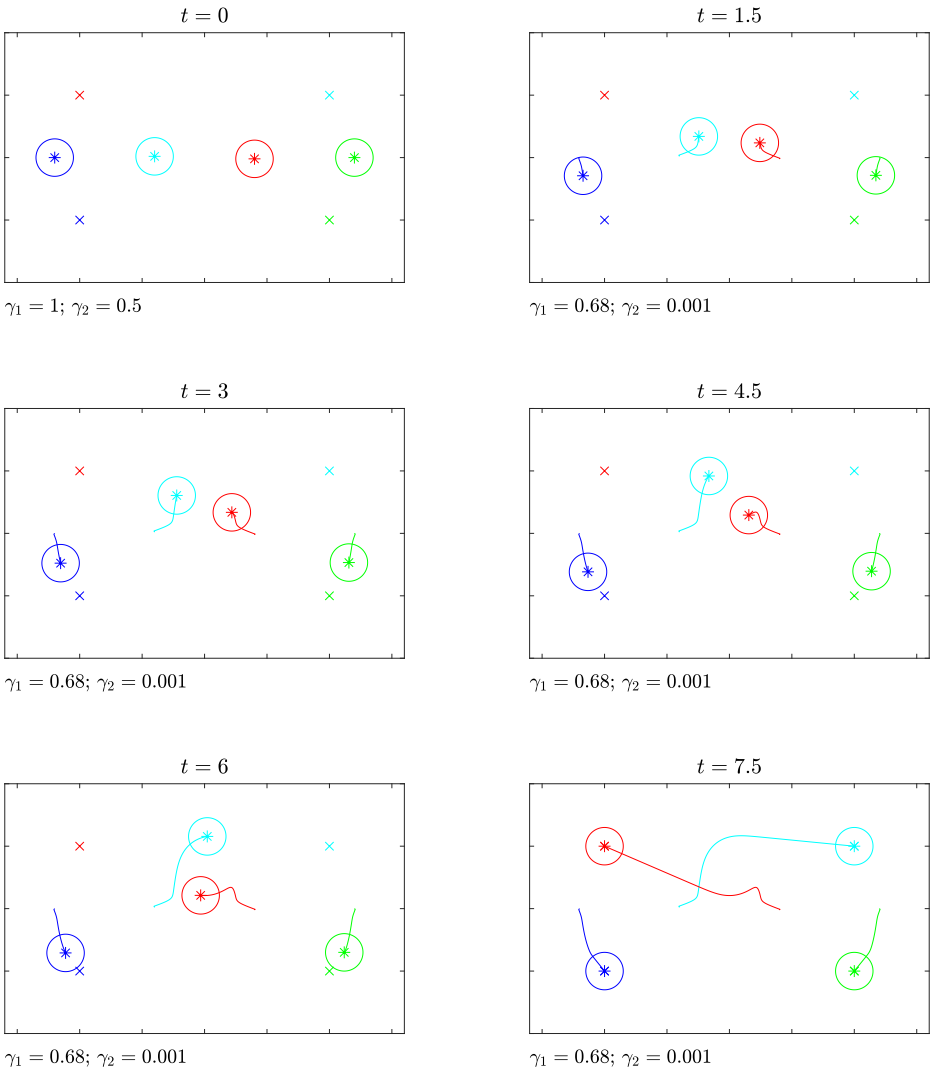


Fig. 1.20 Solution of the hybrid implementation for the problem considered in Example 1.

As shown by Figure 1.21, the hybrid implementation generates collision-free path of motions for the mobile robots that avoids collision and that let the agents patrol the desired curve. The trajectories attained by the agents are dependent just on the initial condition $(x(0), k(0))$, which uniquely determines the state-evolution of the (autonomous) hybrid system. It is worth noticing that the hybrid implementation triggers a change in the parameters k at time $t = 1.4950$ so to guarantee convergence of the path of motion to $\mathscr{V}$. △

Example 3. Let $N = 3$ and suppose that the size of the 3 agents is $r_i = 0.3$, $i = 1, 2, 3$. The objective of this example is to let the first robot (the leader) patrol a circle while letting the other two agents be at prescribed distance, equal to 0.5.

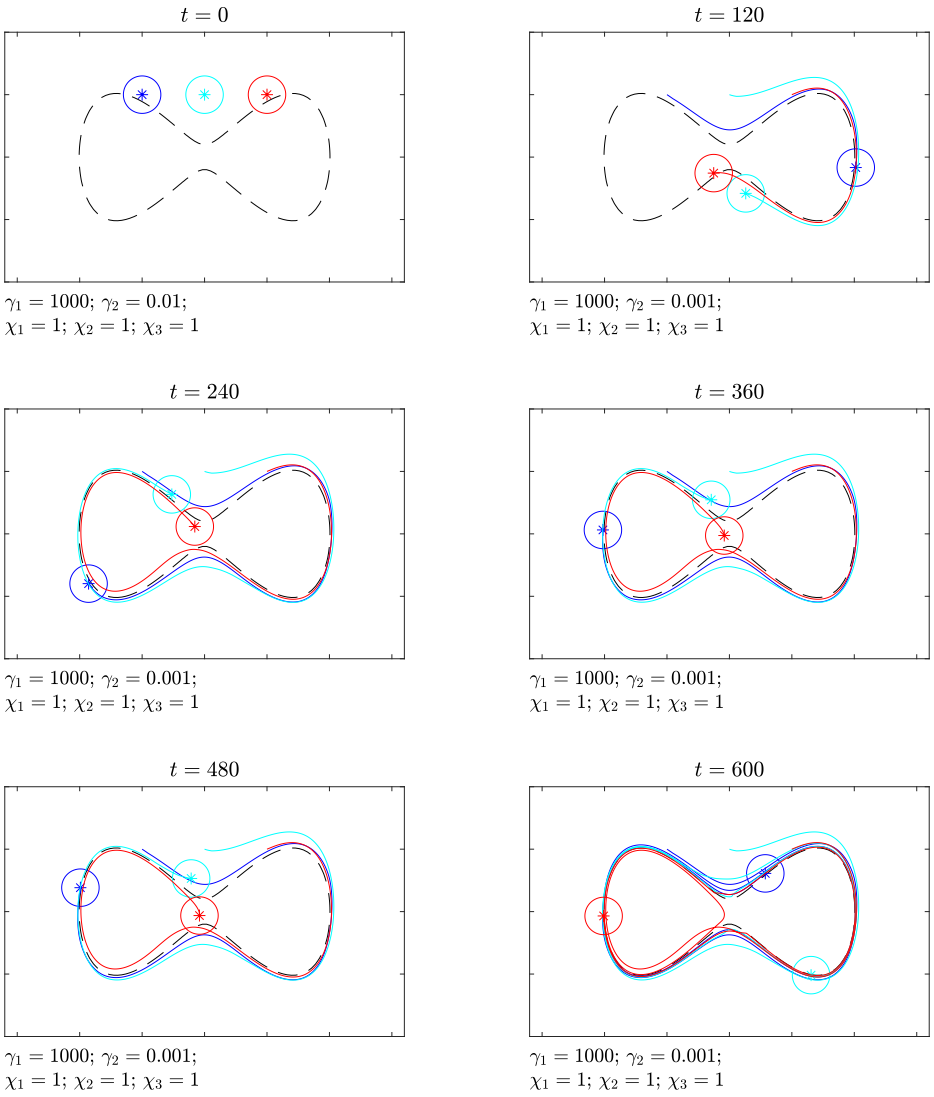


Fig. 1.21 Solution of the hybrid implementation for the problem considered in Example 2.

Let the workspace be as in Examples 1 and 2. The resulting trajectories of the hybrid implementation of system (1.12) are collision-free path of motions for the multi-agent system.

As shown by Figure 1.22, the hybrid implementation generates collision-free path of motions for the mobile robots that avoid collision and that let the leader patrol the desired curve while letting the other agents be in formation. Note how, in such a simulation, the hybrid implementation does not trigger any change in the constants k. △

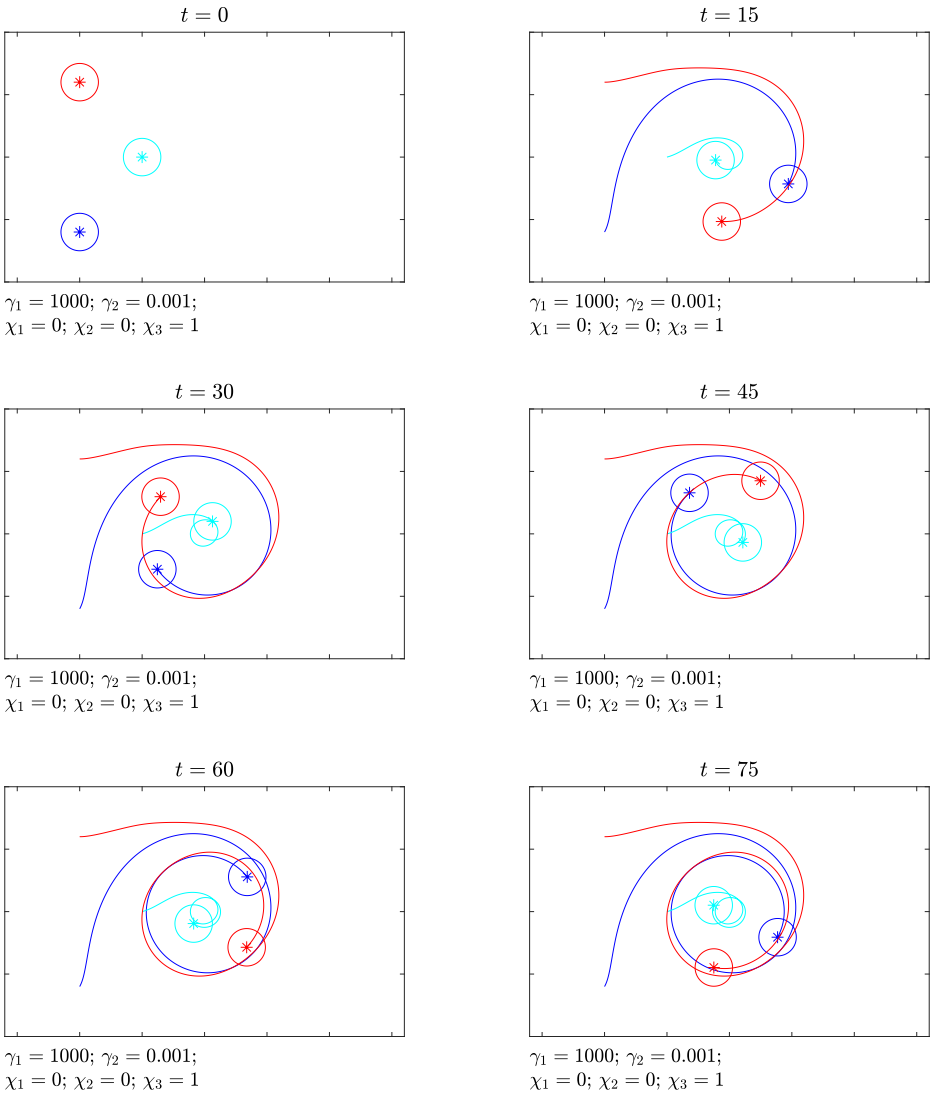


Fig. 1.22 Solution of the hybrid implementation for the problem considered in Example 3.

Experimental results

Experiments have been carried out using the `Robotarium`, a remotely accessible robotic testbed, designed to help users quickly prototype and validate their control strategies through implementation on physical robots without the overhead of setting up their own hardware or high-fidelity simulation.

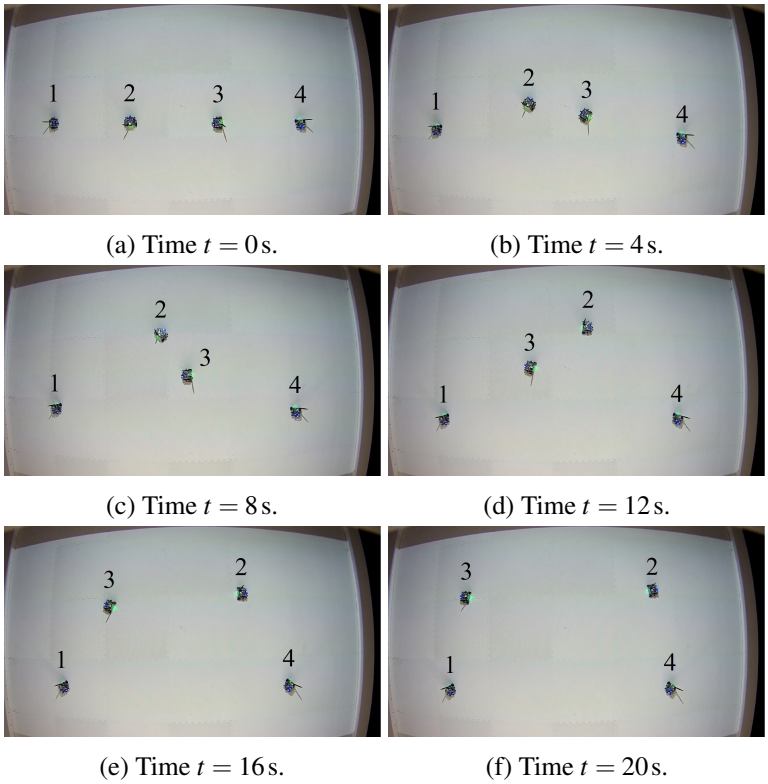
In particular, Experiment 1 shows how the control law given in Example 1 can be employed to steer four mobile robots to a desired target point. In Experiment 2, the control method outlined in Example 2 is employed to make three mobile agents patrol a given curve. Lastly, the control law simulated in Example 3 is validated, letting the leader patrol a circle while keeping the other two agents at a prescribed distance.

To ease the description of the experimental results, relevant snapshots are provided, supported by a proper labeling of the involved robots; see [165] for the full video.

The single-integrator dynamics corresponding to the hybrid implementation have been converted to unicycle dynamics using a suitable linearizing feedback, i.e., the team of mobile robots has been controlled using the centralized control law. Further, since the trajectories of the system $\dot{x}(t) = F(t)f(x(t))$ are the same as those of the system $\dot{x}(t) = f(x(t))$, provided that $F(t) > 0$ for all $t \in \mathbb{R}$, the linear and angular velocities resulting from (1.9) have been uniformly scaled so to meet the actuation constraint. Namely, letting v_i and ω_i be the linear and angular velocities resulting from (1.9) for $i = 1, \ldots, N$, and letting $\bar{v}$ and $\bar{\omega}$ be the maximal admissible linear and angular velocities of the mobile robots, the parameter $F(t)$ above has been selected as

$$F = \min\left\{1, \frac{\bar{v}}{2\max_i\{|v_i|\}}, \frac{\bar{\omega}}{2\max_i\{|\omega_i|\}}\right\}.$$

Experiment 1. The control law given in Example 1 has been used to steer four robots to a target position. The initial poses of the robots are $X_1(0) = -1.2\,\text{m}$, $Y_1(0) = 0\,\text{m}$, $\Theta_1(0) = 0\,\text{rad}$, $X_2(0) = -0.4\,\text{m}$, $Y_2(0) = 0\,\text{m}$, $\Theta_2(0) = \frac{\pi}{2}\,\text{rad}$, $X_3(0) = 0.4\,\text{m}$, $Y_3(0) = 0\,\text{m}$, $\Theta_3(0) = \pi\,\text{rad}$, $X_4(0) = 1.2\,\text{m}$, $Y_4(0) = 0\,\text{m}$, $\Theta_4(0) = \frac{3}{2}\pi\,\text{rad}$. Figure 1.23 reports some snapshots of the corresponding experiment.



(a) Time $t = 0\,\text{s}$. (b) Time $t = 4\,\text{s}$.

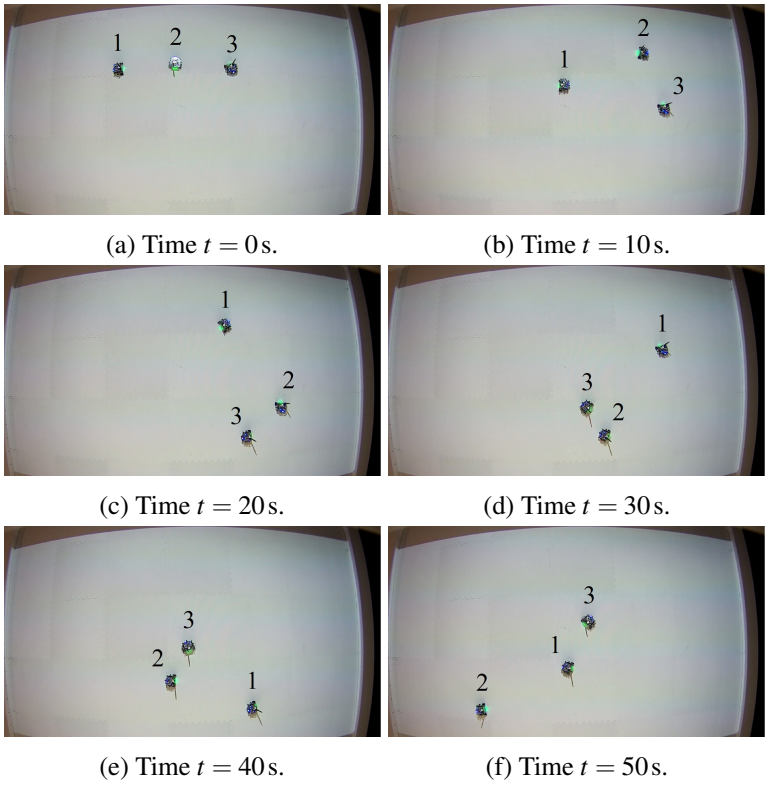
(c) Time $t = 8\,\text{s}$. (d) Time $t = 12\,\text{s}$.

(e) Time $t = 16\,\text{s}$. (f) Time $t = 20\,\text{s}$.

Fig. 1.23 Result of the experiment with the control law given in Example 1.

As shown, the proposed control method allows to steer the four agents to their target positions avoiding collisions among themselves and with fixed obstacles despite the actuators saturation and the robots nonidealities. However, by comparing Figures 1.20 and 1.23, it can be observed a slower convergence to the target points due to the saturation introduced to satisfy the actuators limitations. △

Experiment 2. The control law given in Example 2 has been used to let three robots patrol a given curve. The initial poses of the robots are $X_1(0) = -0.5\,\text{m}$, $Y_1(0) = 0.5\,\text{m}$, $\Theta_1(0) = \pi\,\text{rad}$, $X_2(0) = 0\,\text{m}$, $Y_2(0) = 0.5\,\text{m}$, $\Theta_2(0) = \frac{\pi}{2}\,\text{rad}$, $X_3(0) = 0.5\,\text{m}$, $Y_3(0) = 0.5\,\text{m}$, $\Theta_3(0) = \frac{3}{2}\pi\,\text{rad}$. Figure 1.24 reports some snapshots of the corresponding experiment.



(a) Time $t = 0\,\text{s}$. (b) Time $t = 10\,\text{s}$.

(c) Time $t = 20\,\text{s}$. (d) Time $t = 30\,\text{s}$.

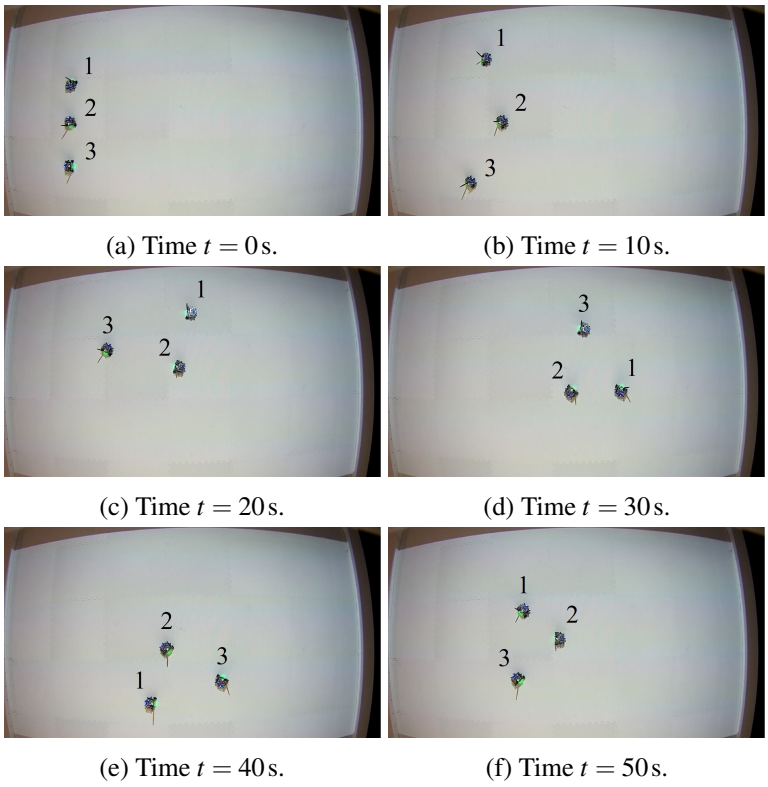
(e) Time $t = 40\,\text{s}$. (f) Time $t = 50\,\text{s}$.

Fig. 1.24 Result of the experiment with the control law given in Example 2.

As demonstrated by such a figure, the proposed control method let the three agents patrol the desired curve avoiding collisions among themselves and with fixed obstacles. However, note that the resulting trajectory is slightly different from the one depicted in Figure 1.21 due to position measurement errors, which, however, do not affect the convergence of the proposed navigation method by the robustness analysis reported in [16]. △

Experiment 3. The control law given in Example 3 has been used to let the leader patrol a circle while letting the other two robots be at a prescribed distance. The

initial poses of the robots are $X_1(0) = -1\,\text{m}$, $Y_1(0) = -0.6\,\text{m}$, $\Theta_1(0) = 0\,\text{rad}$, $X_2(0) = -0.5\,\text{m}$, $Y_2(0) = 0\,\text{m}$, $\Theta_2(0) = 0\,\text{rad}$, $X_3(0) = -1\,\text{m}$, $Y_3(0) = 0.6\,\text{m}$, $\Theta_3(0) = 0\,\text{rad}$. Figure 1.25 reports some snapshots of the corresponding experiment.



(a) Time $t = 0\,\text{s}$. (b) Time $t = 10\,\text{s}$.

(c) Time $t = 20\,\text{s}$. (d) Time $t = 30\,\text{s}$.

(e) Time $t = 40\,\text{s}$. (f) Time $t = 50\,\text{s}$.

Fig. 1.25 Result of the experiment with the control law given in Example 3.

As shown by such a figure, the proposed control method allows the leader to patrol the desired curve while keeping the other agents at a prescribed distance and avoiding collisions among the robots and with fixed obstacles. △

The main contribution of this work compared to the afore mentioned preliminary results is that here the navigation strategies envisioned in [137, 158] are coupled with a hybrid framework, which autonomously selects and updates navigation parameters so to guarantee the achievement of the desired formation and patrolling task while avoiding collisions with fixed obstacles in the environment and among agents. Furthermore, a robustness analysis of the proposed navigation strategy is carried out: even in the case of inaccurate measurements of the agents' positions and imperfect implementation of the nominal strategy, the method ensures patrolling in prescribed formation . It is worth to be noted that, thanks to the centralization of the proposed approach, there is no predefined leader among the robots: the desired patrolling paths can be assigned to some or all the agents of the team, thus enhancing the robustness of the scheme.

1.2.3 Dealing with dynamic obstacles: a costmap layer approach

In previous sections, we have seen how it is possible to control AMRs to autonomously navigate both in environments shared with humans or for the purpose of obtaining formation and patrolling applications in dedicated environments.However, considering an AMR motion in an environment with dynamic obstacles, being able to tune navigation based on the speed of such moving obstacles is useful for improving the performance and safety of the motion itself.

To this aim, the work we have presented in [15] seeks to provide a strategy for dynamic obstacle handling, which can be easily integrated with the current ROS2 Navigation Stack (Nav2), thus being a flexible and modular solution for the problem of navigation in dynamic environments. In particular, our Dynamic Obstacle Layer (DOL) approach can be used as a plug and play solution to integrate the current Nav2 with the dynamic obstacles information coming from a generic 2D LiDAR sensor, starting from some preliminary results available in [166].

The proposed approach is reported here at a high level, omitting details that can be found in [15].

Background

Navigation can be decomposed into the following tasks [167]: (i) modelling the world as a map, (ii) computing collision-free trajectories, and (iii) path following while avoiding collision with obstacles. Tasks (ii) and (iii) make up the *motion planning* problem [168], which over the years boosted the development of several algorithms for its solution, based on different mathematical approaches and technologies [169].

As already stated in Section 1.2.1, path planning algorithms are typically performed within two hierarchy levels: global and local planners.

Within the ROS1 NavStack[1], the local planners made available are:

- *Dynamic Window Approach* (DWA), an online collision avoidance algorithm that takes into account the dynamics of the mobile robot. Taking into account a robot's velocity and acceleration constraints: firstly, a valid velocity search space is generated, and then the optimal solution is selected through a cost function evaluating the trajectories scores [170] (background details in Appendix A).
- *Elastic Band* (EBand) deforms the global path when new obstacles are detected, using an artificial force model [171]. An elastic band is created by a contraction force that pulls the robot towards the goal position, while a repulsive force pushes the path away from obstacles.

[1] ROS currently has two versions: from now on we will refer to the original version as ROS1.

- *Timed Elastic Band* (TEB) is an improvement of the EBand, in which the time information is added to the path computation. Nevertheless, issues arise when the dynamic obstacle intercepts the recomputed path, so three elastic bands are created as alternative paths and the shortest one compliant with the planning requirements is chosen [172].

The selection of the most suitable planner depends on the navigation requirements, and is usually a compromise among precision, speed and performance. For instance, according to [173, 174], DWA planner stands out for its small computing power requirement and repeatability in consecutive tests, while EBand provides more accurate results, and TEB is the fastest to react to the dynamic obstacles, although it requires more computer resources as it tries to optimize multiple trajectories.

For our DOL approach we used ROS2 and the relative Nav2 stack, which still does not embed a strategy for dynamic obstacles handling, as raised by the ROS2 community [175]. Indeed, navigation is performed on the basis of a costmap representation of the environment, where static occupied costmap cells are inflated by an exponentially decreasing cost decay rate, irrespective of whether they are occupied by static or dynamic objects. Thus, the robot plans a more pessimistic trajectory and keeps a larger distance from obstacles than actually required by collision avoidance, but without the awareness of the *temporal evolution* of the occupancy grid. Consequently, the navigation performance is affected by unnecessary delay, leading to lower productivity and preventable downtimes in dynamic industrial setups.

Dynamic path planning: the DOL approach

Being the current flexible production plants highly demanding environments, research is brought on with the aim of enhancing the ROS1 NavStack to make it compliant with industrial highly dynamic and ever-changing environments [176], along with ROS1 to ROS2 migration processes to take advantage and improve well-established frameworks, providing valuable capabilities, e.g., task planning [177].

Several works about laser range finder data processing for dynamic obstacles detection and tracking are available in literature [178–180]. Here, the dynamic obstacles detection is achieved applying some heuristic algorithms directly on the LiDAR pointclouds, clustering the obstacles based on parameters such as the cluster radius or the distance between points.

In the proposed approach, dynamic obstacles are detected and tracked taking advantage of the occupancy grid provided by the Nav2 packages, making it a modular solution such as the ROS1 *costmap_converter* package implemented in [181]. In this way, the DOL is less dependent on the type of used sensor. Then, a policy for translating the obstacles velocity and orientation information into the costmap has been defined exploiting the available Inflation Layer structure and applying a risk level set concept similar to the one presented in [182].

The DOL approach developed in this work is articulated in three steps:

(i) *Object detection.* Starting from the costmap representation of the environment, dynamic obstacles are identified and divided from static ones, applying image processing algorithms and running average filters.

(ii) *Object tracking*. A Kalman Filter is applied to estimate the velocity of detected dynamic obstacles as they are tracked.

(iii) *Cost assignment.* A developed costmap layer assigns costs around each moving obstacle in the *local_costmap*, according to a 2D Gaussian shape, with variances proportional to the obstacle velocity and oriented along its moving direction.

(i) **Object detection.** During this phase, the robot's costmap representation of the surrounding environment is handled as an image. This means that the foreground indicates whatever is moving, while the background is everything that is static. *Background subtraction* is obtained applying two running average filters to each pixel – a "fast" and a "slow" filter.

$$P_f(t+1) = \beta[(1-\alpha_f)P_f(t) + \alpha_f C(t)] + \frac{1-\beta}{8}\sum_{i\in NN} P_{f,i}(t)$$

$$P_s(t+1) = \beta[(1-\alpha_s)P_s(t) + \alpha_s C(t)] + \frac{1-\beta}{8}\sum_{i\in NN} P_{s,i}(t)$$

where $P_f(t)$ and $P_s(t)$ represent the output of the fast and the slow running average filters at time t, respectively. β denotes the ratio between the contribution of the central cell filter and the effect of the neighbouring cells to $P_f(t)$ and $P_s(t)$, so to capture the running average filter of the 8 Nearest Neighbour (NN) cells, since large objects form blocks of cells in the local costmap. The gains α_f and α_s define the effect of the current costmap $C(t)$ on both filters. Therefore, the two filter rates are chosen such that $0 \leq \alpha_s < \alpha_f \leq 1$.

Then, two thresholding steps are performed to filter out the high and low frequency noise and identify those pixels occupied by dynamic obstacles. Namely, the fast filter classifies a cell as foreground if it exceeds threshold c_1, and the difference between the fast and the slow filter has to exceed a threshold c_2 in order to eliminate quasi-static obstacles with low frequency noise:

$$P_f(t) > c_1$$
$$P_f(t) - P_s(t) > c_2$$

The values c_1 and c_2 have been set heuristically based on the range of values that cells can assume in a Nav2 costmap (from 0 to 255), with the same settings adopted in [181]. The output binary map has all the dynamic pixels marked with "1", as it is shown in the general scheme of the DOL approach reported in Figure 1.26.

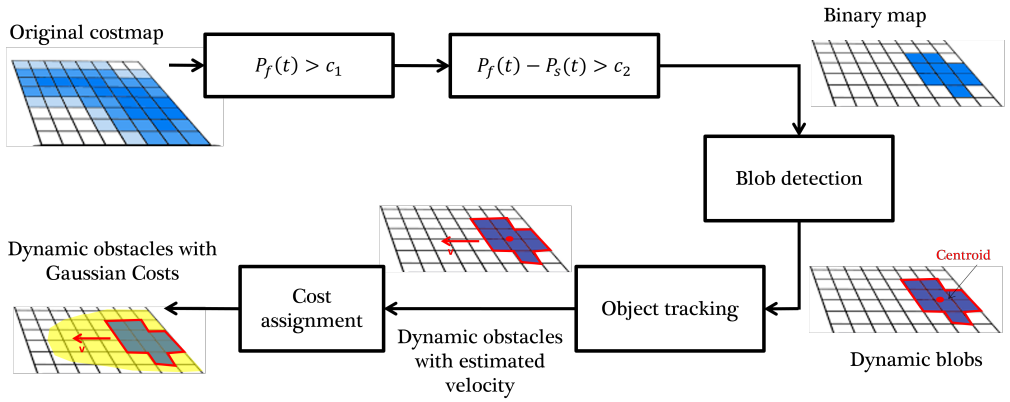


Fig. 1.26 Dynamic Obstacle Layer method steps.

Then, the *SimpleBlobDetector* heuristic algorithm, provided by the OpenCV library [183], clusters the dark pixels in the binary image into blobs representing each dynamic obstacle in terms of contours – a list of the cells which define the blob contours – and centroid – the coordinate of the cell (pixel) in the weighted center of the blob.

(ii) **Object tracking.** The centroid of dynamic obstacles progresses with each costmap update and subsequent foreground detection. The assignment of blobs in the current map to obstacle tracks constitutes a data association problem. In order to disambiguate and track multiple objects over time, the current obstacles are matched with the corresponding tracks of previous obstacles. A new track is created for any new obstacle that appears and is not being tracked yet. Tracks that are not assigned to current objects in the foreground frame are temporarily maintained. If an object is no longer detected for a prolonged period of time, its track is removed.

The assignment problem is solved by the Hungarian algorithm [184], which solves weighted assignment problems by minimizing the total Euclidean distance between the tracks and the current set of obstacle centroids. Then, a Kalman filter estimates the current velocity of tracked obstacles assuming a first order constant velocity model, since it is sufficient to capture the prevalent motion patterns of humans and robots in indoor environments.

(iii) **Cost assignment.** To include the dynamic obstacles information into the costmap, the region around each detected obstacle is inflated with a 2D Gaussian shape, whose peak is associated with the magnitude of the obstacle velocity. This way, faster obstacles are inflated more than slower ones. This has the aim to make the local planning aware of the obstacle with a sufficient heads-up for replanning.

Moreover, the orientation information makes possible to inflate more the cells along the moving direction of the obstacles. To achieve this, two 2D Gaussian shapes are blended: one inflating the cells in front of the obstacle, and the other inflating the cells on its back region. Given an obstacle O with centroid in $c(x,y)$ in the map reference frame, we define a local coordinate system with origin in c, X-axis oriented

along the velocity vector direction, Z-axis pointing outwards the costmap plane and Y-axis set according to the right-hand rule (Figure 1.27a).

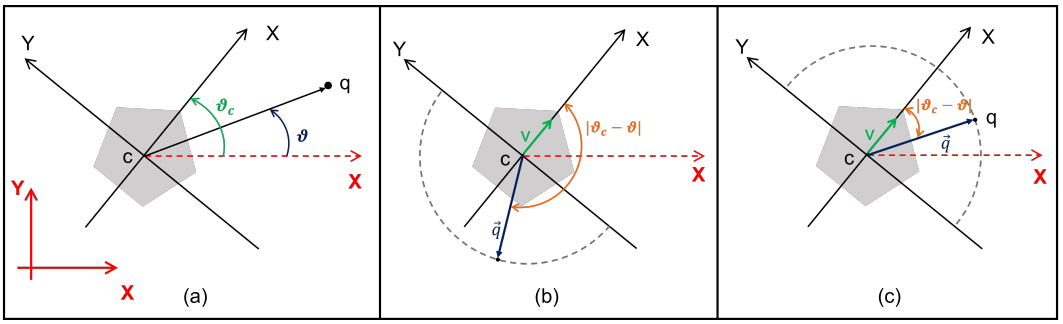


Fig. 1.27 (a) Local obstacle reference frame (black) with respect to the global (map) reference frame (red). (b) A point q within the back area. (c) A point q in the frontal space.

Hence, the obstacle inflation region is represented by the combination of Gaussian functions that inflate the frontal and the back area of the obstacle, respectively. Through the tuning of a specific parameter, the correct Gaussian function is selected, depending on whether the considered cell is in the frontal or back space of the obstacle (Figs. 1.27b and 1.27c).

Each Gaussian function is computed as:

$$\Phi_{c,\Sigma}(q) = A\,exp\left\{ -\frac{[d\,cos(\vartheta - \vartheta_c)]^2}{2\sigma_x^2} - \frac{[d\,sin(\vartheta - \vartheta_c)]^2}{2\sigma_y^2} \right\}$$

where d is the Euclidean distance between q and $c(x,y)$, A is an amplitude parameter set to the maximum cost possible on the costmap, i.e., 255, and σ_x^2, σ_y^2 are the diagonal entries of the Σ covariance matrix, which determines the shape of the inflation region. In particular, the two covariance matrices are defined as follows:

$$\Sigma_{front} = \begin{pmatrix} \sigma_{x\_front}^2 & 0 \\ 0 & \sigma_{y\_front}^2 \end{pmatrix}$$

$$\Sigma_{back} = \begin{pmatrix} \sigma_{x\_back}^2 & 0 \\ 0 & \sigma_{y\_back}^2 \end{pmatrix}$$

Therefore, σ_x and σ_y can be tuned to model a generic shape at will. Here, in order to take into account the obstacle velocity magnitude, a maximum obstacle speed *max_speed* has been set. Then, in order to inflate more the front region, we defined the *speed ratio* $r = \frac{vel}{max\_speed}$, where *vel* is the estimated obstacle speed.

Finally, the variances are modified according to heuristics functions that make the Gaussian shape lengthened in the direction of the obstacle motion and narrowed in the lateral area.

Implementation in ROS2

The presented approach has been implemented exploiting ROS2 Foxy on an Intel NUC8 with a Linux Ubuntu 20.04 environment. For testing purposes, during the development phase, a virtual environment has been created using the Webots simulator, while Rviz has been used for outputs visualisation and debugging. The simulated robot is the TurtleBot3, for which both Webots and Nav2 already provide a physical model and interface packages. Nevertheless, the DOL still remains independent of the robot which is considered.

Nav2 comes with a modular and (run-time) reconfigurable core, consisting in a Behavior Tree (BT) navigator [185] and task-specific asynchronous servers: *Planner*, *Controller* and *Recovery* servers (see Figure 1.28).

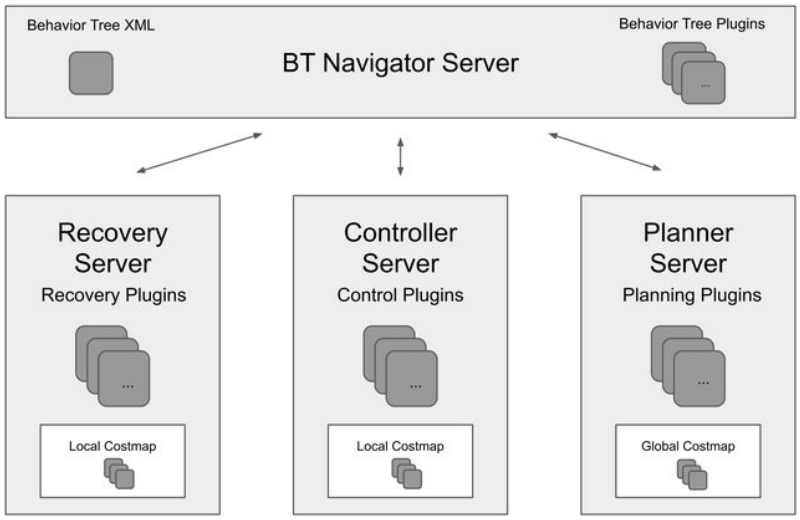


Fig. 1.28 Navigation 2 architecture.

They are action servers hosting the environmental representation used by the algorithm plugins to compute their outputs, under the orchestration of the BT navigator. In particular:

- The *Recovery* behaviours are plugins triggered by the BT when a navigation failure occurs.
- *Controller* plugins replace the *local_planner* of the NavStack of ROS1: they compute a feasible control effort to follow the global plan, based on a local environmental representation, thus performing local planning.
- The task of *Planner* plugins is the computation of a valid, and potentially optimal, path from the current pose to a goal pose, serving the function of global planning.

The *Planner* and *Controller* servers work on two different costmap representations of the environment, the *global_costmap* and the *local_costmap*, respectively.

The dynamic obstacle handling is usually faced in motion planning at a local level, in this case by the *Controller* Server. Among the Controller Server plugins for local planning deployed in Nav2 there are the DWB Controller (*nav2_dwb_controller*), and the TEB Controller (*teb_local_planner*). In particular, the former is the successor to the DWA controller in ROS1.

The huge potential of the costmap representation in Nav2 lies in the adoption of the *costmap layers method* [186]: unlike traditional monolithic costmaps, where all the data are stored in a singular grid of values, in the costmap layers approach, each layer tracks one type of obstacle or constraint, and then modifies a master costmap that is used for the path planning. The base layers in Nav2 are essentially three: (i) static layer – stores the costs associated with the static map provided at launch time, (ii) obstacle layer – continuously marks and clears cells according to sensor data, and (iii) inflation layer – propagates cost values out from occupied cells that decrease with distance, in order to provide a safety margin for the robot navigation.

The DOL has been thought as an additional costmap layer plugin to the *local_costmap*, embedding the information of dynamic obstacles velocity and orientation information into the occupancy grid, together with the existing controller plugins. As shown in Figure 1.29, the *ros2_costmap_to_dynamic_ obstacles* package provides the nodes implementing the obstacle detection functions and publishing the blobs corresponding to detected obstacles through a specific custom ROS2 message type on the */detection* topic. The *kf_hungarian_tracker* package provides a subscription to this topic, so that it can perform object tracking.

Then, it publishes the dynamic obstacles and their estimated velocities on the *local_costmap/tracking* topic, which is created when Nav2 is launched. Finally, the costmap layer, implemented through *nav2_dynamic_costmap_layer_plugin*, processes this information to calculate the Gaussian costs, and then updates the master costmap, which the robot uses for navigation.

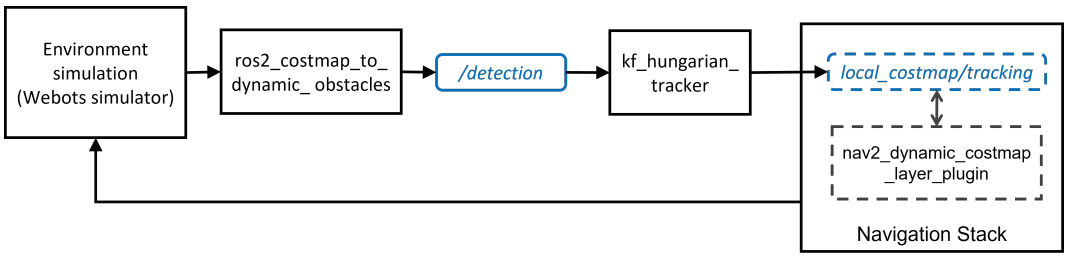


Fig. 1.29 Interaction scheme between internal SW functional blocks and the simulated external environment.

Simulation tests and analysis

The path planning approach available in ROS2 (DWB Controller) and the DWB integrated with the proposed Dynamic Obstacle Layer method (DWB + DOL) are compared through simulations for a preliminary evaluation of the DOL approach.

The simulations have been executed on Webots employing a TurtleBot3 burger robot, a two-wheeled robot equipped with a RPLIDAR A3 LiDAR sensor providing a maximum distance range of $25\,m$, an angular resolution of 0.225° and a scan rate of $15\,Hz$. Webots allows to create realistic 3D virtual worlds including the physical properties of each object. In particular, it is possible to specify the dynamic behaviour of robotic objects (*Robot Nodes*) through a Webots Controller.

The world created for testing is an empty rectangular arena $10\,m \times 6\,m$ where dynamic obstacles are simulated as wooden boxes of $20\,cm \times 20\,cm$ base and $50\,cm$ tall, configured as *Robot Nodes* (Figure 1.30).

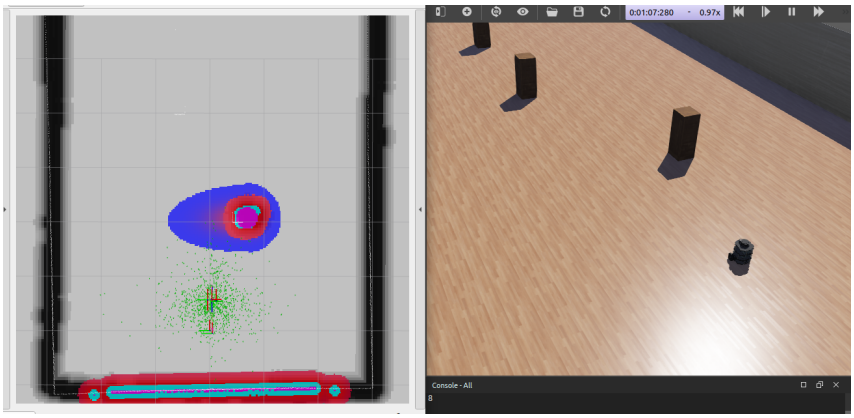


Fig. 1.30 Virtual world on the right and Rviz output on the left with DOL plugin.

Each box is designated an initial position and moves at a constant speed in a back-and-forth motion, covering the entire arena along the short side direction. A Webots Controller has been coded for each obstacle and the speed can be tuned when the world is launched, so that simulations scenarios can be easily modified. To obtain the static map of the virtual environment (without boxes) for navigation, SLAM was performed exploiting the *turtlebot3_cartographer* package.

So as to compare the performances of the baseline DWB with the DWB + DOL configuration, the TurtleBot3 is commanded to navigate in the virtual world N times, from one side of the rectangular arena to the opposite side, covering a total distance of $8\,m$ (Figure 1.31). The Planner plugin used for computing the global path is *nav2_navfn_planner/NavPlanner*. A brief demo video showing the experimental setup and behaviour using DWB alone and DWB + DOL can be found at [187].

The parameters recorded for the evaluation purpose are: the travel time, the number of *wait* recovery behaviours triggered during travel and if any collision occurred. Such data have been collected from experiments conducted in two different conditions: obstacles (boxes) moving at constant speed set to $0.6\,m/s$ (Test set 1) and $0.8\,m/s$ (Test set 2). The performance indices considered for each set are described hereafter.

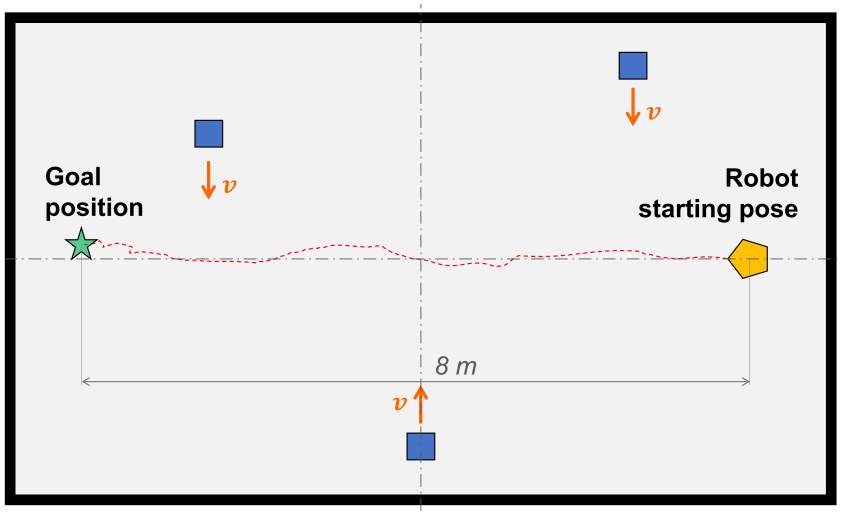


Fig. 1.31 Test scheme.

Smooth navigations. It indicates how many times the TurtleBot3 smoothly navigated to the goal position. It includes also the cases in which the robot stopped briefly to avoid collisions with a moving box in its proximity.

Wait recoveries. It is the number of times the *wait* recovery behaviour has been triggered, still successfully reaching the goal. A *wait* recovery is usually triggered when obstacles come suddenly too close and the Controller Server cannot find an affordable path by a given timeout interval.

Collisions. It corresponds to the percentage of unsuccessful navigation due to a collision of the TurtleBot3 with a moving box.

Successful navigations It is the total amount of times in which the robot successfully reached the goal position, either performing a smooth navigation or after triggering the recovery behaviour.

Test set 1. For the first test set, $N_1 = 50$ simulations have been launched for both DWB and DWB + DOL configurations. Table 1.4 sums up the navigation results.

Table 1.4 Navigation results at $0.6 m/s$ obstacles speed.

	DWB + DOL	**DWB**
Smooth navigations	86,0%	82,0%
Wait recoveries	10,0%	0,0%
Collisions	4,0%	18,0%
Total successful navigations	**96,0%**	**82,0%**

As can be seen, the proposed method combined with DWB reported a greater successful navigation rate than the simple DWB: 96% against 82%, respectively. In particular, the 'smooth navigations' percentage is quite similar with the two approaches, but the DWB + DOL solution reported fewer collisions, because the *wait* recovery was triggered more times. This means that the dynamic obstacle layer provides a safer navigation if it is combined with the actual DWB planner. Indeed, the Gaussian costs forewarn the robot about an approaching obstacle and the Recovery Server is triggered on time if moving on would result in a collision. On the other hand, with the chosen obstacle speed ($0.6 m/s$) and the same Nav2 parameters settings, DWB does not react on time if an obstacle suddenly approaches.

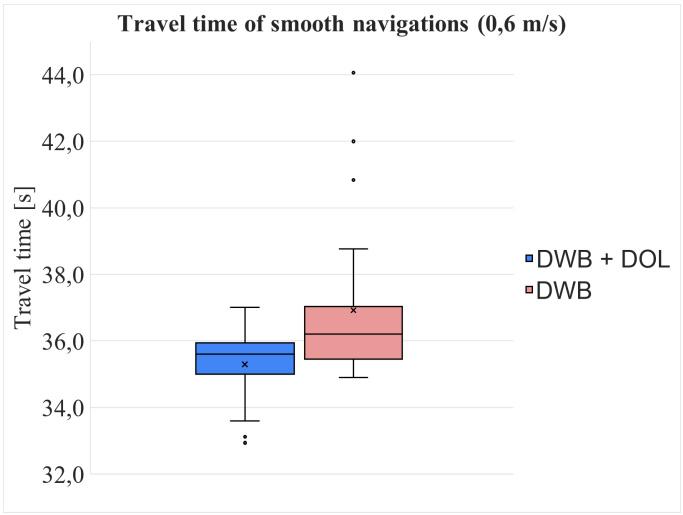
It can be noted that the number of 'smooth navigations' for DWB + DOL and DWB over the total number of tests is similar (86% and 82%, respectively), but the travel time performances are different. Figure 1.32a shows the box plots representing the travel time data of all the 'smooth navigations' achieved by the robot with the two approaches. One can notice that the DWB distribution is more asymmetric and for sure non-Gaussian. Also, the mean travel time reported in DWB + DOL is lower than that of DWB, even if only of $1 s$. Nevertheless, the most important result is that the interquartile range (IQR) for DWB + DOL is smaller than DWB box. Thus, it seems that the proposed method ensures to estimate a more confident travel time for a given environment and settings. Note that outliers lay all below the box plot for the DWB + DOL (shorter travel times), while they are all longer travel times for the DWB plot. This suggests that carrying out more tests might produce less overlapped box plots and so more accurate considerations.

Test set 2. The obstacle speeds have been set to $0.8 m/s$. This value has been chosen in order to push the available DWB Controller to its limits. Indeed, in this case, only $N_2 = 30$ simulations are sufficient to clearly prove the poor performance of the DWB compared to the DWB + DOL approach. Note that with respect to the previous test set, the obstacle speed is the only value that has been changed. Results are shown in Table 1.5.

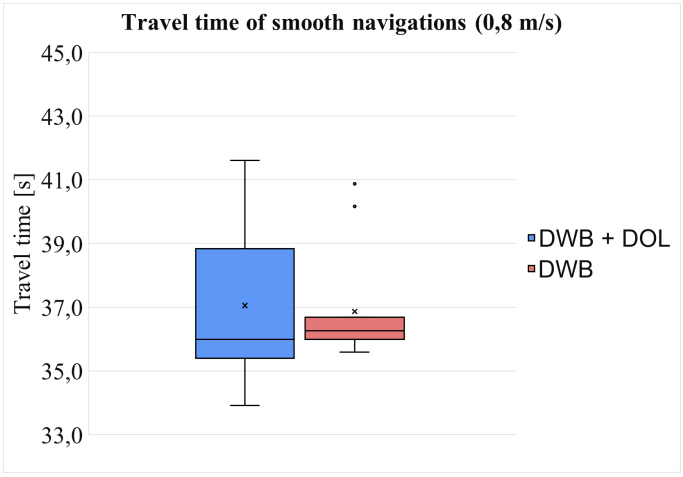
Table 1.5 Navigation results at $0.8 m/s$ obstacles speed.

	DWB + DOL	**DWB**
Smooth navigations	50,0%	43,3%
Wait recoveries	36,7%	0,0%
Collisions	13,3%	56,7%
Total successful navigations	**86,7%**	**43,3%**

It is worth noting that the number of collisions during the simulations launched with only DWB have remarkably increased (from 18,0% at $0.6 m/s$ to 56,7%); likewise, for the DWB + DOL, collisions increased from 4,0% at $0.6 m/s$ to 13,3%. Nevertheless, for the DWB + DOL case, the percentage of triggered recoveries has increased as well, ensuring 86,7% of successful navigations, while the percentage of success in case of DWB has halved with respect to Test set 1. Despite this, for the



(a) Test set 1



(b) Test set 2

Fig. 1.32 Box plot of the travel times during 'smooth navigations' for both test sets.

second test set, travel time data have been collected over $N_2 = 30$ data points (Figure 1.32b). Given that the 'smooth navigations' indices are equal to 50% (DWB + DOL) and 43,3% (DWB), it is worth pointing out that no robust consideration can be made.

Furthermore, median values of both test sets are very similar. The first remarkable difference is that in Test set 2, due to the small amount of data, the DWB + DOLbox plot has a greater variance. This is also because the faster are the obstacles, the more corrective actions are performed by the robot, making the travel time more unpredictable. Concerning the data for DWB, the variance has been considerably reduced with respect to the previous test set. However, the data sample is too small, since the robot performed successfully a smooth navigation 13 times out of 30.

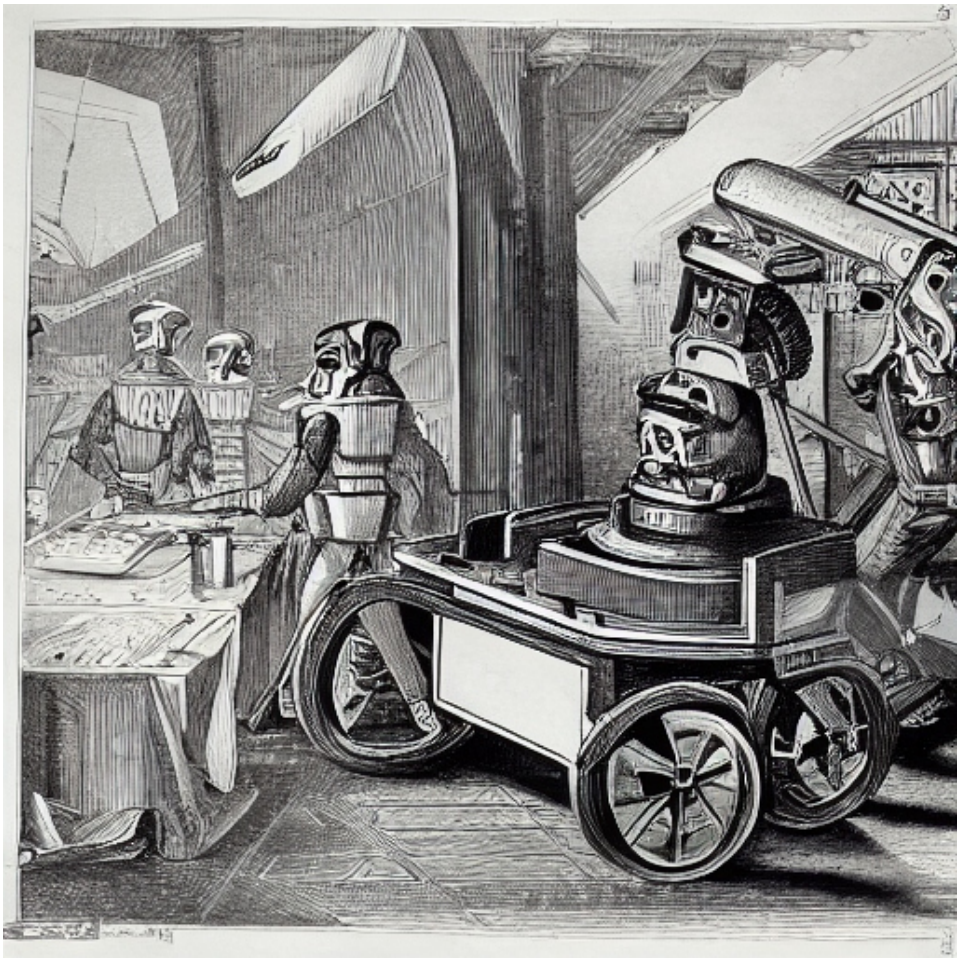
As it is shown by the simulation results, the DOL approach definitely reports some performance improvements in terms of collisions rate, but still collisions occur, even at the lower obstacle speed of $0.6 m/s$. One reason behind this could be the not optimal communication between the Webots virtual environment and the Navigation Stack, that may cause delays in the obstacle detection and costs computation. However, this is strictly related to the computational performances of the whole ROS2 and Linux environment installed on the Intel NUC platform. Yet specific Nav2 parameters could be tuned to reduce the collisions rate with the adopted HW/SW setup (see the complete paper [15] for further details). For what concerns the Gaussian cost assignment, the variances (Gaussian shape parameters), have been set empirically and based on an hypothetical obstacle maximum speed. As alternative, an additional function could be introduced to compute the costs combining the obstacle speeds with the robot speed.

Hence, the proposed Dynamic Obstacle Layer approach seems to provide a safer navigation in presence of dynamic obstacles. At the same time, in the majority of the test cases, the DOL allows to plan a smoother trajectory than the DWB Controller alone, resulting in reduced travel times starting from the same conditions. Indeed, despite the TurtleBot has a maximum speed lower than the set obstacle speeds, it manages to adjust the trajectory when an obstacle is reported by the DOL, dodging it or passing behind it, even if there is still much room for improvements of the travel time performance.

In conclusion, the research contributions described above provided a possible development path towards the autonomy of mobile robots in the industrial sector, considering different application requirements. However, as already stressed, is now essential that IMRs are able to operate in cooperative and/or collaborative environments with human operators. The next chapter will provide a roadmap through the research work undertaken to address autonomous navigation in human-shared environments, while complying with safety and efficiency requirements.

Chapter 2

Mobile robots navigation towards human-shared workspaces



UTONOMOUS mobile robots in industrial plants and warehouses can have a number or roles, ranging from patrolling to material transportation. However, given the need to achieve upgraded capabilities to carry out complex tasks, human and robotic workers must be able to co-exist. This co-existence may imply either *cooperation* or *collaboration*, or both. In the first case, humans and robots work on dedicated tasks while being aware of each other's presence, while the latter case would involve an interaction between the two to achieve a common final goal, represented by the task execution.

To achieve this kind of co-existence, the robots need to be aware of their surroundings in a fashion that goes beyond the awareness needed for autonomous navigation: information on other mobile agents, on human workers and on special areas in the workspace become crucial for decision-making based on the target application. Also, the concept of *team* assumes relevance when shared information lead to improved skills. Teams of mobile robots, either AGVs and/or AMRs, can achieve what a single one would not, unlocking potentialities that bring research a step forward towards teams of humans and robots. Such potentialities have been investigated, leading to the works presented hereafter.

2.1 AMRs as meta-sensors

The work presented in [5], was published within the HuManS – *Human centered Manufacturing Systems* project, funded by Fondo Europeo Sviluppo Regionale (POR FESR Piemonte 2014-2020), relative to the whole concept of new production line. The main goal of the *HuManS* project, which involved COMAU, Politecnico di Torino, and several industrial and academic partners, was the creation of a new manufacturing paradigm, in which the human operator is at the center of the production system, by means of innovative technical solutions that allow the execution of highly complex operations, through a safe and efficient interaction among operators, robots of different nature, and working stations.

One of the contributions to the project dealt with the development of an advanced robotic architecture for a flexible production line. We proposed there for the first time the idea of the AMR as a *meta-sensor*. In this view, the AMRs act as mobile sensors, distributed and reallocated as a supporting net to traditional AGVs, so to reduce as much as possible the need for infrastructural interventions. This way, the AMRs integrate the AGVs knowledge of the environment, so to avoid the use of a

fixed sensor net that would imply infrastructural interventions, limiting the flexibility of the line itself. The proof of concet of such architecture is illustrated in this section.

The current and future trend aims at "setting free" all industrial robots, allowing human-robot cooperation by de-caging the manipulators and by slowly getting rid of the virtual cage enclosing the AGVs. The evolution of conventional AGVs goes toward AMRs, generically indicating infrastructure-free, highly autonomous mobile platforms, which position themselves as a solution to industrial flexibility issues.

Giving a unique and unambiguous definition of AMR is nowadays very difficult due to the current development of the concept, which depends on the field of application and operational requirements. Nevertheless, in order to highlight the features that characterize the new concept of AMR (in an industrial context), the following criteria are pointed out:

Degree of autonomy One of the main characteristics of the AMR is the capability of understanding its surroundings and being able to navigate dynamically. Indeed, the presence of on-board more advanced systems allows for sophisticated decision-making capabilities [188]. An AMR is, at its most autonomous version, capable of avoiding static and dynamic obstacles, including human operators. The degree of autonomy may vary, based on the requirements.

Degrees Of Freedom The definition of AMR not only includes a "simple" moving platform (as standard AGVs), but may include other systems on top of it extending the autonomous mobile robot DOF. Indeed, mobile manipulators, i.e., mobile robots coupled with a robot arm on it, are used for many industrial operations that can involve cooperation with human operators [189]. Thus, the presence of additional automated structures extends an AMR capabilities and application possibilities (Figure 2.1).

Application Depending on the robot kinematics and sensors set, the applications can vary. For example, a mobile manipulator can support factory workers with repetitive and precision industrial processes in a flexible way, transferring the production line where needed (this is not possible with fixed base *cobot arms*, i.e., collaborative robot arms). Mobile robots are employed in logistics tasks (e.g., the transport of different loads between areas) that can be dull, repetitive and/or hazardous for human workers. Autonomous mobile robots are also used for moving products and goods between workers and stations, allowing to minimize idle time. Another utilization is maintenance tasks in restricted access areas and inventory processes at predefined schedules, allowing for optimized inventory process times in huge warehouses [190]. Furthermore some AMRs can implement person following skills, docking to machinery and voice control.

(a) The MiR200 is a safe, cost-effective mobile robot. It can be equipped with customized top modules, including lifts, bins and cobots [191].

(b) The RB-KAIROS is a completely integrated Collaborative Mobile Manipulator (CMM), designed for the development of industrial tasks [192].

(c) Fetch robot AMRs collection [193].

Fig. 2.1 Examples of AMRs.

Ease of integration An automated guided vehicle follows fixed routes, usually along wires or magnets embedded in the ground. This implies that a small change in the working space configuration is constrained by the AGVs road-paths: adding a new AGV to an existing fleet is a difficult process, and extending the working space for the AGVs is not immediate. On the other hand, AMRs are not dependent on the infrastructure since they dynamically determine the best route according to pre-learned maps. In this way, AMRs motion does not depend on the environment setup and allows for fast integration in the factory/warehouse workflow, without the need for expert staff or any re-layout.

Scalability As a consequence of the ease of integration of AMRs in a factory set-up, the number of smart mobile robots can be increased without being hindered by structural changes. A system of AMRs is thus highly and easily scalable.

Safety standards As already mentioned, autonomous mobile robots are accessing spaces usually reserved for human operators and manned vehicles. Indeed, with respect to AGVs and, obviously, the traditional fixed automation elements, the AMRs are free to roam around: no fences and predefined routes imply major safety concerns. Indeed, for several years the available standards addressed AGVs, e.g., the ANSI/ITSDF B56.5-2012 – Safety Standard for Driverless, Automatic Guided Industrial Vehicles and Automated Functions of Manned Industrial Vehicles. This standard "*defines the safety requirements relating to the elements of design, operation, and maintenance of powered, not mechanically restrained, unmanned automatic guided industrial vehicles and the system of which the vehicles are a part*" [194].

Thereby, this standard is defined for classical AGVs and does not take into account the higher degree of autonomy of current AMR systems: no real-time path re-planning and optimized routing is considered. Nevertheless, there are several safety regulations for Industrial Mobile Robots by ISO, e.g., the ISO 3691-4:2020 [94]; also the Robotic Industries Association (RIA) together with the American National Standards Institute (ANSI) developed safety requirements for industrial mobile robots [195].

Artificial Intelligence Even though the motion environment of an AMR is overall structured (factory/warehouse map), it is dynamic due to moving platforms and factory workers. Consequently, the AMR needs a set of sensors whose information, intelligently fused, are the input to sophisticated algorithms providing artificial intelligence to the platform. Object detection, surroundings perception and many other capabilities are critical for the AMR autonomy.

Cost The use of AMR in preexisting workspaces does not imply an infrastructure re-configuration, which translates in less-costly integration. Furthermore, newer technologies result in faster and cost-effective performances. Therefore, AMR solutions are overall less expensive with respect to traditional AGV systems.

Having defined the characteristics of an AMR, some projects can be considered as important reference points for the transition from traditional AGVs to AMR systems.

In [196], the mobile robot, the environment and planning are modeled as automata, exploiting modular supervisory control theory. In this way the robot is able to navigate in the presence of unpredictable obstacles: two supervisors guide the mobile robot enforcing the path it has to follow while ensuring collision avoidance, task management and static versus dynamic obstacles. The research presented in [197] aims at applying cyber-physical systems to the design of AGV systems and fosters efficiency, taking care of urgent tasks, allowing overtaking. In this system, AGVs become smart agents thanks to the physical layer where perception, decision-making and communication are implemented with sensors, a computing module and a Wi-Fi module, respectively.

In [198] the Plug and Navigate (PAN) Robots project is presented, which aims at providing an advanced logistic system involving smart AGVs. These AGVs are able to compute autonomously their motion along a set of virtual paths, called roadmap, pre-computed using a semi-automated map creation process [199]. The mobile platforms are capable of overtaking obstacles found on (or near) the predefined path, exploiting local deviations [200]: the detouring operation is assisted by a centralized data fusion system that gathers together on-board sensing and the data output from an infrastructure-based environment perception system.

The autonomous mobile robot is thus an evolution of the automated guided vehicle, adapted to the imperative need of the industrial evolution to break the now obsolete and inefficient fixed production line paradigma. Moreover, the AMR main intelligence lies in the sensor data fusion, fostered by the use of the latest sensing technologies. Therefore, the AMR sensing system should represent an added value not only for the AMR platform itself, but for the whole smart factory ecosystem, made up of inter-connected, on-line, real-time devices.

In the work-in-progress project presented in this section, the idea is to exploit the mobile platforms as a distributed sensor infrastructure, and as a support net to existing traditional AGV systems. The aim is to design a system of AMRs taking on the role of *meta-sensors*, going beyond the distinction between smart entities and sensing systems, at a higher level of abstraction.

2.1.1 Sen3Bot: a proof of concept

The system is thought for ideally any flexible production line, made up of classical AGVs, workstations, cobots, in spaces accessible to human operators. The system can be described by the macro-elements composing it:

(i) meta-sensor AMR fleet

(ii) Sensors Synergy Center

(iii) AGV Coordination Center Interface

At a high-level, we can identify the goal system as an Industry 4.0 sensor system, supporting non-autonomous agents. Note that, in this case, the word "sensor" comprehends both sensors and meta-sensors. The meta-sensor entities integrate the AGVs knowledge about their surroundings, allowing to smartly react to foreseeable approaching dynamic obstacles, e.g., factory workers. The aim is to provide real-time fully-conscious reactions to the changing environment, which would be unachievable with a fixed sensor net.

Information gathered from the AMRs are combined together within a data fusion center, called the Sensors Synergy Center (SSC), which generates an overall updated picture of the plant traffic status. Furthermore, an ad-hoc interface lets translate significant data into commands, to accordingly modify motions planned by the AGV Coordination Center (AGV CC). Note that, the AGV CC represents the generic pre-existent classical AGV manager. Communication between the two centralized systems (SSC and AGV CC) is bi-directional: indeed, the AGV CC can forward task requests to AMRs.

(i) AMR as meta-sensor

As mentioned above, the meta-sensor AMR aims at increasing the traditional AGV consciousness about obstacles on its route, especially when blind intersections are involved. In this case, the AMR is a sort of AGV extension, an integral part of its sensor system: it is a meta-sensor, whose activity is related to the following issues.

Navigation The AMR exploits the ROS (Robot Operating System) [139] NavStack, which uses relationships between coordinate frames, sensor values and odometry information to perform planning, obstacle avoidance and, at will, localization in a previously created map, using the KLD-sampling (Adaptive) Monte Carlo localization approach [201]. The Global planner can implement the Dijkstra or the A* algorithm while the vehicle task allocation is entrusted to the preexistent AGV CC in order to integrate the AMRs according to known and tested rules.

Vision Recent computer vision developments provide object recognition and detection, fundamental components for sorting out the appropriate reaction for mobile robots. A benchmarking to find the best option for the vision information is currently going on: conventional solutions are compared to new, unexplored alternatives. FireWire (IEEE1394a or IEEE1394b) cameras are well supported by ROS, which provides an image pipeline useful for camera calibration and raw image low-level processing, e.g., distortion rectification and color decoding [202]. In order to get depth information, a stereo vision camera could be a choice. However, the combination, fusion, of two or more sensors can produce a more robust piece of information, since disadvantages can be neutralized and advantages summed up. A solution under evaluation is combining a laser range finder for depth information, with a monocular camera, e.g., a low-cost PTZ IP camera.

Nowadays, PTZ IP cameras can be easily set up since they usually come with a plug-and-play application, which lets the user connect the camera to the preferred wireless access point Wi-Fi signal. An IP camera is a standalone unit, accessible via its IP address. The majority of IP cameras are compatible with the ONVIF protocol, a set of web services-based specifications, using open standards, e.g., XML, to determine how communication occurs between electronic devices over an IP network [203]. The IP camera RTSP (Real Time Streaming Protocol) output stream has been caught using the *GStreamer* tool [204], a pipeline-based media processing framework, and processed so as to obtain a jitter-free, de-payloaded, RGB stream, re-directed as a Linux virtual video device. This video has been read as an ordinary video device, using the ROS tool *gscam* [205], that, leveraging GStreamer, can attach itself to a specially formatted pipeline. Thanks to gscam, the stream can be broadcasted as a standard ROS image message. In this way the image is processed through the ROS image pipeline and can be accessed by other ROS distributed nodes (e.g., the SSC processes). Moreover, the low level parameters of ONVIF compatible devices can be accessed and modified using Python libraries, making it possible to manage via ROS the motion of this kind of cameras. Other options, in combination with laser scanners, will be evaluated, such as omni-directional cameras for wider field of view and stereo cameras, to allow depth information redundancy.

Safety For what concerns safety measures, the idea is to complement safety compliant sensors (e.g., safety-rated laser scanning systems) with non-conventional and/or low-cost sensors that may not be intrinsically safe but providing an overall secure piece of information, as a result of sensor data fusion, performed within the SSC.

Network Communication A further key enabler of advanced manufacturing is wireless communication. Wireless networks comply with many Industry 4.0 requirements, e.g., operational flexibility and easier setup, unlike wired alternatives. Cost-effectiveness is another advantage supporting the low-cost and high-performance aims. Since we consider a factory/warehouse scenario, we expect to work with indoor robots, allowing to consider a Wi-Fi connection for smart elements intercommunication.

For this aim, ROS provides a well-documented network setup, where all distributed nodes are networked via a local Wi-Fi router: robot data are shared among nodes within the same network, through ROS communication paradigms (e.g., ROS topics) [206]. However, ROS heavy dependency on TCP/IP standards leads to issues, e.g., reliability problems, bandwidth limitations, jitter and delays [207]. With regard to these challenging network problems, ROS2 [208] promises to ensure native real-time and multi-robot performance-enhancing features, through the use of Data Distribution Service (DDS) as networking middleware. However, given its state of initial development, ROS2 does not represent a robust choice, at the moment.

Thus, ROS can be a good network choice for the prototyping of the foreseen system, but obviously in the knowledge that (i) there are limitations and shortcomings in the network communication management, which need to be compensated for (through the implementation of ad-hoc nodes or external support frameworks making up for deficiencies) during the development of the real implementation, and (ii) emerging technologies, e.g., 5G networks and 5G architectures, aim at meeting latency, resilience, coverage, and bandwidth requirements, possibly overcoming present issues [209].

In its "Guide to Industrial Wireless Systems Deployments" [210], NIST (National Institute of Standards and Technology) provides manufacturers and users with best practice guidelines, depending on operating requirements and environments. These guidelines have been established following a joint industrial wireless workshop with IEEE, exploring latest and future wireless technologies [211]. The guide points out key features on which to select the proper wireless network, e.g., network throughput, addressing method, reliability, safety, size of the system (number of device), that will be for sure taken into account for the project development.

(ii) Sensors Synergy Center

Sensors data from each meta-sensor AMR are gathered in a centralized system where information is processed to obtain appropriate control commands that, sent to the AGV CC through the dedicated interface, regulate the AGVs motion according to the identified objects. The sensor fusion process comprehends both intra-AMR sensors fusion and inter-AMR data fusion, returning an overall overview of the plant state. Currently, images are gathered and input in the advanced state-of-the-art, real-time, object detection algorithm YOLO (You Only Look Once) v3 [212], in order to identify objects and accordingly perform decisions. YOLO applies a single neural network to the full image, dividing it into regions and predicting bounding boxes and probabilities for each region. Detection can be performed using a pre-trained model (on the COCO detection dataset [213]) and computations can be performed on a GPU, since the used neural network framework, the open source Darknet framework [214], is written in C and CUDA.

The SSC has the aim of representing a centralized computation node, allowing to unload heavy computations from distributed agents. The AGV CC receives the plant traffic overview while informing the SSC about the AGV poses, allowing for object matching. Moreover, when needed, the AGV CC can send task allocation request, for tasks that may not be completed by AGVs due to unavailability or too-long waiting time. Indeed, a meta-sensor AMR, while still providing crucial sensing information, can assist a human operator in collaborative operations or to perform urgent tasks, if requested. Note that tasks should be assigned to the AMR if the requirements are within its payload and/or precision capabilities.A possible implementation for information transmission between computational centers would involve the network communication intrinsically provided by ROS.

The SSC addresses the AMR towards spots where AGVs are supposed to travel (Figure 2.2), according to the route allocation provided by the AGV CC, so as to ensure a safe passage, by advertising the presence of an obstacle/human operator (Figure 2.3).

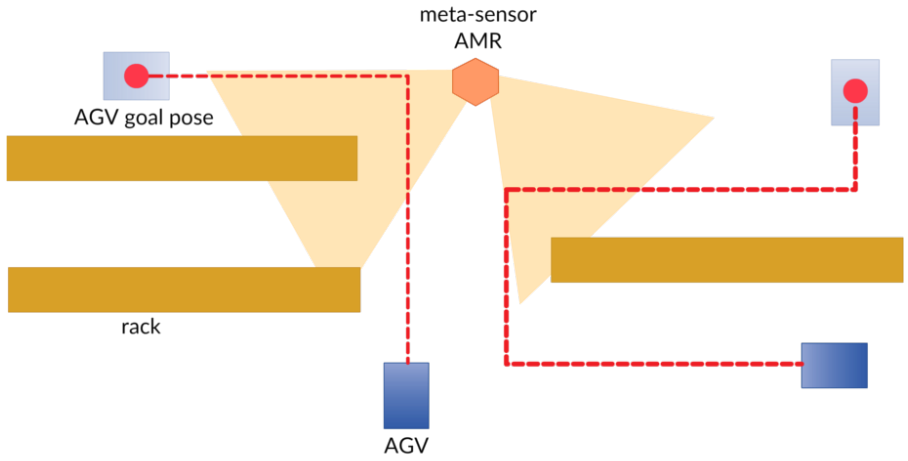


Fig. 2.2 The AMR is sent in the neighborhood of blind intersections, which are foreseen to be travelled by AGVs.

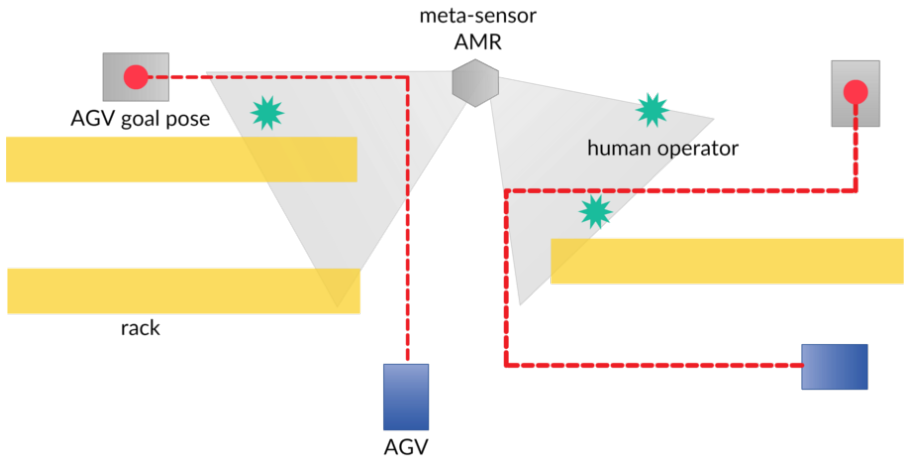


Fig. 2.3 The AMR can detect human operators and obstacles (e.g., racks) and inform the SSC that will warn the AGVs (through the AGV CC), just in time.

AGV Coordination Center Interface

In order to enable communication between the SSC and the AGV CC, an interface has to be implemented. Keeping in mind the intention of using ROS as a software development framework, the idea is to exploit the message-based communication paradigm to easily map information from the AGV CC to the SSC. Examples can be the AGVs poses translated into "*tf*" ROS messages, and a task allocation request implemented as a ROS Action. On the other hand, the plant traffic overview output by the SSC must be translated into information useful for the AGVs route selection. The AGV CC Interface should be made up of adaptable methods, translating data formats from one system to the other. The SSC side can be kept untouched, while the AGV CC interface side implementation depends on the pre-existent coordination system in the plant where the meta-sensor system has to be deployed. In Figure 2.4 are represented the dynamics at play within the meta-sensor and AGV systems.

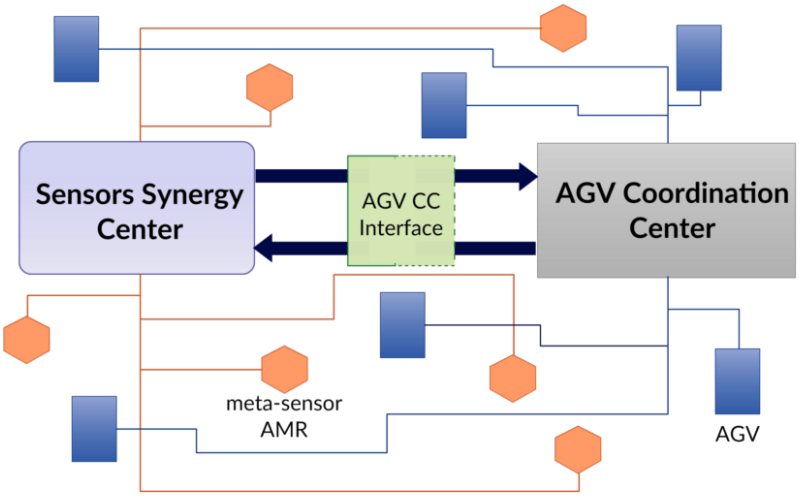


Fig. 2.4 Interactions going on in the foreseen system. Communication between the SSC and the AGV CC is bi-directional: the former generates the overall plant traffic map, by gathering data from the AMR fleet, while the latter updates the AGVs poses in the SSC traffic map and deals with the AGVs coordination.

The aim of the meta-sensors network is to enrich the factory/warehouse ecosystem with a net of autonomous mobile agents, serving the role of blind intersections sentries. Each meta-sensor is equipped with safety-rated sensors, whose data are combined with possibly non-conventional low-cost vision sensors data. The system aims at exploiting data fusion and open source tools to achieve an optimized system behaviour, representing a solution within just anyone's reach.

The working principles of this meta-sensors system, introduced here as a proof of concept, have been detailed in later works, along with a meta-sensor demonstrator implementation, as described in the following sections.

2.1.2 A meta-sensors network: Sen3Bot Net

As metioned in Section 2.1.1, the safe motion of traditional Automated Guided Vehicles in human-shared workspaces can be achieved thanks to the support of a fleet of Autonomous Mobile Robots, acting as a net of meta-sensors, able to detect the human presence and share the information. The localization and perception information of each meta-sensor can be shared among all mobile platforms. In particular, the information about the dynamic detection of human presence can be combined and uploaded in a shared map, increasing the awareness of the mobile robots about their surroundings in a specific working area.

The work reported in this section envisions an architecture integrating a network of meta-sensors with an existing net of Automated Guided Vehicles, with the aim of enhancing systems based on outdated mobile agents that seek for Industry 4.0 solutions without the necessity of a complete renewal. Refer to [9] for more details. A distributed multi-agent system should efficiently split and execute tasks. However, the methodology adopted for this kind of system is usually focused on the algorithms for a particular category, such as motion planning for delivery operations. Nevertheless, the Highway Code implementation proposed in [215] focuses on how to handle safety between robots and human operators. Simulations are highly recommended, since the experimental testing in real world environments would be dangerous and time consuming. In this way, it is possible to perform a list of risks assessments according to the available ISO standards for Human-Robot Collaboration (HRC) operations [216].

A distributed control architecture for multi-robot task allocation is presented in [217]. Distributed messages are used for the communication system between the robots. Each robot has a task tree, that allows the communication with its teammates, the identification of its own tasks and the ones performed by the other robots. In addition, the architecture allocates tasks in such a way that the robots work together to complete the operations, respecting all the motion constraints. Also, in [218], the authors present a multi-Mobile Sensor Agents (MSAs), in which task decomposition and task allocation problems are solved using the "Region allocation tree" method for coordinating the robots. The proposed method was developed in MATLAB. Regarding the multi-sensor data fusion, they applied Bayesian tracking framework, focused on discrete-state implementation.

Ensuring safety in a workspace where mobile robots and human operators work together is crucial, and it is necessary to comply with the strict constraints mandated by international standards. This could pose a challenge in setting up the AGVs. Indeed, industrial manufacturing systems that employ only AGVs may need to find a trade-off between an ad-hoc smarter solution and a more traditional one having a wider application even if less efficient. This becomes necessary since traditional AGVs have technical limitations and, on the other hand, their enhancement is expensive [219]. In contrast, AMRs have a better equipment, which allows them to perform reactive tasks, such as collision avoidance of dynamic obstacles and the relative autonomous path re-planning.

There are alternative methods to ensure safety within the industrial environment where mobile robots and human operators coexist. One approach is to install fixed sensors around the working space, which can monitor the activities of both the mobile agents and the human operators. These sensors create virtual barriers that may deactivate or slower down the robot if the human presence is detected. In [220], particular areas are classified by colors, which indicate the operations carried out by the robots and the relative potential dangerousness for the human operator. Nonetheless, having a fixed network of sensors hampers flexibility.

Cyber-Physical Systems (CPS) are emergent technologies that integrate functionalities for connecting operations in the real world with computing and communication infrastructures through a well-defined network [221]. In this context, sharing relative information about the real world among all agents can facilitate collaborative planning in the manufacturing process and help ensure safety within the working space. When considering CPS, many researchers have developed multi-agent system algorithms that measure and share the relative position of each robot and of all the obstacles within the environment.

The integration of information coming from different sources can be achieved by applying sensor data fusion methods. The combination of data from different sensors allows to overcome the limitations of each sensor, including a limited field-of-view. For example, in [222], a control method that preserves the visibility among robots when they are equipped with limited field-of-view sensors (such as LIDAR, cameras and optical sensors) is proposed. The aim is to maintain multiple lines-of-sight formed by the robots while they are moving. The visibility is modeled using graphs and the edges of the line-of-sight of each robot within the network.

Furthermore, a sensor fusion method for cooperative trail-following tasks is proposed in [223]. The decision making of each robot is based on its own visual data and the shared information from other robots, which is periodically exchanged among them. The authors combined the visual information coming from ground and aerial robots, and tested it in the real-world environment, successfully dealing with the "limited view" problem, typically found in single-robot systems. Similarly in [224], the use of an air-ground robot combined with ground robots is presented. By aligning the data from the Unmanned Aerial Vehicle (UAV) and the ground robots camera frames it is possible to estimate the global pose of each ground robot. However, it is not always feasible to combine ground and aerial robots, as the latter may not be suitable for every indoor environment.

Managing dense information coming from a sensor may lead to false positives. For this reason, there are researchers that combined different methods with Kalman Filters to obtain robust measurements. The multi-sensor fusion strategy proposed in [225] is able to detect and eliminate spurious data before undergoing the fusion procedure, exploiting divergence computation between the a-priori and a-posteriori distributions of the Information Filter. The framework was applied to a multi-robot system moving within an indoor environment with the aim to improve the localization integrity of the overall system, also known as the Collaborative Localization (CL), in which each robot detects the others and computes relative observations.

Cooperative localization consists in environmental measurements performed by multiple robots. In this way, the robots can share their local information to improve their localization. One of the applications is the Cooperative-SLAM , in which the map is created by the combination of each robot independent map. The tests were performed in simulation and in real robots by changing the sensors on-board and the communication rate. An analysis of cooperative localization is presented in [226].

In [227], a hybrid distributed and centralized cooperative fusion is proposed; in particular, the authors refer this methodology as the Edge Cloud Cooperative Localization (ECCL), which combines several distributed Kalman Filters in nodes edge and a centralized system that works as a fusion unit in the cloud server. In order to ensure a robust data fusion, the authors applied a localization validation method called the Cooperative Redundancy Validation (CRV), that takes into account all the available observations.

In [228], a mobile robot sensor network for cooperative localization is proposed. In particular, the mobile robot is used for performing SLAM tasks and simultaneously, each sensor node tracks the robot pose and combine their information in order to obtain an accurate localization for both the robot and the sensor network. The sensor fusion algorithm is based on the Kalman Filter, while the sensor network works under the Parallax-Based Robot Pose Estimation algorithm.

It is worth noting that most of the algorithms for cooperative localization of mobile agents are designed with the idea of introducing new robots with better functionalities to substitute the old ones. However, many worldwide industries are still working with non-collaborative robots and a total replacement would require a huge investment [229]. In contrast to larger firms, a complete renewal for Small, Medium and Micro Businesses (SMMEs) for becoming smart factories may be a problem, due to the high cost and limited resources [230].

By exploiting concepts related to CPS, it is possible to integrate intelligent agents, e.g., AMRs, with the existing elements within the industrial infrastructure without the need of a total technology replacement, while still obtaining the advantages of Industry 4.0 solutions. Moreover, since the future scenario envisages robots and human operators working very closely in the same environment, there is the need for a shared acceptance of the robots as part of the process and feedbacks must be taken into account. Indeed, the probabilistic behaviour of an AMR leads to a skeptical attitude from workers, since they cannot predict the unexpected motions of the robot [231].

As reported in Section 2.1.1, the meta-sensors system introduced in [5] was described at a very high conceptual level, leaving out the expected behaviour specifications and the functional details, which, in fact, are provided in this Section. Additionally, the meta-sensor AMR module, there only hinted at, is here taken into account as a fully functional module, whose desired features have implemented and tested in [7] and will be provided in Section 2.1.3.

Meta-sensors architecture: working principles

In the proposed architecture, each meta-sensor AMR, equipped with a heterogeneous selection of sensors, becomes a sensor itself, with the specific function of facilitating the monitoring of industrial scenarios, as a support to an existing net of traditional fixed-path AGVs. Thereby, the meta-sensor AMRs must not be considered as an evolution of traditional AGVs, but as AGVs enhancers, to enable smart factories benefits. To achieve this, the relative localization information of each AMR along with the information about its surroundings are fused and shared to all the mobile agents. Significant emphasis will be placed on the dynamic detection of human workers; when one or more AMRs detect the presence of human workers in a specific area, the relative position of the latter ones will be updated on the shared map. The AGVs coordination interface will process all the gathered information from the meta-sensors and send the proper commands to the AGVs depending on the dangerousness of the area and on the activities of the operator.

This section outlines the working principles of the envisioned architecture, to describe the coordination and decision-making interface of the AGVs with the measurements obtained from the meta-sensors. The architecture is imagined to be integrated in a flexible production line setting, where traditional AGVs, workstations, cobots, and human operators co-exist. Moreover, it has the following main components: (i) a meta-sensor AMR fleet, (ii) the Sensors Synergy Center (SSC) and (iii) the AGV Coordination Center Interface (AGV CCI).

Henceforth, the terms *Sen3Bot* (Sentry roBot) and *smart AMR* will be used to refer to the meta-sensor AMR. Note that these terms are used interchangeably. The Sen3Bot component, whose implementation details will be provided in Section 2.1.3, has the capability of detecting and identifying humans. Its vision information is integrated with the corresponding distance value through camera-laser data sensor fusion, allowing to correctly place the identified humans within the plant map, and to impose a more conservative behaviour around them (higher inflation radius). The SSC element is in charge of performing sensor-fusion and map traffic updates, receiving the current poses of the AGVs by interfacing the existing AGV Coordination Center (AGV CC), to take decisions for the Sen3Bots task allocation and execution based on the AGVs currently pursued tasks. The AGV CCI then allows to convert the significant data gathered by the SSC (through the Sen3Bots) into proper commands that the AGV CC will use to suitably adjust the AGVs motion. Some implementation details of components (i) and (ii) have already been described in Section 2.1.1, but a more functional overview of both elements, whose behaviour is inevitably strictly correlated, is provided in this work.

The S3B Net (Sen3Bot Network) is a network of autonomous and interacting agents, i.e, intelligent robots that can autonomously perform actions on the basis of a planning algorithm while being able to sense and act when an environmental change occurs. Based on a recognized design workflow pattern [232], we describe the system according to the following characterizing blocks.

Task decomposition

The S3B Net main goal is to ensure a safe motion for each AGV of the pre-existent system. This task can be achieved by the so-called *meta-sensor fusion*, i.e., the integrated information gathered from the involved monitoring Sen3Bots. Note that, apart from its main task, each Sen3Bot can be exploited for moving tools or bringing materials to human operators depending on the availability. In fact, when not busy with its sentry role, the Sen3Bot can take on traditional AMR tasks.

Coalition formation

The coalition formation in the S3B Net is assumed to be defined a-priori and depends on the critical level of the different areas of the factory visited by the AGVs during their tasks. To better understand what is meant here by critical level, some reference to standard definitions are introduced to explain the concept.

According to ANSI/ITSDF B56.5 "Safety Standard for driverless, automatic guided industrial vehicles and automated functions of manned industrial vehicles" [194], we can identify several zones in an industrial environment. Hazardous or restricted areas are not taken into account here, since by standard these are clearly marked areas where personnel has no access. What is relevant for the working scenario are non-restricted areas, defined in ANSI/ITSDF B56.5-2019 as areas where the guide-path is installed and that are shared by personnel. Based on the ANSI/RIA R15.08-1-2020 [195] standard, which defines safety requirements for Industrial Mobile Robots, we consider the following regions:

- *free space*, where the IMR (Industrial Mobile Robot) can plan a path
- *keep-out zone*, excluded from the free space
- *monitored space*, corresponding to the volume around the IMR where perception systems can monitor.

Taking into consideration the co-existence of both AGVs and meta-sensor AMRs in the working scenario, the Sen3Bot monitored space must cover as much as possible the non-restricted area that the supported AGV is going to cross, in order to provide a real-time awareness of the situation *before* the AGV even reaches the location.

Additionally, since today's smart factories and the ones of the very near future have enhanced manual stations and cobots workstation fully integrated within the automated lines [233], we can identify zones where the presence of human operators is highly probable and therefore we consider them to be areas of interest.

We can thus distinguish the following areas:

1. **Critical areas.** This type of areas include: **1.1** Non-restricted areas with limited visibility for the approaching AGV (blind intersections); and **1.2** Areas where human operators are likely to pass, like transition areas. This area type

may include also particular areas that may fall somewhere in between the definitions of areas of type **1.** and **2.**

2. **Sub-critical areas.** These include cobots workstations and manual stations locations where human operators are likely to be present, but expected to be mostly static.

3. **Non-critical areas.** These comprehend restricted areas known to be human free or full visibility areas where human passage is rare. Note that safety is anyway guaranteed by safety-rated sensors on the AGV.

Therefore, our areas of interest are critical and sub-critical ones: as soon as an AGV is assigned its path, if the latter crosses areas of interest, there can be various approaches for the a-priori coalition formation, depending on the level of criticality of the first crossed area. If a critical area (1.1 or 1.2) is crossed, two Sen3Bots are assigned to monitor it. On the other hand, when a sub-critical area is foreseen to be crossed, only one Sen3Bot is sent to the scene, to scout the area. Moreover, the final number of Sen3Bots needed to monitor the scene (until the AGV will overcome the area) also depends on a new real-time information gathered by the meta-sensor AMRs, once they reach their respective monitoring poses: the detected human operators' speeds and directions. These additional data can be used to decide whether the number of Sen3Bots sent to the scene is suitable for the scenario. Note that areas of type 1.1 are the most critical ones, since making up for the lack of visibility is crucial to avoid undesired collisions. For this reason, if the crossed area is of type 1.1, two Sen3Bots are required at all times, until the AGV leaves the area. Less strict policies are adopted when considering types 1.2 and 2 areas. Table 2.1 summarizes the architecture coalition formation policy.

Table 2.1 Coalition formation in the Sen3Bot Net

Area Type	**Area ID**	**Initial # of Sen3Bots**	**Human behaviour**	**Final # of Sen3Bots**
Critical Area	1.1	2	static dynamic	2 2
Critical Area	1.2	2	static dynamic	1 2
Sub-critical Area	2	1	static dynamic	1 2

As shown, the detected human behaviour influences the final number of employed Sen3Bots. Indeed, if the detected humans are quite static, the information to be sent to the AGV does not require redundancy. Instead, a dynamic situation necessitates robust data to be shared: in this case, a redundant information about the human operators is preferred.

In order to define how a *static* and a *dynamic* behaviour are distinguished in the Sen3Bot detection, it is worth recalling how human obstacles are represented in the shared map. As described in Section 2.1.3, human obstacles are enclosed in *virtual cages*, i.e., they are provided a greater safety radius value with respect to other obstacles. In fact, a virtual obstacle is published in correspondence of each detected human, and the extension of this virtual obstacle depends on the human bounding box extension (detected at the image processing stage). This virtual cage remains integral with the operator movements, dynamically adjusting to the detected person bounding box. Moreover, the human obstacle is conservatively enclosed since the safety distance is maintained based on the left and right edges of the vision derived bounding box. The described behaviour is shown in Figure 2.5.

Fig. 2.5 The Sen3Bot human obstacle avoidance behaviour.

Then, for the sake of simplicity, the human behaviour is classified depending on the following criteria: when the center of the human obstacle moves more than 1 m from the first detected position, in either direction, the human operator is considered to be dynamic. Otherwise, it is considered to be static. Furthermore, each Sen3Bot (if more than one are monitoring the scene) will provide such data from different perspectives, thereby allowing to capture a more complete information about the human motion. Note that the threshold value may be adjusted depending on the area features and user needs.

Additionally, depending on the area criticality and the behaviour of the detected human operators, a *priority index* p is assigned to every task pursued by a meta-sensor AMR. This way, it is possible to classify tasks based on their priority, to allow for flexibility in the case that a smart AMR is required for a higher priority task. Suppose a Sen3Bot is pursuing a classical AMR task, e.g., moving material, and the system requires a Sen3Bot to assist the passage of an AGV in a critical zone of type 1.1: if the mentioned Sen3Bot is eligible for being assigned this task, it will interrupt the pursued low priority task (postponing it) and move to the critical area. The maximum priority is given by $p = 0$ and increased when the priority of the task decreases. The flowchart in Figure 2.6 provides a schema of the overall behaviour. As shown, if human operators are detected, the AGV CCI gathers the information from the shared map and through the AGV CC (which we assume corresponds to the generic pre-existent classical AGV manager) slows down the AGV when it approaches the area of interest.

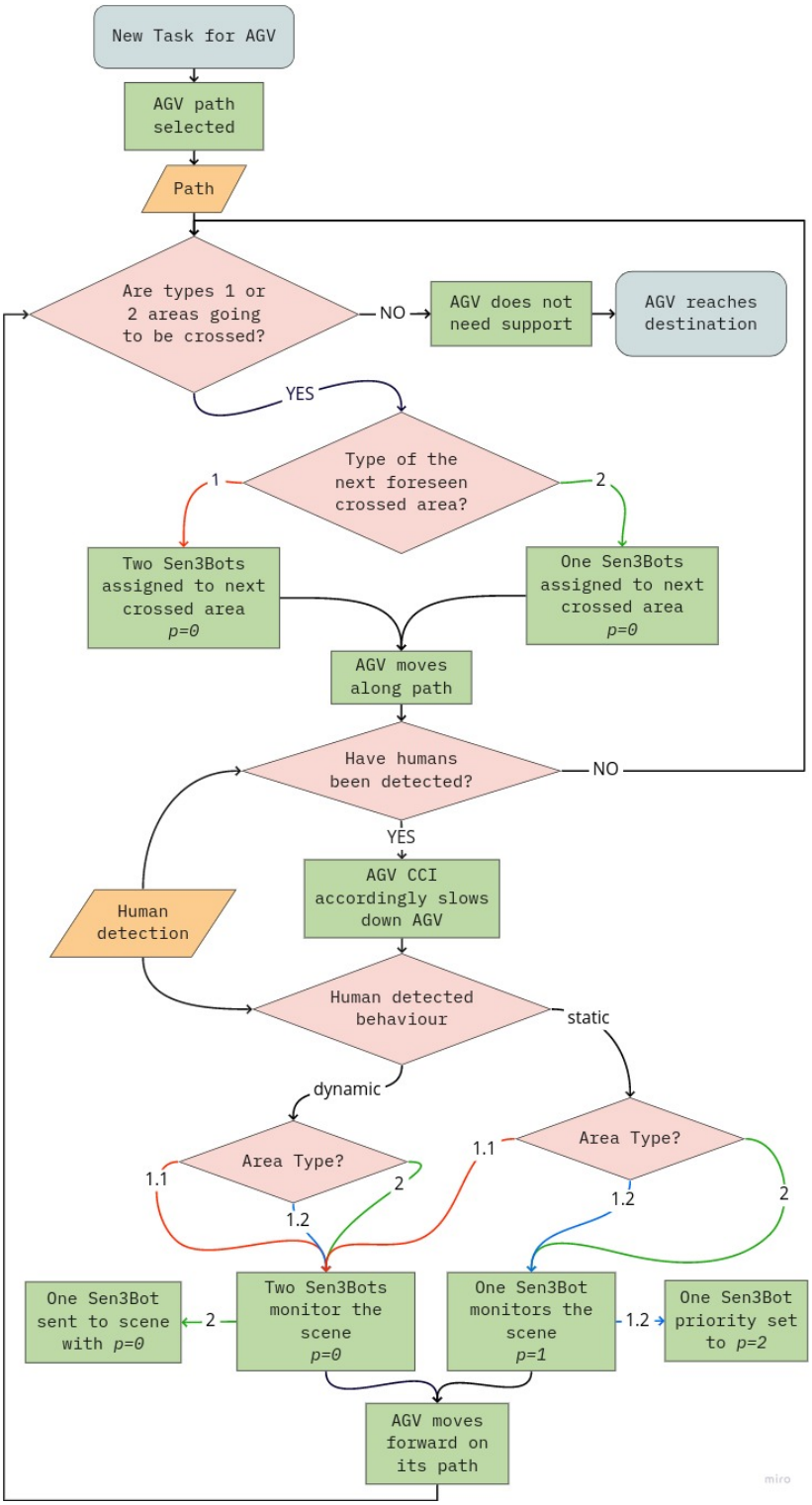


Fig. 2.6 Flowchart for the Sen3Bot Net coalition formation decision process.

Task allocation and execution

Once the system detects that an AGV has been assigned a task, it deploys the necessary number of smart AMRs to monitor the AGV's passage. In our case, task allocation is based on two criteria: (i) distance of the Sen3Bot from the area requiring the monitoring and (ii) the Sen3Bot availability/priority of the task it is busy with.

Before describing how task allocation is managed, it is worthwhile to provide a general assumption about the initial pose of the meta-sensor AMRs. We assume that the power charging stations for the Sen3Bots are positioned in such a way that the Sen3Bots monitor the most critical areas in the factory, while charging or while at *home* (in close proximity to the station). This enables the S3B Net to be responsive when areas requiring the most attention are involved. This allows to get rid of the limitations inevitably introduced by fixed sensors, enabling flexibility while ensuring safety. Note that if the Sen3Bot is not assigned any task, it goes back to its *home* pose and $p = 2$ is set (low priority); of course, $p = 0$ is set when the Sen3Bot is heading to its recharging station due to detected low battery.

The priority value p together with the distance of each Sen3Bot from the interest area are taken into account in order to identify the eligible Sen3Bots for standing sentry in the area. To facilitate a faster S3B Net response, the list of all Sen3Bots is sorted depending on how distant they are from the scene and then selected only if the task they have been assigned to is of priority 1 or 2 (i.e., medium or low priority). In the case the Sen3Bot has been assigned a $p = 1$ task, it is selected but can be replaced if a more distant Sen3Bot with $p = 2$ is found. The selection process iterates until the number N of required Sen3Bots is reached. Refer to Figure 2.7 for more details about what the workflow would be. It should be noted that the total number of Sen3Bots available in the plant is mainly dependent on the number of areas of interest and on the client necessities; a good trade-off could be reached considering that a minimum coverage of all critical areas should be guaranteed.

Furthermore, the proof of principle simulations reported in this work consider the case in which a central system manages the task allocation for a relatively small number of agents. Indeed, the complexity of the overall system increases with the number of smart AMRs, so a distributed control strategy could be preferable in some cases. If a large number of mobile agents is required for the application scenario, the proposed architecture can be implemented over separate distributed modules, with each module operating in a specific area. However, in order to maintain synchronized the whole system, which would include other mechatronic systems within the smart factory, i.e., cobots, these modules still have to communicate with a centralized supervisor that assigns the tasks at a high level. This way, the performance of the entire manufacturing system can be improved through sequences of minimal corrective actions, as in [234], established by integrating the performance indicators of the production process with the capabilities and working functionalities of the robotic systems and agents.

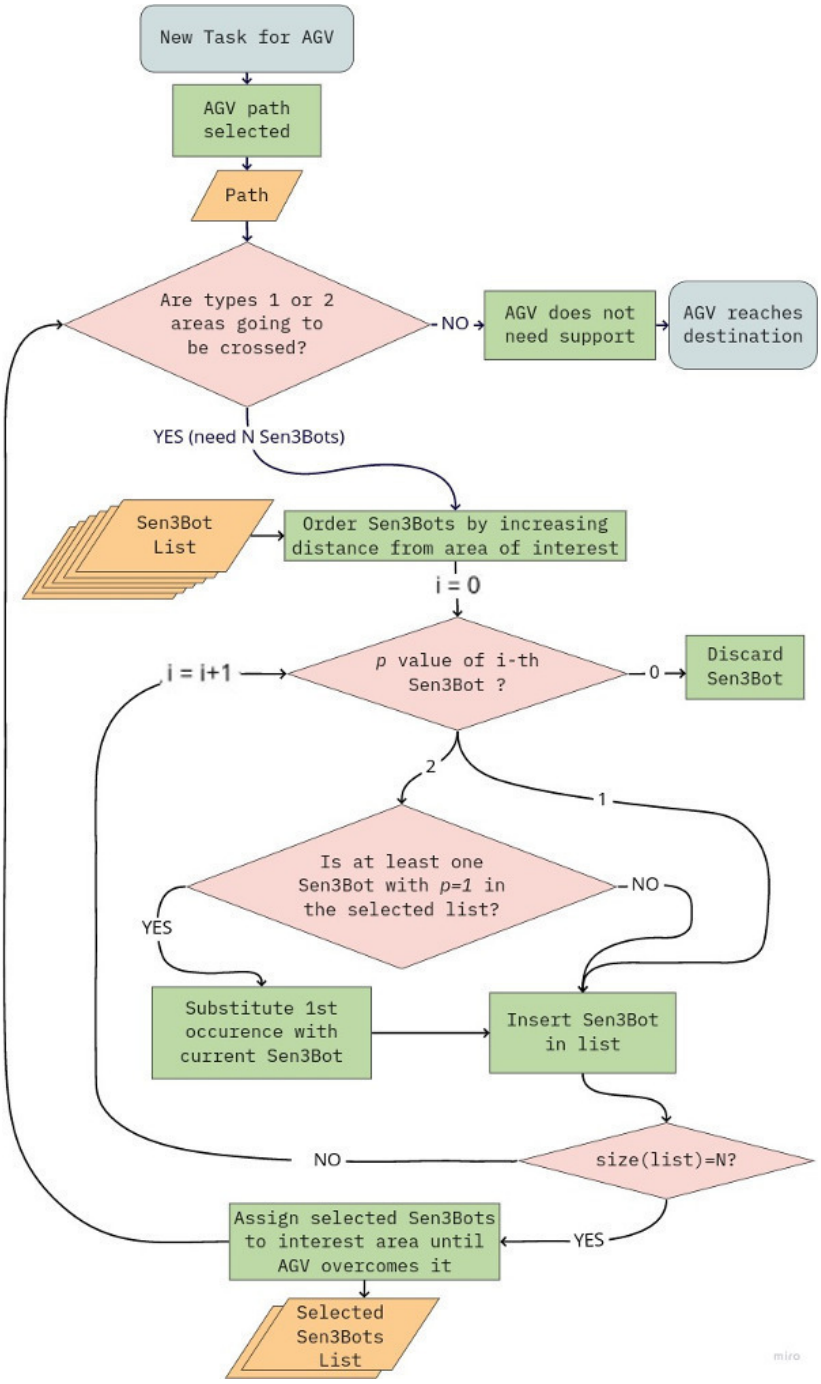


Fig. 2.7 Flowchart for the Sen3Bot Net task allocation decision process.

Simulation: case scenarios

Some proof of principle simulations are have been performed, with the aim of demonstrating the S3B Net behaviour. The video featuring the considered demo cases is available online at [235].

The simulations have been executed on ROS [139] Kinetic Kame, with the corresponding compatible version of the Gazebo Simulator [236]. It must be noted that version 7 of Gazebo does not allow to spawn animated person models (actors) in the simulated environment. However, this does not hinder the simulation aim of demonstrating the architecture behaviour. The created Gazebo *world* represents a portion of a plant accessible to human operators (non-restricted area). Rows of racks and a workstation are present, to provide the minimum conditions for a demonstrative simulation. In Figures 2.8–2.10, the top view for each simulated scenario is shown: on the left the simulated industrial workspace, on the right the *rviz* visualization, where laser data of all mobile robots, along with the cost maps are visible.

Scenario 1, Figure 2.8: the left mobile platform represents a traditional AGV (green laser points), while the right one simulates a meta-sensor agent (red laser points). This proof of principle focuses on the core role of the meta-sensor AMR concept, i.e, integrating the AGV knowledge about the path it is going to travel along before reaching the area of interest. Specifically, as seen in Figure 2.8A, the AGV is aware of the human presence, even if it is out of its scope, thanks to the SSC processed information coming from the S3B Net. Should the AGV on-board obstacle avoidance sensors the only to be considered, the human operator would be detected too late (Figure 2.8B): the Sen3Bots meta-sensor fusion allows for a smart and safer behaviour. Note that the simulation does not consider the coalition formation rules, to simplify the scene for the sake of clarity.

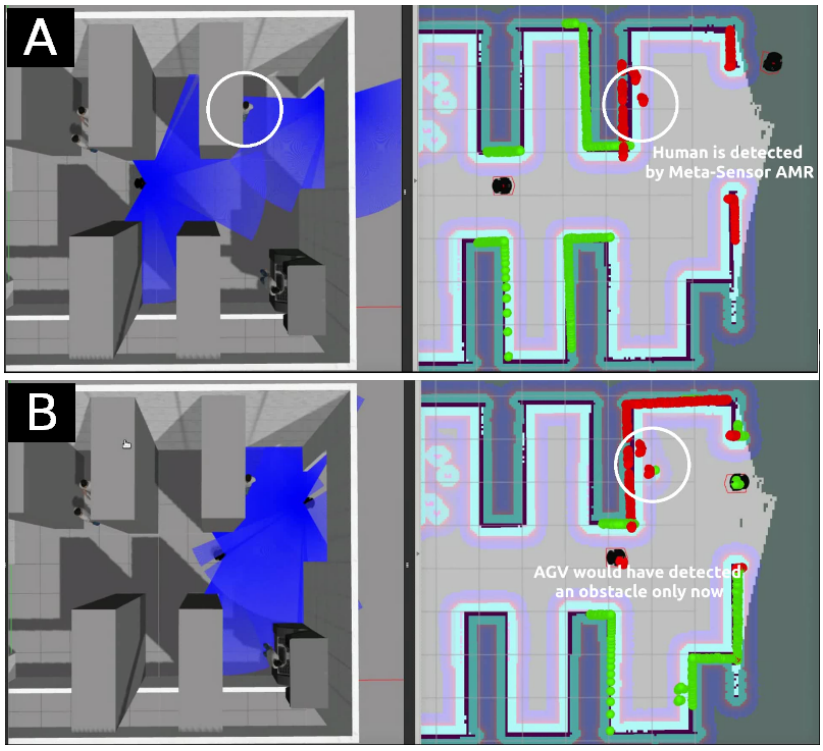


Fig. 2.8 Sen3Bot Net simulation: meta-sensor concept demonstration.

Scenario 2, Figure 2.9: a critical area of type 1.1 (see paragraph 2.1.2 for area types reference) has to be crossed by the AGV: two Sen3Bots are sent to the scene and inform the AGV about the presence of one human operator. Figure 2.9A identifies the different mobile robots involved in the simulation. The AGV CC gathers this information from the SSC and accordingly modifies the AGV motion as it approaches the scene (Figure 2.9B). Then, since the human activity is perceived as dynamic, the two Sen3Bots are instructed to both stand sentry in the area (Figure 2.9C).

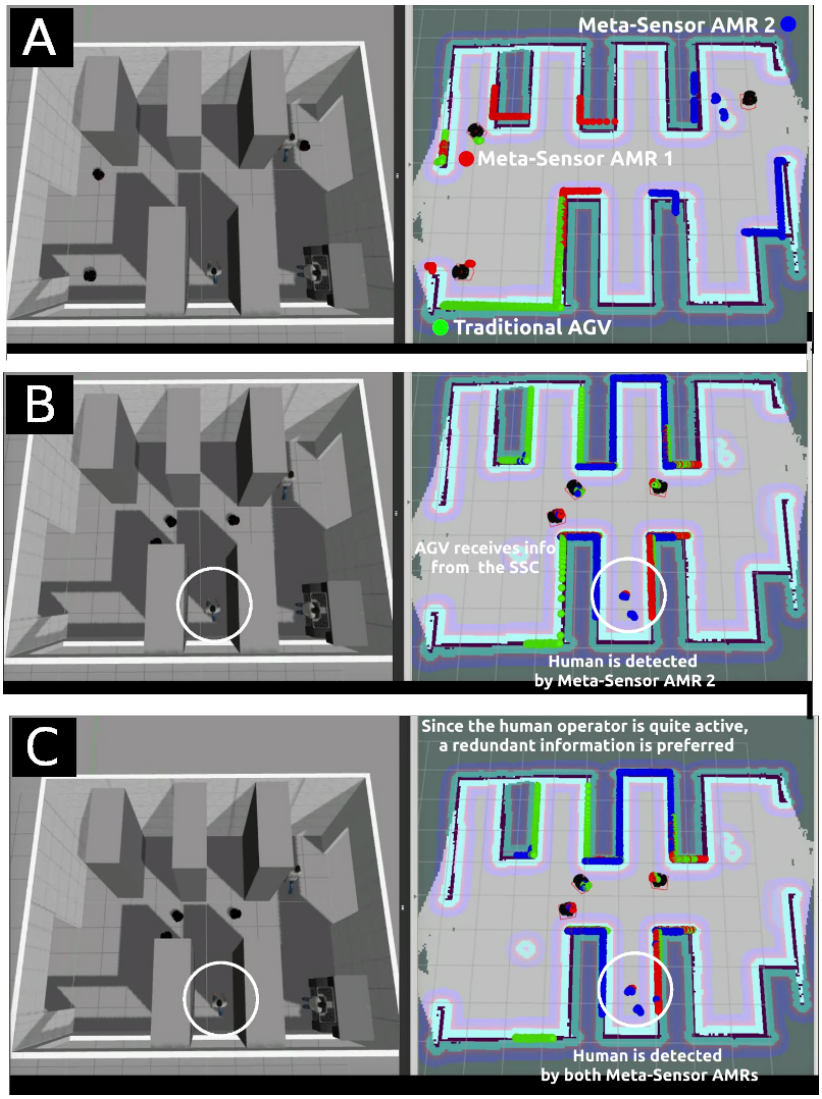


Fig. 2.9 Sen3Bot Net simulation: critical area 1.1 and dynamic human operator.

Scenario 3, Figure 2.10: an AGV has to cross a workstation area, usually classified as area of type 2, which is positioned in a way that visibility is hindered by racks and thus considered a critical area 1.2. (see Figure 2.10A for a recap of each mobile robot role in the simulation). Two Sen3Bots are preliminarly sent to the scene. However, during the AGV passage the human obstacles are detected to

be quite static, allowing for the system to set one of the meta-sensor AMRs to a priority $p = 2$ (Figure 2.10B). Then, as shown in Figure 2.10C, as the Sen3Bot is set available for pursuing other tasks, a monitoring is requested in a very near area and meta-sensor AMR 1 is selected and sent to the requested sentry pose (for reference on task allocation see paragraph 2.1.2).

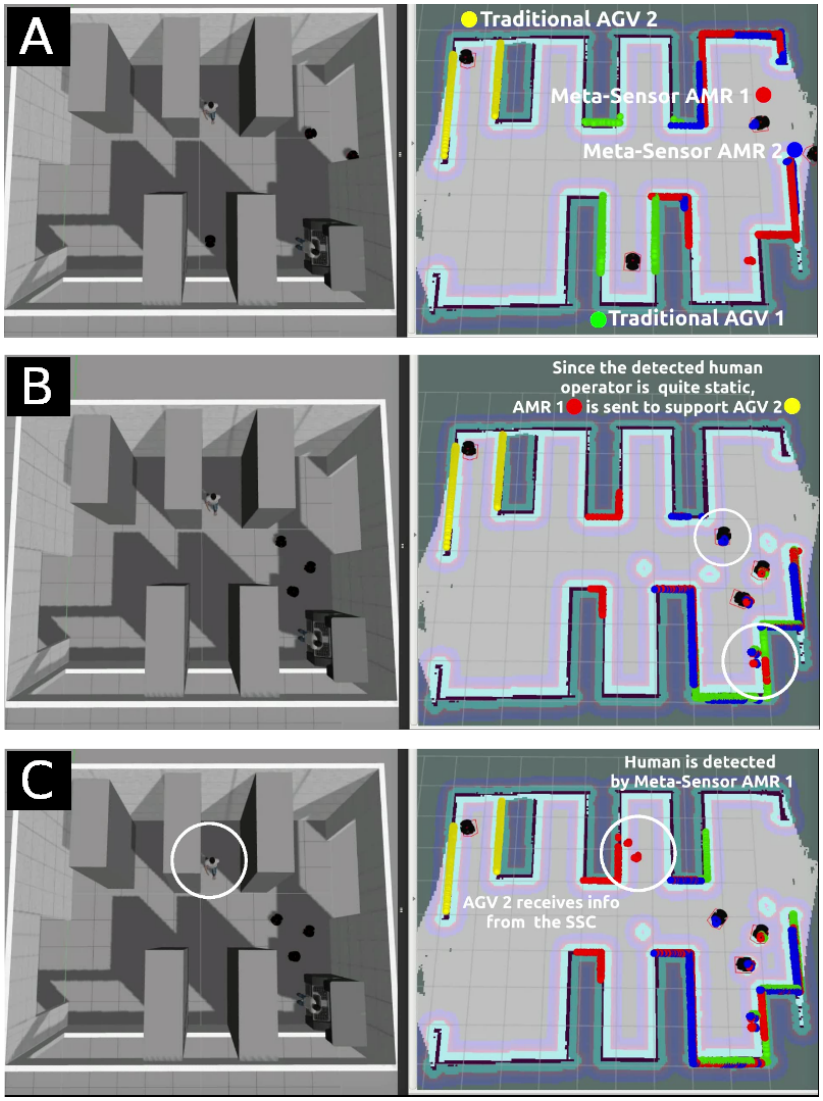


Fig. 2.10 Sen3Bot Net simulation: critical area 1.2 and static human operator.

Our architecture would allow to upgrade obsolete pre-existent systems integrating it with intelligent AMRs. This is in contrast with the on-going trend of entirely substituting the existing mobile robot set-up with a new network of intelligent AMRs.

Given this deeper description of how the proposed meta-sensors network (Sen3Bot Net) would work, a step towards a real framework implementation is to provide a working Sen3Bot demonstrator, detailed hereafter.

2.1.3 Sen3Bot: preliminary implementation

In this section, a preliminary working implementation of one meta-sensor AMR is presented: it exploits camera-laser data sensor fusion techniques to ensure a safe behavior when specific objects are detected, e.g., humans. Human detection has to be guaranteed in a safe way through a proper SW/HW architecture including both *safe* and *not safe* devices, from the industrial point of view.

As thoroughly discussed in Section 1.1.2, depending on the application, one may combine different types of sensors in order to cover a wider range of measurements or duplicate the data to avoid false positives. For instance, a combination of vision and distance sensors is commonly used to equip AMRs working in industrial applications where safe behavior between machinery and human operators must be ensured. These combinations are often cameras with Radio Detection And Ranging (RADAR) systems [237] or cameras with Laser Imaging Detection and Ranging (LIDAR) sensors [238], [239]. The vision sensor is typically used for object recognition or identification of particular geometric patterns, but is influenced by environmental conditions, e.g., the ambient lighting. On the other hand, a distance sensor provides high accuracy on measurements, even though its performance may be affected by (i) the reflection properties of the object to be detected, in the case of LIDARs, and (ii) by external radio wave frequencies, when using RADARs. Inspite of the limitations of each sensor type, by combining them it is possible to associate a detected object with its corresponding distance in the robot coordinate system.

One approach to sensor fusion is the *machine learning* approach. In [240], the authors use the data set coming from RGB-D cameras and compare the performance of several CNNs in order to robustly detect and localize a person. The work in [241] proposes a person detection algorithm based on the linear Support Vector Machines (SVM) learning process that extracts laser features and Histograms of Oriented Gradients (HOG) features from image data. Other approaches, e.g., in [242], combine a Kalman filter with the Global Nearest-Neighbour method, in order to predict and resolve the data association problem for people tracking.

Furthermore, a considerable body of literature treats the LIDAR and vision data fusion as an extrinsic calibration problem: the two sensors' coordinate systems are put in relation through a rigid body transformation, so to align the data derived from both sensors [243]. For the purpose of computing this transformation, usually an external object is required, such as a checkerboard pattern [244], [245] or a trirectangular trihedron [246], to match the correspondences between the two sensors and obtain a mapping that transforms the points from the laser to the camera coordinate system, and thereafter to the image plane.

Although most of the researchers use stereo cameras and 3D LIDARs to model the environment (since they can give a detailed representation of the surroundings), these devices are expensive, and most of the time it is possible to overcome this problem by using transformation matrices when using a 2D LIDAR and an entry-level camera [247].

For our implementation, unlike other commonly adopted methods, where the sensor system is trained at the beginning to combine the data from different sources, we take advantage of a pre-trained NN for human identification. The outputs of the NN are the elements of the image already classified and labeled, which are subsequently used for the sensor fusion algorithm, avoiding the creation of a further dataset. Our aim is then to provide an affordable solution, which takes advantage of sensor fusion to integrate state-of-the-art object detection algorithms taking data from low-cost vision sensors (that may not be intrinsically safe, when used on their own) to complement standard safety compliant sensors (e.g., safety-rated laser scanning systems). Moreover, the use of non-infrastructural sensors provides greater flexibility and scalability compared to other solutions that rely on centralized systems gathering data from infrastructure monitoring.

Sensor data fusion algorithm for a meta-sensor AMR

In order to give a background and motivation to the features that have been built up for this entity, a brief description of the working scenario is provided hereafter. More details on the proposed architecture can be found in Sections 2.1.1 and 2.1.2. Three main components characterize the Sen3Bot Net: (i) a meta-sensor AMR fleet, (ii) the Sensors Synergy Center (SSC) and (iii) the AGV Coordination Center Interface. The work presented here covers points (i) and (ii): note that the developed structure/model for the AMR meta-sensor entity can be replicated for other elements of the fleet.

The smart AMR (Sen3Bot) has to be considered a *feature-enhancer* entity more than a mere evolution of the classical AGV; it is a part of the AGV net, the "brain" behind the system synergy, leveraging sensor data fusion to improve the AGV fleet awareness about the environment's dynamical changes. In this case, we consider the human operator as the object of interest to be detected and advertised to all agents moving within the system.

Solution overview within the meta-sensor system

Having fixed the AMR role in the system, we can identify the behaviour and features we would like the AMR element and SSC to have. In order to achieve smart navigation in a human-shared workspace, we need to gather informative data from the surroundings. With this aim, we decided to equip a mobile robot with a (low cost) monocular camera and a 2D laser range finder in order to perform data association.

So to perform a correct mapping of information, the transformation between the laser and the camera must be computed (*extrinsic calibration*). The human perception and detection is entrusted to the vision part of the sensor system (*human-obstacle detection*). As the obstacle absolute position is computed, we want the Sen3Bot to share this information with other agents and define a reaction rule in the presence of this particular kind of obstacle (*human-obstacle avoidance*).

1) Extrinsic calibration: required to transform the laser points in the camera reference frame and project them onto the image plane. To do so, first the internal and external parameters of the camera are determined, and then the rigid body transformation between the laser and the camera coordinate systems is computed.

- *Camera Calibration.* The camera calibration consists in estimating a relationship between the information of the camera coordinate system and the image frame, along with the relative pose of the camera with respect to the world reference frame [248]. Assuming the pinhole model of the camera, the transformation of the 3D points $\mathbf{C_p} = [X, Y, Z, 1]^T$ to the 2D points $\mathbf{c_p} = [u, v, 1]^T$ is defined as:

$$\mathbf{c_p} \sim \mathbf{K} \cdot \left[\mathbf{R} \,\vdots\, \mathbf{t}\right] \cdot \mathbf{C_p} \tag{2.1}$$

  where $\mathbf{K} \in \mathbb{R}^{3\times3}$ is the so-called camera intrinsic matrix, $\mathbf{R} \in \mathbb{R}^{3\times3}$ and $\mathbf{t} \in \mathbb{R}^{3\times1}$ are the extrinsic parameters of the camera, which relate the world 3D information to the camera coordinate system.

  The intrinsic matrix $\mathbf{K}$ is a projective transformation of the 3D points from the camera coordinates into the 2D image coordinates and is defined as:

$$\mathbf{K} = \begin{bmatrix} \alpha_1 & s & c_x \\ 0 & \alpha_2 & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

  where α_1 and α_2 are the focal lengths in pixel units, c_x and c_y the coordinates of the principal point of the image in pixel units, and s the skew coefficient between the axis of the image.

  To correct the distortion present in the images captured by real cameras, we estimated the radial and tangential distortion coefficients, as the ideal pinhole camera model may not accurately represent the image, given that it does not include a lens [249].

- *Camera-Laser calibration.* After camera calibration, we applied a laser to camera extrinsic calibration algorithm based on [245], taking into account the physical characteristics of the sensors, so to estimate the relative pose of the camera with respect to the laser range finder. Let's consider a point $\mathbf{C_p} \in \mathbb{R}^{4\times1}$ in the camera coordinate system, located in $\mathbf{L_p} \in \mathbb{R}^{4\times1}$ in the laser reference frame. The rigid transformation between the two coordinate systems can be expressed as:

$$\mathbf{L_p} = \left[\begin{array}{c:c} \mathbf{\Phi} & \mathbf{\Delta} \\ \hdashline \bar{0}\,\bar{0}\,\bar{0} & \bar{1} \end{array}\right] \cdot \mathbf{C_p} \tag{2.3}$$

  where $\mathbf{\Phi} \in \mathbb{R}^{3\times3}$ is the rotation matrix of the camera with respect to the laser range finder and $\mathbf{\Delta} \in \mathbb{R}^{3\times1}$ the relative translation vector.

2) Human-obstacle detection: in general, the aim of object identification is to determine the presence, in some given image or frame, of any instance of objects categorized into classes. The recognized objects spatial locations within the image reference frame are returned, e.g., via bounding boxes. Object detection performances have been boosted up with the introduction of deep learning techniques, which let a machine automatically learn feature representations from data. With deep learning, in general, patterns are classified using statistical techniques based on sample data and processing it with multi-layered neural networks [250], [251].

The Convolutional Neural Networks (CNNs) are among the most popular architectures for deep-learning: they are designed to receive multiple arrays data as input, e.g., a three-channel (RGB) image array structure. Many methods handle detection as a classification problem, i.e., object proposals are produced and fed to a classifier. However, some other methods formulate detection as a regression problem, having spatially separated bounding boxes and associated class probabilities as output [252]. Most recent methods are region-based, i.e., they perform a selective research to obtain region proposals, despite this kind of approach often represents a speed bottleneck. Regression-based methods getting rid of the region proposal step have represented a suitable choice for our purposes.

Indeed, we need to identify a specific object class with some index of confidence (i.e., the output class probabilities for the detected objects), and locate these objects in a suitable spatial representation in a reasonable time. Figure 2.11 represents the adopted human detection process at high level.

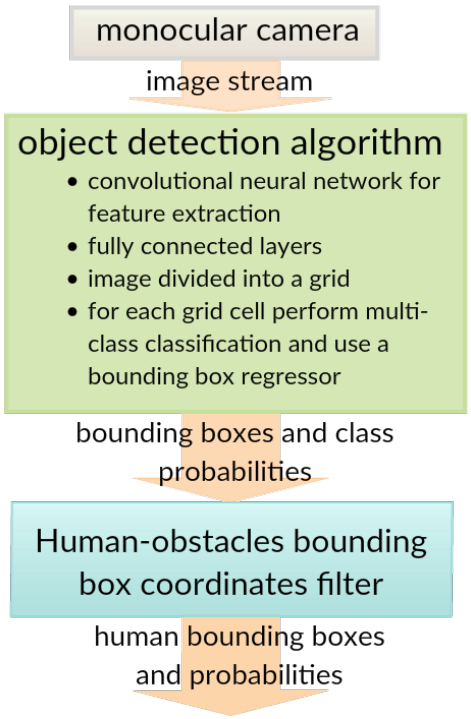


Fig. 2.11 Human obstacle detection process.

3) Human-obstacle avoidance: as a human obstacle is detected, its absolute position in the map reference frame is extracted and shared among all Sen3Bots. The smart AMR reaction to a human obstacle has been made *conservative* in terms of safety distance separating the moving agent and the detected obstacle. The human target positions can then be enclosed in *virtual cages*, i.e., areas considered not accessible by the network of mobile robots, characterized by a greater safety radius value with respect to other obstacles.

Solution implementation

In order to test the sensor data fusion algorithm for human detection and avoidance, we have employed a Pioneer 3DX mobile robot [144] equipped with a SICK LMS200 laser range finder [145] with 10-meter range and scanning angle of 180°, an entry level IP camera (ONVIF [203] standalone unit, accessible via its IP address). As a processing unit for receiving data and controlling the robot, we used a Raspberry Pi [146] 3 Model B mounting an ARM Cortex-A53 (x4 core) CPU (1.2 GHz) and 1-GB RAM; the code that required a higher computational effort has been run on a desktop PC with a Intel Core i7-7700 CPU and a dedicated GTX1060/6GB GPU. The monocamera has been placed above the laser range finder and its orientation set such that the image plane intersects the laser plane as shown in Figure 2.12.

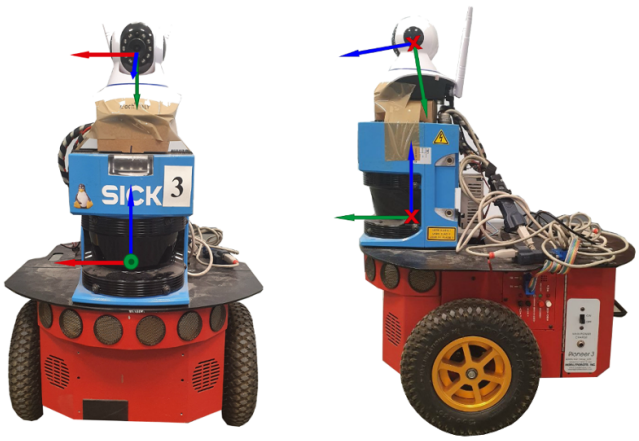


Fig. 2.12 Representation of the sensors reference frames.

The steps taken for the solution software implementation are described below.

1) Laser-Camera transformation computation:

- *Intrinsic Calibration*: the intrinsic calibration of the camera was performed using the MATLAB *Computer Vision Toolbox*, which requires a collection of 20 images to be captured with a checkerboard positioned at various orientations, assuming that all inclination angles are kept below 45° with respect to the camera plane [141]. The camera provides an image resolution of 1280 x 720 pixels, leading to the following intrinsic matrix:

$$\mathbf{K} = \begin{bmatrix} 1.3046 \cdot 10^3 & 0 & 727.7219 \\ 0 & 1.3064 \cdot 10^3 & 241.5278 \\ 0 & 0 & 1 \end{bmatrix}$$

- *Extrinsic calibration and laser point projection*: the extrinsic calibration was computed by collecting simultaneously data from both the vision and range finder sensors, i.e., 20 images of the checkerboard in several orientations and their corresponding laser readings. Considering the specifications of the available sensors, we applied a modified version of the extrinsic calibration algorithm proposed by Zhang and Pless in [245], obtaining the following transformation matrices:

$$\mathbf{\Phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.9848 & 0.1736 \\ 0 & -0.1736 & 0.9848 \end{bmatrix}, \mathbf{\Delta} = \begin{bmatrix} -0.0215 \\ -0.2065 \\ 0.0304 \end{bmatrix}$$

With the rotation matrix $\mathbf{\Phi}$ and translation vector $\mathbf{\Delta}$, it is possible to calculate the laser points in the camera reference frame, and project them in the image plane. In fact, by defining the matrix $\mathbf{H} \in \mathbb{R}^{3\times4}$ as:

$$\mathbf{H}(i,j) = \mathbf{\Phi}^{-1}(i,j), \text{ for } i = 1,...,3 \text{ and } j = 1,...,3$$
$$\mathbf{H}(i,4) = -\mathbf{\Delta}(i), \text{ for } i = 1,...,3$$

and combining (2.2) and (2.3) the vector $\mathbf{c_p}$ is obtained as:

$$\mathbf{c_p} = \mathbf{K} \cdot \mathbf{H} \cdot \mathbf{L_p} \tag{2.4}$$

Note that the world 3D reference frame coincides with the camera coordinates one, making $\mathbf{R}$ the identity matrix and $\mathbf{t}$ a null vector. In order to have a correct representation of the points in the image, the vector $\mathbf{c_p}$ must be normalized with respect to the third component, leading to a vector in the form $\hat{\mathbf{c}}_\mathbf{p} = [u,v]^T$.

Hence, we obtained a projection of the laser points in the image plane (Figure 2.13), which is quite reasonable, since the mapped points are coherent with the hit surfaces.

Fig. 2.13 Laser points projected in the image plane.

2) Relevant bounding box information extraction: since the image processing requires a set of different tools, we have decided to group all the related software within a Docker [253] container, which leverages the GPU computational capabilities and removes the burden of installing ad-hoc tools, fostering portability and scalability. As a human detection system we have chosen the *You Only Look Once* (YOLO) real-time object detection system [254], a commonly used Neural Network (NN) for object classification in an image or video frame. YOLO applies a Convolutional Neural Network (CNN) to the full image, dividing it into regions and performing the bounding boxes prediction and relative probabilities computation for each region. The bounding boxes that have high confidence scores are kept as final predictions, resulting in detections (background details in Appendix A).

This object detection system satisfies our requirements (as specified in Section 2.1.3). YOLO processes the IP camera stream, previously interpreted as a generic webcam output using the *gstreamer* [204] tool. The original code that generates the bounding boxes on screen has been modified, so to save their coordinates information in a text file. This file is used as a source for a ROS [139] node that reads it and wraps into ROS topic messages only the information we are interested in (i.e., the lines identified by a "person" label). All communications between the container and the other ROS distributed nodes happen through the host network. It is worth noting that simply reading the stream output by a low-cost IP plug-and-play camera lightens up the already computationally heavy image processing step.

3) Mapping laser data to the image plane: The information published by the laser range finder is processed within a specific ROS node and projected on the image plane, exploiting the estimated calibration parameters. This same node is also in charge of comparing such information with the messages published on the topic related to the human-obstacles bounding boxes coordinates: data are synchronously gathered from topics, by exploiting the *ApproximateTimeSynchronizer* class that is included in the `message_filters` ROS package.

Once a correspondence between pixel coordinates is identified, the relative points of interest are expressed in the global map reference frame, using the calculated rigid transformation between the sensors and the robot model reference frames (all made available through the `tf` ROS publishing system).

4) Detected humans as virtual obstacles: Thus, combining the human boundary boxes and the laser range finder measurements, the location of the detected operator in world coordinates is obtained, and the safe distance mechanism can be executed. To ensure the desired safe behaviour in the presence of human obstacles, we have taken advantage of the Timed Elastic Band (TEB) local planner [255].

In fact, the TEB plugin also allows to define custom *virtual obstacles* by specifying their location and shape. This local planner can be integrated with the global planner provided by the ROS navigation package. This way, all points falling into each person bounding box are published in the map as circular obstacles with a user-defined radius: the detected human horizontal extension influences the obstacle shape. Thus, a human presence adds a constraint for the re-planning process of the robot, ensuring safety between humans and robots (refer to Appendix A).

Figure 2.14 summarizes the algorithm workflow at a software level: the YOLO C++ software has been edited in order to select all the "person" labeled objects and append them to a *.txt* file, whose content is fed to a ROS node for its translation into ROS messages. Another node is in charge of filtering synchronized data to identify laser points falling into the pixel ranges corresponding to humans. Finally, virtual obstacles are published in correspondence of the human coordinates, with a radius that is added to the *rviz* inflation one.

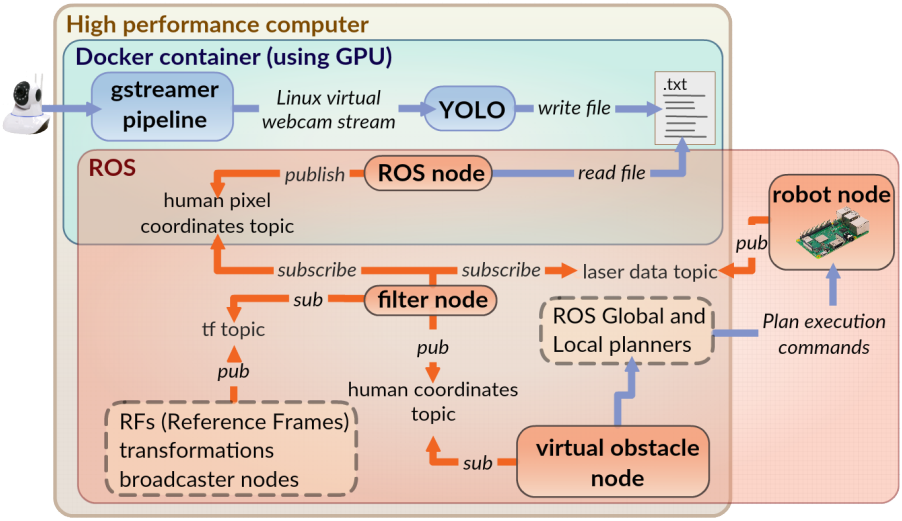


Fig. 2.14 Sketch of the process workflow.

Experimental results

The following section outlines the outcomes achieved using the proposed sensor data fusion technique, which allows for a safe meta-sensor AMR reaction to human presence. In addition, to showcase the achieved results, a sample working scenario is taken into account in the video available at [256]. The key aspects of our algorithm are highlighted in several relevant time frames of the video, which can be summarized as follows:

- In Figure 2.15, a human is detected and its location (perceived by the laser and inflated by ROS) is reinforced by the publication of a set of virtual obstacles (red squared markers). The path computed by the local planner conservatively stays out of the inflation radius imposed by ROS, since the obstacle position is computed not only as detected by the laser range finder, but as the result of the latter and the YOLO processed information.
- Figure 2.16 illustrates how the virtual obstacle publication depends on the size of the bounding box detected during the image processing stage.
- Our algorithm is able to identify, and consequently publish, two human obstacles simultaneously: this influences the re-planning process for the mobile robot, as can be seen in Figure 2.17.

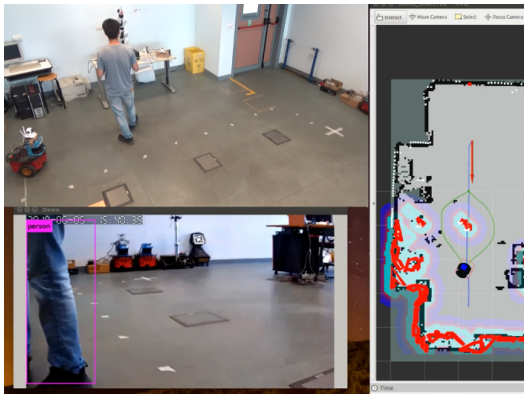


Fig. 2.15 Human detection and relative virtual obstacle publication at 02:37.

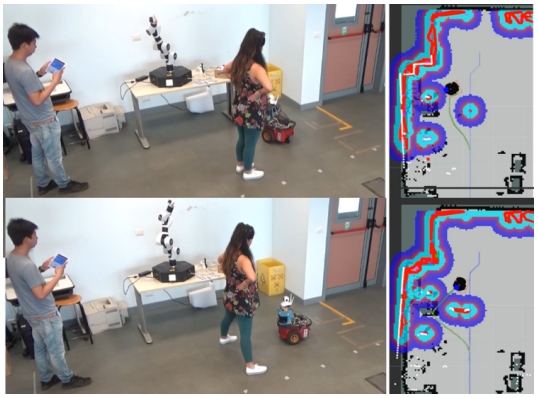


Fig. 2.16 The local planned path depends on the virtual obstacle.

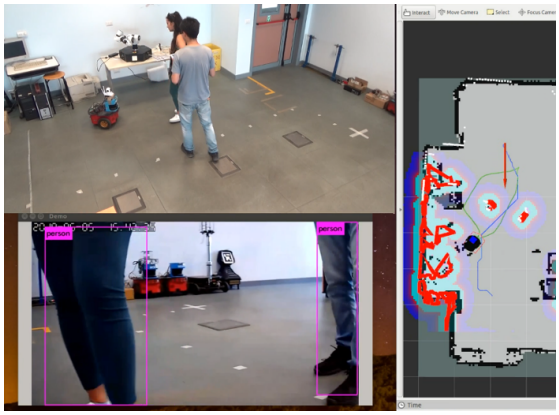


Fig. 2.17 Several human obstacles can be simultaneously detected (04:32).

Note that, since the overall execution time depends on (i) the chosen planner, (ii) the adopted algorithms, and (iii) on the computational capability of the computer executing the sensor data fusion code, the reaction of the robot might fall within the standard definition of soft real-time processes or not. Also, the application is feasible when the relative motion between the robot and the human is slow, so that the robot has enough time to react and re-plan the trajectory, avoiding thus the human operator.

Moreover, the calibration procedure implies an offline data collection phase: despite this is convenient as it has to be performed just once, it relies on the assumption that the involved sensors are well and definitely fixed in place. Enhanced solutions would be given by the implementation of an online calibration procedure, or the application of deep-learning algorithms for data association.

Finally, with the described solution, the information on the human obstacle is considered to be spread to other meta-sensor AMRs, so to influence their *local behaviour*. However, an ideal implementation would trigger a re-planning at a global computation, allowing for a more aware navigation. This feature has been achieved through the contribution described in the next section.

2.2 Applications for meta-sensor AMRs

The works presented in this sections showcase how the meta-sensor AMR, i.e., the Sen3Bot, can be used to improve the Supervised Global Planner algorithm and how it could be exploited as a base for the implementation of collaborative interactions.

2.2.1 Online Supervised global path planning

In this work, the smart AMR implementation is exploited to improve the three-level planning approach presented in 1.2.1. Namely, a set of waypoints belonging to a safe path is computed by the supervisory planner and, depending on the nature of the detected obstacles during navigation, the re-computation of the safe path may be enabled as the collision avoidance action provided by the local planner is triggered. Specifically, the supervisory planner is triggered as the detected human gets sufficiently close to the Sen3Bot, allowing it to follow a new safe virtual path while conservatively overcoming the operator.

This work is thus an improvement of the SGP [6], which integrates an online update of the computed path, exploiting the human detection capability presented in [7]. In the case of static scenarios, the position of the obstacles in the environment are well known, so the generated path can be considered as a safe path. However, when the robot deviates due to the presence of a human operator, the local planner reacts to that obstacle, and the SGP is triggered starting from the current robot pose.

The ability of the smart AMR to follow, when needed (e.g., depending on the area), predefined safe paths in order to minimize the risk of unexpected obstacles (similarly to the fixed paths followed by traditional AGVs) is enriched by the ability of simultaneously handling the presence of dynamic obstacles, guaranteeing a safer avoidance policy in case of humans and the return to the safe path as soon as possible.

The path planning algorithm run by the SGP has a deterministic and repetitive behaviour (i.e., given the initial and final poses, the path will always be the same). It is then considered as safe not only because it automatically avoids all the known static obstacles, but also because it provides a fixed path followed by the mobile robots, if no variation occurs in the environment along such a path. This way, the human operator who is working cooperatively with mobile agents is aware about the trajectory of the robot, and can possibly avoid intersecting the robot safe path. Nevertheless, a human operator may unintentionally cross the dedicated predefined route: the proposed algorithm allows to conservatively avoid obstacles identified as humans even though the latter ones are not expected. The combination of the mentioned features ensures an overall safe behaviour, since the robot (i) follows a safe virtual path, (ii) has the ability of re-planning when a human obstacle is close and (iii) can adapt its behaviour to the environment's changes.

Online Supervisory algorithm with human obstacle avoidance

The original SGP algorithm is based on the collision-free motion planning presented in [137], that generates a path that tends to an algebraic curve. In particular, the SGP takes into account the kinematic model of the unicycle robot, and the safe curve is obtained letting the intersection of the obstacle space and a set of polynomial functions describing the possible trajectories be empty. Note that the equations describing this curve will be omitted here for brevity, since they were already provided and experimentally validated in [6] (background details in Appendix A).

Note that the original supervisory algorithm computed the waypoints *offline*: whenever the LP deviated the robot due to the presence of unknown obstacles, it was not ensured to resume the motion along the SGP computed path. In fact, the modified A* algorithm is intrinsically attracted by the imposed safe curve when the robot is sufficiently near to it, while only the pure A* mechanism is kept otherwise.

Moreover, obstacles were detected but not identified, i.e, a human obstacle was treated in the same way as a generic object. Indeed, thanks to the capabilities introduced by the Sen3Bot implementation, the human operator can be identified and published as a *virtual obstacle* to be overcome conservatively. In terms of safe path conservation, an event-based trigger is added to the SGP.

This planner will be referred in the next sections as the *Online Supervisory Global Planner* (OSGP). The hierarchical planning structure is shown in Figure 2.18.

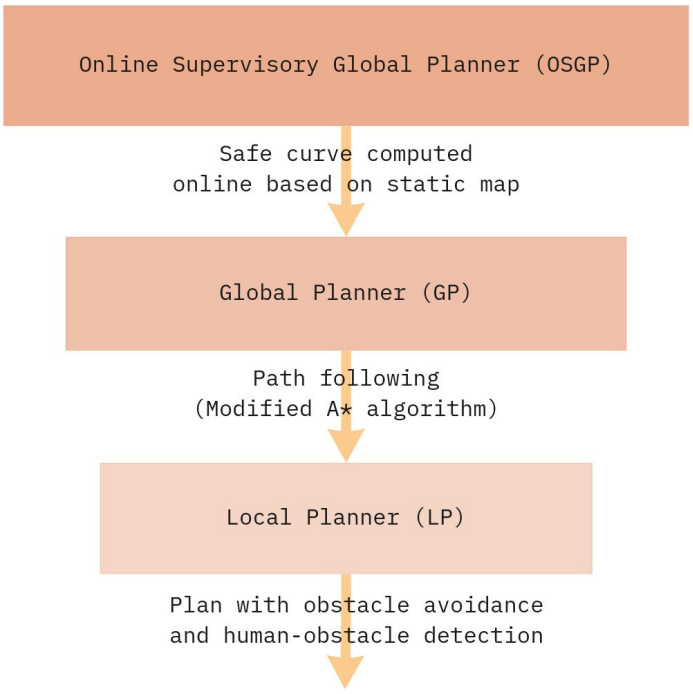


Fig. 2.18 Hierarchical structure of the online supervised global planning.

OSGP algorithm implementation

The SGP is coded in MATLAB and provides as output a set of waypoints from a starting pose towards a final goal. The MATLAB function also registers as a ROS node, in order to convey the planning information onto the ROS Parameter Server. The communication with the robot ROS network is performed through the MATLAB ROS Toolbox, that allows the local machine to compute and send the waypoints, and read the status of the robot in order to trigger the re-planning behaviour. The waypoints values are then suitably namespaced to ease potential multi-robot implementations, and made available to the system ROS nodes. The whole MATLAB part is run in *headless mode*, i.e., the code is run from the command line with specific options that suppress the display server, the splash screen display, and the desktop version modules, which significantly reduces the CPU usage.

Following a system of event-based triggers, the SGP algorithm is re-computed only when strictly necessary, to ensure a deterministic and time-efficient behaviour. As previously mentioned, the path computed by the OSGP can be considered as safe, since it is based on a binary occupancy grid map in which static obstacles are conservatively enclosed by ellipses of minimum radius; also, the GP implements a modified A* algorithm where waypoints calculated by the OSGP are favoured in terms of cost, for the heuristic global plan computation. As a human obstacle is detected and sufficiently close to the AMR, the LP deviates the robot and a trigger is sent to the OSGP for path re-computation. Note that in order to let the robot trigger the OSGP planning while already overcoming the human obstacle, a small delay is introduced. The Sen3Bot current pose used as a starting pose for the collision-free motion planning algorithm. The LP is utilized to steer clear of dynamic obstacles, while YOLO is employed to detect and avoid human obstacles applying a conservative (more details can be found in Section 2.1.3). The overall algorithm flow is showed in Figure 2.19.

Moreover, in order to better highlight the core difference from the SGP algorithm proposed in [6], the expected behaviour is represented in Figure 2.20. When using the offline SGP version, a significant deviation from the supervisory path could result in the activation of the pure A* mechanism, which computes the shortest path (grey dotted line) to reach the final goal. However, this is undesirable, since we want the Sen3Bot to resume its plan along the path considered safe. To make up for this unwanted behaviour, the SGP safe virtual path is re-planned online, taking as starting pose the current one (blue solid line).

The proposed OSGP algorithm presents some implementation measures that can improve the technology criticality level [257] and can be considered as catalyst for an eased transferability to real industrial contexts.

- *Containerization as portability facilitator*. The OSGP algorithm with human obstacle detection and avoidance has been containerized using Docker [253]. Robot-agnostic elements have been grouped based on their main task, e.g., AMR vision, AMR navigation, and SGP computations. Therefore, to ease

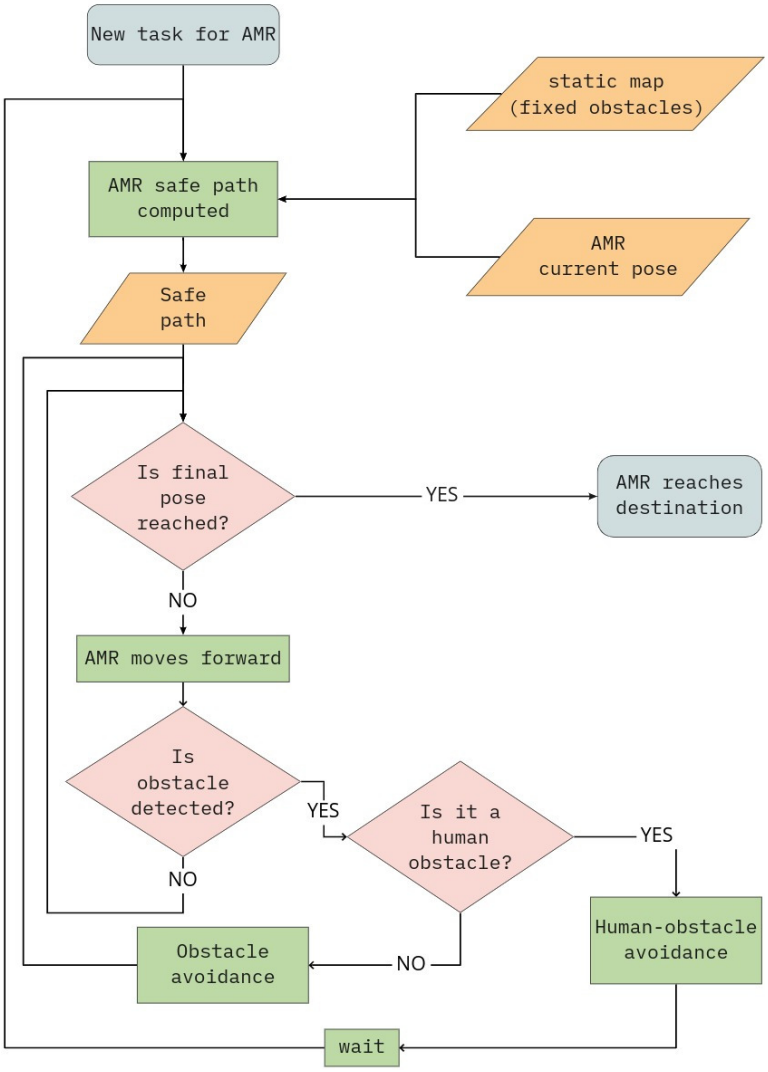


Fig. 2.19 Flowchart for the OSGP algorithm.

up the migration to other hardware specifications and software environments, all the robot-specific elements and tuning parameters have been grouped to enable suitable edits. The use of containers can speed up the transfer from laboratory demonstrators to commercially available AMRs: in fact, the Linux containers technology is considered a lightweight alternative to virtual machines [258], and allows the end-user to run the containerized application having the installation of Docker on the target machine as only constraint.

- *Cost-efficient improvements as technology transfer enablers*. Using cost-efficient sensors enhanced by deep learning algorithms can be considered a key enabler for technology transfer in contexts that would, as a matter of principle, not consider it. By exploiting sensors which may not fall within the classification of high-end and/or high tech devices, the focus is moved

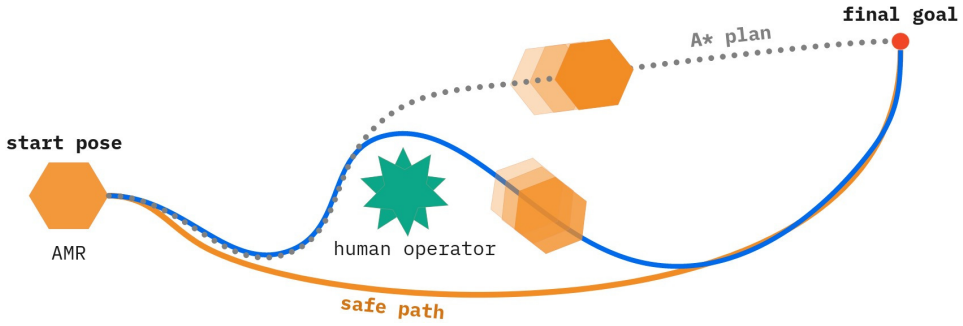


Fig. 2.20 SGP behaviour versus OSGP behaviour.

from the economic value to the innovation meaning of a resulting solution. Upgrading obsolete equipment can favour the adoption of new technologies avoiding to exacerbate low technology transfer rates that may affect Small and Medium Enterprises [230].

Experimental results

The testing scenario emulated an industrial context, involving a mobile platform moving in a closed space shared with human operators, as warehouse corridors with racks or assembly workstations with conveyors would be. The tests working space is shown in Figure 2.21, reporting the MATLAB occupancy grid map and OSGP path plot, and the corresponding GP path plan visualization on ROS *rviz*.

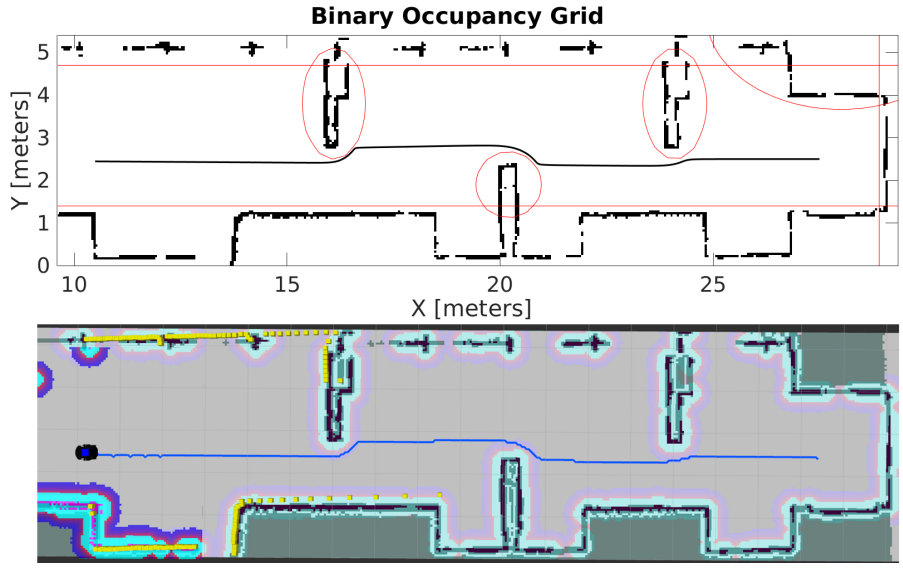


Fig. 2.21 Top: MATLAB OSGP path plot. Bottom: *rviz* visualization of the static map used for SLAM navigation.

We employed a Pioneer 3DX mobile robot equipped with a SICK LMS200 laser range finder (10 m range, 180° scanning angle). The video stream from an entry level IP camera served as vision source, fed to the YOLO real-time object detection system. The core processes were performed on a desktop PC with a Intel Core i7-7700 CPU and a dedicated GTX1060/6GB GPU.

It is worth pointing out that some open issues identified in the practical implementation of the offline SGP algorithm, have been solved. Among these, a smoother behaviour have been achieved by substituting the `TrajectoryPlannerROS` local planner with the `TebLocalPlannerROS`, implementing the TEB algorithm [172]. As anticipated, the MATLAB code has been run with its GUI switched off (headless mode), since no interaction is needed during the algorithm execution. In particular, the supervisory planner function is run according to specific trigger flags which let MATLAB and the ROS system interact and automate the re-computation process.

The overall high level setting representation is shown in Figure 2.22; the execution of the online SGP algorithm in different test scenarios is showcased in the video available at [259], whose relevant timestamps can be found in each test description.

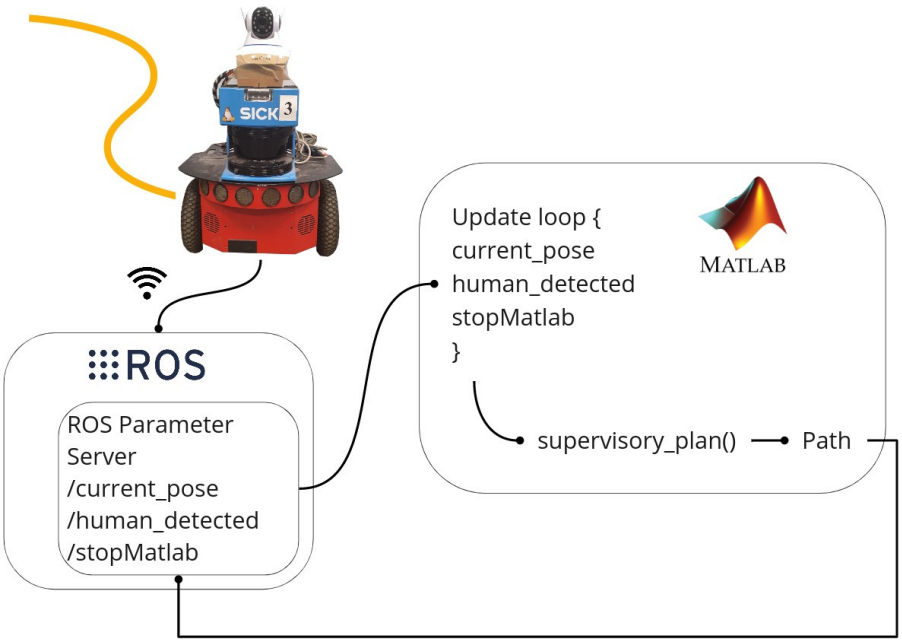


Fig. 2.22 High level algorithm implementation schema.

Test scenario 1 – OSGP re-planning behaviour (Figure 2.23): we tested the re-planning behaviour of the OSGP when a human obstacle is identified. In this case, the person represents a human worker performing some operations in front of a workstation. At timestamp 00:41, the AMR starts moving along the SGP safe path, tracked by the GP plan (blue solid line); as a person is detected, the LP (orange solid line) starts its standard obstacle avoidance mechanism. As the human gets closer, the OSGP is eventually triggered (00:58) to compute a new reference path for the GP.

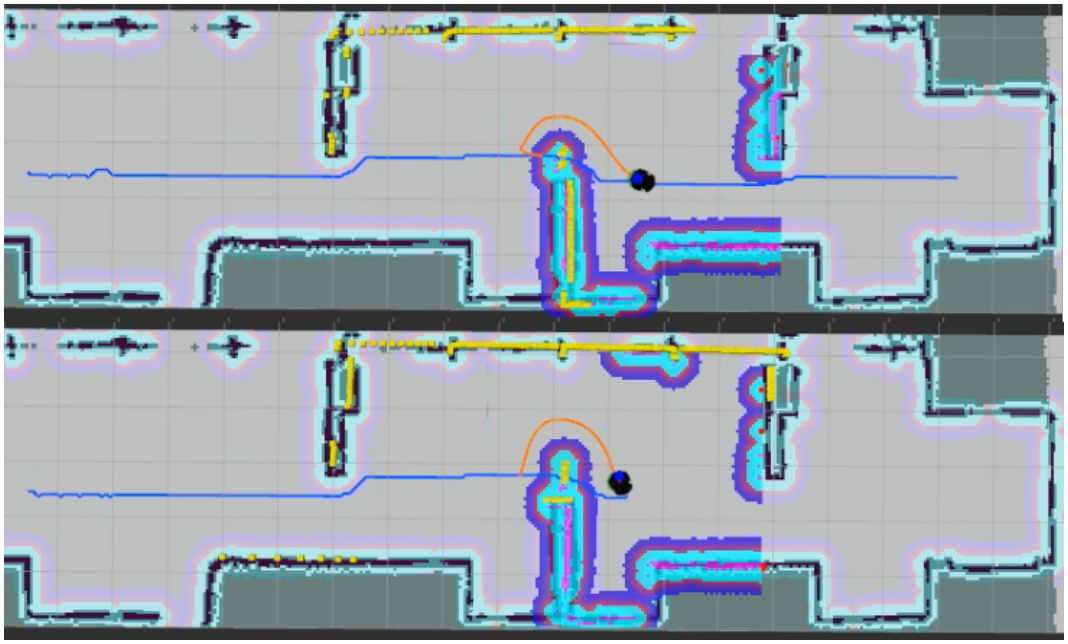


Fig. 2.23 Test scenario 1: *rviz* view of the OSGP re-planning behaviour after human detection.

In such a way, the Sen3Bot is always brought back towards the safe path even if its motion is significantly deviated. As expected, the robot overcomes a person maintaining a conservative distance imposed by the virtual obstacle publication.

Test Scenario 2 – human detection without re-planning (Figure 2.24): at 01:15 of the video, an operator performing some quick checks along a rack is detected but, being far enough from the Sen3Bot, the OSGP path re-computation is not triggered.

In this case the human presence does not disturb the AMR activities and a re-planning would just introduce unnecessary overhead. Note that the operator pose is published as a virtual obstacle in the map (red dots).

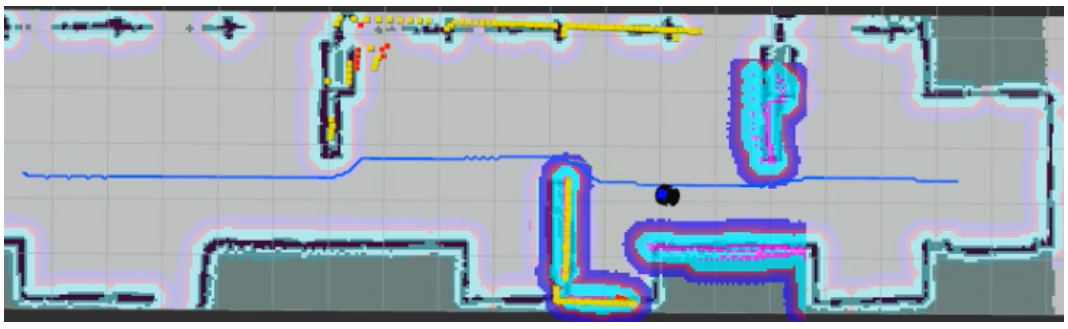


Fig. 2.24 Test scenario 2: OSGP behaviour when a human obstacle is sufficiently far from the AMR.

Test Scenario 3 – generic obstacle avoidance (Figure 2.25): the last test scenario, as shown at 01:37 of the video, considers the case in which a generic obstacle (not present in the static map) is encountered: the smart AMR is able to avoid it without triggering a new safe path computation, which helps to save computation time. In fact, the absence of a human operator in the robot vicinity does not impose a reactive resume of the safe virtual path. In conclusion, the presented online supervised planning algorithm allows to track a path imposed by the supervised planner towards a safe path, while ensuring a conservative overtaking of human obstacles. Indeed, in the absence of changes in the environment, the motion of the AMR always follows the SGP plan to the safe virtual path. This feature can potentially increase the workers' confidence in sharing the workspace with mobile robots, since their overall motion can be predicted. Besides, using YOLO, the AMR can distinguish if an obstacle is a generic object or a human operator, and gather its distance through camera-laser data fusion.

Fig. 2.25 Test scenario 3: OSGP algorithm reaction to a generic obstacle detection.

Starting from this degree of autonomy, suitable for a cooperative application, i.e., a situation in which humans and robot co-exist working on specific and independent tasks, a step toward collaboration would be introduced by the capability of the Sen3Bot to interpret the surrounding environment to act accordingly.

2.2.2 Sen3CoBot: a proof of concept

The proof of concept presented here and proposed in [11] features a collaborative extension of the Sen3Bot. In the envisaged extension, the Sen3Bot becomes a *Sen3Cobot*, acting as a collaborative mobile robot able to recognize trained operators and perform collaborative actions, directly requested by the operators through a pre-defined sequence of movements. First of all, to lay the groundwork for a collaborative approach, the Sen3Bot human avoidance behavior could be incorporated with relevant factors coming from the *proxemics theory*, e.g., speed adjustment and direction of approach. It is undeniable that safety is the foremost consideration in the design of IMRs that operate alongside human workers [260]. Note that for the proposed POC, the workspace is divided in areas according to the definitions provided in 2.1.2.

When taking in consideration the cooperative scenario where a Sen3Bot moves according to the OSGP plan, the inflation radius assigned to the identified human can possibly be so large that the robot may have problems to move in narrow spaces. An improvement to this system could be the identification of human operators so as to distinguish trained operators from general staff and, in the first case, reduce the safety radius surrounding the operator. Also, collaborative interactions can be achieved by improving the surroundings interpretation capabilities of the smart AMR. To illustrate the idea, the following assumptions will be considered:

(i) In the light of the envisioned collaborative extension:

- If the area has the highest criticality level (area of type 1), the Sen3Bot must work in cooperative mode, implying that conservative avoidance of humans is implemented.
- If the area has the medium criticality (area of type 2), the Sen3Bot can switch between two modes, cooperation or collaboration with human operators.

(ii) We consider an area with criticality level equal to 2, i.e., a sub-critical area, corresponding to a zone that includes cobots workstations and manual stations where human operators are likely to be present, but expected to be mostly static.

(iii) In such a sub-critical area, the human operators are assumed to be mainly trained ones, i.e., they are aware that the area is shared with AMRs and know how to interact with them. We assume that operators of this type are identified by a tag, e.g., a QR code, on the front and the back of a wearable leg band.

(iv) According to the Sen3Bot Net rules, if a critical area of type 2 is foreseen to be crossed by an AGV, two Sen3Bots are sent to the scene if the human operator is moving within the environment.

Collaborative extensions of the Sen3Bot Net can then be developed in the scenario described hereafter, under the above listed assumptions. The presence of two monitoring Sen3Bots in a sub-critical area ensures redundancy and, in principle, both AMRs can act as collaborative mobile robots. Nevertheless, once on the scene, only one enters a *wait4col* state, i.e., an idle state where the AMR waits for a triggering command, signalling its state via a visual indicator, e.g., a led. As the operator is recognized as trained, proximity rules can be less conservative: the robot can reduce the inflation radius around the detected human, since the latter is supposed to be aware of the former's behavior. If the operator needs assistance from one of the Sen3Bots, he/she will have to perform a pre-defined sequence of movements:

- The operator approaches the AMR in *wait4col* idle mode.
- The operator stops at a fixed distance *Df* in front of the mobile robot to let it read the front leg band tag, for a time *Tf*. The leg tag contains relevant information (e.g., the operator ID).

- If the operator turns around, letting the robot read the back tag for a time *Tb*, then the collaborative mode of the Sen3Bot is activated. We imagine the above sequence to trigger several collaborative applications, e.g., Follow-Me, assistance with materials or tools.

Feasible values for *Df* are in the range of 0.5 m–1 m, while *Tf* and *Tb* can be chosen between 2 and 3 seconds. Notice that the time and distance parameters considered in the sequence take into account a suitable tolerance. Furthermore, if the operator executes such sequence for the second time in front of the same robot, the collaborative mode of the mobile agent is deactivated; in which case, the Sen3Bot can be re-assigned a new task, for example monitoring a different area.

The described Front-Back sequence intends to emulate the human interaction that is likely to take place when two persons are conversing. In this way, the mobile agent can understand this common human intention demonstration, allowing a narrowing of the gap between the different cognitive skills between human and robots. A schematic representation of the Sen3Bot modes characterizing the proposed collaborative approach is given in Figure 2.26. In particular, the HUMAN REQUEST flag is set by default to 0, since the robot starts with the cooperative mode and becomes 1 when the conditions of the first Front-Back sequence are valid, allowing the robot to switch to the collaborative mode. The flag is reset to 0 when the human operator performs a valid Front-Back sequence for the second time to the same robot.

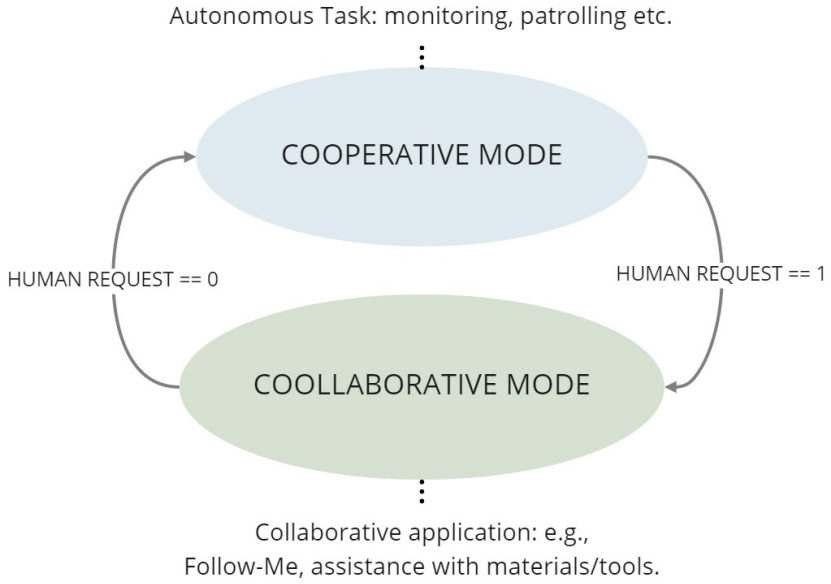


Fig. 2.26 Modes switching schema for a Sen3Bot monitoring an area of type 2, enabled to wait for collaborative task triggering, i.e., with $wait4col == 1$.

The proposed concept could potentially reduce the time needed for human workers to be trained in working with mobile robots within an industrial environment. This is because the interaction with the robots would mimic human-like behavior.

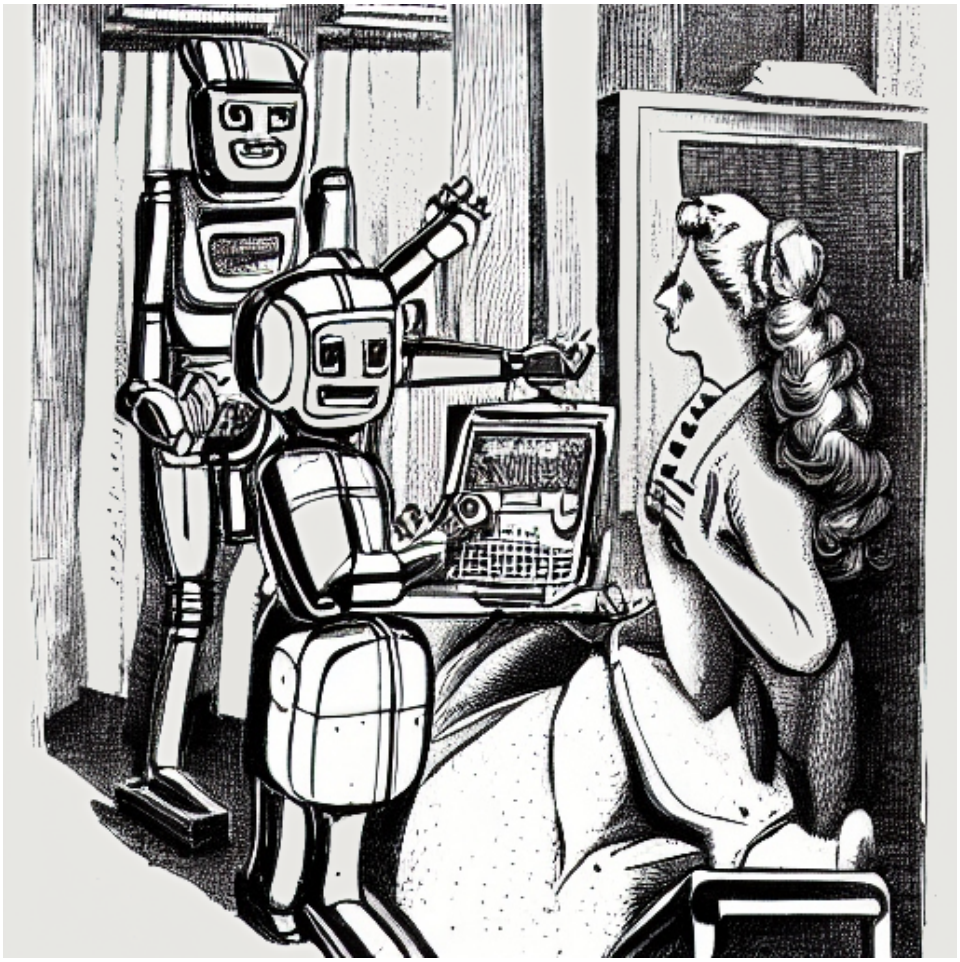
Moreover, wearing a leg band with a tag represents a low-cost solution for identifying operators, to track their activities. In fact, a tag such as a QR code may contain all the needed information for the robot and the overall system. However, it is not possible to state how robust the solution is, at least, not until a real implementation in a real industrial scenario is tested. Furthermore, running an algorithm that identifies the human operator, tracks his/her location within the map and at the same time filters and processes relevant information coming from the QR code, may need a high-performance system for computing the perception algorithm.

Finally, taking into consideration the current implementation of the Sen3Bot agent, such envisioned collaborative module could be implemented by taking advantage of tools for tag recognition such as Zbar, for which a ROS (Robot Operating System) wrapper node is available [261]. Also, other tools are present as Open-Source material, mainly based on the OpenCV library, whose ROS compatibility is well-established as well. Please note that this additional collaboration module would not imply the need for further sensors, since the already available IP camera video stream would allow the application of the afore-mentioned vision tools.

In this section, we have seen how a network of mobile agents can contribute to monitoring workspaces shared between human operators and robots, and how, depending on the needs, it is possible to switch from supervised navigation to a fully autonomous one. A homogeneous team of AMRs can help a homogeneous fleet of AGVs by providing the opportunity to upgrade the overall mobile robots network, which takes advantage of the diversity of both types of mobile robots. Similarly, the heterogeneity between robots and humans is a source of richness in terms of skills, and this can be the trump card to achieve fruitful collaboration. The next chapter will show how the ability of robots to learn from humans is crucial to achieve such goal.

Chapter 3

Robot learning for improved industrial collaborative applications



EARNING is the transformative process resulting from experience that enhances the potential for better performance and future learning [262]. We, as human beings, are used to learn from birth, through experience and trial-and-error reinforcing our behaviour in a specific environment while pursuing a certain task, also taking into account the overall context. The concept of improving learning by doing what we are actually trying to learn to do dates back to 1897 with the principles of the pedagogic theory of *learning by doing*, according to which "an individual must interact with the environment and implement action in order to learn, and thinking with the aim of problem solving occurs only when confronted with a problem" [263]. The so-called *artificial intelligence* (AI), as opposed to natural intelligence, is the concept within which the learning process for a machine falls; the idea of *machine learning* (ML) originated from a mathematical framework proposed in 1943 by the logician Walter Pitts and the neuroscientist Warren McCulloch [264], and then popularized in 1959 by an American pioneer in the field of computer gaming and artificial intelligence, Arthur Samuel.

Machine learning has then a relatively long history behind it, but its application to robotics leads to a relevant learning outcome difference: *robot learning* differs from traditional machine learning in that it places greater emphasis on producing *actions* as output while taking in the environment as input [265]. To go back to the considered context of human-robot collaboration in manufacturing, we recall what has been introduced in Chapter 1. As said, the fourth industrial revolution marked the turning point towards the industrial integration of all the technological advances which allowed the implementation of a series of new types of systems, leveraging the increase in computational power. Moreover, *lean* principles have shaped technology and decision-making processes to reduce material waste and non value-added activities [266], [267]. What has deeply transformed the structure of production lines is today's market demand in developed countries, which shifted its interest from mass production to *products on request*.

Intelligent collaborative robotics can represent a key enabler of custom manufacturing production, to try and top mass production costs and production times, evolving toward a smart custom manufacturing system. To do so, Human-Robot Collaboration (HRC) and Interaction (HRI) should aim at identifying a balanced synergy between the human and the robot (mobile or fixed base cobot) so as to value and enhance the cognitive skills and experience of the former and the accuracy, strength and efficiency of the latter. In fact, through performing essential but less complex tasks, cobots enable humans to concentrate on assignments that demand advanced reasoning and decision-making abilities.

This increased involvement of human workforce in automated industrial processes characterizes the Industry 5.0 concept, identifying humans as the enabling element for mass customization [268]. Proactive HRC can be obtained with bi-directional empathy and holistic understanding during the collaborative execution, spatio-temporal cooperation prediction, estimating the interaction among all elements involved (human, robot and workpiece - if any), and teamwork self-organization learning as a result of converging knowledge [269]. Therefore, to make robots the ideal human collaborators, they should possibly behave as human team members. This is because making the robot capable of human-like behaviors and decision-making would allow for smooth interaction with human workers. Indeed, if robots were to learn how to behave with levels of artificial intelligence as comparable to natural intelligence, this kind of human-robots collaboration could revolutionize the integration of robots not only in the industrial field but in our everyday life.

To achieve such an advanced artificial intelligence there is still a long way to go, but gradual changes are pursued towards such direction. Among the hindering causes of smooth HRC there is traditional programming or rule-based programming to make robots behave in a desired way per each specific case. In fact, in traditional programming, a programmer writes code that defines the exact steps that a machine should take to complete a task, making difficult to generalize and requiring a level of expertise from the user in case the code has to be edited for ad-hoc online adjustments.

The transition to ML-based methods enables the robot to learn patterns and make decisions *based on data*, emulating the human capability of learning from doing and improving the performance over time through *experience*. Moreover, these kind of methods also allow to teach robots tasks that are difficult to execute using traditional programming methods.The cognitive elements characterizing the human learning process, namely attention, memory, perception, metacognition and motivation, can all be mapped to robot learning aspects:

- *Attention* has similarities with the capability of a robot to identify patterns in the incoming data, i.e., selectively attending to certain information in the environment while ignoring others.
- *Memory* can be assimilated to the ability of a robot to avoid forgetting past experiences for better future performance.
- *Perception* can be easily re-mapped to the perception capability of the robot of its surrounding.
- *Metacognition* of human learning can be compared with a robot ability to learn how to learn while being unsupervised.
- *Motivation* can be represented by cost functions that a learning robot tries to minimize or the reward it tries to maximize.

Artificial intelligence methods are usually grouped as shown in Figure 3.1. All definitions made for machine learning can be transposed on robot learning, as robots are simply machines that *act in* and *interact with* the surrounding environment.

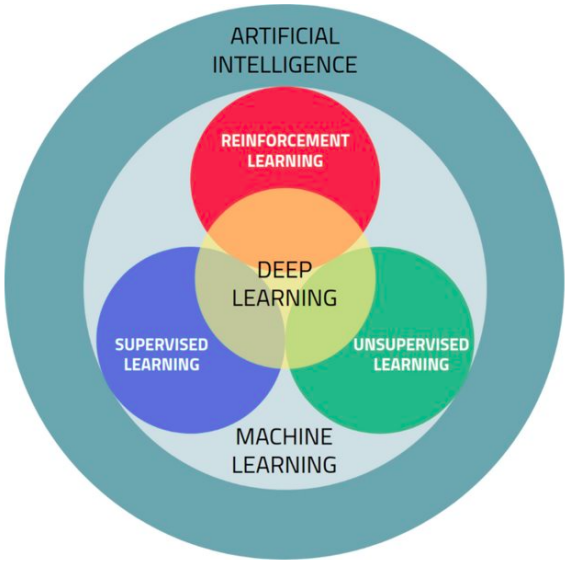


Fig. 3.1 Conventional AI learning types schema [270].

Thereby, based on the level of human intervention required we can identify the following types of robot learning methods [271]:

Supervised learning

In supervised robot learning, the robot is trained using labeled data, meaning it is provided with inputs and the corresponding desired outputs or labels. The goal is to learn a function that maps inputs to outputs based on the training data, and use this function to make predictions on unseen data, which typically results in performing some action or behaviour adjustment. Among the supervised learning algorithms there are linear regression, logistic regression and decision trees. The main drawback of supervised learning is the need for labeled data, whose collection can be expensive and time-consuming.

Unsupervised learning

In unsupervised robot learning, the training data consists of unlabeled examples, i.e., no specific desired outputs or actions to take are specified. The goal is to learn the underlying structure or find patterns in the data, and make predictions without any guidance or supervision. Example algorithms include clustering, anomaly detection, and generative models such as autoencoders and variational autoencoders. The main drawback of unsupervised learning is that it can be difficult to evaluate the learned model performance, i.e., determine whether the model has learned useful and relevant information or not.

Reinforcement learning (RL)
: This recalls the *learning by doing* theory: the robot learns through trial-and-error, without requiring explicit programming for every possible scenario. In fact, the robot is provided with a goal or objective, and then it must take actions to achieve that goal: each taken action is either rewarded or punished, based on whether it brings it closer to or further away from the objective. The robot reinforces its behaviour given its own experiences, gradually learning how to achieve the goal more effectively in potentially dynamically changing environments. The main drawback is that learning can require a large amount of data and time, leading to the exploration-exploitation dilemma.

All these learning methods can have a *deep learning* (DL) version, grounded on deep neural networks (DNNs) that automatically learn and extract relevant features from raw data without the need for manual feature engineering, which is a time-consuming and often challenging task in traditional learning. This way, complex patterns and relationships in data can be captured. Nevertheless, DL methods require a large amount of data and computational resources. As can be noted, the leitmotiv that unites the existing learning methods is the extensive use of *data*. As such, one can straightforwardly understand how big data and learning algorithms are tightly connected. Nevertheless, the need for a great amount of data to achieve good and generalizable models, often represents a bottleneck for learning applications. It is thus necessary to find a trade-off between model complexity, data quantity and quality and required generalization, which all depend on the application requirements.

The contributions reported in this chapter seek to exploit robot learning in the industrial context, in particular offering possible solutions to enable improved human-robot collaboration in manufacturing flexible production lines.

3.1 Data-driven framework for robots collaboration with human workers

As thoroughly analysed, the manufacturing assembly lines of the future are foreseen to dismiss fully unmanned systems in favour of anthropocentric solutions. However, bringing in the human complexity leads to modeling and control questions that only data can answer. The work first outlined in [10] (as a proof of concept, along with its development feasibility analysis) –and presented hereafter– showcases a data-driven framework to let human-robot collaborative processes overcome the barriers to successful interaction. In a holistic vision, the human-robot system can be interpreted as a component within the context of the well-known and well-established concept of Cyber-Physical System (CPS).

In [272], the authors present an architecture to be used as a framework to achieve self-organization and self-adaptation using a multi-agent based technology, in order to achieve auto-reconfiguration without human intervention to comply with product

customization requirements while reducing costs. The human operator is involved in the task execution but not considered as part of the multi-agent system. The work presented in [273] reviews the role of Multi-Agent systems (MASs) as main tool for CPS implementation. According to the analysed works, a MAS technology reference architecture should be technology and system agnostic to ease its instantiation and to favour its co-existence and integration with pre-existing automation. The agent-based solution reconfiguration is interpreted as not strictly related to physical modification of the system but rather as a multi-objective reconfiguration, which could potentially include sustainability aspects as energy saving, reuse of equipment and optimal use of human resources.

In [274], the human operator interacts with CPSs exploiting a modular robotic cell able to reconfigure itself according to the requested operation. In [275], the Decentralised Manufacturing System is presented as an enabler of distributed production layouts. To achieve optimal HR work distribution, the authors propose an algorithm that assigns a collaboration potential to each movement in order to distribute process capabilities. Moreover, in the context of MASs in smart manufacturing systems, the integration of human actions in the control system, also referred as human-in-the-loop (HITL), allows to consider the human-machine cell as a single agent with the ability of self-arranging its work locally [276]. The authors of [277] claim that fully unmanned factories cannot be implemented not only due to ethical or social reasons, but mainly to the unfeasibility of such complex control systems. To achieve interoperability, they propose augmented reality as a rapid learning solution and the use of machine learning (ML) techniques to enable self-predict capabilities for implementing dynamic reconfiguration.

As a matter of fact, Human Cyber Physical Systems (HCPSs) are envisioned to become the core of the factory of the future [43], suggesting an increasingly clear transition towards anthropocentric approaches. Therefore, a desirable solution would leverage human cognition skills and machines' processing and data mining capabilities to implement human capabilities augmentation. HRC potential could be unlocked if the different cognition models of human operators and robotic agents were suitably interfaced to allow effective and efficient communication [278]. One possibility is the emulation of human interaction paradigms that typically imply an anticipatory behaviour, proven to be the human natural and dominant mode when performing interactive activities, e.g., driving [279]. This, intuitively, is implementable if the system has prediction capabilities on its future states. However, being a human agent part of the considered system, inferring the overall dynamics of the system is not easy.

Indeed, much of the available literature on human intention recognition deals with the problem of modeling and interpreting the human motion itself. In order to infer the user intentions, many works consider a nonlinear joint model of the human to estimate muscle activations by monitoring electromyography (EMG) signals using wearable sensors [82]. However, given the measurement limitations linked to human physiological parameters, other solutions, as the one proposed in [280], utilise ML to estimate a neural network (NN) state model relating the human joint angles and

EMG signals. In general, data-driven approaches using datasets to train activity models that link events to activities allow to take into account the uncertainties of the model with the disadvantage that (i) large dataset are needed to learn the model and (ii) interpretability and generalization are limited [281]. Note that the time required to produce such large datasets is not compliant with short setup times.

A large and growing body of literature has investigated the role of data synthesis and relative data augmentation to solve the data scarcity problem. In fact, this issue is becoming more and more relevant as the majority of AI-based methods heavily depend on data. Most works concern low-data regime image classification problems where small training datasets cause bad generalization, leading to models affected by overfitting. The latter can usually be reduced through regularization and drop-out techniques that, nevertheless, do not tackle the problem at its roots: here is where data augmentation comes into play. Among the available techniques for data augmentation, Generative Adversarial Networks (GANs) have demonstrated their power allowing for much larger invariance space with respect to typical data augmentation techniques that usually rely on linear transformations [282]. The use of GANs as data augmentation method ranges from medical applications to emotion classification and, when considering the industrial context, several examples can be found that apply this technique to improve fault detection systems [283], [284].

To place the data-driven framework we propose, it is worth pointing out that HRC in manufacturing applications have showed to have numerous facets depending on specific parameters, such as the degree of collaboration, dictated by the product characteristics [285], and the human operator long and short term attributes [286]. Keeping in mind these aspects, several approaches when considering close-proximity interactions between human operators and robotic systems can be found in the literature on HITL. The framework presented in [287] provides a remote human-in-the-loop control mechanism that brings an enhanced reality visualization tool and a gesture recognition system together to implement a natural interaction between the human operator and two manipulator collaborative robots. The method provided in [288] aims at recognising patterns by identifying maneuvers or motifs in time series trajectory data generated in HITL applications. In particular, to perform pattern extraction, the authors exploit ML algorithms on symbolic representations obtained by clustering feature vectors, which in turn are derived from trajectory segments. The work in [289] provides a deep learning (DL)-based visual prediction method in which a multilayered neural network realizes recognition and prediction of various human manipulation actions. Specifically, the algorithm uses partial sequences of motion data, allowing to recognize and predict motions before the action takes place.

In [290], the human arm trajectory is segmented and partial segments are used for human intention inference. Specifically, they apply time series classification, which finds similarities between the current partial trajectory and each representative trajectory of each task, corresponding to the average of all training trajectories. Moreover, the methods in [291] exploit gaze approximations and skeletal data to estimate the reaching goal intention. This information initializes a learning algorithm on a nonlinear state-space model with the uncertain system dynamics modeled

using a dynamic NN; the estimated human trajectory is then used as a reference for safe and efficient robot motion. These examples highlight that, while some research has been carried out on HITL collaborative applications involving standard cobots, i.e., collaborative manipulators, frameworks involving Autonomous Mobile Robots (AMRs) and mobile manipulators are not taken into account, supposedly due to the lack of clear mobile-cobots safety regulations [11]. Nevertheless, mobile robotics represents one of the key enablers of flexible manufacturing and should be considered in HITL applications. Namely, in passive HITL, i.e., when the human does not exercise control over the system [292], e.g., applications engaging the robotic system for monitoring the human operator and autonomously adapting to the situation.

Below we lay out the specifications for the proposed data-driven framework, involving collaborative robotic systems at large, with the role of distributed sensors. The system goal is to recognise the executed process, inferring it from the human operator state, and make it known to the robotic system, with the aim of implementing a suitable control to maximize execution performances. The outlined solution seeks to reduce the complexity of available collaborative systems capable of behaviour recognition, to investigate if less information can be more significant than what has been considered so far.

Framework Outline

In this section, a preliminary definition of the proposed HITL framework is carried on. To better describe its components, a use case scenario posed as a problem scenario is considered and described hereafter. We consider a custom manufacturing context where flexible production lines are implemented using both fixed-base and mobile robots. This means that the execution of manufacturing processes is distributed among agents, with the aim of complying with the flexibility requirements of custom products and products on request, whose assembly often heavily relies on cognitive and experience-based capabilities of human operators.

As mentioned before, fully-automated operations do not always represent the most suitable solution to certain operations. In fact, several operations are still necessarily brought on by hand, especially in the case of small-scale job shops. For this reason, the robotic system collaborating with the human operator can be considered as an apprentice, i.e., it exploits information coming from the human counterpart to autonomously learn and gain awareness about the executed operation or task. Note that we consider an operation as composed of a set of tasks, where tasks are performed by the operator in collaboration with cobots, moving from one workstation to the other.

An envisioned use case is that of a job shop operation where a human operator is needed for the successful finalization of the product. The operation to manipulate or assemble a product requires her/his presence at different manufacturing workstations placed in the shop floor, depending on the product to be delivered (Figure 3.2).

Fig. 3.2 A robot-assisted assembly use case for the presented collaborative framework. The human operator needs to carry out several tasks performed at ad-hoc workstations. The robotic system observes the operator to determine the current process to infer its own desired behaviour.

Given the above scenario, the problem statement can be defined as follows: given a collaborative operation, the robotic system should implement proactiveness to maximize execution effectiveness and efficiency, complying with lean manufacturing requirements. To face this problem, the idea is to have a robotic system able to autonomously understand the on-going process to behave accordingly. This improves the interaction during collaboration (the robotic system awareness allows for anticipatory behaviour) leading to effective execution. Furthermore, it reduces waiting times of the human operator, corresponding to a minimization of the overall process execution time, hence of the execution efficiency.

The solution implementing the described idea has the following main objectives: using human information to learn a model for process (operation or task) recognition, and controlling the robotic system based on the process recognition output to implement anticipatory behaviour. The resulting framework is composed of the following two physical components: (i) the human operator, representing the main source for process classification, and (ii) the robotic system, serving as sensor data source and used for anticipatory behaviour.

The framework considers the human-robot comprehensive system as a CPS that implements a passive HITL manufacturing application by abstracting from the mentioned two physical components what we refer to as the *system functions*. The combination of such functions determines two levels of abstraction: the *global*

functions of the system consist in operation-oriented classification and associated control of AMRs and cobotic manipulators, and the *local functions*, in charge of task recognition and related actions on behalf of standard cobots. From a practical point of view, the system provides a global recognition of the human motion between workstations of the flexible production line (operation execution), and a local classification of human manipulation and actions (task execution). To stress the need to involve mobile robots in HITL manufacturing applications, the focus is put on the description of the global functions of the framework. The framework functions can be summarized as follows:

(i) *Human operator modeling*. From the global point of view, the human operator is approximated by a 2D point on the plant map. Indeed, human modeling is complex and with the goal of operation recognition, considering the 2D motion of the human operator should be sufficient.

(ii) *Data collection*. In order to identify the 2D point position corresponding to a human operator, an algorithm filters sensor data about the surrounding environment to retrieve the relevant piece of information. In this case, data are gathered by the robotic system itself, serving as a distributed network of sensors.

(iii) *Robotic system awareness*. Globally, the estimated sequence of 2D positions of the human operator is fed to a classification algorithm. Based on the classification capabilities of the model, the ongoing operation should be identified and a reference trajectory for the mobile robotic system generated.

(iv) *Robotic system control*. The computed robot reference trajectory is employed for controlling the mobile platform, which is then able to anticipate the expected motion of the human operator.

An overview of the framework is shown in Figure 3.3.

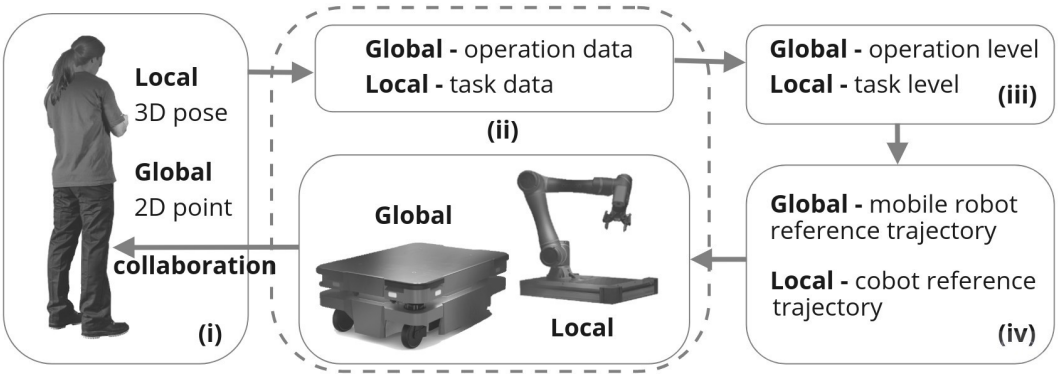


Fig. 3.3 Framework high level schema.

Development feasibility analysis

In this section, the development analysis of the proposed framework is tackled, mainly concerning the technical limitations and feasibility of the project itself. Here, the possible implementation choices evaluation is reported. In particular, as anticipated, we focus on the description of the *global functions* of the framework and the physical components involved in it.

The following analysis can be interpreted as the evaluation of possible instantiations of the abstract global functions of the framework. The authors consider functions (i) and (ii), namely *Human operator modeling* and *Data collection*, as already available. In fact, the Sen3Bot meta-sensor [7] has the specific function of monitoring industrial scenarios while detecting human operators, implementing a safe behaviour when approaching them. The Sen3Bot can be considered a sensor itself, able to publish the 2D position of the human operators on a shared map, in order to make it available to all the robotic systems' components In [9] the possibility of a collaborative behaviour has been introduced: within this HITL framework the Sen3Bot would be upgraded to a collaborative Sen3Bot (Sen3CoBot) able both to autonomously navigate and be aware of the executed process. As a side note, to lighten the computer vision computational effort of the Sen3CoBot, downgrading the Sen3Bot original software choices to favour edge computing should be considered.

The *Robotic system awareness* (global function (iii)) development and implementation should take into account several feasibility aspects. Given the intention of employing Artificial Intelligence for this function, some technical details intrinsically influence the algorithmic choice. First of all, data availability must be considered: ML methods typically require structured data, implying accurate manipulation if not available; on the other hand, DL algorithms often require a large quantity of training data to avoid overfitting. Moreover, another aspect that must be taken into account is the available hardware. Operations performed within DL models, such as convolutions, are best executed on a GPU parallel architecture. A possible distributed solution could involve the use of edge computing systems, e.g., embedded GPUs. Additionally, Cloud deep learning services offer several solutions, which take advantage of shared powerful hardware.

Properly choosing an AI algorithm also implies selecting an unsupervised or a supervised learning technique. The function requires the recognition of the executed operation, thus a classification is necessary. Supervised learning requires a lot of labelled data to train the model. In this case, however, there are no readily available labelled datasets on 2D human trajectories during an operation, and labelling should be done for each new recorded trajectory, which can be quite time demanding. Conversely, unsupervised learning would allow to train and use the clustering model with unlabelled data but, because there are no labels, there is no way to identify the process and retrieve the associated reference trajectory for the robot. To face this problem, a mixed method supported by GAN-based data augmentation could provide a feasible solution. Clustering could be performed on large sets of synthetic data generated by a GAN, then the obtained clustered data could be used for training a

classification model. Note that we assume the number of clusters c, i.e., the number of operations, to be known a-priori. However, depending on the required accuracy, it could be necessary to bring on the labelling and feature engineering processes for supervised classification.

Hence, to implement the operation recognition function, a suitable trajectory time series data classification algorithm should be identified. Ideally the model is trained and fed with partial trajectory data to implement the desired anticipatory behaviour and, based on the confidence with which the trajectory is classified, use the robot reference trajectory associated to the recognized operation to control the robotic system. The first development step should identify the best data structure to be used, since it must contain both the human operator trajectory and the Sen3CoBot trajectory. In fact, during training the robot should be teleoperated to assist the human operator when necessary, so as to record both the sampled human trajectory $\boldsymbol{x_h}$ and the Sen3CoBot trajectory $\boldsymbol{x_r}$. Once the operation is recognized by feeding the model partial trajectory data, the Sen3CoBot is given $\boldsymbol{x_r}$ from that instant k on, as the reference to be tracked. The sampling interval, at first, could be fixed and assigned, while a more dynamic tuning could be implemented to discretize the trajectory based on some relevant criteria. A summarizing schema of the *Robotic system awareness* function instantiation is shown in Figure 3.4.

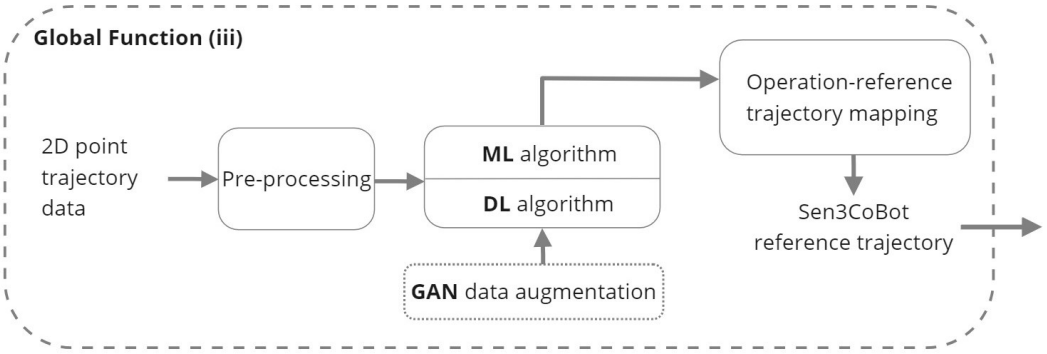


Fig. 3.4 Robotic system awareness summarizing schema.

Finally, the *Robotic system control* function will receive the Sen3bot reference trajectory $\boldsymbol{x_r}$ associated with the recognized operation. The Sen3CoBot control could implement a mission planner to deal with exceptions, such as an obstacle present in the next goal position in the reference trajectory. Moreover, if no robot motion is predicted to be necessary within the recognized operation, the Sen3CoBot could go in an energy saving mode/state to minimize battery consumption. A further functionality that would be interesting for the Sen3CoBot control, is a sort of proactiveness of the robotic system, that through visual or sound outputs could cause a reaction in the human behaviour, for example to signal possible delays with respect to the expected operation execution time.

The next section reports the implementation we have brought on as a solution for one of the main data-driven framework global functions.

3.1.1 Sen3CoBot learns how to recognize operations based on 2D map human motions

The solution unfolded in this section, and presented in [12], aims at achieving operation recognition by monitoring the human collaboration motion on a 2D map. Many human-robot collaborative applications in flexible manufacturing involve manipulator cobots, whereas little attention is given to the role of mobile robots. In our case we consider the information provided by a monitoring Sen3Bot (2.1) to demonstrate that human behaviour prediction for improving human-robot collaboration applications can attain satisfying results by classifying the corresponding human path on a map, taken as an image data sample. Note that, this ability enables the Sen3Bot to achieve collaboration with a human operator, making it an additional feature of the collaborative Sen3Bot, i.e., the Sen3CoBot, introduced in Section 2.2.2.

Human-robot collaborative applications are generally based on some kind of co-working of the human operator and the robot in the execution of a given task. A disruptive change in the collaborative modalities would be given by the capability of the robot to anticipate how it could be of help for the operator. In case of an Autonomous Mobile Robot (AMR), this would imply not only a safe navigation in the presence of a human operator, but the automatic adaptation of its motion to the specific operation carried out by the operator. To this aim, we want to enable a Sen3CoBot monitoring a human operator to translate his/her motion into a path plotted on a 2D map, and then classify the operation based on the image sample.

A human trajectory tracking algorithm mixing human-oriented Global Nearest Neighbour (GNN) data association and Kalman filter-based human tracking is proposed in [293]. Vision-based approaches to detect humans and objects are widely used; however, they do not provide accurate range information. For this reason, the authors of [293] combined the information of a 2D lidar and a RGB-D-based YOLO (You Only Look Once) system to correct missed information while tracking the human motion. A dataset containing human motion trajectory and eye gaze data is presented in [294]. The data of humans moving in a room are collected mainly through a motion capture system enriched with 3D lidar, eye gaze detectors and a RGB-D camera information; the recorded trajectories are available in 2D maps, and used for training and motion prediction models of human motion. In [295], human body pose and gaze are analysed to obtain an accurate prediction of the human's intentions. A Recurrent Neural Network (RNN) is used to predict sequences of multiple and variable length actions, based on gaze and skeleton datasets.

A Multiple Predictor System (MPS) for human motion prediction is proposed in [296]. Depending on the context, it automatically switches between three individual classifiers: velocity-based position projection, time series classification and sequence prediction. In order to enhance the robot's ability to adapt its behaviour in environments shared with humans, the MPS is added in the path planning algorithm. In particular, the human's head 2D coordinates are used as features for the predictors [297]. In [298], a human motion prediction algorithm is proposed using a Hidden Markov Model (HMM). The HMM learns a set of movements executed

by a human operator in an assembly task and then generates motion transition and observation probability matrices. In this way, it is possible to predict the motion of the human operator and perform assistive motion planning in Human-Robot Collaborative applications. Similarly, HMM is proposed in [299] to recognise human activities based on the principle object affordances, i.e., the relationship between the activity and a particular object/tool.

AlexNet, a Deep Convolutional Neural Network (DCNN) is modified employing transfer learning-enabled algorithm in [300], to enhance the robot's capability to learn human's actions. In particular, human actions can be divided in: (i) generic body motions, e.g., grasping or holding a tool and (ii) specific movements related to a context, e.g., actions performed while using a tool. The training procedure involves two separated deep neural networks, that analyse the human motion and identify the tools associated to the tasks. A multisensor framework exploiting online transfer learning techniques for human tracking is presented in [238]. The performance for all possible combinations of 3D lidar, 2D lidar and RGB-D cameras are evaluated, and in particular, the solution that combines 2D lidar and RGB-D camera achieved the best results in terms of performance and precision to learn people's movements in the environment. In fact, the sensor's choice may enhance the robot's perception of the human [11], and therefore improve its learning curve about human intentions.

Position-based Operation Recognition

This section briefly recalls the definitions and main concepts reported in Section 3.1 – ***Framework Outline***. The main objective of the outlined framework is to exploit human information to learn a model for operation and task recognition, to make a mobile cobotic platform or manipulator aware of the ongoing process, allowing to enable anticipatory behaviour for improved collaboration along a flexible production line.

Problem and Idea

This work develops a *global function* of the framework, namely the recognition of the executed operation. We consider a set of tasks as composing an operation, and each task is brought on – at a "local" level – in collaboration with cobots at specific workstations. Given a collaborative application, the present work aims at allowing the system to learn to recognise the operation – at a "global" level – by collecting information on the human operator motion: at each set of human poses on the map corresponds a specific operation. Note that, learning from data with sufficient generalization capabilities requires data availability. Moreover, being able to straightforwardly identify the relevant information that a human gives back while moving around and predict the often unpredictable human behaviour are both quite complex objectives.

The idea is to emulate how a human being usually perceives its surroundings: the decision making anticipating an action is performed based on an approximated observation of the surrounding environment, in favour of efficiency and resource saving, i.e., when we look around we do not usually catch every single information coming our way before taking a decision. The goal of the presented solution is to demonstrate that the path traversed by a human operator is a sufficient information to identify the performed operation, to possibly anticipate human behaviour in the described restricted context scenario.

Solution

To capture a series of positions occupied by a human operator on a map, the proposed solution exploits the Sen3Bot meta-sensor implementation (see Section 2.1), a smart AMR whose role is to monitor and safely cooperate with humans. Within the data-driven framework the project is brought towards a collaborative evolution, the Sen3CoBot. Indeed, as specified in Section 3.1 – Framework Outline, the global functions (i) and (ii), i.e., *human operator modeling* and *data collection* are resolved by considering only the positions of humans, which are detected by a state-of-the-art computer vision object detection algorithm.

The proposed solution tracks the set of positions and represents it as a path, without including time information, as it is not relevant for operation recognition. Rather, dropping the time information (i.e., preferring the path data with respect to the trajectory data) enables the recognition of operations performed by different operators, who may take varying amounts of time to complete each operation. Thereby, the chosen solution considers that the digital representation of path data, when plotted on 2D map, is simply a matrix; by dropping the time information and given the duality of images as matrices, we translate a spatio-temporal data recognition problem into an image classification problem. Interpreting the data in this way has proven to be crucial, as it has expanded the pool of potential methods for solving the problem by providing access to a wide variety of established and well-documented architectures, libraries, and tools for implementing deep learning models. As such, this has enabled us to tap into a large community of developers who often make these tools available as open source material.

In the proposed solution the Sen3CoBot stack is extended by adding the capability of understanding the executed human operation, i.e., implementing the framework global function (iii) – *robotic system awareness*. In fact, the AMR currently implements passive HITL behaviour, as it monitors the area to gather position information from the detected human, and interprets it as an operation to be recognized. However, in the context of the overall data-driven framework, recognizing operations through this passive step is key for the mobile cobot to make informed decisions before taking any action: based on the confidence of the classification, the robot will be given different trajectories to follow. To this end, the robot will need to act according to the probabilities associated to each class of operations. This means the system will iteratively check, while gathering new data, if the guess has changed

and send a different reference to the mobile cobot accordingly. With the aim of dealing with the data scarcity problem, the solution takes data augmentation through simple transformations as a first step toward model improvement. Opting for image datasets instead of video datasets reduces the complexity of data and, to some extent, minimizes the presence of noise in the dataset, since the images only contain relevant information, i.e., the map and the path taken by the human operator.

Tools

We provide here a brief introduction to the main open source software tools that form the basis of the presented solution. First, it is worth recalling that the data gathering provided by the Sen3Bot is ROS 1-based (Robot Operating System), featuring a containerized vision module exploiting YOLO (refer to Appendix A). Also, the additional recognition capability introduced by the proposed implementation has been containerized. Containerization has been achieved using Docker [301], which allows to build, run and manage Linux Containers. The development and testing of self-contained applications can be done in a lightweight, clean environment. Allowing to leverage GPU within containers, local resources are exploited for running – hundreds of times faster than a regular CPU – Neural Network (NN) based algorithms, while avoiding lag problems derived from using Cloud available GPUs, and security issues.

To solve the operation recognition/image classification problem, the `fastai` deep learning library, specifically `fastai v2` [302], has been chosen. This library provides low, mid and high-level APIs to intuitively create deep learning models, either from scratch exploiting the Python libraries it is built on (PyTorch, NumPy, PIL, pandas and others), or allowing to use state-of-the-art architectures and techniques made available following best-practices to get the most out of the available hardware. The authors of `fastai` also made available an interactive book written with Jupyter [303], an open source project providing notebooks. A notebook is an interactive programming environment – whose cells' code can be modified and run straightaway – capable of working with different language backends, kernels, allowing to use several different programming languages. For the sake of the recognition solution validation, notebooks represented a simple playground for code development and testing. We built our solution upon [304], which provides a docker image with pre-installed `fastai v2` libraries and notebooks, which has been adapted to solve the considered problem. Figure 3.5 summarizes the overall structure and tool choices.

Before moving on to the description of the implementation choices, it is fundamental to put in evidence some assumptions, which influenced the development of this preliminary solution. In order to develop a baseline model for our image classification problem, we make the assumption that there is only one human operator present in the monitored area, without any other dynamic obstacles. Moreover, to begin with a simple classification problem, the number of operation classes is limited to two. Note that we refer to *classes* of operations since this enables the possibility of fine-tuning pre-trained models using new operations, which may be variants of the main class. In fact, since we defined an operation as a group of tasks

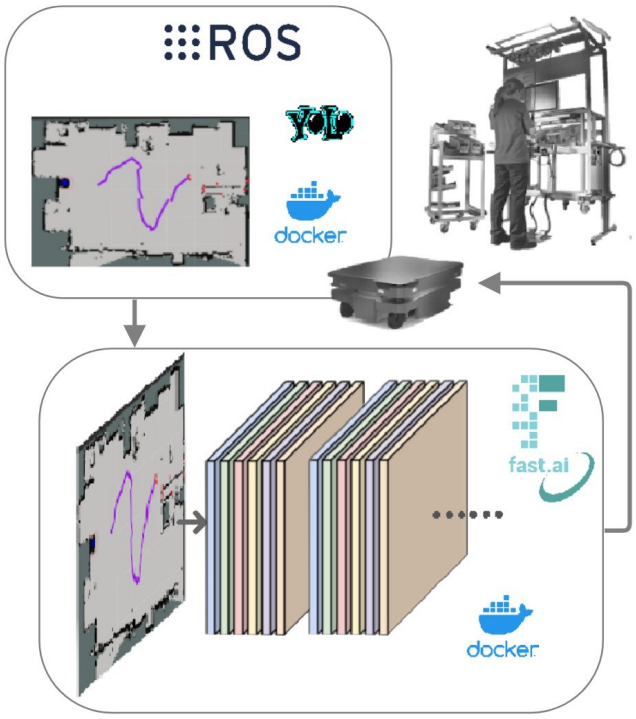


Fig. 3.5 The proposed solution aims at recognizing – through image classification – the operation executed by the human operator, by tracking the human position on a 2D map. This information will then be passed as input to the *robotic system control* global function.

performed at specific workstations (providing the needed machinery/cobot to bring on the necessary task), it is reasonable to assume that there may be variations in the tasks performed within a single operation, leading to slightly different variants of the main operation classes. Furthermore, the number of operations in a shop-floor is indeed usually limited to the available equipment and setup for the manufacturing of a certain product.

Implementation

This subsection illustrates the solution development and implementation details.

Positions collection In the Sen3Bot stack, the positions associated with the detected human operator are published as virtual obstacles in the navigation local costmap of the AMR: this allowed to enable safe overcoming of the human obstacle. Within this work context, the published ROS topic provides a source of position messages to be plotted graphically on the *rviz* visualization tool.

Path plotting The messages were subjected to filtering, since all points falling within the detected human obstacle bounding box are published as virtual obstacles: laser data may correspond to points behind the human obstacle surroundings

(see Figure 3.6). Those points have been filtered out and, among the points covering the bounding box width, only the nearest one has been maintained at each sampling instant.

Each human 2D position is then collected by a ROS node that pushes it in a type `Path` ROS message – a standard message type in the navigation stack – which is then published on an ad-hoc ROS topic.

As can be observed in Figure 3.6, the resulting path is not very smooth. Nevertheless, since it tracks a human motion, this can be considered an expected output, due to the non-smooth human motion and detection noise. Also, the resulting path is inevitably dependent on the detection and messages publication (ROS node spin) frequencies.

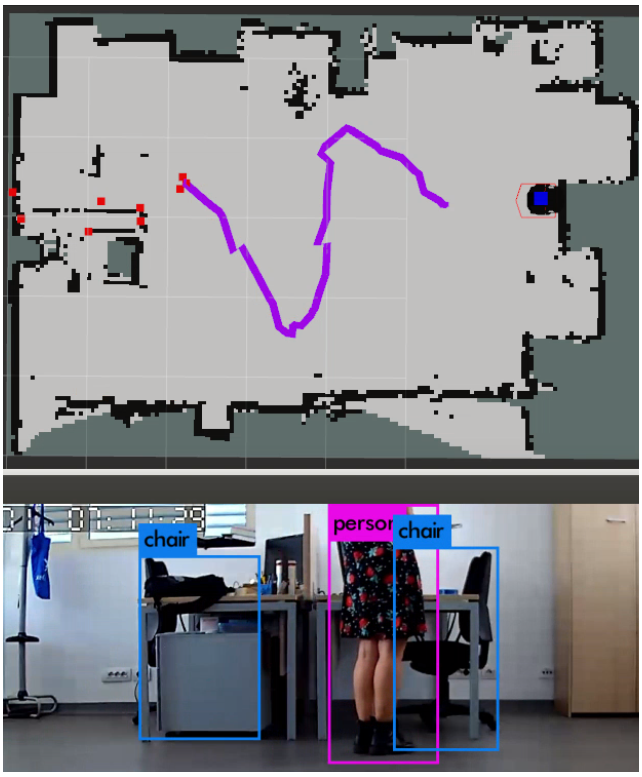
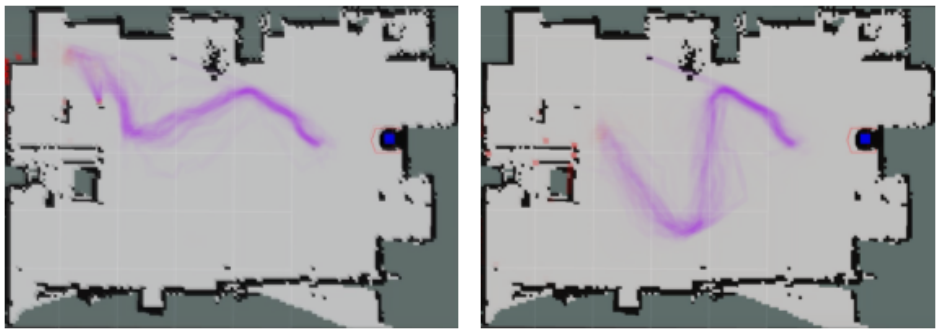


Fig. 3.6 After data filtering, the detected human path is published, and its content plotted on the 2D map by passing the type `Path` topic to an *rviz* Display.

Data collection Although simulation has been considered, a collection of real data has been preferred. This is because training a learner on purely synthetic data will unavoidably affect its capabilities of performing classification on real data samples, to the extent that it might not be able to recognize any operation at all. Nevertheless, real data collection combined with data augmentation can improve overfitting issues. With the purpose of speeding up the data collection, the operation executions have been video recorded and periodic screenshots of the *rviz* map visualization have been generated.

Dataset creation Disjoint subsets of such collected samples have been used for training, validation, and testing of the algorithm, respectively. To enhance the learning algorithm's ability to generalize, samples were collected from the motions of two distinct human operators. Data samples have been saved in the dataset `/train` and `/valid` folders. Note that only images representing completed operations are included in the training and validation sets. The reason for including only completed operations images for training and validation is to allow the architecture to adjust its parameters using representative samples of each operation class. Instead, to test the model capabilities to recognize an operation from the very beginning of its execution, testing is performed on samples of ongoing operation executions: this allows to observe the model capability to improve its confidence as the operation/path goes towards completion. Figure 3.7 shows the mean values for each training set for each operation, so as to give a compact representation of the collected dataset content. The mean of all the image tensors corresponding to a certain class of operations is obtained by taking the mean along dimension 0 of the stacked rank-4 tensor (RGB images).



(a) Class `A` operations mean value. (b) Class `B` operations mean value.

Fig. 3.7 Mean values images for each of the considered classes of operations, computed among the samples in each corresponding collected dataset.

Architecture Since the operation recognition problem (interpreted as a plotted path recognition) is now an image classification problem, the well-known "go to" NN architectures for this kind of learning problems are Convolutional Neural Network (CNN) models. Specifically, a Deep ResNet (Residual Network) architecture has been used. ResNets address the degradation of training accuracy (vanishing gradient) problem, i.e., the degradation of training accuracy, associated with network depth [305]. A `ResNet18` has been selected as learning architecture, and for the optimization step, we choose cross-entropy as our loss function, as it is the most common loss function used for binary classification problems. This function will be minimized by the stochastic gradient descent procedure during weight stepping. The learning rate has been set to 0.002, as suggested by `lr_find()`, a `fastai` function that plots the loss against learning rate values and outputs a suggested value, corresponding to the point where the gradient is the steepest.

Data augmentation A total of 300 image samples per class of operations have been collected. Despite being aware of the problem of overfitting due to data scarcity, the choice have been dictated by the purpose of testing the preliminary solution recognition capabilities keeping the number of labelled images low, for the sake of achieving a low complexity setup. Nevertheless, to improve generalization capabilities and avoid overfitting, data augmentation methods have been employed. As a pre-processing step, all samples within the dataset have been resized to reduce their dimension, as – in our case – size reduction does not seem to affect the model performance. Then, a set of transformations have been selected, namely small rotations and warping, and lighting editing. This set of transformations are defined along with their relative application probabilities, i.e., the probability with which the transformation will be applied to random batch elements during training. The batch size have been set to 64 and the training algorithm will take care of shuffling in a random way across the training data set when choosing candidates for each mini-batch. Moreover as a callback for every tweak of the training loop, we chose to apply the MixUp method [306]. MixUp generates new data during the learning procedure through convex combination of random pairs of images and associated labels. A random sample can be seen in Figure 3.8.

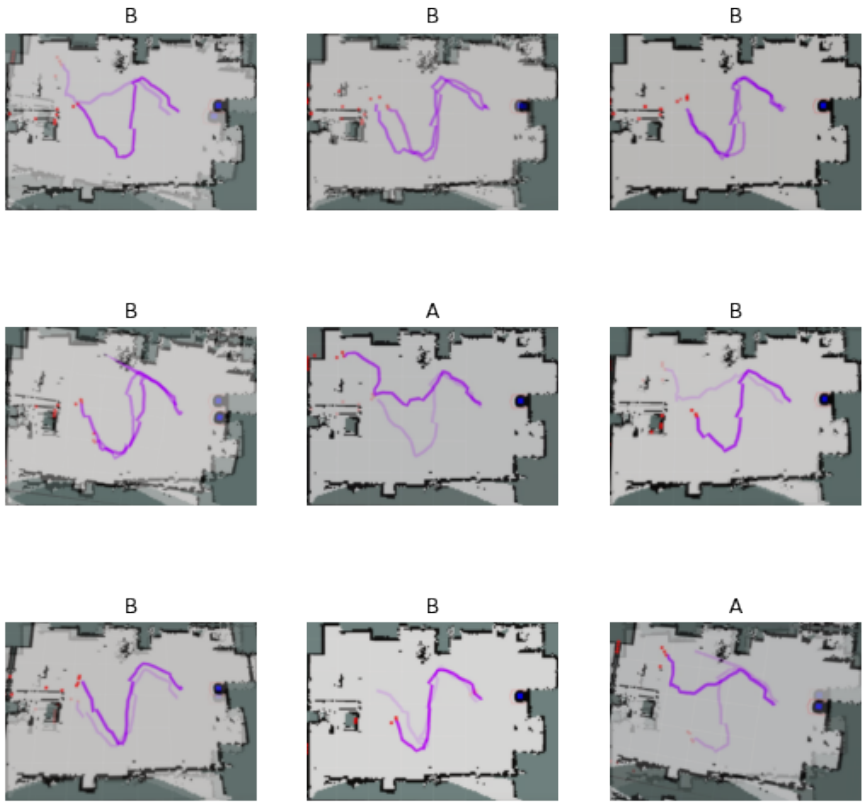


Fig. 3.8 Sample of batch elements generated by MixUp algorithm. As can be seen, shuffled samples are also affected by the randomly applied transformations for data augmentation.

Model After setting the described hyperparameters, a one-cycle training policy have been performed [307], which is a commonly used method for training `fastai` models from scratch, i.e., without transfer learning. As expected, most times the accuracy saturated to 1 during the first couple of epochs, suggesting overfitting issues. However, the main aim of the developed work is to test the obtained model on images representing the sequence of sub-paths corresponding to an operation. Then, the number of epochs for training has been limited to 1. An accuracy of about 0.93 has been achieved, with a training time of 4 s, running on a PC equipped with a 4GB GDDR6 NVIDIA GeForce GTX 1650 GPU. The trained model has been saved as a baseline model for the considered problem. Figure 3.9 shows a subset of the top losses peaked during training.

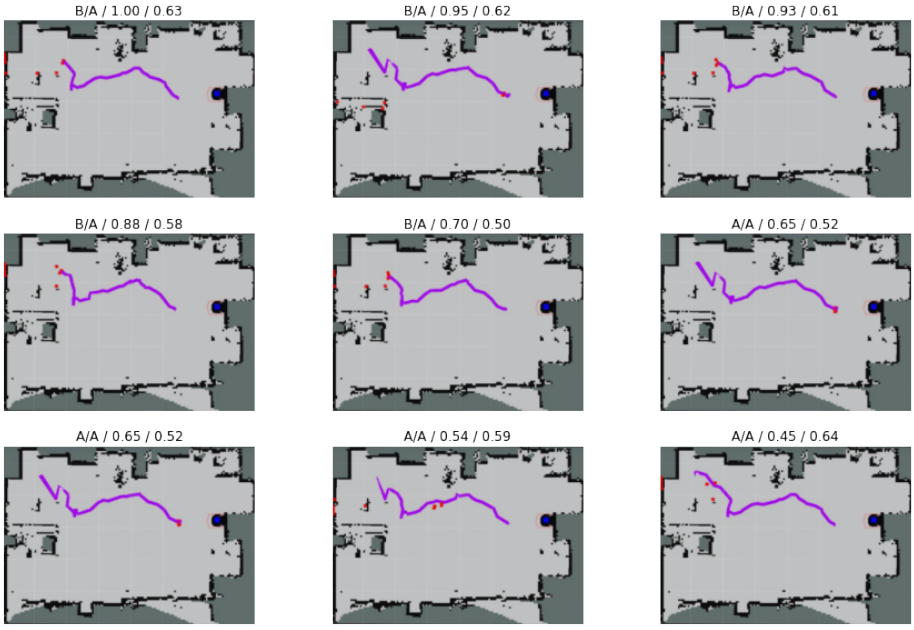


Fig. 3.9 Subset of samples that generated top losses. The title of each image shows: Predicted class / Actual class / Loss / Probability of actual class.

Testing

It is worth recalling that, in order to demonstrate the feasibility of the proposed solution for operation recognition problems to improve collaborative applications, the obtained baseline model should have the ability to differentiate between various operation classes. Additionally, it should ideally improve its confidence in guessing the class of an operation as it is provided with a sequence of images that represent the progression of the execution of the operation. In fact, in the proposed data-driven framework, according to the currently recognized operation, a specific reference trajectory will be fed to the mobile robot control system. In particular, a reference

trajectory should be generated, resulting from the weighted combination of candidate reference trajectories, were the weights are proportional to the associated operation class probabilities. The baseline model has been first feed with progressive screenshots from a class `A` operation execution. Then, the same has been performed for a class `B` operation.

Figure 3.10 reports the obtained testing results for class `A` operations testing samples. As can be seen, for both sequences the model initially struggles to correctly

```
(PILImage mode=RGB size=282x200,
 [('A', TensorBase(0), TensorBase([0.7406, 0.2594])),
  ('B', TensorBase(1), TensorBase([0.1507, 0.8493])),
  ('B', TensorBase(1), TensorBase([0.2263, 0.7737])),
  ('A', TensorBase(0), TensorBase([0.6505, 0.3495])),
  ('A', TensorBase(0), TensorBase([0.6896, 0.3104])),
  ('B', TensorBase(1), TensorBase([0.4127, 0.5873])),
  ('A', TensorBase(0), TensorBase([0.6742, 0.3258])),
  ('A', TensorBase(0), TensorBase([0.9912, 0.0088])),
  ('A', TensorBase(0), TensorBase([0.9985, 0.0015])),
  ('A', TensorBase(0), TensorBase([0.9965, 0.0035]))])
```

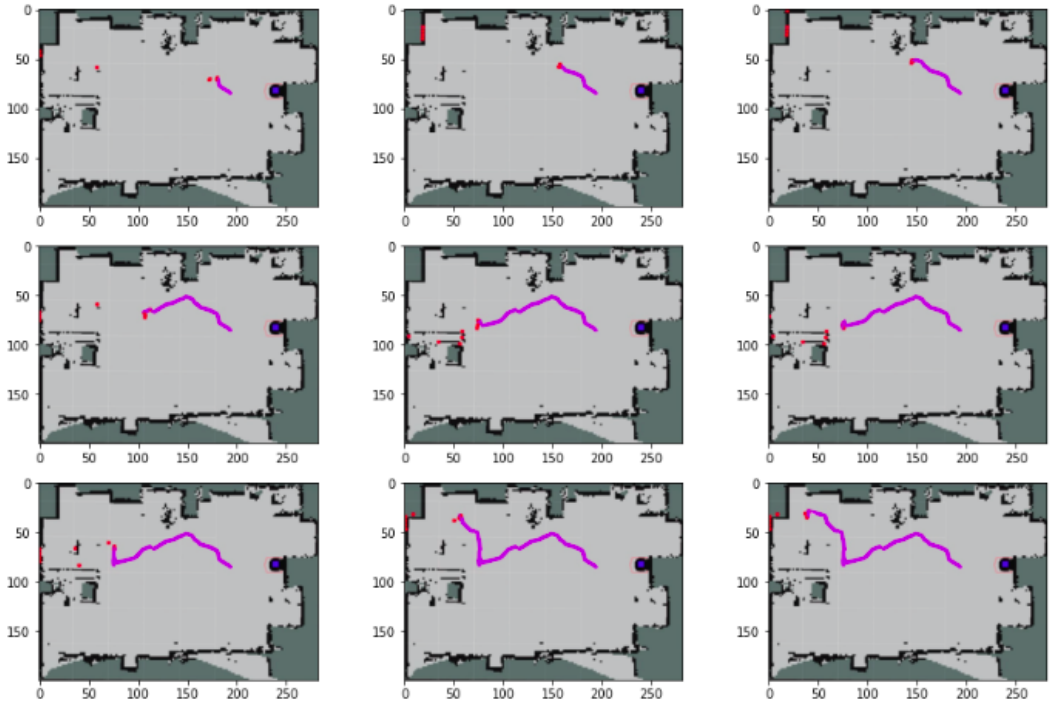


Fig. 3.10 Prediction labels and probabilities, along with the set of testing sample images, corresponding to an operation `A` execution.

recognise the ongoing operation. This is expected, since the first part of each representative path is identical for both classes of operations, as the first visited workstation is the same. Nevertheless, in the case of operation `A` recognition the classification results start with a couple of switching predictions, 74% `A` at step 1, passing through 59% `B` at step 6, and reaching 67% `A` at step 7, which lead to a prediction above 99% from step 8.

As seen in Figure 3.11 the recognition of operation B generated similar results, starting from high probabilities associated to the wrong class (95% probability for A at step 1) and progressively switching to increasing probabilities for class B, eventually reaching above 95%, starting from step 8.

```
[('A', TensorBase(0), TensorBase([0.9536, 0.0464])),
 ('A', TensorBase(0), TensorBase([0.5815, 0.4185])),
 ('A', TensorBase(0), TensorBase([0.8506, 0.1494])),
 ('B', TensorBase(1), TensorBase([0.4887, 0.5113])),
 ('A', TensorBase(0), TensorBase([0.5541, 0.4459])),
 ('A', TensorBase(0), TensorBase([0.5660, 0.4340])),
 ('B', TensorBase(1), TensorBase([0.3165, 0.6835])),
 ('B', TensorBase(1), TensorBase([0.0485, 0.9515])),
 ('B', TensorBase(1), TensorBase([0.0019, 0.9981])),
 ('B', TensorBase(1), TensorBase([0.0207, 0.9793]))]
```

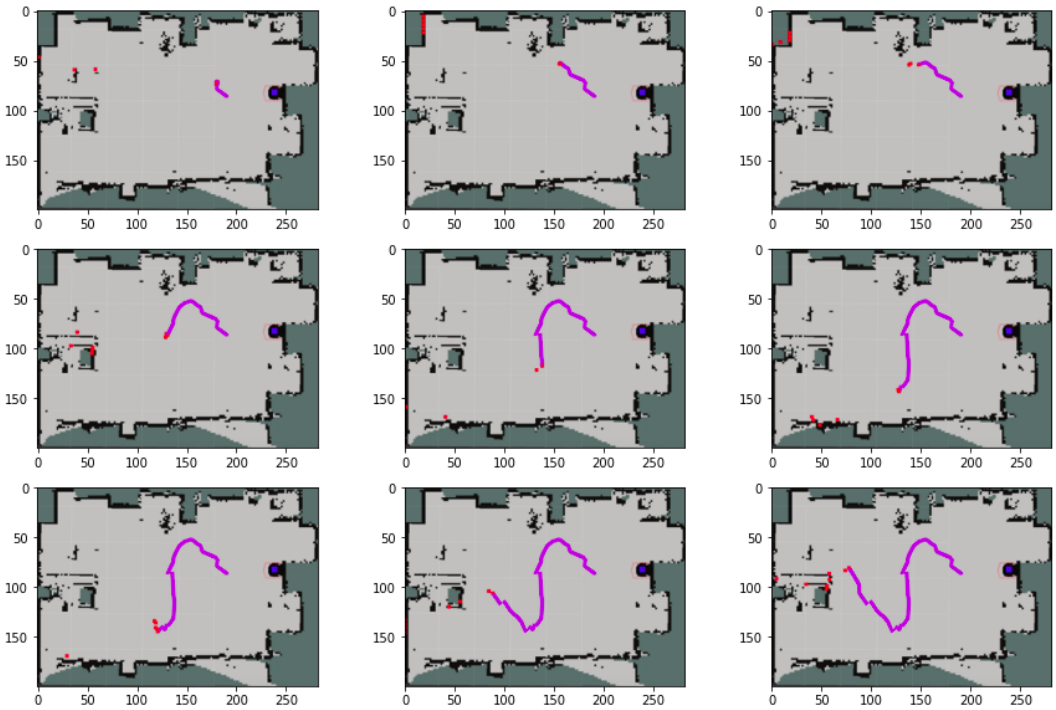


Fig. 3.11 Prediction labels and probabilities, along with the input set of testing sample images, corresponding to an operation B execution.

The reported contribution proposes a 2D human motion-based solution to perform operation recognition in the context of human-robot collaborative applications. To mainly demonstrate the solution feasibility with the tackled problem, a small dataset have been prepared for training purpose. The obtained results showed signs of overfitting issues, as expected due to data scarcity for training, but performed well in generalization capabilities when tested on images containing sub-path of the complete operation to be recognised. The results are nonetheless promising, and the possibility to tune the context assumptions, the hyperparameters and the datasets dimension, leaves room for significant improvements.

The next section introduces a proof of concept solution that could implement the local functions of the HITL data-driven framework just described.

3.1.2 Manipulator-equipped Sen3CoBot: a proof of concept

In Section 3.1.1, the focus was put on the implementation of *global functions* of the proposed framework. The following work, presented in [14], proposes a general framework for mobile manipulators assisting human workers, in a context of mass customization. This could be seen as a possible implementation of *local functions* to improve collaboration at task level. In fact, according to the definitions given in Section 3.1 – ***Framework Outline***, each task is brought in collaboration with cobots at specific workstations. In this case, the cobot would be a Sen3CoBot equipped with a collaborative manipulator, hence a mobile manipulator Sen3CoBot, which could potentially comply with both *global functions* and *local functions* implementations, as it can move to collaborate according to the human operation and also provide assistance for workstations tasks.

In order to develop a robotic assistant for manufacturing collaborative tasks, it is crucial to incorporate functionalities for object/tool recognition and grasping. In [308], a mobile manipulator able to autonomously navigate while detecting humans and objects to automate small and medium-sized enterprises production is proposed. A pointvoxel-region based convolutional neural network is used to detect objects in a robust way, allowing to get rid of 3D point cloud data uncertainty issues, while a 2D camera is employed to calibrate the cobot relative position to workstations. In [309], an event-based robotic grasping framework for known and unknown objects in a cluttered scene is presented. In particular, the model-based solution consists of 3D scene reconstruction, object clustering according to Euclidean distance, position-based visual servoing, and grasp planning of objects whose shape is known a-priori. The mentioned works are part of a substantial body of literature that exploits AI to implement object recognition and manipulation.

The problem of human action recognition have been investigated intensively as the human role in AI adoption is getting more and more relevant. The use of monitoring vision-based 2D and 3D data for action recognition has gained popularity due to the advancement and accessibility of vision oriented AI methods [310], but an interesting new direction exploits first person or egocentric videos to train deep learning models [311]. Egocentric videos are recorded from the point of view of the wearer, allowing to change perspective to what the operator sees during the execution of the task. This kind of approach poses interesting and challenging directions for computer vision action recognition. Also, approaches based on other types of data are available. An example is presented in [312], where the focus is on assembly operations recognition, achieved by exploiting two convolutional neural network models with identical architectures, each one independently trained on inertial measurements of the right and left operator's hands.

Depending on the context and the specific operation being executed, the collaborators involved may have varying levels of decision-making authority. For example, the work presented in [313] uses a mobile manipulator as a flexible solution for tending operations on computer-numerical-controlled and additive manufacturing

machines. A semi-automated execution mode is preferred, due to the cost of the required equipment. In this mode, the robot automatically computes low-level decisions and grasping planning, while the human operator supervises high-level decisions and pre-grasping poses. This way, the operator can perform risk-informed decisions to accept, refine, or decline system-generated plans.

It is worth mentioning that recent research on human-robot collaborative tasks often involves approaches that involve AI-enabled mobile manipulators, since they embody the main capabilities demanded to a robotic assistant: flexibility, autonomous mobility and improved dexterity. Even though some solutions opt for supervised or semi-supervised collaboration, decision making data provided by the human operator during this kind of applications could be collected with the aim of providing a dataset to the AI-enabled cobot. In such a way, by training on human-generated past collected information, a cobot would be able to perform high level decisions on its own. In fact, the cobot can learn from monitoring data, which can be used to either emulate the human operator's motion or acquire it for future use; this is done with the purpose of action recognition and prediction.

In the proposed framework, the mobile manipulator observes, analyses, learns, and attempts to help the human operator working in an assembly line, by understanding the intentions of the human operator with/without verbal communication.

Framework proposal

To the best of our knowledge, current solutions separately solve human action prediction, object recognition, object affordance manipulation, safe motion control, however, there is still a gap for what concerns the safe human-robot interaction for assistive applications that involve object affordance. The role of the robotic system in this case is to give assistance to the human, which can enhance the performance of the overall production time. In particular, the proposed framework will be focused on mobile manipulators as they offer increased versatility compared to fixed base manipulators. This flexibility can be useful for tasks such as material or tool handling, where items may need to be delivered to or from the human operator. In order to develop this framework, the robotic system should have at least the following features:

(I) An AI-based perception system for recognition and learning of human actions, tools/instruments, context, working spaces, whose data come from the sensors mounted on the robot and the ones used to monitor the workstation. The idea is that the robot should exploit monitoring data captured while the operator is performing some task, e.g., assembly or inspection tasks. Then, based on the context, a sequence of tasks to be performed can be predicted. For instance, in an assembly task, the human worker may use a type of screwdriver and then use another kind of screwdriver or wrench. Once the human actions are classified, the robot should be able to predict the next tool the human would like to use and then take the tool that the human is no longer using.

(II) A control system involving three blocks: (i) a safe trajectory generator, whose output varies according to the tool that the robot is holding, (ii) a safe trajectory tracking system, and (iii) a safe trajectory replanner/corrector for those cases in which the robot needs to perform a trajectory correction or replanning its trajectory while avoiding in a safe manner the human operator, in order to avoid unintentional harm to him/her.

Framework feasibility analysis

This section examines the currently available solutions that would help develop the key functional blocks of the envisioned framework. In this preliminary stage of the framework development, some assumptions are made for the sake of simplicity and to show correctly its workflow. It is assumed that the tasks carried out by the human operator involve a limited number of tools and consist of a sequence of brief actions.

Human action recognition. In an assembly task in which the human may open a device/product, inspect/modify it, and then seal it up, it is likely that the human needs different tools in order to complete the task. The sequence in which the human will use the next tool can be predicted by the robot. Hidden Markov Model (HMM) algorithms are popular for human motion prediction, since the hidden state transitions can be used when there are uncertainties due to weak motion recognition. In [298], HMM is used to predict a sequence of human actions by generating motion and observation probability matrices. Moreover, the principle of object affordance is included in the HMM model proposed by [299] in order to give some context to the human actions.

Tool identification. The tools can be classified a-priori by tool type, based on their utility and also on how they are usually grasped/held/used by the human operator. Identifying the tool may allow the robot to grasp it correctly and deliver it in a safe manner. In fact, affordance detection [314] allows to localize, classify and label the affordance of the detected objects, such that the graspable and non-graspable parts of a tool can be identified. A possible solution is provided in [315], where an object-agnostic affordance recognition neural network is presented. In particular the system analyses and predicts affordances of object parts in an image.

Workspace organization. In a complex assembly task, a worker may require multiple tools, but it's preferable to have a clear workspace with only frequently used tools within reach. On the other hand, dangerous, heavy, or rarely used tools should be stored in a dedicated box or space that can be accessed when needed. In order have a working space that can be easily classified and interpreted by the robotic AI system, for convenience, the workspace can be divided in the following areas: (i) a *human workspace*, where the human can perform the tasks normally, (ii) a *tool area*, which in turn can be divided into two sub-areas, namely, a *release area* – where the predicted next tool is released by the robot), and a *collect area* – where unused tools are left and collected by the cobot to be put back in place in the tool deposit.

Mobile manipulator control system. Especially in the context of collaborative tasks, safety is a relevant issue. As such, hazardous situations must be avoided by taking into account not only what is being executed but also how. For example, tool-path smoothing methods as the one proposed in [316], demonstrated to ensure better tracking performance in the motion control process. This could be exploited to improve tracking of generated paths according to recognized human actions, to ensure a reliable behaviour. Moreover, in order to ensure a safe interaction between the robot and the human, a safe trajectory generator should be used for each tool type, e.g., screwdrivers, hammers, pliers, as these tools can cause harm if they are not handled correctly, in particular while being held by a moving robot. Then, based on the object affordance information, the cobot should be able to act accordingly. To do so, control and manipulation approaches can be borrowed from non-collaborative frameworks. For instance, the work in [317] provides a high-accuracy method for estimating a workpiece pose for workpiece exchange, which consists in compensating the potentially poor positioning of the mobile base with a marker-based alignment of the manipulator. In a preliminary stage of the proposed framework, the trajectory tracking would consider only the kinematic model of the mobile manipulator, while the safety issue will be dealt with using the onboard sensors, so as to avoid any contact or unintended harm to the human. Future developments should consider the dynamic model of the robot in case of force/torque interactions; for instance, an impedance control architecture for collaborative material handling could be compliant [318].

The next section analyses the evolution of industrial fixed-based manipulators interfaces and how this allowed to achieve today's collaborative manipulators and how, leveraging recent learning research directions, robots can be taught to behave directly by humans.

3.2 Towards industrial manipulators 5.0

As introduced in Chapter 1, robots have been populating and starring in the industrial production lines since the third industrial revolution, when industrial manipulators –also known as robotics robotics arms– helped automatizing manufacturing processes. In fact, robotic arms could perform repetitive tasks at high speed and high precision, without getting tired or making errors – considered typical human weaknesses. As such, for a long time industrial manipulators seemed to represent the future human substitutes. However, as described before, the research direction suggests otherwise, as humans will be irreplaceable and robots will be human teammates. Early industrial manipulators were the Unimate robots, developed by George Devol and Joseph Engelberger in the late 1950s [319], and first installed in a General Motors plant in 1961. Bulky and stationary, these primordial versions of the industrial manipulator were utilized to bring on repetitive and hazardous tasks, e.g., painting and welding and controlled by computers programmed using a punched paper tape, which was fed into the robot's controller. Thinking about this being a bit more than half a

century ago, is really eye-opening on the fast pace technology is going forward. Indeed, thanks to sensors development and sophisticated methods led industrial manipulators to evolve towards today's collaborative manipulators. A key aspect of collaboration between humans and robots is the way the bidirectional communication and interaction occurs.

Human-Robot interfaces and interactions modes changed hand in hand with hardware and software improvements and new requirements, starting from offline control to interactive robot learning. Interfaces play a crucial role in human-robot interaction, as they allow communication between humans and robots, facilitating information sharing and feedback, which are essential aspects for successful collaboration. In fact, in any relationship involving collaboration, the teammates have a common goal to reach –e.g., a task to perform– and information exchange along with coordination of actions are paramount for achieve such aim. Overall, interfaces evolved from physical devices to more abstract software layers. Yet, physical interfaces are still necessary especially to ensure safety measures, as the human operator can hard-stop the robot through physical interfaces if an action that could potentially damage equipment or harm the operator or other humans in the workspace is being performed. Also, safety can be achieved by providing visual or sound cues to anticipate the robot motion, or to indicate when it is safe to enter the workspace.

Human-robot interfaces can be mono-directional or bi-directional, based on whether either the human or the robot use the interface as an input source or an output source, or both. For instance, interfaces can allow the operator to provide feedback on the robot performance, to adjust its behavior or modify the task based on the changing requirements, or they can enable the robot to signal or display some kind of information to be interpreted by the human.

Considering manufacturing applications, the most common human-robot interfaces, aggregated by the involved human *senses*/robot *modalities*, are:

- *Touch/Haptics*:
  - → Physical devices, including buttons, switches, and joysticks allow the human operator to control the robot's movement and actions. A typical device to manually control industrial manipulators is the teach pendant.
  - → Haptic interfaces can provide feedback to the human operator through touch or force feedback. The provided information can be either related to the robot surrounding environment, e.g., some kind of feedback on the contact surface the robot is interacting with.
- *Vision/Visual signals*:
  - → Graphical user interfaces (GUIs) may provide a visual representation of the robot's current status, allowing the user to input commands/adjust parameters. Indeed, GUIs also fall within the tactile interfaces.
  - → Virtual reality interfaces let the human operator control the robot through a safe virtual environment, and can be used for complex tasks simulation.

  - → Augmented reality interfaces allow to render virtual visual representations over the real environment, allowing for a combined visual and touch experience. Indeed, AR interfaces also fall within the tactile interfaces.
  - → Visual/Light signalling of the robot is a possible way for it to communicate to the operator a change of status.
  - → Gestures interpretation can also be an intuitive way for the human to provide control commands or feedbacks.
- *Hearing/Speech/Sound signals*:
  - → Natural language interfaces allow communication from the user to the robot using spoken or written language.
  - → Sound signalling of the robot is a possible way for it to communicate to the operator a change of status.

To the best of the author's knowledge, at the time of publication of this manuscript, the human senses of *smell* and *taste* have still not been exploited for human-robot interfaces in manufacturing applications, leaving an interesting open research direction. Note that, overall, the choice of human-robot interface will depend on the application requirements and could involve a combination of different interface modalities to avoid information ambiguities. In general, friendly human-robot interfaces can be achieved by implementing interactions perceived as intuitive and natural by the human operator, and providing personalized interfaces with custom configurations and settings based on his/her preferences, work habits, and culturalization [320].

Another interesting classification consists in sorting by increasing degree of autonomy of the robot the HR interaction modes, from a fully supervised to an autonomously learning robot:

- *Teleoperation* The robot is controlled, hence fully supervised, by a human operator using a remote interface such as a joystick, keyboard, or haptic device.
- *Shared control* The human and the robot share the control of the latter. Tasks are performed autonomously by the robot, while the operator provides high-level guidance or corrections when necessary.
- *Task-level programming* It enables the robot to perform a specific task or set of tasks using high-level instructions. Task-level programming can be used for tasks that have a well-defined structure or sequence of steps.
- *Learning from demonstration* A human teacher performs a task which is learned by the robot by either watching it or experiencing it. The underlying structure of the task is typically inferred using ML.
- *Autonomous decision-making* In autonomous decision-making, the robot is capable of making decisions and taking actions without human intervention, thanks to a advanced environment perception and task interpretation.

The choice of HR interface and interaction mode, depends on the specific application. In particular, when the aim is human-robot collaboration, all the described interaction modes and interfaces can contribute to achieve such goal, depending on the level of autonomy required to the robot.

In the context of flexible manufacturing, the collaborative assembly task represent a manufacturing operation of high practical relevance. Having a heterogeneous team of workers working collaboratively and simultaneously on a common operation allows to achieve efficient labor division: the robot is typically allocated repetitive or power demanding tasks, while the human perform highly dexterous operations [321]. When considering teleoperation, it can be interpreted as a sort of human-manipulator collaboration as the capabilities of the latter are made available to the former in the case of hazardous environments or to perform operations that can weary a human operator in the long run. Taking into consideration an assembly task, the robot can be instructed through an appropriate interface to preassemble pieces of a custom product to be then finalized or inspected by the human operator.

The following section briefly reports a work that seeks to achieve a low cost human-robot interface for remotely controlled preassembly tasks. In particular, in the context of this thesis, the following work is re-framed to be applicable to a collaborative assembly task where the human supervises the pieces selection and then intervenes, for inspection or finalization, on the parts preassembled by the robot. This perspective provides an example application that would take advantage of learning from a human, as the latter is giving useful information about the assembly parts selection from which the robot could learn some kind of pattern.

3.2.1 Human-robot interface for remotely controlled assembly tasks

The work presented in [13], and summarized here, proposes a touchscreen interface for supervised assembly tasks, using an LCD screen and a hand-tracking sensor. Typical interfaces employed to teleoperate the robot are not so intuitive to use. Difficulties in using a robot's interface can prolong the learning and control processes, and may even lead to increased stress and mental effort for the operator. The aim of this work is to provide an intuitive remote controlled system offering a flexible way to perform an assembly task, where high-level decisions are made by the human, while the robot performs pick-and-place operations for the preassembly phase.

A typical device to manually control industrial manipulators is the Teach Pendant, whose interface is similar to a joystick. However, this kind of interface is not very intuitive, and it may require a lot of practice in some applications [322]. Approaches based on conventional human-to-human communication interfaces like speech or gestures can also be utilized to teleoperate robots. Industrial environments are usually very noisy, so it is difficult to operate the robot using vocal commands, and gestures are preferred over speech [101]. A common gesture used in human-human

interaction is to point a part, a location or indicating a position. This is as simple as it is useful to codify real-world assembly tasks [323].

There exist various other teleoperation interfaces available that can be adopted to achieve a fast response, thereby enhancing the usability. In [324], it was seen that people perceive more accurately and prefer a system they are more comfortable with, such as a touchscreen. Nowadays, most of the human-robot interfaces include a touch screen, becoming easy and intuitive to use [325]. It is important to consider that in an industrial setting, operators may wear gloves for safety purposes. This can make the use of Teach Pendants with touch screen interfaces less practical, as the screens are typically small and difficult to operate with gloves on. A possible solution to such problem could be that of enlarging the touch screen device. However, this approach may not be suitable if the operator is wearing gloves for safety reasons, and also, larger touch screen devices can be costly.

In [326], an infrared-matrix sensory system is used to emulate a touch screen interface that recognizes the user's actions and therefore, to control the manipulator remotely. Furthermore, the interaction quality between a human and a teleoperated manipulator can be enhanced by incorporating confirmation feedback into their communication, as explained in [327]. Their study has shown that incorporating confirmation feedback in human-robot communication can lead to improved interaction quality and higher task selection accuracy compared to scenarios without any feedback.

The scenario that suits the proposed solution could either consider (i) a production environment that does not allow the presence of human operators due to safety reasons but, however, requires the robotic system to be controlled onsite, or (ii) a collaborative assembly task where the preassembly phase is entrusted to the manipulator but needs to be supervised by a human operator as each next piece must be selected according to the specific custom product. In fact, a considerable literature has grown up around the theme of hybrid make-to-stock/make-to-order manufacturing strategies [328]; make-to-order manufacturing involves high level decisions aiming at the customization of final products. When some functional features are available on the robot's system, e.g., low level planning, sensory system to monitor the workspace, status update and the possibility to receive tasks inputs from the operator, it is possible to command the robot with high level tasks [329]. An interface that allows to control the robot with high level tasks reduces the human operator workload, as demonstrated in [330, 331], since the robot is capable of calculating the most efficient trajectories for performing the required actions without requiring input on the motion of individual joints.

The proposed system emulates a touch screen behaviour exploiting a hand-tracking sensor along with a liquid-crystal display (LCD). The latter is also employed to show a visual feedback to the operator. A monocular camera streams the workspace where objects to be identified and grasped lie.

Concept

The PoinTap interface exploits a simple LCD screen upgraded to an emulated touchscreen, externally enabled using a hand-tracking device, inspired by the work presented in [332]. Then, the emulated touchscreen is enriched with the possibility of pointing to the screen while receiving a live visual feedback, so as to improve the interaction experience of the human operator. In fact, the screen shows the robot workspace scene, provided by a top-view camera, based on which the human operator interacts through a "point at part" gesture, which is interpreted and output it as a feedback mark on the screen. The system verifies that the pointed/tapped area on the streamed image is actually corresponding to a recognized object and transforms its coordinates into coordinates interpretable by the robot. Such a region is then identified as an area of interest for a desired task by the robot.

The term PoinTap, introduced to denote the proposed solution, is just related to such a capability of translating the operator pointing/tapping gesture. The block diagram of the PoinTap interface is presented in Figure 3.12, illustrating the flow of data (arrows), and indicating the input information required for each output.

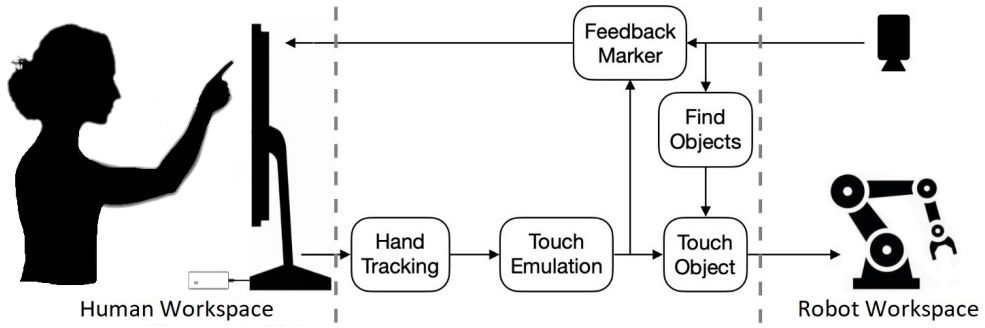


Fig. 3.12 Block diagram of the proposed system.

The main system blocks act as an interface between the human and robot workspaces. Note that several elements of the system can be customized (Table 3.1) to adapt the generic system to both the specific resources and application.

Implementation

As anticipated, the considered use case is an assembly task where the robot performs a set of pick and place operations to preassemble some parts. The human operator visualizes the set of pieces on the screen, and by using a "point at part" gesture, the user points at the piece of interest and, supported by the feedback mark, chooses corresponding area on the screen. This is interpreted by PoinTap, which passes the information to the robot as corresponding to the first object used in the preassembly task. This operation is performed one or more times, according to the number of pieces to be assembled.

Table 3.1 PoinTap Blocks and relative features.

Block	Features
Hand Tracking, Touch Emulation (TE), TE Setup	The information needed as input is the position of the tip of the index, independently of the source sensor.
Camera	A custom camera can be chosen, but note that its specifications affect the quality of the streamed image of the robot working space.
Find Objects	This block's implementation depends on the selected object recognition tool.
Touch Objects	This block should be used as it is. Some adaptations may be needed, depending on the software choices for the previous and next blocks.
Feedback Marker	This block strongly depends on the framework used to implement the PoinTap Interface.
Robot	The manipulator can be chosen according to the requirements specific to the application or depending on the available configuration.
LCD	The LCD screen specifications (length, width and height) are imply saved as parameters.

The interface exploits ROS as a middleware and the touch emulation uses the Layered Touch Panel [332] software, with a Leap Motion (LM) sensor [333] as a hand-tracking sensor. Both the LCD screen and the camera are both low cost devices. The objects recognition task was entrusted to the `find_object_2d` [334], a ROS package which integrates the Find-Object application [335] provided by OpenCV [336] with the image stream coming from the webcam. The ROS visualization tool, *rviz*, was exploited for the live feedback mark visualization, while the well-established integration with the Gazebo simulator [236] allowed for a smooth integration of the interface with a simulated version of the chosen robot manipulator. More details on hardware and software can be found in [13].

The implemented interface, even if imagined for the mentioned industrial use case, was tested on a research manipulator, given the possibility to validate it in a laboratory environment. To this end, a Niryo One manipulator [337] was used. It is worth mentioning that, to improve performances, it was decided to distribute the system among several machines: the simulation software was run on a PC, the camera and vision nodes were executed on a Raspberry Pi (RPI) board [146], and a further RPI board ran the nodes in charge of interfacing and controlling the robotic arm, adapted to our case. A summarizing schema of the blocks implementation of the PoinTap system is reported in Figure 3.13.

As shown, the human workspace is composed of the LCD screen (serving both as an input and an output device) and the LM sensor, both placed on a flat surface.

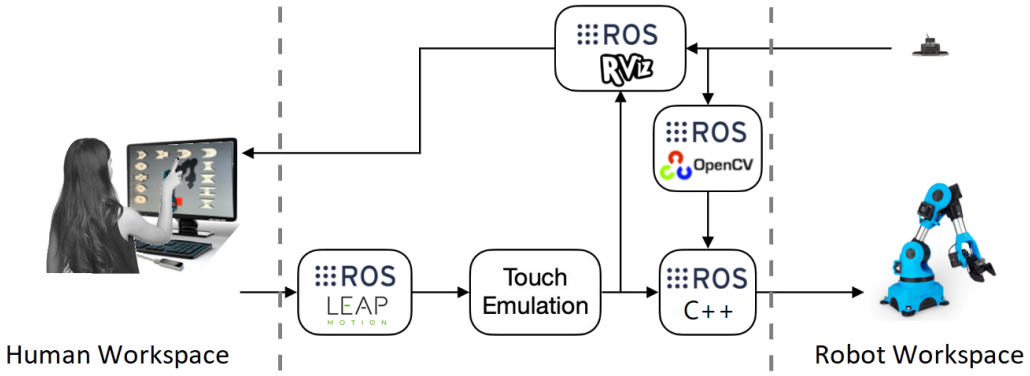
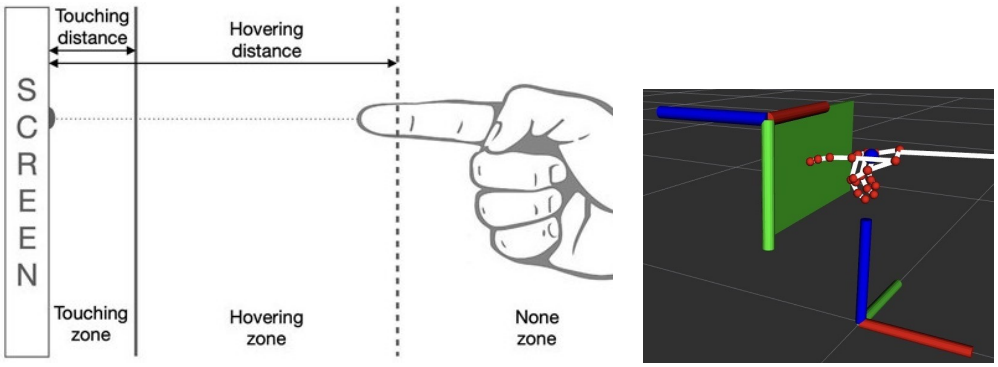


Fig. 3.13 Block diagram of the PoinTap system implementation.

The latter was considered to be perpendicular to the plane where the LCD screen lied, which is a common configuration for a working desk. In addition to its use as an output device for displaying images from the camera and feedback marker, the screen was also used as a reference for defining the touch and virtual panels. Note that the only requirement for the chosen screen is that it must lie inside the LM sensor field of view. For what concerns the robot workspace, the robot was positioned at the center of the world frame while the camera was placed above it, in such a way to visualise the whole workspace. Moreover, the image plane axes were set to match the plane where the robotic arm and pieces are placed, so as to have a direct correspondence of coordinates. The most relevant features of each block are reported hereafter (the interested reader can find all the details in [13]):

Touch Emulation (TE) the development of this block was led by the identification of the following problems in gesture recognition: (i) recognizing a pointing finger gesture may require the operator to achieve a high level of precision, (ii) timing associated to the pointing gesture is relevant, and a tradeoff must be identified to allow recognition in an acceptable time, and (iii) the pointing line was not easy to define and could lead to ambiguous and unnatural interaction. To provide a solution independent of such issues, the TE system proposed in this work provides a 1:1 scale among the scene image and the LCD screen, which corresponds to the touching plane. Exploiting the concepts related to the Layered Touch Panel [332], the real time feedback was implemented by adding two virtual panels parallel to the LCD: one at a distance equal to the *touching distance* – to catch touching events, the other at an *hovering distance* – to capture an hovering event. (Figure 3.14). Notice that hovering and touching distances are received as input.

Feedback Marker The LCD is used to (i) show the real workspace and, together with the LM hand-tracking sensor, emulate a touchscreen, and (ii) display a feedback marker in correspondence of the finger tip. The Feedback Marker block is designed to receive two inputs: the image stream from the camera and the hovering position from the TE block. When an image is available, the



(a) Graphical representation of the two virtual panels. (b) Reference frames in *rviz*.

Fig. 3.14 The touch emulation implementation schema and setup reference frames.

block displays it. If the TE block sends a message with the hovering position, a virtual marker (black circle) is added to the image. To do so, the position of the tip of the index is scaled to the screen range and converted into pixels in order to have a direct correspondence.

Find Objects The Find Object block receives the image stream provided by the camera and a set of photos to be recognised in the workspace. During the setup phase, the user can use the Qt-based GUI provided by the `find_object_2d` package to load images of the objects to be recognized. Note that images of the *simulated* pieces where used for recognition, both in the simulated case and the experimental testing. As an object is recognised, a coloured polygon is drawn around it, a unique numerical ID is assigned to it. Moreover, as soon as an object is detected to be new in the scene, a message containing all the pieces positions is generated.

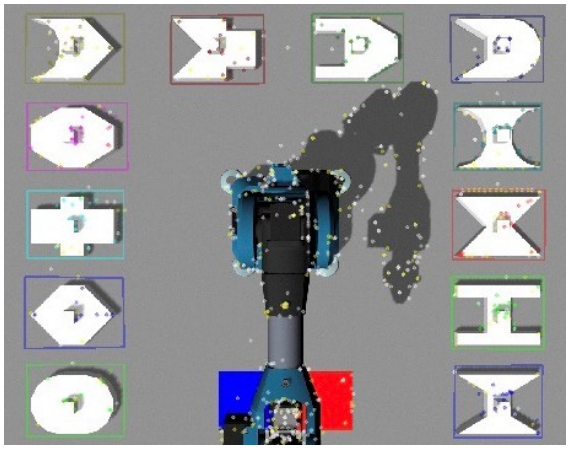
Touch Object The Touch Object (TO) block receives as input the data generated by the Find Objects and the Touch Emulation blocks. In particular, when the message containing the positions of the identified objects is published, the TO block stores the information in an internal variable. Then, as a new message containing the touch position is made available by the TE block, the TO block verifies if the position of the tip of the index lies on a piece and, if the latter is true, it generates a message containing the relative object ID. This information is then read by the robot and used for the task execution. The condition to be checked is considered to be satisfied if the index's tip position is inside the rectangle inscribed in the polygon.

Testing

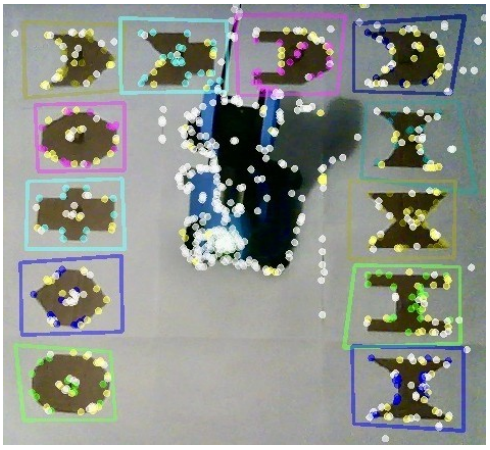
The main results achieved during the testing and validation phase of the PoinTap interface, in the considered use case, are hereafter presented, obtained through experiments conducted both in simulation and with an actual robot. In fact, being

the system implemented using the ROS framework, it was possible to exploit such implementation for both the simulated case and the experimental one. Twelve pieces were placed around the robot along a horseshoe shape. Specifically, the pieces used within the simulated environment (in Gazebo) were reproduced as accurately as possible using cardboard, to perform the experimental validation.

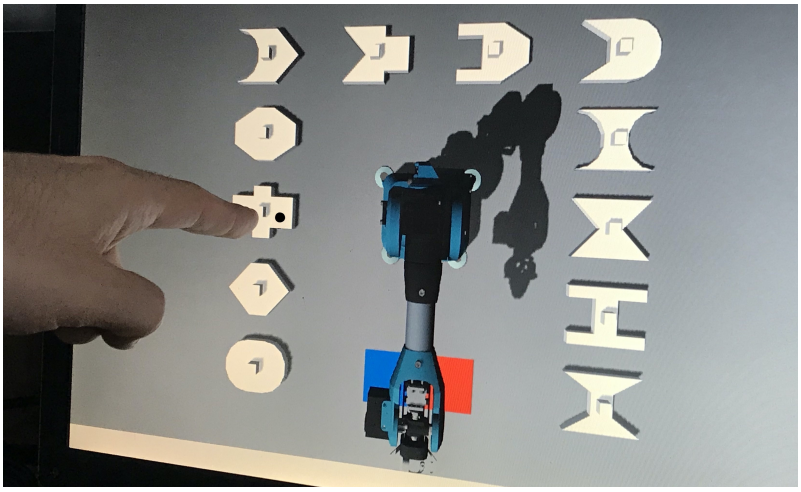
As shown in Figures 3.15a and 3.15b, displayed objects were enclosed in coloured polygons, indicating they were correctly recognized. As the human operator "point-tapped" the first object on the touch-emulated screen, the feedback marker appears (Figure 3.15c). Note that the simulated and real tests share the human workspace configuration, the devices within it, and the core PoinTap system including the robotic arm control and pick-and-place planning nodes. Indeed, simulation only involved the robot workspace and the devices within it (robotic arm and camera).



(a) Visualisation of recognised objects in simulation.



(b) Recognised objects in the real-world.



(c) Live feedback on the simulated workspace.

Fig. 3.15 (a),(b): object recognition visualization. (c): feedback marker feature.

Simulation: to perform the assembly task, two pieces were selected by touching the screen to be picked and placed by the simulated robot. As soon as the first object was point-tapped by the operator and identified by the system, the simulated Niryo One received the command and performed the first pick-and-place operation, then waiting for the second object to be touched. As the second piece was recognized and selected, the robot grasped and placed it next to the previous piece.

Experimental validation: two different pieces were selected by the operator, which were then handled by the real robot to carry out the assembly task (Figure 3.16). The experimental validation results can be seen at [338].

(a) The human operator point-taps a piece.

(b) The robot picks the selected piece.

Fig. 3.16 Interface and pick-and-place procedure in the real-world scenario.

The PoinTap system can enhance flexibility when a re-organization of the assembly task is required. Indeed, the sequence of objects to be assembled can be intuitively selected by point-tapping the LCD screen, without the need of trained operators. Also, our method could overcome the limit of using touch screen devices while wearing gloves. Given the possibility to train the object recognition system as desired, the pieces shape is not relevant. Finally, since the touch panel distance from the screen is tunable, PoinTap can be useful to avoid touching the screen so as to improve hygienic conditions. This is a topic of interest when considering health emergency situations such as the one we have and are witnessing. The limitations of the interface are dictated by the sensors intrinsic limitations, e.g., the field of view and accuracy, and by the available hardware combination. Moreover, due to the use of a monocular camera to monitor the scene, the robot may not be aware of the actual position of each piece. As a result, if some pieces are initially placed in positions that differ from the expected ones, the robot may not be able to grasp them.

The reported contribution, as presented here, seeks to provide a low cost interface for potentially collaborative assembly operations. On the other hand, the next section presents an attempt to improve the process of providing demonstrations to an intelligent collaborative manipulator, able to learn from human demonstrations interactively.

3.2.2 Exploiting simulation to improve interactive imitation learning

As anticipated, the emerging concept of Industry 5.0 envisages flexible production lines that prioritize human-centered manufacturing, with collaborative robots providing support to human workers. One relevant aspect is that to facilitate seamless collaboration between intelligent machines and their human counterparts, it is crucial to design solutions that can be easily used by non-expert users. Learning from demonstration has emerged as a promising approach to tackle this challenge. Nevertheless, there is a need to prioritize the development of safe solutions that can optimize the cost associated with the demonstrations collection, resulting in being less demanding for the operator. The user effort can be measured in terms of (i) the number of needed demonstrations to learn an acceptable policy, and (ii) the level of understanding about the underlying learning process, required to provide efficient and informative demonstrations. This contribution, introduced in a paper currently under review [339], presents a preliminary framework called EValueAction (EVA) that aims at overcoming the issues above.

EVA is designed to support humans in collecting interactive demonstrations exploiting simulation evaluations and predictions, ensuring safety and avoiding failures. The significance of this work is further strengthened by a case study that highlights the criticality of informative demonstrations in improving generalization.

To enable real-time monitoring of processes and safe decision-making, simulation and virtual representations of physical systems are increasingly taking hold as a way to gather insight into the behaviour of assets in real-world conditions. Among the projects pushing towards this direction, it is worth mentioning ASSISTANT [340], an H2020 European project that seeks to lay the foundations to achieve a manufacturing environment able to continuously learn from data, through AI-based digital twins, towards sustainable and safe learning by integrating human experience with machine intelligence.

In the context of smart manufacturing, the use of autonomous robots permits to streamline processes, along flexible production lines shared with human operators. Despite fully-automated plants have always been considered the main endpoint of industrial developments, recent trends in market demand towards the realization of custom products, has brought out the relevance of human complex and creative reasoning, which is yet to be achieved by machines. Indeed, whilst factories are gradually integrating Industry 4.0 paradigms into their processes, current and forecoming research is investigating a human-centric vision of production plants and warehouses, the so-called *Industry 5.0*. Built upon the previous industrial revolutions, Industry 5.0 envisions an industrial era in which skills, creativity, and problem-solving abilities of humans are combined with robots and smart processes, to create more sustainable and value-driven production processes [29].

When considering human-robot collaborative applications, a collaborative robot (*cobot*) is expected to behave as a proper teammate able to perceive collaboration

and interaction as another human would do. To do so and to overcome the cognitive mismatch between humans and machines, the use of AI-based algorithms can upgrade robots to interpret and adapt to human behavior. However, in established human-robot collaborative frameworks, the user is typically required to have the expertise to both understand the robot's behaviour in correlation to the underlying algorithm, and -when needed- to adjust it to change such behaviour. Such limitation can be improved by having a cobot capable of learning on its own from human demonstrations. *Learning from Demonstration* (LfD) techniques, also known as *imitation learning* (IL), involves training a machine learning model to perform a task by showing it examples of how the task should be executed. This of course implies that the human operator is involved in the process of data collection, being asked to provide informative demonstrations to be fed to a machine learning algorithm.

There are two major problems associated with this type of learning: (i) *dataset bias*, which is caused by the teacher's specific behavior or the lack of diversity in the demonstrated situations, and (ii) *overfitting*, linked to models that are too finely tuned or data scarcity in terms of the number of demonstrations provided. These problems can limit the ability of LfD algorithms to generalize (also known as *extrapolation*), resulting in difficulty adapting to new situations.

To address these challenges, human workers must be capable of providing (i) informative and diverse demonstrations that are free of bias, as well as (ii) a sufficient number of demonstrations. This would require additional mental and physical effort on the part of the human teacher: a relevant problem, in light of the renewed centrality of humans in manufacturing, where ethical and social considerations, including the human workload, are given increasing relevance. Indeed, the time spent for demonstrations, which could potentially turn out to be not informative demonstrations, can be foreseen to be included in the future *non-value adding activities*, i.e., activities hindering the transition towards lean manufacturing paradigms [341]. To enforce this new perspective, the authors of [342] suggest that the metrics for evaluating robot learning should prioritize time efficiency in order to accurately reflect the *true cost for humans*.

Hence, achieving generalization using a limited number of informative demonstrations remains a key research focus in the field of LfD. Seamless extrapolation capabilities would let a robot adapt in unfamiliar situations, based on experience and grounded on complex reasoning, as a human is capable to do. We briefly outline the existing body of research tackling the problem of generalization, by aggregating them based on how they deal with two aspects that can affect the generalization capabilities of a learnt model, i.e., the quantity and the quality of demonstrations.

The literature review highlights the crucial role of the quantity and quality of demonstrations in achieving successful generalization, irrespective of the inherent capability of the methods used. Therefore, two primary goals can be identified to minimize the "cost of generalization" from a human perspective: decreasing the number of demonstrations required and enhancing their quality.

The works below seek to enhance generalization while maintaining a minimal number of demonstrations. In [343], the authors propose a system that first learns,

from few demonstrations, the distribution of the input trajectories conditioned on the user set task parameters as a latent representation. If the system is queried outside the learnt space, it attempts to adapt the skill representation through RL, and then the simultaneous training is brought on over the demonstrated trajectories and the newly explored ones, allowing for skills extension while preserving learnt ones. In [344], goal oriented exploration is achieved through RL after a first phase of learning from demonstration. In particular, their method couples a policy gradient method with parameter perturbation in the framework of evolutionary algorithms, reaping the benefits of both allowing for efficient goal-oriented exploration, as the demonstrations guide the evolutionary process. The authors of [345] exploit adaptive dynamic movement primitives, where RL is taking care of learning the coupling terms and forcing terms' weights of the model formulation. In such a way, the trajectory profiles are effectively adapted to new tasks characterized by topologically similar trajectories, alleviating human repetitive demonstration burdens. Despite the simulation to reality mismatch, the learnt skill policies provided a good initialization and improved the sample efficiency of the learning process in reality.

The authors of [346] achieve adaptability by incorporating into the model a set of task variables, which describe the context under which demonstrations are performed. They propose a framework that exploits variations in the replications, i.e., repeated demonstrations for identical task variables, to retrieve an adaptive and robust policy. The approach proposed in [347] seeks to facilitate generalization by conditioning the learning process on different information sources conveying the task, e.g., pre-trained embeddings of natural language or videos of humans performing the task.

Other works exploit dataset augmentation to achieve policy improvements for learning task-parametrized skills without increasing the number of demonstrations. For instance, in [348], the learning dataset is augmented by adding noise on recorded paths. This allows to enhance demonstrations by scattering data-clouds and expand the demonstration area, so that excessive demonstration paths are not necessary. Also, the approach presented in [349] enables learning task-parameterized skills with few demonstrations by augmenting the original training dataset with generated synthetic data. Note that reducing the number of required human demonstrations also trim the chance of ambiguous demonstrations.

Learning from offline human data is often challenged by the quality of the demonstrations, which is considered one of the most critical factors [350]. In some cases, informative demonstrations can determine the ability or not to achieve good extrapolation [351]. In [352], authors assess the generalization ability of demonstrations by measuring the performance of the learning model in the task it has learned. They also point out that consistency in teaching does not directly reflect the demonstration quality, as the latter may be affected by consistent errors due to the teacher limited expertise; on the other hand, as the explicit definition and quantification of the quality of demonstrations are still open questions, in [353], the ability of the learning model to generalize is used as a measure to evaluate the quality of demonstrations. Several other works try to exploit as much as possible all available demonstrations. For instance, the framework proposed in [354] brings

forth another interesting perspective on informative demonstrations, as it learns a well-performing policy from demonstrations with varying optimality: by taking into account confidence-reweighted non optimal demonstrations, it is possible to exploit a larger number of data. The method proposed in [355] uses a generative model to compute a generalized trajectory and builds attractor landscapes to reproduce the motion over different start and end joint angles. The work in [356] takes advantage of available expert demonstrations to train a proximity function. Task progress, in terms of goal proximity, is used as a dense reward for the agent training, leading it to reach high-proximity states. In [357], complex goal-oriented tasks are solved by fully exploiting the demonstration dataset to build a compositional task latent representation space, from which novel subtasks can be generated. This allows for one-shot generalization to previously unseen tasks, as a single expert trajectory as reference to guide the new task is sufficient. The authors of [358] propose a continual learning from demonstration approach in which the robot learns individual motions sequentially; skills are learned from demonstrations of different tasks and are embedded into a unified model.

The framework introduced in [359], assesses various representations based on a specified similarity metric, using only one demonstration, and recommends the most similar replication of the demonstrated skill for novel conditions along with its corresponding generalization capability. Furthermore, the researchers in [360] intend to enhance the extrapolation abilities of LfD techniques by utilizing virtual demonstrations produced through the use of the invariants method. Such demonstrations offer improved consistency and quality compared to those generated by human teachers. Additionally, the approach introduced in [361] utilizes epistemic and aleatoric uncertainty data to identify feedbacks' ambiguities, which helps in selecting the best learning samples. Meanwhile, the algorithm proposed in [362] leverages topological persistence to detect ambiguity in a trained policy, prompting for user demonstrations only when necessary and avoiding the collection of demonstrations for already known scenarios. However, to achieve high-quality and sufficient quantity of demonstration data for effective generalization, a system could be developed to optimize the time cost for humans while ensuring generalization on novel situations. This can be achieved by guiding the user's interactive demonstrations.

The framework presented here, namely EValueAction (EVA), assists the user in collecting informative demonstrations to enhance policy generalization. The system also ensures safety by conducting simulations before executing actions in the real world, to prevent failures.

Preliminaries

To provide some context for the proposed EVA system, it is important to describe the case study that inspired the concept. This includes the assumptions made and the outcomes of the trials conducted. The problem contextualization is described as follows. In flexible manufacturing, collaborative assembly tasks are of high practical relevance. Typically, robots are assigned repetitive or physically demanding

tasks, while humans perform highly dexterous operations [321]. Adaptability to new situations is crucial in collaborative assembly tasks, where workspace configurations, positions, and shapes of assembly parts often change. According to the assembly task subdivision introduced in [363], we consider an *approaching phase* and an *assembling phase*. Such division is performed so as to prevent the potential underfitting caused by the variability of the demonstration data between the two assembly stages. The approaching phase of collaborative assembly tasks is highly affected by environmental constraints and part configurations, making extrapolation capability crucial. As a result, an IL algorithm with good generalization capabilities, learned from a few demonstrations, would be most beneficial for this stage, particularly in scenarios where a human is teaching a cobot how to perform the desired approaching phase. For instance, in a peg-in-hole collaborative assembly task, the approaching phase could be simplified to a pick-and-place or hovering task.

As extensively investigated, the issue of poor generalization capabilities in IL has been widely addressed in the research community. The following case study offers a clear illustration of how the way demonstrations are performed can significantly impact the generalization capabilities of IL methods.

A simplified learning scenario

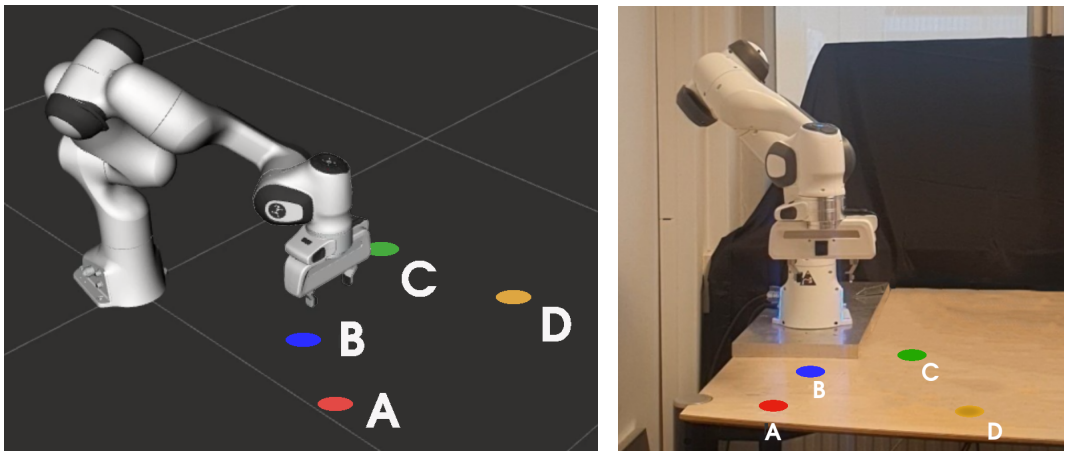
For the execution of the trials, we have taken advantage of a combination of algorithms presented in [364] and [365]. In particular, the first algorithm has been edited to achieve a version that incorporated some features of the second. The Interactive Learning of Stiffness and Attractors (ILoSA) framework allows to learn both the desired attractor position and the desired stiffness as a function of the robot position, by using a nonlinear feedback policy that is learned from kinesthetic demonstrations and teleoperated corrections. The use of Gaussian Processes for policy learning provides good generalization in the neighbourhoods of the demonstrations, while providing confidence level of the corresponding prediction. The authors exploit this information to understand if the robot state is currently in unvisited regions, and implement a stabilizing attractive field to lead the robot towards minimum variance regions. Since the latter algorithm recorded demonstrations within a global frame, generalization was limited around the visited area. Hence, we borrowed from [365] the use of the goal local reference frame so to be able to test over different final positions. Note that we exploited this local reference frames version of ILoSA as a kinesthetic motion imitation learning framework, i.e., without taking advantage of the interactive corrections feature, and didn't take advantage of the learned stiffness, as we wanted to keep a simplified motion learning scenario.

In summary, we exploited ILoSA to learn from human demonstrations the attractor distance for the robot impedance control, learned with respect to the final position reference frame. The main objective of the trials was to achieve as many new goals as possible by using only one informative demonstration.

Demonstrations setup

The trials and demonstrations were conducted using a 7 degree-of-freedom Franka-Emika Panda robot equipped with an impedance controller and a ROS communication network. The demonstration involved moving the robot from a reference home position to a final position of interest. In the approaching phase scenario, it was assumed that an assembly part was picked up from a location and brought to the reference home position at each assembly task iteration. The learned policy was then used to reach the final desired position before the assembly stage.

As expected, the employed IL algorithm was able to generalize in the neighbourhood of a demonstrated trajectory task. Therefore, to assess the generalization capabilities of the employed IL algorithm beyond the neighbourhood of the demonstrated trajectory task, four goals of interest were selected (Figure 3.17).



(a) Points of interest in *rviz*.

(b) Points in the real setup.

Fig. 3.17 Four goals of interest on the working plane have been chosen for the generalization check trials.

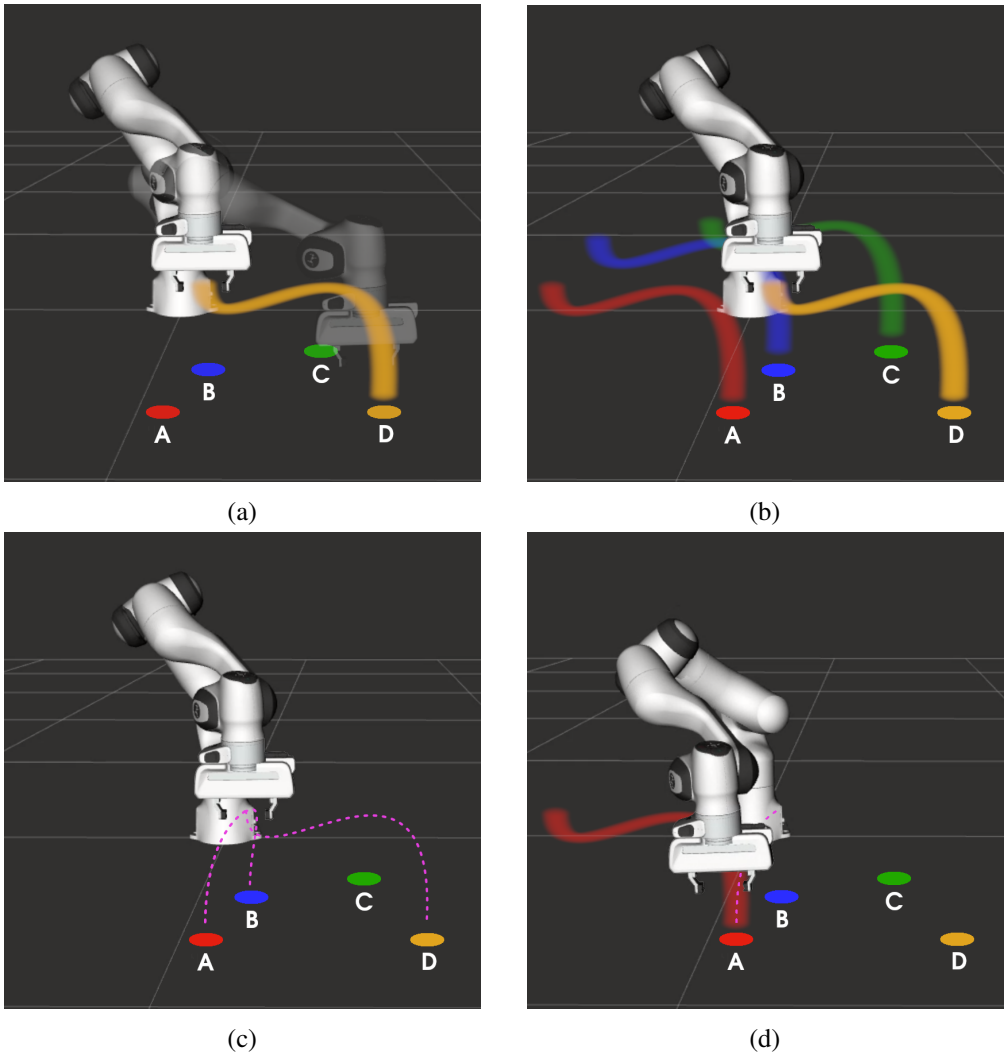
Automatic demonstration recording

The ILoSA Jupyter interactive Python code (accessible online [366]) has been adapted to allow the user to input a final goal point interactively. The input data is used to name saved data and trained GPs models files/structures, and generate a `.txt` file to record the tests and their outcomes. This `.txt` file can be later accessed, which facilitates automatic tracking of the trials. Additionally, after recording, the modified code enables the iterative testing of the learned policy over the set of interest points, thereby accelerating data collection and consultation.

Trials outcomes

Multiple trials were conducted, leading to the identification of three significant cases that are detailed below. It was assumed that demonstrations with final destination points `A` and `B` were the least generalizing, based on the results of the trials, thus were not considered as demonstration points.

Medium expertise demonstration Destination hovering point is one among `A`, `B`, `C`, and `D` (refer to Figure 3.18).



(a) (b)

(c) (d)

Fig. 3.18 Medium expertise demonstration on `D`.

A human teacher performed a demonstration with the position above point `D` as the final position to be reached (Figure 3.18a), and the height recorded during during the demonstration was kept as a hovering height for all desired

goals on the worktable. After the policy training, the motion was learned in the local reference frame leading the robot to areas with high confidence for each final goal, as shown in Figure 3.18b. When the policy was tested on all the points of interest, it generalized well on `A` and `B`, since the IL algorithm attracted the robot towards the high confidence areas thanks to its stabilization prior (Figure 3.18c). Then, when `A` was the input for policy testing, the cobot was able to reach it (Figure 3.18d).

Expert knowledge demonstration Destination hovering point is not among the points of interest (Figure 3.19).

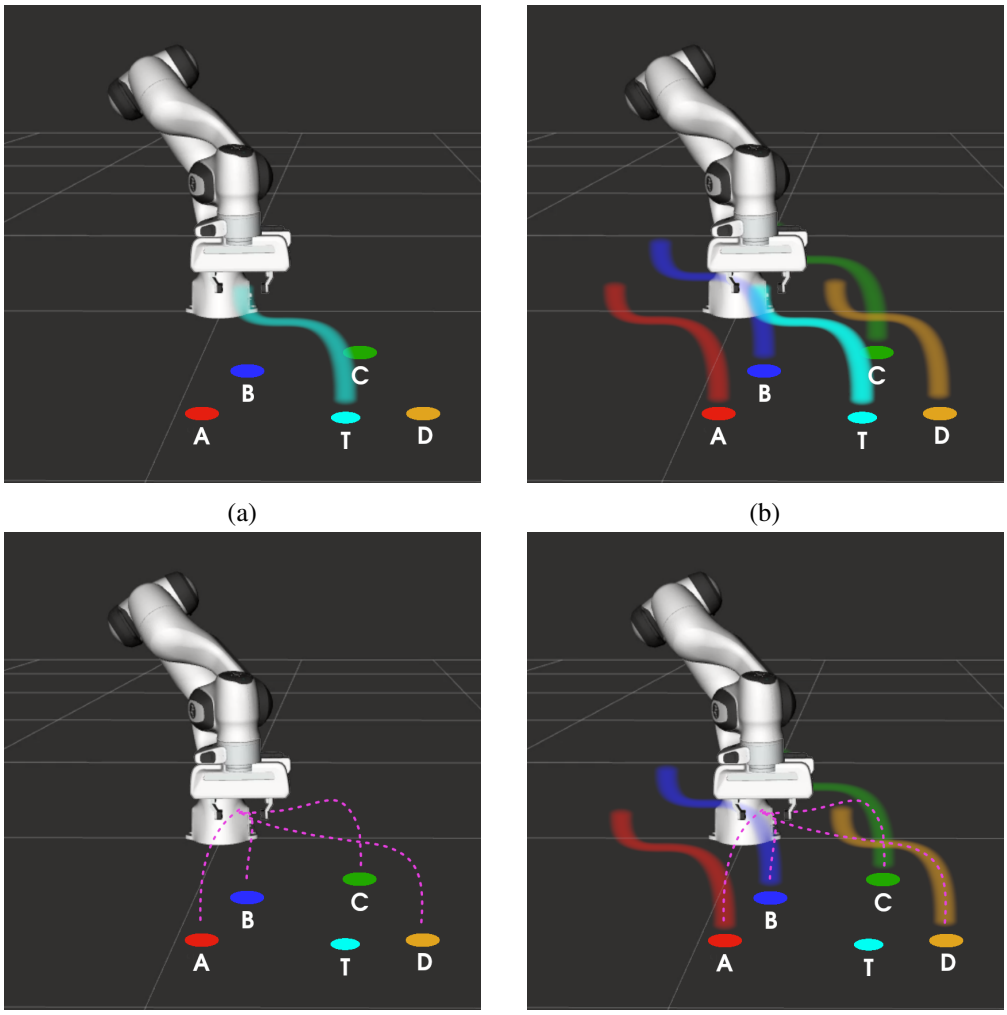


Fig. 3.19 Expert demonstration on `T`.

The motion to reach the position above an empirically chosen test point `T` was demonstrated by a skilled human teacher who had a good understanding of the IL algorithm used for training (Figure 3.19a). Such demonstration intuitively

would have allowed the trained policy to generalize on all points of interest. Indeed, after GPs training, the policy generalized on `A`, `B`, `C` and `D` (Figure 3.19b–3.19d). It should be noted that in this case, the generalization ability is impacted by the motion shape and stabilization fields. Depending on these factors, the robot may or may not be drawn to the locally learned motion.

Little knowledge demonstration We proceeded to have a human teacher, who was not informed about the IL algorithm used in the training process, demonstrate a motion towards point `D`. Since the user was not aware of the influence that the motion shape could have on the generalization capability, the resulting policy surprisingly failed to generalize to the other points of interest, learning only how to hover over `D`. It is worth noting that although the human teacher had robotics expertise, they had no information on the learning process.

Main takeaways

The trials showed that if the operator is expert and well aware about the learning process, a single demonstration can be enough to generalize over four points of interest and their surrounding areas (including the testing point and its surrounding area). However, if the operator is non-expert, it may happen that only one task can be learned from a single demonstration. Regardless of the LfD method used and the complexity of the learning task, these insights shed light on how robots can enhance the quality of demonstrations and decrease their quantity. By leveraging its knowledge of the policy, the machine could offer recommendations on which areas would benefit most from new demonstrations and even simulate demonstrations virtually, thereby reducing the number of demonstrations required from the human. This led to the idea of utilizing simulations for policy evaluation, which could also enhance the safety of collaborative tasks.

The proposed system

The proposed system, EVA, seeks to provide a framework to guide the human teacher during the interactive demonstrations to improve the generalization over new states of a learned policy. The main elements of the system would be: (i) the LfD method of choice, (ii) automatic recording of demonstrations to populate a dataset, (iii) a policy evaluation algorithm exploiting a digital twin, (iv) human-robot interface for bidirectional information exchange.

Formalisation of the learning problem

The overall goal is to let the robot successfully perform a task by taking actions $\boldsymbol{a} \in \mathscr{A}$, given observations $\boldsymbol{o} \in \mathscr{O}$, where $\mathscr{A}$ and $\mathscr{O}$ are the robot's action and observation spaces, respectively. These actions could, for example, be reference end-effector configurations, while observations could comprise joint angle measurements

and camera images. These observations result from states $\boldsymbol{s} \in \mathscr{S}$, i.e., $\boldsymbol{o} = O(\boldsymbol{s})$ where $\mathscr{S}$ is the state space and O the observation mapping $O : \mathscr{S} \rightarrow \mathscr{O}$. We make this distinction between states and observations, since full state information is often not available in real world scenarios. Since actions follow from observations, the aim is to find a policy π (which is a mapping from observations to actions, i.e., $\pi : \mathscr{O} \rightarrow \mathscr{A}$) such that a task is performed successfully. Furthermore, we would like this policy to be successful starting from a set of initial states $\boldsymbol{s}_0 \in \mathscr{S}_0 \subset \mathscr{S}$. That is to say, the robot policy does not have to be successful starting from all possible states, but should be performing well for the actual distribution over initial states it encounters, which we denote by $p(\boldsymbol{s}_0)$. Furthermore, successful completion of a task can be determined by defining a goal state set $\mathscr{S}_g \subset \mathscr{S}$. Note that goals could be dynamic and part of the state $\boldsymbol{s}$. We can quantify the optimality of a policy by defining a reward function $R(\boldsymbol{s})$. Given the reward function, we can define the value of a state [367]:

$$V_\pi(\boldsymbol{s}) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R(\boldsymbol{s}_{t+k+1}) \middle| \boldsymbol{s}_t = \boldsymbol{s} \right], \tag{3.1}$$

where $\mathbb{E}_\pi[\cdot]$ is the expected value given that policy π is executed, t is any time step and γ is the discount rate with $0 \leq \gamma \leq 1$. Intuitively, the value of a state says how well the policy performs on average starting from that state following policy π. We define the optimal policy π^* as the one that has the highest expected value given our distribution of initial states:

$$\pi^* = \arg\max_{\pi \in \Pi} \mathbb{E}_{\boldsymbol{s}_0 \sim p(\boldsymbol{s}_0)}[V_\pi(\boldsymbol{s}_0)], \tag{3.2}$$

where Π is the policy space and $\mathbb{E}_{\boldsymbol{s}_0 \sim p(\boldsymbol{s}_0)}[\cdot]$ denotes the expectation given that $\boldsymbol{s}_0$ is drawn from distribution $p(\boldsymbol{s}_0)$. However, in practice it is often not trivial to come up with an appropriate reward function $R(\boldsymbol{s})$. This would particularly be the case if we were to solve this problem with RL. In that case, the reward function should ideally guide the robot towards successful behaviour [368], which would require *reward shaping* [369]. This can be a time consuming process that could also lead to suboptimal behaviour. Robotic RL also results in challenges related to data efficiency and safety [368, 370]. Therefore, we choose to find π^* by LfD. In this setting, it is not required that $R(\boldsymbol{s})$ guides the agent and we can only reward success:

$$R(\boldsymbol{s}) = \begin{cases} 1 & \text{if} \quad \boldsymbol{s} \in \mathscr{S}_g \\ 0 & \text{else} \end{cases}. \tag{3.3}$$

Worth noting is that with this reward function and $\gamma = 1$, the optimal policy π^* will simply be the policy with the highest success rate. By setting $0 < \gamma < 1$, not only success will be rewarded, but the robot learner will also be stimulated to solve the task in minimum-time. This is desirable in our scenario, as time is related to cost in most industrial applications.

Our method falls within the realm of Interactive Imitation Learning (IIL), since demonstrations are given interactively. Compared to standard IL training, IIL can

be more sample-efficient since demonstrations are also collected while executing the novice policy, rather than the teacher's policy alone [371]. This results in demonstrations for states that the robot learner actually encounters, rather than only demonstrations for states the teacher encounters. We can collect a dataset with N demonstration trajectories $\mathscr{D} = \{\boldsymbol{\tau}_0, \boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_N\}$ and try to imitate the expert policy. We consider demonstrations that consist of observation vectors, i.e., $\boldsymbol{\tau} = [\boldsymbol{o}_0, \boldsymbol{o}_1, \ldots, \boldsymbol{o}_T]$ where T is the last time step of demonstration $\boldsymbol{\tau}$. An estimate of the optimal policy can be obtained by minimizing a loss between the demonstrated trajectories and trajectories resulting from the policy:

$$\hat{\pi}^* = \underset{\pi \in \Pi}{\operatorname{argmin}} L(\pi, \mathscr{D}). \tag{3.4}$$

A popular choice for L is the Kullback–Leibler divergence [372] between the distribution of observations induced by the learner policy and teacher policy [373]. We should however take into consideration that demonstrations are costly, since they can be time consuming for the human and ideally we would like the system to have a high level of autonomy to maximize efficiency. Therefore, we wish to optimize task success while minimizing the number of demonstrations:

$$\begin{aligned} &\max_{\mathscr{D}} \mathbb{E}_{\boldsymbol{s}_0 \sim p(\boldsymbol{s}_0)}[V_{\hat{\pi}^*}(\boldsymbol{s}_0)] - \lambda |\mathscr{D}| \\ &\quad \text{s.t.} \quad \hat{\pi}^* = \underset{\pi \in \Pi}{\operatorname{argmin}} L(\pi, \mathscr{D}), \end{aligned} \tag{3.5}$$

where $|\mathscr{D}|$ is the cardinality of $\mathscr{D}$ (the number of demonstrated trajectories) and λ a regularization parameter. In this way, we have arrived at an expression for the optimal set of demonstrations. The maximization of the expectation of the state values in (3.5) ensures that we maximize task success, while the regularization term penalizes the number of demonstrations in our dataset. We choose to penalize the number of trajectories $\boldsymbol{\tau}$ rather than their length, since a longer demonstration is preferred over multiple short ones, both by the human and considering the potential cost per demonstration for preparing and processing it.

Concept idea

Finding the optimal solution to the optimization problem in (3.5) is difficult, because the problem does not have an analytical solution or gradient. It may be possible to use numerical optimization methods that iteratively improve a candidate solution until it satisfies some stopping criterion, such as reaching a certain tolerance level (e.g., success rate) or a maximum number of iterations. However, when evaluations are costly, the optimization problem can quickly become intractable, because it may take a large number of evaluations to find a good solution. In the context of the optimization problem in (3.5), evaluating the performance of a candidate solution (i.e., a dataset of demonstrations) requires collecting human demonstrations and physical experimentation to evaluate the performance, which can be time-consuming and expensive. Then, it is important to design an optimization algorithm that balances

the need for exploration and exploitation and is efficient in terms of the number of evaluations required to find a good solution.

Simulation is a cost-effective means of evaluating candidate solutions without the risks and expenses associated with real-world experimentation. Recent advancements in parallelized physics simulation on accelerated platforms have enabled fast simulations [374, 375]. However, the dissimilarities between the simulator and real-world environments can hinder the transferability of policies learned in simulation to real-world settings, due to modeling errors and differences in physical properties. As learning methods tend to exploit these differences to maximize simulated rewards, simulators' conventional use results in overestimation of real-world performance. This may lead to safety hazards and unexpected failures during deployment. Overestimating the real-world performance of a robot policy trained in simulation can lead to safety hazards and unexpected failures during deployment.

To address this issue, we propose to swap the real-world and simulator's roles to synthesize policies using human demonstrations and evaluate them using accelerated physics simulation. By doing so, discrepancies between simulation and reality lead to an underestimation of real-world performance. Failures in simulation may be attributed to either a sub-optimal policy or discrepancies. In case of success, the policy was robust enough to achieve the goal despite the discrepancies. The proposed solution does not rely on gradient information and instead uses a simulation based search strategy to find a good solution for the problem in (3.5). The system comprises two phases: pre-training and lifelong learning. During the pre-training phase, real-world demonstrations are continuously provided by the human operator, until a certain success rate is achieved in simulation or a maximum number of iterations is reached (see Algorithm 2). After the pre-training phase, the robot enters the lifelong learning phase, where it executes the policy independently. However, it will halt and request a human-demonstration if it encounters a state that resulted in a low success rate in simulation (see Algorithm 3).

Algorithms 2 and 3 provide the workflow of these two phases, while the involved functions are detailed below.

`evaluate`: this function estimates the policy's performance by computing the value function V_π, as defined in (3.1), which reflects the policy's performance over the induced state distribution, given a reward function R and the initial state distribution $p(\boldsymbol{s}_0)$. Computing V_π may be impractical due to the large number of real-world evaluations required and limited access to the full state $\boldsymbol{s}$. Monte Carlo sampling and an accelerated physics simulator can overcome such challenges (e.g., [374, 375]), as both are suitable for parallel implementation, to estimate an approximate value function $\hat{V}_\pi(\boldsymbol{o})$ that is a function of the observations $\boldsymbol{o}$, serving as a proxy for the real value function. Depending on the problem, different function approximators can be employed to represent $\hat{V}_\pi(\boldsymbol{o})$. For simpler problems, a tabular representation may be sufficient. However, for more complex problems, more expressive function approximators such as artificial neural networks or Gaussian processes may be used. If there are significant differences between the real-world and simulator, a policy trained with real-world demonstrations may, in some cases, always fail to solve the

Algorithm 2 Pre-training phase.

Input: Reward function: $R(\boldsymbol{s})$ // See Eq. (3.3)
LfD method: $L(\pi, \mathscr{D})$ // See Eq. (3.4)
Initial states: $\boldsymbol{s}_0 \sim p(\boldsymbol{s}_0)$
Initial policy: π_0
Success threshold: α
Output: Pre-trained policy: π
Dataset: $\mathscr{D}$

$\mathscr{D} \leftarrow \emptyset$ // Initialize empty dataset
$\pi \leftarrow \pi_0$ // Initialize policy
$\hat{V}_\pi \leftarrow \texttt{evaluate}(R, p(\boldsymbol{s}_0), \pi)$ // In simulation
do
  $\boldsymbol{s}_{\text{start}} \leftarrow \texttt{suggest}(\hat{V}_\pi)$ // Start state of demo
  $\boldsymbol{\tau} \leftarrow \texttt{demonstrate}(\boldsymbol{s}_{start})$ // In real-world
  $\mathscr{D} \leftarrow \mathscr{D} \cup \{\boldsymbol{\tau}\}$ // Aggregate data
  $\pi \leftarrow \texttt{optimize}\ (L, \mathscr{D})$ // Update policy
  $\hat{V}_\pi \leftarrow \texttt{evaluate}(R, p(\boldsymbol{s}_0), \pi)$ // In simulation
until $\mathbb{E}_{\boldsymbol{s}_0 \sim p(\boldsymbol{s}_0)}[\hat{V}_\pi(O(\boldsymbol{s}_0))] > \alpha$ // Success rate

Algorithm 3 Lifelong learning phase.

Input: Reward function: $R(\boldsymbol{s})$ // See Eq. (3.3)
LfD method: $L(\pi, \mathscr{D})$ // See Eq. (3.4)
Initial states: $\boldsymbol{s}_0 \sim p(\boldsymbol{s}_0)$
Pre-trained policy: π // See Alg. 2
Dataset: $\mathscr{D}$ // See Alg. 2

$\hat{V}_\pi \leftarrow \texttt{evaluate}(R, \mathscr{S}_{start}, \pi)$ // In simulation
$\boldsymbol{s} \leftarrow \texttt{reset}\ (p(\boldsymbol{s}_0))$ // In Real-world
while *running* **do**
  $\boldsymbol{o} \leftarrow O(\boldsymbol{s})$ // Read sensor observations
  if $\hat{V}_\pi(\boldsymbol{o}) > \alpha$ **then** run
    $\boldsymbol{a} \leftarrow \pi(\boldsymbol{o})$ // Get action
    $\boldsymbol{s} \leftarrow \texttt{act}(\boldsymbol{a})$ // In real-world
  else request demonstration
    $\boldsymbol{s}_{start} \leftarrow \boldsymbol{s}$ // Demo from current state
    $\boldsymbol{\tau} \leftarrow \texttt{demonstrate}(\boldsymbol{s}_{start})$ // In real-world
    $\mathscr{D} \leftarrow \mathscr{D} \cup \{\boldsymbol{\tau}\}$ // Aggregate data
    $\pi \leftarrow \texttt{optimize}\ (L, \mathscr{D})$ // Update policy
    $\hat{V}_\pi \leftarrow \texttt{evaluate}(R, p(\boldsymbol{s}_0), \pi)$ // In simulation
  if *done* **then**
    $\boldsymbol{s} \leftarrow \texttt{reset}\ (p(\boldsymbol{s}_0))$ // In real-world

task in simulation. In this case, we propose to use on-policy RL algorithms such as [376] to address the discrepancies. Specifically, an agent can be trained to solve the task with RL while constraining its policy to remain close to the real-world policy. By allowing slight deviations from the real-world policy, we can mitigate the differences between the real-world and simulator.

`suggest`: This function proposes a new starting state $\boldsymbol{s}_{\text{start}}$ for the next demonstration based on the policy's simulated performance, which is reflected by the estimated value function $\hat{V}_\pi(\boldsymbol{o})$. Note that the starting state does not necessarily have to lie in the set of initial states $\mathscr{S}_0$. Proper selection of starting states can significantly reduce the number of demonstrations needed to achieve adequate performance, a task that is typically performed by an expert user with knowledge of the chosen LfD method. Alternatively, meta learning, a technique for learning to learn and adapt to new tasks, can be employed [377]. In this case, a meta learning model would be trained to suggest the next starting state for a demonstration expected to improve the current value function $\hat{V}_\pi(\boldsymbol{o})$ the most. The system could be trained on examples of demonstrations and their incremental impact on improving $\hat{V}\pi(\boldsymbol{o})$.

`demonstrate`: This function requests the human to provide a human-demonstration from a given starting state $\boldsymbol{s}_{\text{start}}$. Depending on the task, this could occur via, for example, kinesthetic teaching or teleoperation [371]. Then, `optimize` estimates a policy $\hat{\pi}^*$, as defined in (3.4), by minimizing the loss function determined by the selected method. Finally, `act` applies the action proposed by the robot's policy π to the system, and `reset` resets the real-world system to an initial state. In other words, it samples a new state $\boldsymbol{s}_0 \sim p(\boldsymbol{s}_0)$.

In summary, this work is proposed as a means to reduce the cost of the demonstrations process for the human teacher, while simultaneously achieving a balance between exploration and exploitation for optimal generalization. The current implementation of the system serves as a POC for what the finalized version may look like. For instance, a first implementation need to make the system user-friendly, and the information presentation to the user would need to be conveyed in an intuitive and easily understandable manner.

Conclusions

The research work brought on in the last three years, features the integration of open source tools, libraries and data, while taking advantage of well-established algorithms or slightly modified versions of these, to create new solutions. This research journey has a main bring away: knowledge sharing, cross-field contamination and tool integration are key to research innovative approaches, and allow to speed up their validation on laboratory demonstrators.

The AMR as a meta-sensor, or Sen3Bot, was first only imagined (Section 2.1.1), and then implemented (Section 2.1.3), using cutting-edge object recognition methods on a low-cost camera video sources, combined with distance information to detect and avoid human obstacles. From the path planning point of view the AMR first emulates AGVs fixed path, following a virtual safe path (Section 1.2.1), then integrating the human-recognition capabilities (Section 2.2.1) to envision a network of Sen3Bots (Section 2.1.2). This set of works provided an example of integrating new technologies to upgrade existing and established equipment, which may be obsolete but still working. Instead of completely replacing older systems, companies could upgrade and integrate them with new technologies to increase efficiency and reduce waste. This approach appear to be compliant with the circular economy principle of re-use, allowing to favour sustainability of renewing processes, yet keeping in mind that a trade-off between reuse and energy-efficient equipment adoption must be found.

Moreover, the framework envisioned in Section 3.1 and implemented as reported in Section 3.1.1 exploits deep learning models to recognize a human operation from map images featuring the human 2D motion. This work demonstrates that minimal but useful results can be achieved even in case of data scarcity, exploiting data that is independent of the specific features of the operator. In fact, the mobile robot actually learns by looking at human operations demonstrations, in order to be able to accordingly make decisions about its own motion. The human is reduced to a meaningful point on a map, stripped of details that are not useful for recognizing the action. In some way, we want to make the robot learn from a human as another human would do. In fact, when we learn, we do not actually overfit on the action source but rather on the action itself, which of course is intrinsically characterized by how that specific human executes it, but the learning process should get rid of features that go beyond the teaching demonstration execution.

Finally, the research idea introduced in Section 3.2.2 brings into the picture simulation to regulate interactive demonstrations in human-in-the-loop imitation learning, i.e., interactive imitation learning. IIL is a good example of how new learning paradigms try to make the robot learn as a human would do. The robot first learns from demonstrations, then learns further from the corrective actions it's subjected to while executing the task according to its own policy, hence actually learning by doing. Moreover this work represent an effort towards reducing the burden on humans, with the aim of making the teaching process less intense in terms of teacher's time and mental fatigue. For a direct mapping of thesis sections to co-authored and published research papers, the reader is redirected to the **Preface** of this manuscript.

The lessons learned from this research investigation are the following:

- *Less is more*.
  When testing research ideas, simple initial scenario are enough to demonstrate your point. As the aim is to validate a concept, overly complex situations may hinder the purpose. Also, when considering the learning process, we ourselves don't start off with the most difficult situation or task. Similarly, it is important to respect the robot's learning process, gradually making a simple initial task increasingly more complex, to build skills upon a solid base.

- *Laboratory demonstrators are not prototypes*.
  The tested implementations are often not mature enough technologies, as their aim is to validate a research idea in a controlled laboratory setting, within use cases that rarely correspond to the real-world conditions ones. The same is valid for the simulation testing phase. Hence, many aspects need to be reconsidered in case a prototype has to be delivered, e.g., working conditions, communication standards, proprietary software limitations, etc.

- *Not all hardware needs to be high-end*.
  In particular, sensors don't necessarily need to have advanced specifications, unless they are algorithmically or computationally integrating the capabilities of the onboard computer. In fact, the main performance minimal requirements are related to the computational power necessary to execute machine learning algorithms and to process a large amount of data. Low-cost sensors' inaccuracy, in some cases, can be compensated through sensor data fusion.

- *Open-source is a precious source*.
  Open-source tools open the door to a worldwide shared knowledge. As such, the open-source world is a bit cumbersome but full of information that, if appropriately sourced, can be exploited for creating innovative material. ROS is one of these tools, which really has represented a turning point for democratizing robotics, beyond the research community. This way, the technological development of fields the open-source philosophy reaches is accelerated, as a whole community of coders and academics have your back while learning and shared knowledge is freely accessible.

In the context of the reported thesis, when looking at future directions of robotics in manufacturing and production lines, I would like to share the following insights.

- *Industrial robotics at the time of metaverse*.
  Simulation, as a model-based tool to predict the behavior of a system and its outputs under different scenarios and conditions, has always been a key tool for industrial processes preliminary safe testing and optimization. A step forward has been made with the use of *digital twins*, i.e., simulation of physical systems using real-time data, mainly used for predictive and evaluation, e.g., real-time monitoring, optimization, and maintenance. Considering a future era where everything or everyone have a digital representation, at which the metaverse concept hint, industrial robotics will most likely have digital twins as a standard approach to prevent failures. Many projects are going this direction, for example the NVIDIA Omniverse [378] platform, which aims at facilitating collaborative, real-time, and immersive 3D simulation and visualization of physical objects in a shared virtual environment, easing the creation of powerful industrial digital twins.

- *Human-centric factories and KPIs*.
  Future smart production processes will have to cope with the presence of humans in the loop of production as a main component while satisfying production KPIs (Key Performance Indicators). The manual execution of some operations within the manufacturing process, as well as the presence itself of human operators in the environment shared with robots, will have to become an added value for the quality of the results, and not a potential cause of low efficiency, to try and maintain acceptable KPIs. This goal can be achieved only through a proper choice of sensors and techniques, suitable for each particular kind of robotic system and application.

- *Dataset creation for industrial robotics*.
  A possible further enhancement for the training process of learning robots, would be represented by extending the existing dataset pool for industrial robotics. For instance, industrial open dataset should be focusing on data collection for operation recognition of standard operations or recognition of objects specific to the industrial environment. Standardization of processes and plants would help in the creation of datasets that could used as pre-training phase of available models. Also, when considering human related data collection, unbiased solutions transcending the specific person or situation will be key to achieve adaptable learning robots.

In conclusion, robotics in collaborative manufacturing applications needs to take into considerations many safety and performance requirements. A learning collaborative robot acts and works closely to a human teammate to support his/her work, so it needs to be cautiously designed for this aim. Robots nowadays are proving to be important resources to support humans not only in the industrial production environment but also in everyday household and social life, working around people.

Precisely because many open challenges are there for robot learning, there is plenty of room for research opportunities. The research community is not letting slip such opportunities away and already started integrating cutting-edge technologies with robotics. Just think of the "Chatgpt for robotics" project [379]: this project aims at using the Chatgpt language model developed by OpenAI, to exploit natural language processing in robotics tasks. The journey that will take robots to learn how to behave is thus tough, but nonetheless it will be very exciting.

References

[1] D. Sinha and R. Roy, "Reviewing cyber-physical system as a part of smart factory in industry 4.0," *IEEE Engineering Management Review*, vol. 48, no. 2, pp. 103–117, 2020.

[2] J. Juergensen, J. Guimón, and R. Narula, "European smes amidst the covid-19 crisis: assessing impact and policy responses," *Journal of Industrial and Business Economics*, vol. 47, no. 3, pp. 499–510, 2020.

[3] S. Phuyal, D. Bista, and R. Bista, "Challenges, opportunities and future directions of smart manufacturing: a state of art review," *Sustainable Futures*, vol. 2, p. 100023, 2020.

[4] K. Winkle, D. McMillan, M. Arnelid, K. Harrison, M. Balaam, E. Johnson, and I. Leite, "Feminist human-robot interaction: Disentangling power, principles and practice for better, more ethical hri," in *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 72–82, 2023.

[5] M. Indri, L. Lachello, I. Lazzero, F. Sibona, and S. Trapani, "Smart sensors applications for a new paradigm of a production line," *Sensors*, vol. 19, no. 3, p. 650, 2019.

[6] M. Indri, C. Possieri, F. Sibona, P. D. Cen Cheng, and V. D. Hoang, "Supervised global path planning for mobile robots with obstacle avoidance," in *24th IEEE International Conference on Emerging Technologies and Factory Automation*, (Zaragoza, Spain), pp. 601–608, 2019.

[7] M. Indri, F. Sibona, and P. D. Cen Cheng, "Sensor data fusion for smart amrs in human-shared industrial workspaces," in *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, vol. 1, pp. 738–743, IEEE, 2019.

[8] M. Indri, F. Sibona, P. D. Cen Cheng, and C. Possieri, "Online supervised global path planning for amrs with human-obstacle avoidance," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, pp. 1473–1479, IEEE, 2020.

[9] M. Indri, F. Sibona, and P. D. Cen Cheng, "Sen3bot net: a meta-sensors network to enable smart factories implementation," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, pp. 719–726, IEEE, 2020.

[10] F. Sibona and M. Indri, "Data-driven framework to improve collaborative human-robot flexible manufacturing applications," in *IECON 2021–47th Annual Conference of the IEEE Industrial Electronics Society*, pp. 1–6, IEEE, 2021.

[11] A. Bonci, P. D. Cen Cheng, M. Indri, G. Nabissi, and F. Sibona, "Human-robot perception in industrial environments: A survey," *Sensors*, vol. 21, no. 5, p. 1571, 2021.

[12] F. Sibona, P. D. C. Cheng, and M. Indri, "How to improve human-robot collaborative applications through operation recognition based on human 2d motion," in *IECON 2022–48th Annual Conference of the IEEE Industrial Electronics Society*, pp. 1–6, IEEE, 2022.

[13] F. Sibona, P. D. C. Cheng, M. Indri, and D. Di Prima, "Pointap system: a human-robot interface to enable remotely controlled tasks," in *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 01–08, IEEE, 2021.

[14] P. D. Cen Cheng, F. Sibona, and M. Indri, "A framework for safe and intuitive human-robot interaction for assistant robotics," in *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1–4, IEEE, 2022.

[15] P. D. cen Cheng, M. Indri, F. Sibona, M. De Rose, and G. Prato, "Dynamic path planning of a mobile robot adopting a costmap layer approach in ros2," in *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1–8, IEEE, 2022.

[16] P. D. Cen Cheng, M. Indri, C. Possieri, M. Sassano, and F. Sibona, "Path planning in formation and collision avoidance for multi-agent systems," *Nonlinear Analysis: Hybrid Systems*, vol. 47, p. 101293, 2023.

[17] E. I. Delivered, "The trl scale as a research & innovation policy tool, earto recommendations," *Earto Impact Delivered*, 2014.

[18] J. C. Mankins *et al.*, "Technology readiness levels," *White Paper, April*, vol. 6, no. 1995, p. 1995, 1995.

[19] Fiorella Sibona, "Operation recognition based on human 2D motion." Available online: https://youtu.be/JY1ihYM8Pks (accessed: April 2023).

[20] M. C. Lucas-Estañ, M. Sepulcre, T. P. Raptis, A. Passarella, and M. Conti, "Emerging trends in hybrid wireless communication and data management for the industry 4.0," *Electronics*, vol. 7, no. 12, 2018.

[21] K. Schwab, " The Fourth Industrial Revolution." Available online: https://www.foreignaffairs.com/world/fourth-industrial-revolution (accessed: April 2023), 2015.

[22] "Groupe psa - the multi-silhouette production line." Available online: https://www.youtube.com/watch?v=HMIOi5FIePc (accessed: April 2023).

[23] "KUKA launches a pilot plant for matrix production." Available online: https://www.hannovermesse.de/en/news/kuka-launches-a-pilot-plant-for-matrix-production-73418.xhtml (accessed: April 2023).

[24] B. Chen, J. Wan, L. Shu, P. Li, M. Mukherjee, and B. Yin, "Smart factory of industry 4.0: Key technologies, application case, and challenges," *IEEE Access*, 2018.

[25] S. Wang, J. Wan, D. Li, and C. Zhang, "Implementing smart factory of industrie 4.0: an outlook," *International Journal of Distributed Sensor Networks*, vol. 12, no. 1, 2016.

[26] Y. Lu, H. Zheng, S. Chand, W. Xia, Z. Liu, X. Xu, L. Wang, Z. Qin, and J. Bao, "Outlook on human-centric manufacturing towards Industry 5.0," *Journal of Manufacturing Systems*, vol. 62, pp. 612–627, 2022.

[27] S. Nahavandi, "Industry 5.0—a human-centric solution," *Sustainability*, vol. 11, no. 16, p. 4371, 2019.

[28] S. K. Yadav, K. Tiwari, H. M. Pandey, and S. A. Akbar, "A review of multimodal human activity recognition with special emphasis on classification, applications, challenges and future directions," *Knowledge-Based Systems*, vol. 223, p. 106970, 2021.

[29] P. K. R. Maddikunta, Q.-V. Pham, B. Prabadevi, N. Deepa, K. Dev, T. R. Gadekallu, R. Ruby, and M. Liyanage, "Industry 5.0: A survey on enabling technologies and potential applications," *Journal of Industrial Information Integration*, vol. 26, p. 100257, 2022.

[30] Richard Matheson, Director and Market Development at Nickel Institute, "Customising the future – The next industrial revolution." Available online: https://nickelinstitute.org/en/blog/2020/november/customising-the-future-the-next-industrial-revolution/ (accessed: April 2023).

[31] P. Zheng, H. wang, Z. Sang, R. Y. Zhong, Y. Liu, C. Liu, K. Mubarok, S. Yu, and X. Xu, "Smart manufacturing systems for industry 4.0: Conceptual framework, scenarios, and future perspectives," *Frontiers of Mechanical Engineering*, 2018.

[32] "Sarissa assistenzsysteme." Available online: https://www.sarissa.de/en/ (accessed: April 2023).

[33] "ulixes robotersysteme." Available online: http://ulixes.de/ (accessed: April 2023).

[34] "Ops solutions llc." Available online: https://lightguidesys.com/ (accessed: April 2023).

[35] "Z-laser." Available online: https://z-laser.com/en/ (accessed: April 2023).

[36] "Smart robots." Available online: http://smartrobots.it/ (accessed: April 2023).

[37] X. Wu, V. Goepp, and A. Siadat, "Cyber physical production systems: A review of design and implementation approaches," in *2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pp. 1588–1592, IEEE, 2019.

[38] A. Bonci, M. Pirani, and S. Longhi, "An embedded database technology perspective in cyber-physical production systems," *Procedia Manufacturing*, vol. 11, pp. 830–837, 2017.

[39] A. Bonci, M. Pirani, A. F. Dragoni, A. Cucchiarelli, and S. Longhi, "The relational model: In search for lean and mean cps technology," in *2017 IEEE 15th International Conference on Industrial Informatics (INDIN)*, pp. 127–132, IEEE, 2017.

[40] A. Bonci, M. Pirani, A. Cucchiarelli, A. Carbonari, B. Naticchia, and S. Longhi, "A review of recursive holarchies for viable systems in cpss," in *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*, pp. 37–42, IEEE, 2018.

[41] Y. Yildiz, "Cyberphysical human systems: An introduction to the special issue," *IEEE Control Systems Magazine*, vol. 40, no. 6, pp. 26–28, 2020.

[42] A. P. Dani, I. Salehi, G. Rotithor, D. Trombetta, and H. Ravichandar, "Human-in-the-loop robot control for human-robot collaboration: Human intention estimation and safe trajectory tracking control for collaborative tasks," *IEEE Control Systems Magazine*, vol. 40, no. 6, pp. 29–56, 2020.

[43] M. Pantano, D. Regulin, B. Lutz, and D. Lee, "A human-cyber-physical system approach to lean automation using an industrie 4.0 reference architecture," *Procedia Manufacturing*, vol. 51, pp. 1082–1090, 2020.

[44] A. Bonci, S. Longhi, G. Nabissi, and F. Verdini, "Predictive maintenance system using motor current signal analysis for industrial robot," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1453–1456, IEEE, 2019.

[45] A. Hentout, M. Aouache, A. Maoudj, and I. Akli, "Human–robot interaction in industrial collaborative robotics: a literature review of the decade 2008–2017," *Advanced Robotics*, vol. 33, no. 15-16, pp. 764–799, 2019.

[46] M. J. Rosenstrauch and J. Krüger, "Safe human-robot-collaboration-introduction and experiment using iso/ts 15066," in *2017 3rd International conference on control, automation and robotics (ICCAR)*, pp. 740–744, IEEE, 2017.

[47] I. ISO, "Iso/ts 15066: 2016: Robots and robotic devices–collaborative robots," *Geneva, Switzerland: International Organization for Standardization*, 2016.

[48] P. Aivaliotis, S. Aivaliotis, C. Gkournelos, K. Kokkalis, G. Michalos, and S. Makris, "Power and force limiting on industrial robots for human-robot collaboration," *Robotics and Computer-Integrated Manufacturing*, vol. 59, pp. 346–360, 2019.

[49] B. Yao, Z. Zhou, L. Wang, W. Xu, and Q. Liu, "Sensor-less external force detection for industrial manipulators to facilitate physical human-robot interaction," *Journal of Mechanical Science and Technology*, vol. 32, no. 10, pp. 4909–4923, 2018.

[50] X. Wang, C. Yang, Z. Ju, H. Ma, and M. Fu, "Robot manipulator self-identification for surrounding obstacle detection," *Multimedia Tools and Applications*, vol. 76, no. 5, pp. 6495–6520, 2017.

[51] F. Flacco and A. De Luca, "Real-time computation of distance to dynamic obstacles with multiple depth sensors," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 56–63, 2016.

[52] T. Brito, J. Lima, P. Costa, and L. Piardi, "Dynamic collision avoidance system for a manipulator based on rgb-d data," in *Iberian Robotics conference*, pp. 643–654, Springer, 2017.

[53] E. Matsas and G.-C. Vosniakos, "Design of a virtual reality training system for human–robot collaboration in manufacturing tasks," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 11, no. 2, pp. 139–153, 2017.

[54] A. Vit and G. Shani, "Comparing rgb-d sensors for close range outdoor agricultural phenotyping," *Sensors*, vol. 18, no. 12, p. 4413, 2018.

[55] J. Zabalza, Z. Fei, C. Wong, Y. Yan, C. Mineo, E. Yang, T. Rodden, J. Mehnen, Q.-C. Pham, and J. Ren, "Smart sensing and adaptive reasoning for enabling industrial robots with interactive human-robot capabilities in dynamic environments—a case study," *Sensors*, vol. 19, no. 6, p. 1354, 2019.

[56] H. Su, S. E. Ovur, Z. Li, Y. Hu, J. Li, A. Knoll, G. Ferrigno, and E. De Momi, "Internet of things (iot)-based collaborative control of a redundant manipulator for teleoperated minimally invasive surgeries," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9737–9742, IEEE, 2020.

[57] H. Su, J. Sandoval, P. Vieyres, G. Poisson, G. Ferrigno, and E. De Momi, "Safety-enhanced collaborative framework for tele-operated minimally invasive surgery using a 7-dof torque-controlled robot," *International Journal of Control, Automation and Systems*, vol. 16, no. 6, pp. 2915–2923, 2018.

[58] Y. Ding and U. Thomas, "Collision avoidance with proximity servoing for redundant serial robot manipulators," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10249–10255, IEEE, 2020.

[59] V. Villani, F. Pini, F. Leali, and C. Secchi, "Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications," *Mechatronics*, vol. 55, pp. 248 – 266, 2018.

[60] M. Sakr, W. Uddin, and H. F. M. Van der Loos, "Orthographic vision-based interface with motion-tracking system for robot arm teleoperation: A comparative study," in *Companion of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, p. 424–426, 2020.

[61] S. Yang, W. Xu, Z. Liu, Z. Zhou, and D. T. Pham, "Multi-source vision perception for human-robot collaboration in manufacturing," in *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, 2018.

[62] P. Neto, M. Simão, N. Mendes, and M. Safeea, "Gesture-based human-robot interaction for human assistance in manufacturing," *The International Journal of Advanced Manufacturing Technology*, vol. 101, no. 1, pp. 119–135, 2019.

[63] E. Digo, M. Antonelli, V. Cornagliotto, S. Pastorelli, and L. Gastaldi, "Collection and analysis of human upper limbs motion features for collaborative robotic applications," *Robotics*, vol. 9, no. 2, p. 33, 2020.

[64] M. Ragaglia, A. M. Zanchettin, and P. Rocco, "Trajectory generation algorithm for safe human-robot collaboration based on multiple depth sensor measurements," *Mechatronics*, vol. 55, pp. 267–281, 2018.

[65] M. Köseoğlu, O. M. Çelik, and Ö. Pektaş, "Design of an autonomous mobile robot based on ROS," in *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, pp. 1–5, IEEE, 2017.

[66] H. B. Mitchell, *Multi-sensor data fusion: an introduction*. Springer Science & Business Media, 2007.

[67] L. Lynch, T. Newe, J. Clifford, J. Coleman, J. Walsh, and D. Toal, "Automated ground vehicle (agv) and sensor technologies-a review," in *2018 12th International Conference on Sensing Technology (ICST)*, pp. 347–352, IEEE, 2018.

[68] G. Fedorko, S. Honus, and R. Salai, "Comparison of the traditional and autonomous agv systems," in *MATEC Web of Conferences*, vol. 134, p. 00013, EDP Sciences, 2017.

[69] S. Zhou, G. Cheng, Q. Meng, H. Lin, Z. Du, and F. Wang, "Development of multi-sensor information fusion and agv navigation system," in *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, vol. 1, pp. 2043–2046, IEEE, 2020.

[70] J. Theunissen, H. Xu, R. Y. Zhong, and X. Xu, "Smart agv system for manufacturing shopfloor in the context of industry 4.0," in *2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, pp. 1–6, IEEE, 2018.

[71] E. A. Oyekanlu, A. C. Smith, W. P. Thomas, G. Mulroy, D. Hitesh, M. Ramsey, D. J. Kuhn, J. D. Mcghinnis, S. C. Buonavita, N. A. Looper, *et al.*, "A review of recent advances in automated guided vehicle technologies: Integration challenges and research areas for 5g-based smart manufacturing applications," *IEEE Access*, vol. 8, pp. 202312–202353, 2020.

[72] Y. Zhang, C.-H. Zhang, and X. Shao, "User preference-aware navigation for mobile robot in domestic via defined virtual area," *Journal of Network and Computer Applications*, vol. 173, p. 102885.

[73] K. Chandan, X. Zhang, J. Albertson, X. Zhang, Y. Liu, and S. Zhang, "Guided 360-degree visual perception for mobile telepresence robots," in *The RSS-2020 Workshop on Closing the Academia to Real-World Gap in Service Robotics*, 2020.

[74] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[75] J. Luo, Z. Lin, Y. Li, and C. Yang, "A teleoperation framework for mobile robots based on shared control," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 377–384, 2019.

[76] C. Mavrogiannis, A. M. Hutchinson, J. Macdonald, P. Alves-Oliveira, and R. A. Knepper, "Effects of distinct robot navigation strategies on human behavior in a crowded environment," in *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 421–430, IEEE, 2019.

[77] S. Siva and H. Zhang, "Robot perceptual adaptation to environment changes for long-term human teammate following," *The International Journal of Robotics Research*, p. 0278364919896625, 2020.

[78] F. Han, S. Siva, and H. Zhang, "Scalable representation learning for long-term augmented reality-based information delivery in collaborative human-robot perception," in *International Conference on Human-Computer Interaction*, pp. 47–62, Springer, 2019.

[79] L. Benos, A. Bechar, and D. Bochtis, "Safety and ergonomics in human-robot interactive agricultural operations," *Biosystems Engineering*, vol. 200, pp. 55–72, 2020.

[80] J. P. Vasconez, G. A. Kantor, and F. A. A. Cheein, "Human–robot interaction in agriculture: A survey and current challenges," *Biosystems engineering*, vol. 179, pp. 35–48, 2019.

[81] W. Wang, R. Wang, and G. Chen, "Path planning model of mobile robots in the context of crowds," *arXiv preprint arXiv:2009.04625*, 2020.

[82] J. Li, L. Lu, L. Zhao, C. Wang, and J. Li, "An integrated approach for robotic sit-to-stand assistance: Control framework design and human intention recognition," *Control Engineering Practice*, vol. 107, p. 104680, 2021.

[83] W. K. Juel, F. Haarslev, E. R. Ramírez, E. Marchetti, K. Fischer, D. Shaikh, P. Manoonpong, C. Hauch, L. Bodenhagen, and N. Krüger, "Smooth robot: Design for a novel modular welfare robot," *Journal of Intelligent & Robotic Systems*, vol. 98, no. 1, pp. 19–37, 2020.

[84] P. Kolar, P. Benavidez, and M. Jamshidi, "Survey of datafusion techniques for laser and vision based sensor integration for autonomous navigation," *Sensors*, vol. 20, no. 8, p. 2180, 2020.

[85] S. B. Banisetty and D. Feil-Seifer, "Towards a unified planner for socially-aware navigation," *arXiv preprint arXiv:1810.00966*, 2018.

[86] F. Marques, D. Gonçalves, J. Barata, and P. Santana, "Human-aware navigation for autonomous mobile robots for intra-factory logistics," in *International Workshop on Symbiotic Interaction*, pp. 79–85, Springer, 2017.

[87] M. A. Kenk, M. Hassaballah, and J.-F. Brethé, "Human-aware robot navigation in logistics warehouses.," in *ICINCO (2)*, pp. 371–378, 2019.

[88] A. Blaga, C. Militaru, A.-D. Mezei, and L. Tamas, "Augmented reality integration into mes for connected workers," *Robotics and Computer-Integrated Manufacturing*, vol. 68, p. 102057.

[89] J. Berg, A. Lottermoser, C. Richter, and G. Reinhart, "Human-robot-interaction for mobile industrial robot teams," *Procedia CIRP*, vol. 79, pp. 614–619, 2019.

[90] D. Chen, J. He, G. Chen, X. Yu, M. He, Y. Yang, J. Li, and X. Zhou, "Human-robot skill transfer systems for mobile robot based on multi sensor fusion," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 1354–1359, IEEE, 2020.

[91] C. Röhrig and D. Heß, "Omniman: A mobile assistive robot for intralogistics applications.," *Engineering Letters*, vol. 27, no. 4, 2019.

[92] C. Röhrig, D. Heß, C. Röhrig, D. Heß, C. Röhrig, D. Heß, J. Bleja, U. Grossmann, B. Horster, A. Roß, *et al.*, "Mobile manipulation for human-robot collaboration in intralogistics," in *IAENG Transactions on Engineering Sciences-Special Issue for the International Association of Engineers Conferences 2019*, vol. 24, pp. 459–466, World Scientific, 2019.

[93] I. ISO, "Iso 10218-1:2012-01, "robots and robotic devices - safety requirements for industrial robots - part 1: Robots (iso 10218-1:2011)," *International Organization for Standardization, Standard DIN EN ISO*, 2012.

[94] I. ISO, "ISO 3691-4:2020 - 53.060-53-ICS Industrial trucks — Safety requirements and verification — Part 4: Driverless industrial trucks and their systems," standard, International Organization for Standardization, Geneva, CH, 2020.

[95] I. ISO, "ISO 19649:2017 - Mobile robots — Vocabulary," standard, International Organization for Standardization, Geneva, CH, 2017.

[96] "ISO 19649:2017 updates." Available online: https://www.iso.org/standard/65658.html (accessed: April 2023).

[97] A. Cofield, Z. El-Shair, and S. A. Rawashdeh, "A humanoid robot object perception approach using depth images," in *2019 IEEE National Aerospace and Electronics Conference (NAECON)*, pp. 437–442, IEEE, 2019.

[98] P. A. Lasota, T. Fong, J. A. Shah, *et al.*, *A survey of methods for safe human-robot interaction*. Now Publishers, 2017.

[99] J. Saenz, C. Vogel, F. Penzlin, and N. Elkmann, "Safeguarding collaborative mobile manipulators-evaluation of the valeri workspace monitoring system," *Procedia Manufacturing*, vol. 11, pp. 47–54, 2017.

[100] M. Diab, M. Pomarlan, D. Beßler, A. Akbari, J. Rosell, J. Bateman, and M. Beetz, "Skillman—a skill-based robotic manipulation framework based on perception and reasoning," *Robotics and Autonomous Systems*, vol. 134, p. 103653, 2020.

[101] G. H. Lim, E. Pedrosa, F. Amaral, N. Lau, A. Pereira, P. Dias, J. L. Azevedo, B. Cunha, and L. P. Reis, "Rich and robust human-robot interaction on gesture recognition for assembly tasks," in *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pp. 159–164, IEEE, 2017.

[102] G. H. Lim, E. Pedrosa, F. Amaral, R. Dias, A. Pereira, N. Lau, J. L. Azevedo, B. Cunha, and L. P. Reis, "Human-robot collaboration and safety management for logistics and manipulation tasks," in *Iberian Robotics conference*, pp. 15–27, Springer, 2017.

[103] N. Kousi, G. Michalos, S. Aivaliotis, and S. Makris, "An outlook on future assembly systems introducing robotic mobile dual arm workers," *Procedia CIRP*, vol. 72, pp. 33–38, 2018.

[104] A. Schlotzhauer, L. Kaiser, and M. Brandstötter, "Safety of industrial applications with sensitive mobile manipulators–hazards and related safety measures," in *Austrian Robotics Workshop 2018*, p. 43, 2018.

[105] H. Karami, K. Darvish, and F. Mastrogiovanni, "A task allocation approach for human-robot collaboration in product defects inspection scenarios," in *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 1127–1134, IEEE, 2020.

[106] K. Darvish, B. Bruno, E. Simetti, F. Mastrogiovanni, and G. Casalino, "Interleaved online task planning, simulation, task allocation and motion control for flexible human-robot cooperation," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 58–65, IEEE, 2018.

[107] M. Chen, C. Liu, and G. Du, "A human–robot interface for mobile manipulator," *Intelligent Service Robotics*, vol. 11, no. 3, pp. 269–278, 2018.

[108] G. A. Al, P. Estrela, and U. Martinez-Hernandez, "Towards an intuitive human-robot interaction based on hand gesture recognition and proximity sensors," in *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pp. 330–335, IEEE, 2020.

[109] W. Kim, P. Balatti, E. Lamon, and A. Ajoudani, "Moca-man: A mobile and reconfigurable collaborative robot assistant for conjoined human-robot actions," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10191–10197, IEEE, 2020.

[110] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.

[111] M. Indri, A. Grau, and M. Ruderman, "Guest Editorial Special Section on Recent Trends and Developments in Industry 4.0 Motivated Robotic Solutions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1677–1680, 2018.

[112] SPARC, "Robotics 2020 Multi-Annual Roadmap." Available online: https://www.eu-robotics.net/cms/upload/topic_groups/H2020_Robotics_Multi-Annual_Roadmap_ICT-2017B.pdf (accessed: April 2023).

[113] P. J. Mattaboni, "Autonomous mobile robot," Jan. 20 1987. US Patent 4,638,445.

[114] "IFR Executive Summary World Robotics 2018 Service Robots." Available online: https://ifr.org/downloads/press2018/Executive_Summary_WR_Service_Robots_2018.pdf (accessed: April 2023).

[115] IFR - International Federation of Robotics, "IFR World Robotics 2020 Service Robots Report Presentation." Available online: https://ifr.org/downloads/press2018/Presentation_WR_2020.pdf (accessed: April 2023).

[116] "New Robotics and Drones 2018-2038: Technologies, Forecasts, Players." Available online: https://www.idtechex.com/research/reports/new-robotics-and-drones-2018-2038-technologies-forecasts-players-000584.asp (accessed: April 2023).

[117] "Mobile Robotics in Logistics, Warehousing and Delivery 2022-2042." Available online: https://www.idtechex.com/en/research-report/mobile-robotics-in-logistics-warehousing-and-delivery-2022-2042/855 (accessed: April 2023).

[118] M. Bader, A. Richtsfeld, M. Suchi, G. Todoran, W. Holl, and M. Vincze, "Balancing centralised control with vehicle autonomy in AGV systems for industrial acceptance," in *11th Int. Conf. on Autonomic and Autonom. Sys*, 2015.

[119] G. Fragapane, D. Ivanov, M. Peron, F. Sgarbossa, and J. O. Strandhagen, "Increasing flexibility and productivity in Industry 4.0 production networks with autonomous mobile robots and smart intralogistics," *Annals of Operations Research*, pp. 1–19, 2020.

[120] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[121] J. Yao, C. Lin, X. Xie, A. J. Wang, and C.-C. Hung, "Path planning for virtual human motion using improved A* star algorithm," in *2010 Seventh international conference on information technology: new generations*, pp. 1154–1158, IEEE, 2010.

[122] M. Korkmaz and A. Durdu, "Comparison of optimal path planning algorithms," in *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, pp. 255–258, IEEE, 2018.

[123] S. Koenig and M. Likhachev, "Dˆ* Lite," *Aaai/iaai*, vol. 15, 2002.

[124] D. Ferguson, M. Likhachev, and A. Stentz, "A guide to heuristic-based path planning," in *Proceedings of the international workshop on planning under uncertainty for autonomous systems, international conference on automated planning and scheduling (ICAPS)*, pp. 9–18, 2005.

[125] L. Kavraki and J.-C. Latombe, "Randomized preprocessing of configuration for fast path planning," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 2138–2145, IEEE, 1994.

[126] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms In MATLAB® Second, Completely Revised*, vol. 118. Springer, 2017.

[127] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[128] J. Kuffner and S. LaValle, "An efficient approach to path planning using balanced bidirectional RRT search," *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep*, 2005.

[129] E. Taheri, M. H. Ferdowsi, and M. Danesh, "Fuzzy Greedy RRT Path Planning Algorithm in a Complex Configuration Space," *International Journal of Control, Automation and Systems*, vol. 16, no. 6, pp. 3026–3035, 2018.

[130] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[131] Z. Tahir, A. H. Qureshi, Y. Ayaz, and R. Nawaz, "Potentially guided bidirectionalized RRT* for fast optimal path planning in cluttered environments," *Robotics and Autonomous Systems*, vol. 108, pp. 13–27, 2018.

[132] J. Tu and S. X. Yang, "Genetic algorithm based path planning for a mobile robot," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 1, pp. 1221–1226, IEEE, 2003.

[133] B. Hernández and E. Giraldo, "A Review of Path Planning and Control for Autonomous Robots," in *2018 IEEE 2nd Colombian Conference on Robotics and Automation (CCRA)*, pp. 1–6, IEEE, 2018.

[134] G. Antonelli, S. Chiaverini, and G. Fusco, "A fuzzy-logic-based approach for mobile robot path tracking," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 2, pp. 211–221, 2007.

[135] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.

[136] C. Possieri and M. Sassano, "Patrolling and collision avoidance beyond classical navigation functions," in *European Control Conference (ECC)*, pp. 1821–1826, IEEE, 2018.

[137] C. Possieri and M. Sassano, "Motion Planning, Formation Control and Obstacle Avoidance for Multi-Agent Systems," in *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 879–884, IEEE, 2018.

[138] C. Possieri and A. Tornambè, "On polynomial vector fields having a given affine variety as attractive and invariant set: application to robotics," *International Journal of Control*, vol. 88, no. 5, pp. 1001–1025, 2015.

[139] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, 2009.

[140] "ROS Navigation Stack Official Documentation." accessed: April 2023.

[141] "MATLAB Robotics System Toolbox ROS Interface Documentation." accessed: April 2023.

[142] "ROS actionlib Official Documentation." Available online: http://wiki.ros.org/actionlib (accessed: April 2023).

[143] S. Baldi, N. Maric, R. Dornberger, and T. Hanne, "Pathfinding Optimization when Solving the Paparazzi Problem Comparing A* and Dijkstra's Algorithm," in *2018 6th International Symposium on Computational and Business Intelligence (ISCBI)*, pp. 16–22, IEEE, 2018.

[144] M. Inc, *Pioneer 3 Operations Manual*.

[145] S. A. Waldkirch, *PLMS200/211/221/291 Laser Measurement Systems*.

[146] "Raspberry Pi site." accessed: April 2023.

[147] "ROS gmapping Official Documentation." accessed: April 2023.

[148] G. Grisetti, C. Stachniss, W. Burgard, *et al.*, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, p. 34, 2007.

[149] "ROS rviz Official Documentation." accessed: April 2023.

[150] B. P. Gerkey and K. Konolige, "Planning and control in unstructured terrain," in *ICRA Workshop on Path Planning on Costmaps*, 2008.

[151] "Test Cases with the Pioneer3DX." accessed: April 2023.

[152] J. Yu and S. M. LaValle, "Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1163–1177, 2016.

[153] J.Yu, "Average case constant factor time and distance optimal multi-robot path planning in well-connected environments," *Autonomous Robots*, vol. 44, pp. 469–483, 2020.

[154] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, 1992.

[155] M. C. De Gennaro and A. Jadbabaie, "Formation control for a cooperative multi-agent system using decentralized navigation functions," in *2006 American Control Conference*, (Minneapolis, MN, USA), 2006.

[156] H. G. Tanner and A. Boddu, "Multiagent navigation functions revisited," *IEEE Transactions on Robotics*, vol. 28, no. 6, pp. 1346–1359, 2012.

[157] K. K. Oh, M. C. Park, and H. S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.

[158] C. Possieri and M. Sassano, "Patrolling and collision avoidance beyond classical navigation functions," in *European Control Conference*, (Limassol, Cyprus), pp. 1821–1826, 2018.

[159] B. Siciliano and O. Khatib, *Handbook of Robotics*. Springer Verlag, 2008.

[160] F. Martinelli, C. Possieri, and A. Tornambe, "Motion planning for a unicycle-like mobile robot, using algebraic attractive curves," in *22nd Mediterranean Conference on Control and Automation*, (Palermo, Italy), pp. 1335–1340, IEEE, 2014.

[161] T. Mylvaganam, M. Sassano, and A. Astolfi, "A differential game approach to multi-agent collision avoidance," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 4229–4235, 2017.

[162] H. K. Khalil, *Nonlinear Systems*. New Jersey: Prentice-Hall, 2002.

[163] T. Mylvaganam, C. Possieri, and M. Sassano, "Global stabilization of nonlinear systems via hybrid implementation of dynamic continuous-time local controllers," *Automatica*, vol. 106, pp. 401–405, 2019.

[164] S. Wilson, P. Glotfelter, L. Wang, S. Mayya, G. Notomista, M. Mote, and M. Egerstedt, "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems," *IEEE Control Systems Magazine*, vol. 40, no. 1, pp. 26–44, 2020.

[165] Experiment 2 Demo Video. Available online: https://youtu.be/Bc43Eh8PTf0 (accessed: April 2023).

[166] M. De Rose, "LiDAR-based Dynamic Path Planning of a mobile robot adopting a costmap layer approach in ROS2," Master's thesis, Politecnico di Torino, 2021.

[167] F. Rubio, F. Valero, and C. Llopis-Albert, "A review of mobile robots: Concepts, methods, theoretical framework, and applications," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, p. 172988141983959, 2019.

[168] S. M. LaValle, *Planning Algorithms*, ch. Introductory Material. USA: Cambridge University Press, 2006.

[169] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, "A review of motion planning for highway autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, pp. 1826–1848, May 2020.

[170] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[171] S. Quinlan and O. Khatib, "Elastic bands: connecting path planning and control," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, pp. 802–807 vol.2, IEEE Comput. Soc. Press, 1993.

[172] C. Rösmann, F. Hoffmann, and T. Bertram, "Integrated online trajectory planning and optimization in distinctive topologies," *Robotics and Autonomous Systems*, vol. 88, pp. 142–153, 2017.

[173] B. Cybulski, A. Wegierska, and G. Granosik, "Accuracy comparison of navigation local planners on ROS-based mobile robot," in *2019 12th International Workshop on Robot Motion and Control (RoMoCo)*, pp. 104–111, IEEE, 2019.

[174] I. Naotunna and T. Wongratanaphisan, "Comparison of ROS Local Planners with Differential Drive Heavy Robotic System," in *2020 International Conference on Advanced Mechatronic Systems (ICAMechS)*, pp. 1–6, IEEE, 2020.

[175] "Navigation Dynamic Obstacle Integration." Student Program challenge launched in 2021 by the Nav2 community. Accessed: April 2023.

[176] I. H. Savci, A. Yilmaz, S. Karaman, H. Ocakli, and H. Temeltas, "Improving Navigation Stack of a ROS-Enabled Industrial Autonomous Mobile Robot (AMR) to be Incorporated in a Large-Scale Automotive Production," *The International Journal of Advanced Manufacturing Technology*, pp. 1–22, 2022.

[177] F. Martín, J. G. Clavero, V. Matellán, and F. J. Rodríguez, "Plansys2: A planning system framework for ROS2," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9742–9749, IEEE, 2021.

[178] H. Dong, C.-Y. Weng, C. Guo, H. Yu, and I.-M. Chen, "Real-Time Avoidance Strategy of Dynamic Obstacles via Half Model-Free Detection and Tracking With 2D Lidar for Mobile Robots," *IEEE/ASME transactions on mechatronics*, vol. 26, no. 4, pp. 2215–2225, 2021.

[179] M. Przybyła, "Detection and tracking of 2D geometric obstacles from LRF data," in *2017 11th International Workshop on Robot Motion and Control (RoMoCo)*, pp. 135–141, 2017.

[180] T. Mori, T. Sato, H. Noguchi, M. Shimosaka, R. Fukui, and T. Sato, "Moving objects detection and classification based on trajectories of LRF scan data on a grid map," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2606–2611, 2010.

[181] F. Albers, C. Rösmann, F. Hoffmann, and T. Bertram, *Online Trajectory Optimization and Navigation in Dynamic Environments in ROS*, pp. 241–274. Springer, 01 2019.

[182] A. Pierson, W. Schwarting, S. Karaman, and D. Rus, "Navigating congested environments with risk level sets," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5712–5719, 2018.

[183] "Blob Detection Using OpenCV." Accessed: April 2023.

[184] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[185] M. Colledanchise and P. Ögren, *Behavior Trees in Robotics and AI: An introduction*. CRC Press, Jul 2018.

[186] D. V. Lu, D. Hershberger, and W. D. Smart, "Layered costmaps for context-sensitive navigation," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 709–715, IEEE, 2014.

[187] "DOL: A costmap-based Dynamic Path Planning approach in ROS2." Accessed: April 2023.

[188] "AMRs vs. AGVs: The Difference Between a Robot and a Guided Vehicle." Available online: https://fetchrobotics.com/fetch-robotics-blog/amrs-vs-agvs-whats-the-difference/ (accessed: April 2023).

[189] A. Owen-Hill, "Mobile Collaborative Robots: The Next Big Thing." Available online: https://blog.robotiq.com/mobile-collaborative-robots-the-next-big-thing (accessed: April 2023).

[190] "4 types of autonomous mobile robots, and their warehouse use cases." Available online: https://www.supplychaindive.com/news/4-types-of-autonomous-mobile-robots-and-their-warehouse-use-cases/529548/ (accessed: April 2023).

[191] "MiR Official Site." Available online: https://www.mobile-industrial-robots.com/it/ (accessed: April 2023).

[192] "Robotnik Official Site." Available online: https://www.robotnik.eu/ (accessed: April 2023).

[193] "fetch robotics Official Site." Available online: https://fetchrobotics.com/ (accessed: April 2023).

[194] A. ANSI, "ANSI/ITSDF B56.5-2019, Safety Standard for Driverless, Automatic Guided Industrial Vehicles and Automated Functions of Manned Industrial Vehicles (Revision of ANSI/ITSDF B56.5-2012)," standard, American National Standards Institute/Industrial Truck Standards Development Foundation, New York (NY), 2019.

[195] I. ISO, "ANSI/RIA R15.08-1-2020 – Industrial Mobile Robots - Safety Requirements - Part 1: Requirements For The Industrial Mobile Robot," standard, American National Standards Institute/Robotic Industries Association, Geneva, CH, 2020.

[196] A. G. C. Gonzalez, M. V. S. Alves, G. S. Viana, L. K. Carvalho, and J. C. Basilio, "Supervisory Control-Based Navigation Architecture: A New Framework for Autonomous Robots in Industry 4.0 Environments," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1732–1743, 2018.

[197] Y. Zhang, Z. Zhu, and J. Lv, "CPS-Based Smart Control Model for Shopfloor Material Handling," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1764–1775, 2018.

[198] L. Sabattini, M. Aikio, P. Beinschob, M. Boehning, E. Cardarelli, V. Digani, A. Krengel, M. Magnani, S. Mandici, F. Oleari, C. Reinke, D. Ronzoni, C. Stimming, R. Varga, A. Vatavu, S. C. Lopez, C. Fantuzzi, A. Mayra, S. Nedevschi, C. Secchi, and K. Fuerstenberg, "The PAN-Robots Project: Advanced Automated Guided Vehicle Systems for Industrial Logistics," *IEEE Robotics Automation Magazine*, vol. 25, pp. 55–64, March 2018.

[199] P. Beinschob, M. Meyer, C. Reinke, V. Digani, C. Secchi, and L. Sabattini, "Semi-automated map creation for fast deployment of AGV fleets in modern logistics," *Robotics and Autonomous Systems*, vol. 87, pp. 281–295, 2017.

[200] V. Digani, F. Caramaschi, L. Sabattini, C. Secchi, and C. Fantuzzi, "Obstacle avoidance for industrial AGVs," in *2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pp. 227–232, Sept 2014.

[201] D. Fox, "Adapting the sample size in particle filters through KLD-sampling," *The international Journal of robotics research*, vol. 22, no. 12, pp. 985–1003, 2003.

[202] A. Mahtani, L. Sanchez, E. Fernandez, and A. Martinez, *Effective Robotics Programming with ROS - Third Edition*. Packt Publishing, 3rd ed., 2016.

[203] "ONVIF Official Site." Available online: https://www.onvif.org/.

[204] "GStreamer official webite." Available online: https://gstreamer.freedesktop.org/.

[205] "gscam ROS tool." Available online: http://wiki.ros.org/gscam (accessed: April 2023).

[206] S. S. H. Hajjaj and K. S. M. Sahari, "Establishing remote networks for ROS applications via Port Forwarding: A detailed tutorial," *International Journal of Advanced Robotic Systems*, vol. 14, no. 3, 2017.

[207] D. Tardiolo, R. Parasuraman, and P. Ögren, "Pound: a ROS node for reducing delay and jitter in wireless multi-robot networks," *arXiv preprint arXiv:1707.07540*, 2017.

[208] "ROS 2 documentation." Available online: https://github.com/ros2/ros2/wiki (accessed: April 2023).

[209] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE Industrial Electronics Magazine*, 2017.

[210] R. Candell, M. T. Hany, K. B. Lee, Y. Liu, J. T. Quimby, and C. A. Remley, "Guide to Industrial Wireless Systems Deployments," tech. rep., 2018.

[211] R. Candell Jr and R. Candell, *Industrial Wireless Systems Workshop Proceedings*. US Department of Commerce, National Institute of Standards and Technology, 2017.

[212] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[213] "COCO Dataset." Available online: http://cocodataset.org/ (accessed: April 2023).

[214] J. Redmon, "Darknet: Open Source Neural Networks in C." Available online: http://pjreddie.com/darknet/ (accessed: April 2023).

[215] A. Liaqat, W. Hutabarat, D. Tiwari, L. Tinkler, D. Harra, B. Morgan, A. Taylor, T. Lu, and A. Tiwari, "Autonomous mobile robots in manufacturing: Highway Code development, simulation, and testing," *The International Journal of Advanced Manufacturing Technology*, vol. 104, no. 9-12, pp. 4617–4628, 2019.

[216] R. Inam, K. Raizer, A. Hata, R. Souza, E. Forsman, E. Cao, and S. Wang, "Risk assessment for human-robot collaboration in an automated warehouse scenario," in *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, pp. 743–751, IEEE, 2018.

[217] J. Blankenburg, S. B. Banisetty, S. P. H. Alinodehi, L. Fraser, D. Feil-Seifer, M. Nicolescu, and M. Nicolescu, "A distributed control architecture for collaborative multi-robot task allocation," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pp. 585–592, IEEE, 2017.

[218] E. Adamey, A. E. Oğuz, and Ü. Özgüner, "Collaborative multi-msa multi-target tracking and surveillance: A divide & conquer method using region allocation trees," *Journal of Intelligent & Robotic Systems*, vol. 87, no. 3-4, pp. 471–485, 2017.

[219] G. Demesure, D. Trentesaux, M. Defoort, A. Bekrar, H. Bril, M. Djemaï, and A. Thomas, "Smartness versus embeddability: a tradeoff for the deployment of smart AGVs in industry," in *Service Orientation in Holonic and Multi-Agent Manufacturing*, pp. 395–406, Springer, 2018.

[220] Q. Liu, P. Hua, A. Sultan, L. Shen, E. Mueller, and F. Boerner, "Study of the integration of robot in cyber-physical production systems," in *2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 367–370, IEEE, 2019.

[221] Y. Lu, "Industry 4.0: A survey on technologies, applications and open research issues," *Journal of Industrial Information Integration*, vol. 6, pp. 1–10, 2017.

[222] H. A. Poonawala and M. W. Spong, "Cooperative visibility maintenance in se (3) for multi-robot-networks with limited field-of-view sensors," *Control Theory and Technology*, vol. 15, no. 4, pp. 246–257, 2017.

[223] M. Geng, S. Liu, and Z. Wu, "Sensor fusion-based cooperative trail following for autonomous multi-robot system," *Sensors*, vol. 19, no. 4, p. 823, 2019.

[224] J. Zhang, R. Liu, K. Yin, Z. Wang, M. Gui, and S. Chen, "Intelligent collaborative localization among air-ground robots for industrial environment perception," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 12, pp. 9673–9681, 2018.

[225] J. Al Hage, M. E. El Najjar, and D. Pomorski, "Multi-sensor fusion approach with fault detection and exclusion based on the kullback–leibler divergence: Application on collaborative multi-robot system," *Information Fusion*, vol. 37, pp. 61–76, 2017.

[226] N. Sullivan, S. Grainger, and B. Cazzolato, "Analysis of cooperative localisation performance under varying sensor qualities and communication rates," *Robotics and Autonomous Systems*, vol. 110, pp. 73–84, 2018.

[227] L. Yin, Q. Ni, and Z. Deng, "Intelligent multisensor cooperative localization under cooperative redundancy validation," *IEEE transactions on cybernetics*, 2019.

[228] J. Yuan, J. Zhang, S. Ding, and X. Dong, "Cooperative localization for disconnected sensor networks and a mobile robot in friendly environments," *Information Fusion*, vol. 37, pp. 22–36, 2017.

[229] A. Khalid, P. Kirisci, Z. Ghrairi, K. Thoben, and J. Pannek, "Towards implementing safety and security concepts for human-robot collaboration in the context of Industry 4.0," in *39th International MATADOR Conference on Advanced Manufacturing (Manchester, UK)*, pp. 0–7, 2017.

[230] L. Gumbi and H. Twinomurinzi, "SMME Readiness for Smart Manufacturing (4IR) Adoption: A Systematic Review," in *Conference on e-Business, e-Services and e-Society*, pp. 41–54, Springer, 2020.

[231] A. Markis, M. Papa, D. Kaselautzke, M. Rathmair, V. Sattinger, and M. Brandstötter, "Safety of mobile robot systems in industrial applications," 05 2019.

[232] Y. Rizk, M. Awad, and E. W. Tunstel, "Cooperative heterogeneous multi-robot systems: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 2, pp. 1–31, 2019.

[233] "SPARC. Robotics 2020 Multi-Annual Roadmap." accessed: April 2023.

[234] A. Bonci, M. Pirani, and S. Longhi, "Robotics 4.0: Performance improvement made easy," in *22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2017)*, 2017.

[235] "Meta-Sensor AMRs - DEMO." accessed: April 2023.

[236] "Gazebo Official site." accessed: April 2023.

[237] X. Wu, J. Ren, Y. Wu, and J. Shao, "Study on target tracking based on vision and radar sensor fusion," tech. rep., SAE Technical Paper, 2018.

[238] Z. Yan, L. Sun, T. Duckctr, and N. Bellotto, "Multisensor online transfer learning for 3d lidar-based human detection with a mobile robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7635–7640, IEEE, 2018.

[239] V. De Silva, J. Roche, and A. Kondoz, "Robust fusion of LiDAR and wide-angle camera data for autonomous mobile robots," *Sensors*, vol. 18, no. 8, 2730, 2018.

[240] T. Linder, D. Griesser, N. Vaskevicius, and K. O. Arras, "Towards Accurate 3D Person Detection and Localization from RGB-D in Cluttered Environments," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Workshop on Robotics for Logistics in Warehouses and Environments Shared with Humans*, 2018.

[241] L. Spinello and R. Siegwart, "Human detection using multimodal and multidimensional features," in *2008 IEEE International Conference on Robotics and Automation*, pp. 3264–3269, IEEE, 2008.

[242] A. Leigh, J. Pineau, N. Olmedo, and H. Zhang, "Person tracking and following with 2D laser scanners," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 726–733, IEEE, 2015.

[243] V. D. Silva, J. Roche, and A. M. Kondoz, "Fusion of LiDAR and Camera Sensor Data for Environment Sensing in Driverless Vehicles," *CoRR*, vol. abs/1710.06230, 2017.

[244] K. Ahmad Yousef, B. Mohd, K. Al-Widyan, and T. Hayajneh, "Extrinsic calibration of camera and 2D laser sensors without overlap," *Sensors*, vol. 17, no. 10, 2346, 2017.

[245] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder (improves camera calibration)," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3, pp. 2301–2306, IEEE, 2004.

[246] Z. Hu, Y. Li, N. Li, and B. Zhao, "Extrinsic calibration of 2-D laser rangefinder and camera from single shot based on minimal solution," *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 4, pp. 915–929, 2016.

[247] J. Li, X. He, and J. Li, "2D LiDAR and camera fusion in 3D modeling of indoor environment," in *2015 National Aerospace and Electronics Conference (NAECON)*, pp. 379–383, IEEE, 2015.

[248] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, 2000.

[249] J. Heikkila, O. Silven, *et al.*, "A four-step camera calibration procedure with implicit image correction," in *cvpr*, vol. 97, p. 1106, 1997.

[250] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *arXiv preprint arXiv:1809.02165*, 2018.

[251] G. Marcus, "Deep Learning: A Critical Appraisal," *CoRR*, vol. abs/1801.00631, 2018.

[252] J. Han, D. Zhang, G. Cheng, N. Liu, and D. Xu, "Advanced deep-learning techniques for salient and category-specific object detection: a survey," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 84–100, 2018.

[253] "Docker official website." Available online: https://www.docker.com.

[254] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *CoRR*, vol. abs/1804.02767, 2018.

[255] P. Marin-Plaza, A. Hussein, D. Martin, and A. d. l. Escalera, "Global and local path planning study in a ROS-based research platform for autonomous vehicles," *Journal of Advanced Transportation*, vol. 2018, 2018.

[256] "Human detection and avoidance with the Pioneer 3DX."

[257] Z. Juan, L. Wei, and P. Xiamei, "Research on technology transfer readiness level and its application in university technology innovation management," in *2010 International Conference on E-Business and E-Government*, pp. 1904–1907, IEEE, 2010.

[258] M. Chae, H. Lee, and K. Lee, "A performance comparison of linux containers and virtual machines using Docker and KVM," *Cluster Computing*, vol. 22, no. 1, pp. 1765–1775, 2019.

[259] "Online supervised global planning algorithm for AMRs with human obstacle avoidance."

[260] A. Zacharaki, I. Kostavelis, A. Gasteratos, and I. Dokas, "Safety bounds in human robot interaction: A survey," *Safety science*, vol. 127, p. 104667, 2020.

[261] "Zbar ROS Node Documentation page." Available online: http://wiki.ros.org/zbar_ros (accessed: April 2023).

[262] S. A. Ambrose, M. W. Bridges, M. DiPietro, M. C. Lovett, and M. K. Norman, *How learning works: Seven research-based principles for smart teaching*. John Wiley & Sons, 2010.

[263] J. Dewey, *My pedagogic creed*. No. 25, EL Kellogg & Company, 1897.

[264] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, pp. 115–133, 1943.

[265] Z. Liu, Q. Liu, W. Xu, L. Wang, and Z. Zhou, "Robot learning towards smart robotic manufacturing: A review," *Robotics and Computer-Integrated Manufacturing*, vol. 77, p. 102360, 2022.

[266] G. L. Tortorella, G. Narayanamurthy, and M. Thurer, "Identifying pathways to a high-performing lean automation implementation: An empirical study in the manufacturing industry," *International Journal of Production Economics*, vol. 231, p. 107918, 2021.

[267] I. P. Vlachos, R. M. Pascazzi, G. Zobolas, P. Repoussis, and M. Giannakis, "Lean manufacturing systems in the area of industry 4.0: a lean automation plan of agvs/iot integration," *Production Planning & Control*, pp. 1–14, 2021.

[268] P. K. R. Maddikunta, Q.-V. Pham, B. Prabadevi, N. Deepa, K. Dev, T. R. Gadekallu, R. Ruby, and M. Liyanage, "Industry 5.0: A survey on enabling technologies and potential applications," *Journal of Industrial Information Integration*, vol. 26, p. 100257, 2022.

[269] S. Li, R. Wang, P. Zheng, and L. Wang, "Towards proactive human–robot collaboration: A foreseeable cognitive manufacturing paradigm," *Journal of Manufacturing Systems*, vol. 60, pp. 547–552, 2021.

[270] A. Karthikeyan and U. D. Priyakumar, "Artificial intelligence: machine learning for chemical sciences," *Journal of Chemical Sciences*, vol. 134, pp. 1–20, 2022.

[271] D. Mukherjee, K. Gupta, L. H. Chang, and H. Najjaran, "A survey of robot learning strategies for human-robot collaboration in industrial settings," *Robotics and Computer-Integrated Manufacturing*, vol. 73, p. 102231, 2022.

[272] Y. Zhang, C. Qian, J. Lv, and Y. Liu, "Agent and cyber-physical system based self-organizing and self-adaptive intelligent shopfloor," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 737–747, 2016.

[273] P. Leitao, S. Karnouskos, L. Ribeiro, J. Lee, T. Strasser, and A. W. Colombo, "Smart agents in industrial cyber–physical systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1086–1101, 2016.

[274] C. Cristalli, S. Boria, D. Massa, L. Lattanzi, and E. Concettoni, "A cyber-physical system approach for the design of a modular smart robotic cell," in *IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society*, pp. 4845–4850, IEEE, 2016.

[275] L. Bochmann, T. Bänziger, A. Kunz, and K. Wegener, "Human-robot collaboration in decentralized manufacturing systems: An approach for simulation-based evaluation of future intelligent production," *Procedia CIRP*, vol. 62, pp. 624–629, 2017.

[276] C. Cimini, F. Pirola, R. Pinto, and S. Cavalieri, "A human-in-the-loop manufacturing control architecture for the next generation of production systems," *Journal of manufacturing systems*, vol. 54, pp. 258–271, 2020.

[277] B. Dafflon, N. Moalla, and Y. Ouzrout, "The challenges, approaches, and used techniques of cps for manufacturing in industry 4.0: a literature review," *The International Journal of Advanced Manufacturing Technology*, pp. 1–18, 2021.

[278] F. Yan, L. Shiqi, Q. Kan, L. Xue, C. Li, and T. Jie, "Language-facilitated human-robot cooperation within a human cognitive modeling infrastructure: A case in space exploration task," in *2020 IEEE International Conference on Human-Machine Systems (ICHMS)*, pp. 1–3, IEEE, 2020.

[279] K. Tanida, M. Paolini, E. Pöppel, and S. Silveira, "Safety feelings and anticipatory control: An fmri study on safety and risk perception," *Transportation research part F: traffic psychology and behaviour*, vol. 57, pp. 108–114, 2018.

[280] L. Sun, H. An, H. Ma, and J. Gao, "Real-time human intention recognition of multi-joints based on myo," *IEEE Access*, vol. 8, pp. 4235–4243, 2019.

[281] J. Rafferty, C. D. Nugent, J. Liu, and L. Chen, "From activity recognition to intention recognition for assisted living within smart homes," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 3, pp. 368–379, 2017.

[282] J. Wang, L. Perez, *et al.*, "The effectiveness of data augmentation in image classification using deep learning," *Convolutional Neural Networks Vis. Recognit*, vol. 11, pp. 1–8, 2017.

[283] S. K. Alabugin and A. N. Sokolov, "Applying of generative adversarial networks for anomaly detection in industrial control systems," in *2020 Global Smart Industry Conference (GloSIC)*, pp. 199–203, IEEE, 2020.

[284] J. Luo, J. Huang, and H. Li, "A case study of conditional deep convolutional generative adversarial networks in machine fault diagnosis," *Journal of Intelligent Manufacturing*, vol. 32, pp. 407–425, 2021.

[285] M. Faccio, R. Minto, G. Rosati, and M. Bottin, "The influence of the product characteristics on human-robot collaboration: a model for the performance of collaborative robotic assembly," *The International Journal of Advanced Manufacturing Technology*, vol. 106, no. 5, pp. 2317–2331, 2020.

[286] S. Muhammad, "Modeling operator performance in human-in-the-loop autonomous systems," *IEEE Access*, vol. 9, pp. 102715–102731, 2021.

[287] M. A. R. Garcia, R. Rojas, L. Gualtieri, E. Rauch, and D. Matt, "A human-in-the-loop cyber-physical system for collaborative assembly in smart manufacturing," *Procedia CIRP*, vol. 81, pp. 600–605, 2019.

[288] M. A. Mabrok and A.-H. Abdel-Aty, "Pattern detection for time series trajectories in human in the loop applications," *Journal of Intelligent & Fuzzy Systems*, vol. 37, no. 1, pp. 115–123, 2019.

[289] Z. Liu, Q. Liu, W. Xu, Z. Liu, Z. Zhou, and J. Chen, "Deep learning-based human motion prediction considering context awareness for human-robot collaboration in manufacturing," *Procedia CIRP*, vol. 83, pp. 272–278, 2019.

[290] Q. Li, Z. Zhang, Y. You, Y. Mu, and C. Feng, "Data driven models for human motion prediction in human-robot collaboration," *IEEE Access*, vol. 8, pp. 227690–227702, 2020.

[291] A. P. Dani, I. Salehi, G. Rotithor, D. Trombetta, and H. Ravichandar, "Human-in-the-loop robot control for human-robot collaboration: Human intention estimation and safe trajectory tracking control for collaborative tasks," *IEEE Control Systems Magazine*, vol. 40, no. 6, pp. 29–56, 2020.

[292] M. A. Mabrok, H. K. Mohamed, A.-H. Abdel-Aty, and A. S. Alzahrani, "Human models in human-in-the-loop control systems," *Journal of Intelligent & Fuzzy Systems*, vol. 38, no. 3, pp. 2611–2622, 2020.

[293] H. Bozorgi, X. T. Truong, H. M. La, and T. D. Ngo, "2D laser and 3D camera data integration and filtering for human trajectory tracking," in *2021 IEEE/SICE International Symposium on System Integration (SII)*, pp. 634–639, IEEE, 2021.

[294] A. Rudenko, T. P. Kucner, C. S. Swaminathan, R. T. Chadalavada, K. O. Arras, and A. J. Lilienthal, "Thör: Human-robot navigation data collection and accurate motion trajectories dataset," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 676–682, 2020.

[295] P. Schydlo, M. Rakovic, L. Jamone, and J. Santos-Victor, "Anticipation in human-robot cooperation: A recurrent neural network approach for multiple action sequences prediction," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5909–5914, IEEE, 2018.

[296] P. A. Lasota and J. A. Shah, "A multiple-predictor approach to human motion prediction," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2300–2307, 2017.

[297] V. V. Unhelkar, P. A. Lasota, Q. Tyroller, R.-D. Buhai, L. Marceau, B. Deml, and J. A. Shah, "Human-aware robotic assistant for collaborative assembly: Integrating human motion prediction with planning in time," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2394–2401, 2018.

[298] H. Liu and L. Wang, "Human motion prediction for human-robot collaboration," *Journal of Manufacturing Systems*, vol. 44, pp. 287–294, 2017.

[299] M. Cramer, J. Cramer, K. Kellens, and E. Demeester, "Towards robust intention estimation based on object affordance enabling natural human-robot collaboration in assembly tasks," *Procedia CIRP*, vol. 78, pp. 255–260, 2018.

[300] P. Wang, H. Liu, L. Wang, and R. X. Gao, "Deep learning-based human motion recognition for predictive context-aware human-robot collaboration," *CIRP annals*, vol. 67, no. 1, pp. 17–20, 2018.

[301] D. Merkel, "Docker: lightweight Linux containers for consistent development and deployment," *Linux journal*, vol. 2014, no. 239, p. 2, 2014.

[302] J. Howard and S. Gugger, "Fastai: a layered API for deep learning," *Information*, vol. 11, no. 2, p. 108, 2020.

[303] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, C. Willing, and Jupyter development team, "Jupyter notebooks – a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pp. 87–90, IOS Press, 2016.

[304] A. Tabb, "Docker fastai repository." Available online: https://github.com/amy-tabb/fastai-docker-example (accessed: April 2023).

[305] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[306] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *arXiv preprint arXiv:1710.09412*, 2017.

[307] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," in *Artificial intelligence and machine learning for multi-domain operations applications*, vol. 11006, p. 1100612, International Society for Optics and Photonics, 2019.

[308] Z. Zhou, L. Li, A. Fürsterling, H. J. Durocher, J. Mouridsen, and X. Zhang, "Learning-based object detection and localization for a mobile robot manipulator in sme production," *Robotics and Computer-Integrated Manufacturing*, vol. 73, p. 102229, 2022.

[309] X. Huang, M. Halwani, R. Muthusamy, A. Ayyad, D. Swart, L. Seneviratne, D. Gan, and Y. Zweiri, "Real-time grasping strategies using event camera," *Journal of Intelligent Manufacturing*, pp. 1–23, 2022.

[310] A. Sarkar, A. Banerjee, P. K. Singh, and R. Sarkar, "3D Human Action Recognition: Through the eyes of researchers," *Expert Systems with Applications*, p. 116424, 2022.

[311] A. Núñez-Marcos, G. Azkune, and I. Arganda-Carreras, "Egocentric vision-based action recognition: A survey," *Neurocomputing*, vol. 472, pp. 175–197, 2022.

[312] M. Al-Amin, R. Qin, W. Tao, D. Doell, R. Lingard, Z. Yin, and M. C. Leu, "Fusing and refining convolutional neural network models for assembly action recognition in smart manufacturing," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 236, no. 4, pp. 2046–2059, 2022.

[313] S. Al-Hussaini, S. Thakar, H. Kim, P. Rajendran, B. C. Shah, J. A. Marvel, and S. K. Gupta, "Human-supervised semi-autonomous mobile manipulators for safely and efficiently executing machine tending tasks," *arXiv preprint arXiv:2010.04899*, 2020.

[314] M. Hassanin, S. Khan, and M. Tahtali, "Visual affordance and function understanding: A survey," *ACM Computing Surveys (CSUR)*, vol. 54, no. 3, pp. 1–35, 2021.

[315] F.-J. Chu, R. Xu, C. Tang, and P. A. Vela, "Recognizing object affordances to support scene reasoning for manipulation tasks," *arXiv preprint arXiv:1909.05770*, 2019.

[316] H. Sun, J. Yang, D. Li, and H. Ding, "An on-line tool path smoothing algorithm for 6R robot manipulator with geometric and dynamic constraints," *Science China Technological Sciences*, vol. 64, no. 9, pp. 1907–1919, 2021.

[317] Y. Oba, K. Weaver, A. Parwal, H. Nagasue, and M. Fujishima, "High-accuracy pose estimation method for workpiece exchange automation by a mobile manipulator," *CIRP Annals*, vol. 70, no. 1, pp. 357–360, 2021.

[318] M. Logothetis, C. P. Bechlioulis, and K. J. Kyriakopoulos, "Decentralized impedance control of mobile robotic manipulators for collaborative object handling with a human operator," in *2021 29th Mediterranean Conference on Control and Automation (MED)*, pp. 741–746, IEEE, 2021.

[319] A. Gasparetto and L. Scalera, "From the unimate to the delta robot: the early decades of industrial robotics," in *Explorations in the History and Heritage of Machines and Mechanisms: Proceedings of the 2018 HMM IFToMM Symposium on History of Machines and Mechanisms*, pp. 284–295, Springer, 2019.

[320] N. Gasteiger, M. Hellou, and H. S. Ahn, "Factors for personalization and localization to optimize human–robot interaction: A literature review," *International Journal of Social Robotics*, pp. 1–13, 2021.

[321] D. K. Jha, S. Jain, D. Romeres, W. Yerazunis, and D. Nikovski, "Generalizable human-robot collaborative assembly using imitation learning and force control," *arXiv preprint arXiv:2212.01434*, 2022.

[322] K. Krot and V. Kutia, "Intuitive methods of industrial robot programming in advanced manufacturing systems," in *International Conference on Intelligent Systems in Production Engineering and Maintenance*, pp. 205–214, Springer, 2018.

[323] B. Gleeson, K. MacLean, A. Haddadi, E. Croft, and J. Alcazar, "Gestures for industry intuitive human-robot communication from human observation," in *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 349–356, IEEE, 2013.

[324] A. Dünser, M. Lochner, U. Engelke, and D. R. Fernandez, "Visual and manual control for human-robot teleoperation," *IEEE computer graphics and applications*, vol. 35, no. 3, pp. 22–32, 2015.

[325] J. Berg and S. Lu, "Review of interfaces for industrial human-robot interaction," *Current Robotics Reports*, vol. 1, no. 2, pp. 27–34, 2020.

[326] S. Bier, R. Li, and W. Wang, "A full-dimensional robot teleoperation platform," in *2020 11th International Conference on Mechanical and Aerospace Engineering (ICMAE)*, pp. 186–191, IEEE, 2020.

[327] C. P. Quintero, R. Tatsambon, M. Gridseth, and M. Jägersand, "Visual pointing gestures for bi-directional human robot interaction in a pick-and-place task," in *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 349–354, IEEE, 2015.

[328] T. S. Kuthambalayan and S. Bera, "A review of the literature on mixed make-to-stock/make-to-order production systems: major findings and directions for future research," *International Journal of Services and Operations Management*, vol. 37, no. 3, pp. 372–406, 2020.

[329] V. Annem, P. Rajendran, S. Thakar, and S. K. Gupta, "Towards remote teleoperation of a semi-autonomous mobile manipulator system in machine tending tasks," in *International Manufacturing Science and Engineering Conference*, vol. 58745, p. V001T02A027, American Society of Mechanical Engineers, 2019.

[330] T. Stoyanov, R. Krug, A. Kiselev, D. Sun, and A. Loutfi, "Assisted telemanipulation: A stack-of-tasks approach to remote manipulator control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9, IEEE, 2018.

[331] A. Singh, S. H. Seo, Y. Hashish, M. Nakane, J. E. Young, and A. Bunt, "An interface for remote robotic manipulator control that reduces task load and fatigue," in *2013 IEEE RO-MAN*, pp. 738–743, IEEE, 2013.

[332] Y. Tsukada and T. Hoshino, "Layered touch panel: the input device with two touch panel layers," in *CHI'02 Extended Abstracts on Human Factors in Computing Systems*, pp. 584–585, 2002.

[333] "Leap motion developer website." Available online: https://developer-archive.leapmotion.com/documentation/v2/cpp/index.html (accessed: April 2023).

[334] "find-object ROS package." Available online: https://github.com/introlab/find-object (accessed: April 2023).

[335] M. Labbé, "Find-Object interface." Available online: http://introlab.github.io/find-object (accessed: April 2023).

[336] "Opencv website." Available online: https://opencv.org (accessed: April 2023).

[337] NIRYO, "Niryo one mechanical specifications," 2018.

[338] "Experimental test video demo." Available online: https://youtu.be/7HvFw0sa3Lg.

[339] F. Sibona, J. Luijkx, B. van der Heijden, L. Ferranti, and M. Indri, "EValue-Action: a proposal for policy evaluation in simulation to support interactive imitation learning," in *IEEE International Conference on Industrial Informatics (INDIN23), submitted*, 2023.

[340] G. Castañé, A. Dolgui, N. Kousi, B. Meyers, S. Thevenin, E. Vyhmeister, and P.-O. Östberg, "The ASSISTANT project: AI for high level decisions in manufacturing," *International Journal of Production Research*, pp. 1–19, 2022.

[341] D. Ramesh Kumar, S. Devadasan, and D. Elangovan, "Mapping of non-value adding activities occurring in classical manufacturing companies with lean strategies," *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering*, vol. 236, no. 3, pp. 894–906, 2022.

[342] E. Johns, "Back to reality for imitation learning," in *Conference on Robot Learning*, pp. 1764–1768, PMLR, 2022.

[343] M. Akbulut, E. Oztop, M. Y. Seker, X. Hh, A. Tekden, and E. Ugur, "Acnmp: Skill transfer and task extrapolation through learning from demonstration and reinforcement learning via representation sharing," in *Conference on Robot Learning*, pp. 1896–1907, PMLR, 2021.

[344] J. Cao, W. Liu, Y. Liu, and J. Yang, "Generalize robot learning from demonstration to variant scenarios with evolutionary policy gradient," *Frontiers in Neurorobotics*, vol. 14, p. 21, 2020.

[345] Y. Wang, C. C. Beltran-Hernandez, W. Wan, and K. Harada, "An adaptive imitation learning framework for robotic complex contact-rich insertion tasks," *Frontiers in Robotics and AI*, p. 414, 2022.

[346] M. Arduengo, A. Colomé, J. Borràs, L. Sentis, and C. Torras, "Task-adaptive robot learning from demonstration with gaussian process models under replication," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 966–973, 2021.

[347] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "Bc-z: Zero-shot task generalization with robotic imitation learning," in *Conference on Robot Learning*, pp. 991–1002, PMLR, 2022.

[348] Y. Wang, Y. Hu, S. El Zaatari, W. Li, and Y. Zhou, "Optimised learning from demonstrations for collaborative robots," *Robotics and Computer-Integrated Manufacturing*, vol. 71, p. 102169, 2021.

[349] J. Zhu, M. Gienger, and J. Kober, "Learning task-parameterized skills from few demonstrations," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4063–4070, 2022.

[350] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," *arXiv preprint arXiv:2108.03298*, 2021.

[351] T. Xue, H. Girgin, T. S. Lembono, and S. Calinon, "Guided optimal control for long-term non-prehensile planar manipulation," *arXiv preprint arXiv:2212.12814*, 2022.

[352] M. Sakr, Z. J. Li, H. M. Van der Loos, D. Kulić, and E. A. Croft, "Quantifying demonstration quality for robot learning and generalization," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9659–9666, 2022.

[353] M. Sakr, H. M. Van der Loos, D. Kulić, and E. Croft, "What makes a good demonstration for robot learning generalization?," in *Companion of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 607–609, 2021.

[354] S. Zhang, Z. Cao, D. Sadigh, and Y. Sui, "Confidence-aware imitation learning from demonstrations with varying optimality," *Advances in Neural Information Processing Systems*, vol. 34, pp. 12340–12350, 2021.

[355] L. Panchetti, J. Zheng, M. Bouri, and M. Mielle, "Team: a parameter-free algorithm to teach collaborative robots motions from user demonstrations," *arXiv preprint arXiv:2209.06940*, 2022.

[356] Y. Lee, A. Szot, S.-H. Sun, and J. J. Lim, "Generalizable imitation learning from observation via inferring goal proximity," *Advances in neural information processing systems*, vol. 34, pp. 16118–16130, 2021.

[357] X. Bian, O. Mendez, and S. Hadfield, "Generalizing to new tasks via one-shot compositional subgoals," *arXiv preprint arXiv:2205.07716*, 2022.

[358] S. Auddy, J. Hollenstein, M. Saveriano, A. Rodríguez-Sánchez, and J. Piater, "Continual learning from demonstration of robotic skills," *arXiv preprint arXiv:2202.06843*, 2022.

[359] B. Hertel and S. R. Ahmadzadeh, "Similarity-aware skill reproduction based on multi-representational learning from demonstration," in *2021 20th International Conference on Advanced Robotics (ICAR)*, pp. 652–657, IEEE, 2021.

[360] R. Burlizzi, M. Vochten, J. De Schutter, and E. Aertbeliën, "Extending extrapolation capabilities of probabilistic motion models learned from human demonstrations using shape-preserving virtual demonstrations," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10772–10779, IEEE, 2022.

[361] C. Celemin and J. Kober, "Knowledge-and ambiguity-aware robot learning from corrective and evaluative feedback," *Neural Computing and Applications*, pp. 1–19, 2023.

[362] J. Luijkx, Z. Ajanovic, L. Ferranti, and J. Kober, "Partnr: Pick and place ambiguity resolving by trustworthy interactive learning," *arXiv preprint arXiv:2211.08304*, 2022.

[363] H. Hu, X. Yang, and Y. Lou, "A robot learning from demonstration framework for skillful small parts assembly," *The International Journal of Advanced Manufacturing Technology*, vol. 119, no. 9-10, pp. 6775–6787, 2022.

[364] G. Franzese, A. Mészáros, L. Peternel, and J. Kober, "Ilosa: Interactive learning of stiffness and attractors," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7778–7785, IEEE, 2021.

[365] A. Mészáros, G. Franzese, and J. Kober, "Learning to pick at non-zero-velocity from interactive demonstrations," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6052–6059, 2022.

[366] "ILoSA Github repository." Available online: https://github.com/franzesegiovanni/ILoSA.

[367] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[368] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[369] A. D. Laud, *Theory and application of reward shaping in reinforcement learning*. University of Illinois at Urbana-Champaign, 2004.

[370] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, and S. Levine, "How to train your robot with deep reinforcement learning: lessons we have learned," *The International Journal of Robotics Research*, vol. 40, no. 4-5, pp. 698–721, 2021.

[371] C. Celemin, R. Pérez-Dattari, E. Chisari, G. Franzese, L. de Souza Rosa, R. Prakash, Z. Ajanović, M. Ferraz, A. Valada, J. Kober, *et al.*, "Interactive imitation learning in robotics: A survey," *Foundations and Trends® in Robotics*, vol. 10, no. 1-2, pp. 1–197, 2022.

[372] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[373] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters, *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends® in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[374] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[375] C. D. Freeman, E. Frey, A. Raichuk, S. Girgin, I. Mordatch, and O. Bachem, "Brax–a differentiable physics engine for large scale rigid body simulation," *arXiv preprint arXiv:2106.13281*, 2021.

[376] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[377] F. Alet, T. Lozano-Pérez, and L. P. Kaelbling, "Modular meta-learning," in *Conference on robot learning*, pp. 856–868, 2018.

[378] "NVIDIA Omniverse." Available online: https://developer.nvidia.com/nvidia-omniverse.

[379] S. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, "Chatgpt for robotics: Design principles and model abilities," 2023.

[380] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The international journal of robotics research*, vol. 5, no. 1, pp. 90–98, 1986.

[381] O. Khatib, *Commandc dynamique dans l'espace operational des robots manipulateurs en presence d'obstacles, Doc.-Ing*. PhD thesis, thesis French National Superior School of Aeronautics and Astronautics, 1980.

[382] E. Rimon and D. E. Koditschek, "Exact robot navigation in geometrically complicated but topologically simple spaces," in *Proceedings., IEEE International Conference on Robotics and Automation*, pp. 1937–1942, IEEE, 1990.

[383] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robotics Autom.*, vol. 8, pp. 501–518, 1992.

[384] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[385] S. J. Russell, *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[386] C. Liu, Q. Mao, X. Chu, and S. Xie, "An improved a-star algorithm considering water current, traffic separation and berthing for vessel path planning," *Applied Sciences*, vol. 9, no. 6, p. 1057, 2019.

[387] B. J. Thibodeau, S. W. Hart, D. R. Karuppiah, J. D. Sweeney, and O. Brock, "Cascaded filter approach to multi-objective control," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 4, pp. 3877–3882, IEEE, 2004.

[388] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[389] C. Rösmann, W. Feiten, T. Wösch, F. Hoffmann, and T. Bertram, "Trajectory modification considering dynamic constraints of autonomous robots," in *ROBOTIK 2012; 7th German Conference on Robotics*, pp. 1–6, VDE, 2012.

[390] J. S. Smith, R. Xu, and P. Vela, "egoteb: Egocentric, perception space navigation using timed-elastic-bands," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2703–2709, IEEE, 2020.

Appendix A
Methods and Tools

Artificial potential functions

Artificial potential functions applied to the robot motion control problem, first introduced by Oussama Khatib in [380], [381], are now a widely used approach for robot motion planning and obstacle avoidance. The method uses a virtual potential field to guide a robot towards a goal while avoiding obstacles. The robot is attracted to the goal and repelled by the obstacles based on a potential field function. Navigation functions, a class of potential functions originally defined in [382], [383] by Elon D. Rimon and Daniel E. Koditschek, automatically provide a feedback control law for the robot which also respects the robot's actuators limits. In particular, navigation functions are artificial potential energy functions encoding the motion task of a robot towards an arbitrary destination without any collisions with obstacles, defined on the robot's configuration space.Using potential fields for navigation, allows to solve the path planning, trajectory planning and robot control problems, by describing a suitable navigation function, constructed on the a-priori known geometric description of the free configuration space (Figure A.1).

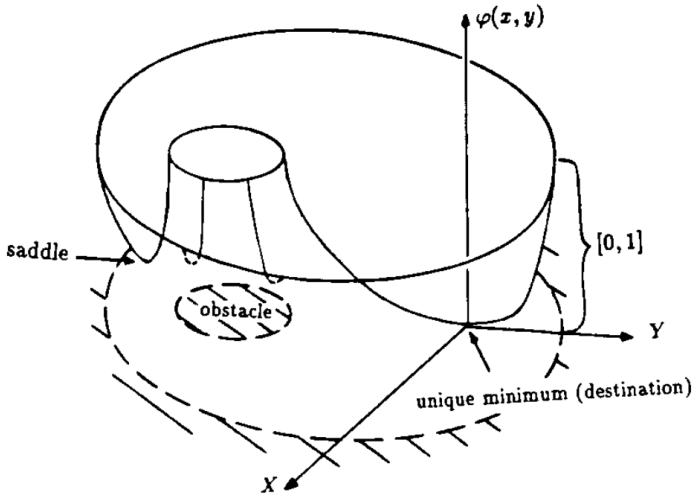


Fig. A.1 A suitable potential function constructed on the robot free configuration space [383].

Once the a suitable navigation function is constructed, the robot can follow the gradient to move towards the goal while avoiding obstacles (energy is minimized by following the negative gradient). As can be noted, the constructed potential function global minimum corresponds to the motion task destination; the potential function is designed such that the robot is attracted to low potential regions (the goal position) and repelled from high potential regions (e.g., obstacles).

Artificial potential functions in this thesis

The navigation strategy proposed in [136], and reported as a Supervisory Global Planner in **Section 1.2.1** and **Section 2.2.1**, is based on techniques inspired by the potential functions navigation approach. In particular, algebraic geometry tools are exploited to compute a set of vector fields such that a specific affine variety $\mathscr{V}$, a geometric objects defined by polynomial equations, is attractive and invariant for the associated dynamic system. This affine variety $\mathscr{V}$ denotes the desired patrolling path, and a feedback control is designed by constructing a Lypunov certificate of the convergence of the mobile robot to $\mathscr{V}$, i.e., constructing a Lyapunov function arbitrarily monotonically decreasing along the trajectories of the mobile robot.

However, attractiveness and invariance of $\mathscr{V}$ do not ensure a-priori a collision free motion path. Then, an inverse optimal control formulation for collision avoidance is provided with the objective of adjusting the feedback control input in such a way that the agent avoids collisions with obstacles. That is, the algorithm selects a vector field that minimizes the increase of a barrier function parameter, leading the robot to stay as far as possible from the obstacles. The motion planning and patrolling with collision avoidance problem is then solved by a feedback control input that steers the robot from its initial position to a patrolling path $\mathscr{V}$, and makes it move along it with a prefixed patrolling speed, while avoiding collisions. Each obstacle occupies a region in the configuration space, $\mathscr{O}_k \subset \mathbb{R}^2$. Moreover, the obstacle $\mathscr{O}_k$ is characterized by a boundary $\partial\mathscr{O}_k$ assumed to be elliptical. Hence, the obstacles present in the static map are enclosed within an associated ellipse of minimal size, resulting in a line in the case of straight walls (Figure A.2).

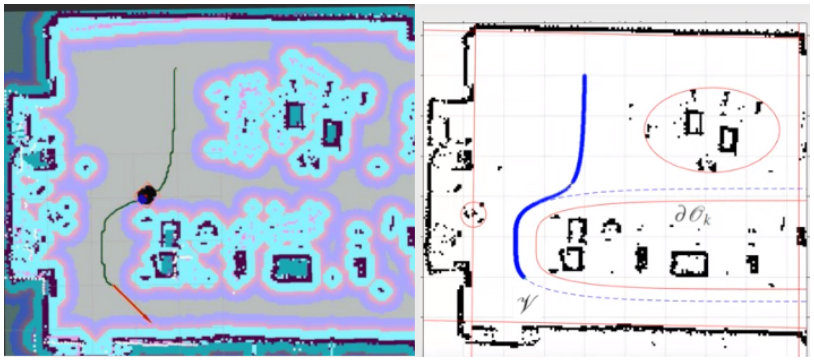


Fig. A.2 The obstacles in the static map used for navigation are enclosed by ellipses boundaries $\partial\mathscr{O}_k$ (red lines). The supervisory global planner computes a path (thick blue line), which tends to the desired patrolling path $\mathscr{V}$ (fine dashed blue line).

Furthermore, the contribution reported in Section 1.2.2 of this thesis, takes advantage of the same theoretical framework extending it to the multi-agent case. Specifically, each agent has a desired patrolling path $\mathscr{V}_i$ and a formation $\mathscr{F}$ that agents aim at, i.e., a set characterizing the potential relative displacement from the desired patrolling path. Then, a hybrid control strategy has been proposed to automatically tune the parameters of a parametric vector fields to ensure that the set $\mathscr{V} \times \mathscr{A}$ (with $\mathscr{A}$ being the compact set of admissible values for the parameters) is asymptotically stable for the corresponding hybrid implementation of the dynamical system.

In practice, the overall hybrid framework allows for the automatic selection and updating of navigation parameters to ensure the successful completion of the desired formation and patrolling task. It achieves this while also preventing collisions with both fixed obstacles in the environment and other agents.

A* algorithm

The A* algorithm, first introduced by Hart et al. in [384], is a search algorithm for finding the shortest path between two points in a graph, guided by an heuristic function. The name of the algorithm was chosen by the authors as it was easy to remember and unlikely to be confused with other mathematical symbols.

In the case of shortest path finding, the graph nodes are the grid map cells and edges costs are distances between cells. The A* algorithm explores the map cells by starting from an initial node (cell) and evaluating its neighboring nodes to select the next node to move to, based on the lowest cost. As reported in (A.1), the cost to reach a particular node from the start cell c is the sum of the edge costs traversed to that cell (known as $g(c)$), to which is added a heuristic function that approximates the cost of moving from the current cell to the goal one (known as $h(c)$).

$$f(c) = h(c) + g(c) \tag{A.1}$$

This way, the heuristic component leads the search towards the goal by prioritizing the exploration of nodes (cells) that are closer to the goal itself [385]. Note that $h(c)$ must satisfy the admissibility and consistency properties. An admissible heuristic is one that never overestimates the true cost of reaching the goal node, while a consistent heuristic satisfies the triangle inequality, meaning that the estimated cost to reach the goal node from a given node is never greater than the actual cost plus the estimated cost of reaching any successor node (neighbour cell).

Let $h(c)$ be the estimated cost of the cheapest path from cell c to the goal cell and let $e(c,c')$ be the cost of the edge from c to its neighbour cell c'. Then, a heuristic function $h(c)$ is consistent if:

$$h(c) \leq e(c,c') + h(c') \tag{A.2}$$

for every cell c and its neighbour c'. This means that the estimated cost of reaching the goal cell from c is less than or equal to the actual cost of reaching c' plus the estimated cost of reaching the goal cell from c'.

An example grid map and computed shortest path is shown in Figure A.3.

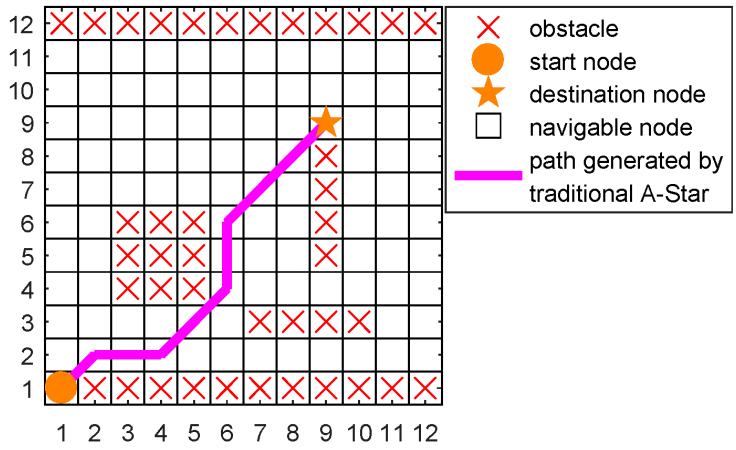


Fig. A.3 An example grid map and path generated by the A* algorithm [386].

A* in this thesis

In general, the advantage of this approach is that the distance metrics used can be modified or extended to incorporate other factors such as time, energy consumption, or safety. In the case of the supervised global path planning algorithm proposed in [6] and reported in **Section 1.2.1** of this thesis, we considered the Euclidean distance as heuristic function, namely:

$$h(c) = \sqrt{(c.x - dest.x)^2 + (c.y - dest.y)^2}, \tag{A.3}$$

where *dest* is the destination cell and the expressions $.x$ and $.y$ extract the fields corresponding to the x and y coordinates of the considered cell, respectively. Then, to favour the passage on each way point (WP) included in the safe path computed by the Supervisory Global Planner (SGP), we introduced an additional cost h_{WP} as:

$$h_{WP}(c) = \sqrt{(c.x - WP.x)^2 + (c.y - WP.y)^2}, \tag{A.4}$$

This way, the overall cost associated with cell c is thus:

$$f(c) = g(c) + h(c) + h_{WP}(c), \tag{A.5}$$

which allows to obtain a global planning algorithm that acts as A* when far from the SGP path, and associates a low cost to the SGP generated positions, so to guarantee their presence in the output plan.

Dynamic Window Approach (DWA)

The Dynamic Window Approach (DWA), first introduced by Fox et al. in [170], is a motion planning algorithm for obstacle avoidance considering the robot's current state and the environment to generate a set of admissible velocities, among which the best velocity is selected based on the robot current goal and velocity constraints.

The dynamic window refers to the time window dynamically adjusted to limit the admissible velocities to those that the robot can achieve between two steering commands, considering its limited accelerations. First the algorithm generates a velocity space including reachable velocities within the dynamic window taking into account the robot's current state and goal (see Figure A.4). Then the trajectories obtainable from the admissible velocities are evaluated individually considering their distance to obstacles, distance to the final goal, and closeness to the desired velocity.

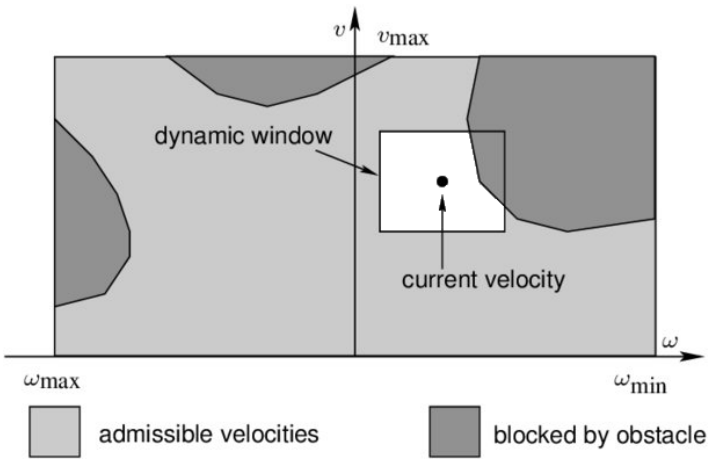


Fig. A.4 The overall search space (light gray) is reduced to the dynamic window (white), which contains only the velocities that can be reached within the next time interval [387].

All the velocities that would lead to a collision are excluded and, among the remaining velocities, the best candidate is chosen by maximizing a cost function $G(v,\omega)$ over the discretized velocity search space (Equation (A.6)).

$$G(v,\omega) = \sigma(\alpha\, heading(v,\omega) + \beta \cdot dist(v,\omega) + \gamma \cdot velocity(v,\omega)), \tag{A.6}$$

where $heading(v,\omega)$ evaluates the alignment of the robot with the desired direction, $dist(v,\omega)$ indicates the distance of the nearest obstacle intersecting with the curvature, and $velocity(v,\omega)$ is utilized to assess the advancement of the robot along its intended path as a projection on the translational velocity v. In practice, the best trajectory is the one that is closest to the goal, farthest from obstacles, and closest to the desired velocity.

DWA in this thesis

In order to perform obstacle avoidance, the information on detected obstacles must be available to the local motion planner and is typically embedded in the ROS costmap layers mechanism. In the contribution presented in **Section 1.2.3**, dynamic obstacles are detected through background subtraction of costmaps, tracked estimating their velocity using a Kalman filter, and assigned a cost determining a 2D Gaussian inflation shape proportional to the obstacle velocity and oriented in its moving direction. This information has been integrated within the local costmap used for obstacle avoidance by the local planner.

In particular the employed local planner is the DWB planner in ROS2, an highly-configurable extension of the DWA implementation (DW "B" being incrementally named). The integration of our dynamic obstacle layer, incorporating each object direction and velocity in the Gaussian shape inflation, allowed for a more reactive and informed selection of the best control input candidate (and relative trajectory) among the ones predicted by the DWB planner.

You Only Look Once (YOLO) framework

YOLO (You Only Look Once) is a Convolutional Neural Network (CNN)-based framework for object detection, introduced by Redmon et al. in [74] and improved in later versions [388], [212], which can identify objects within an image or video frame and localize them by drawing a bounding box around each object.

The YOLO system models detection as a regression problem: the input image is divided into $S \times S$ grid cells, each of which predicts B bounding boxes and the confidence that determines if the cell contains an object. Additionally, the model predicts the likelihood of an object belonging to a particular class for each grid cell (Figure A.5), making it possible to detect and classify multiple objects in an image.

By utilizing convolutional features, potential bounding boxes and their corresponding scores are generated while imposing spatial constraints on the grid cell proposals. This technique helps to alleviate the problem of multiple detections of the same object. YOLO predicts the coordinates of bounding boxes directly using fully connected layers on top of the convolutional feature extractor. With YOLOv2 [388], the authors incorporated batch normalization and eliminated the fully connected layers from YOLO architecture, exploiting anchor boxes to predict bounding boxes. Anchor boxes are a set of predefined bounding boxes of different sizes and aspect ratios that are used as reference templates during object detection, representing a sort of prior knowledge to the algorithm, allowing it to better predict the location and size of objects in the image. On top of this upgrades, YOLOv3 [254], among many enhancements, improved accuracy by using a deeper feature extraction network, and introduced a technique to capture objects at different scales.

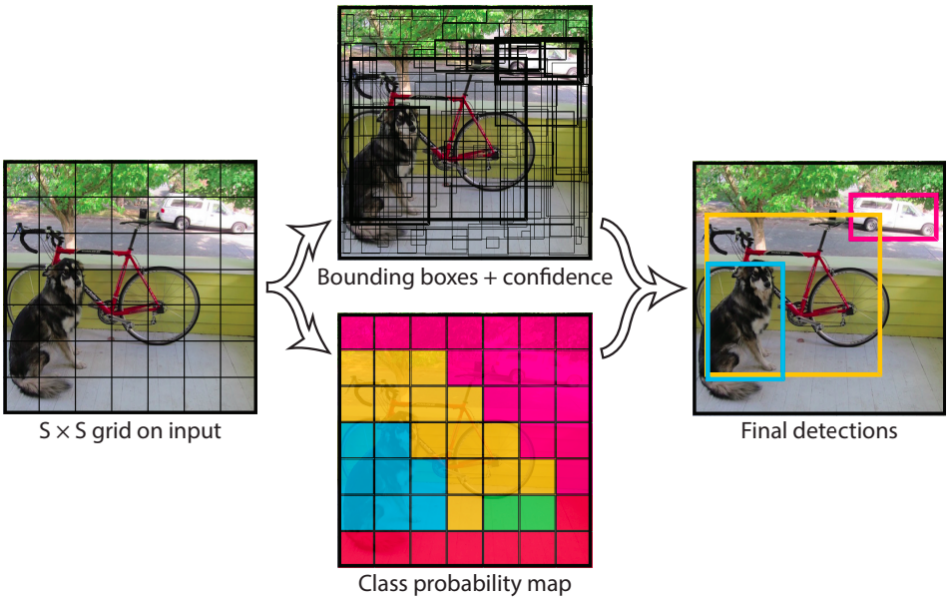


Fig. A.5 Each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities [74].

For each bounding box, the network predicts four coordinates: t_x, t_y, t_w, t_h. The position of the cell relative to the image's top-left corner is denoted by the offsets (c_x, c_y), while the bounding box dimension prior has a width and height of p_w and p_h, respectively. The predicted coordinates correspond to the following:

$$\begin{aligned} b_x &= \sigma(t_x) + c_x \\ b_y &= \sigma(t_y) + c_y \\ b_w &= p_w e^{t_w} \\ b_h &= p_h e^{t_h} \end{aligned} \tag{A.7}$$

The height and width of the bounding box is predicted as offset from the cluster centroid, while the center coordinates of the box relative to the location of filter application is predicted using a sigmoid function (refer to Figure A.6).

YOLO in this thesis

In the context of this thesis, we used a YOLOv3 pre-trained model, to achieve human obstacles detection for the Sen3Bot laboratory demonstrator implementation (**Section 2.1.3**, **Section 2.2.1**, and **Section 3.1.1**). In particular, the bounding box coordinates b_x, b_y, b_w and b_h and relative label of detected objects have been suitably formatted and conveyed into a text file. This detection output file is then accordingly read and filtered by a ROS node that wraps into ROS topic messages only the information we are interested in (i.e., the lines identified by a "person" label. This way the information about the location of a human obstacle in the image reference

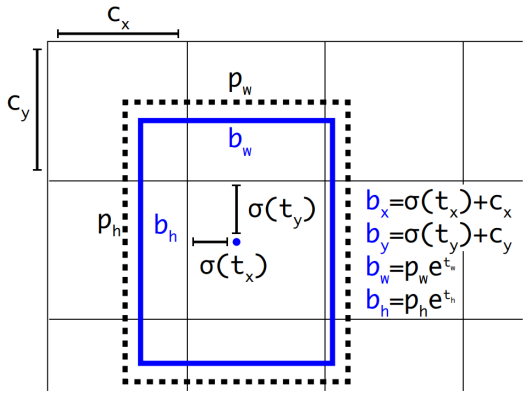


Fig. A.6 Bounding boxes with dimension priors and location prediction [254].

frame has been fused with the corresponding laser distance information, to obtain coordinates in the navigation map reference frame. Virtual obstacles have been placed in correspondence of such positions on the map, enclosing humans in virtual cages, as the distance between the planned motion and the human is kept conservative with respect to generic obstacles.

Timed Elastic Band (TEB) algorithm

The Timed Elastic Band (TEB) algorithm, introduced by Rösmann et al. in [389], is a motion planning algorithm that represents the robot's trajectory as a time-elastic band in the configuration space. The TEB consists of a central trajectory and two boundaries that define the robot's maximum and minimum possible positions at any given time. The TEB algorithm includes temporal information in addition to the "elastic band" concept. By doing so, it is capable of accounting for the dynamic constraints of the robot, and enables the modification of trajectories rather than just paths.

The conventional "elastic band" is defined as a series of n robot configurations, each consisting of the robot's position (x_i, y_i) and orientation in a particular frame.

$$Q = \{\mathbf{x}_i\}_{i=0\ldots n} \quad n \in \mathbb{N} \tag{A.8}$$

Augmented by the time intervals between consecutive configurations, the timed elastic band represents a sequence of $n-1$ time differences ΔT_i.

$$\tau = \{\Delta T_i\}_{i=0\ldots n-1} \tag{A.9}$$

Each time difference represents the duration of time that the robot needs to move from one configuration to the next configuration in the sequence.

TEB is defined as a tuple of both sequences:

$$B := (Q, \tau) \tag{A.10}$$

The TEB algorithm aims to optimize and adapt both the configurations and time intervals in real-time using a weighted multi-objective optimization approach:

$$\begin{aligned} f(B) &= \sum_k \gamma_k f_k(B) \\ B^* &= \underset{B}{\operatorname{argmin}} f(B), \end{aligned} \tag{A.11}$$

in which B^* denotes the optimized TEB, $f(B)$ denotes the objective function, defined as a weighted sum of components $f_k(B)$. In particular, two objectives are considered simultaneously: reaching the intermediate waypoints of the original path and avoiding static or dynamic obstacles. Specifically, the intermediate waypoints attract the elastic band, while obstacles repel it.

In practice, the algorithm works by first generating a set of candidate trajectories that satisfy the start and goal constraints, and then optimizing the trajectory using a cost function that considers both the time and the clearance (distance to obstacles) along the trajectory. The cost function is designed to favor trajectories that have high clearance from obstacles, but also to penalize trajectories that deviate significantly from the central trajectory.

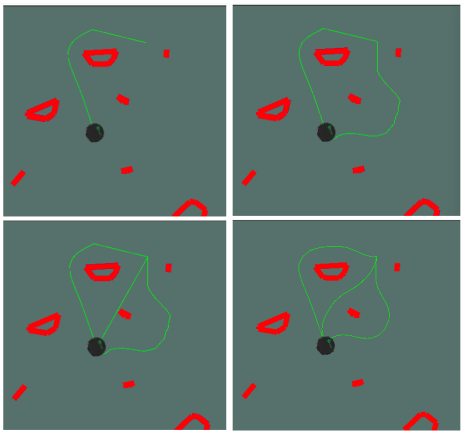
A subset of admissible trajectories with distinctive topologies (candidate paths) is optimized in parallel to allow the local planner to switch to the current globally optimal trajectory among the candidate set in case the robot gets stuck in a locally optimal trajectory due to obstacles. The distinctive topologies of the candidate paths are obtained by utilizing the concept of homology/homotopy classes, which ensures that each candidate path represents a unique way of navigating around obstacles while reaching the desired goal [172]. A graphical representation of these steps is represented in Figure A.7a.

TEB in this thesis

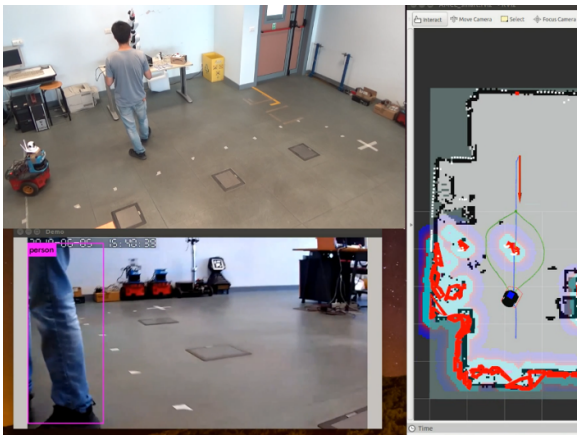
In the context of this thesis, the TEB algorithm has often been chosen as a local planner as it generates a trajectory that optimizes both the robot's motion and the time intervals between consecutive configurations, which allows it to avoid obstacles and navigate in a smooth and efficient way.

In particular, in the sensor data fusion algorithm for smart AMRs (Sen3Bots) proposed in [7] and reported in **Section 2.1.3** of this thesis, we exploited the TEB algorithm as a local planner where virtual obstacles are overlaid on the detected human obstacles (exploiting the YOLO framework), ensuring a conservative behaviour by inflating the radius around the human. Figure A.7b shows how the TEB-generated candidate paths (light green line) suggest two possible overtaking paths, while the

global plan (dark blue) does not take into account obstacles that were not known at mapping time as its costmap is constructed upon the static map.



(a) TEB algorithm steps representation [390].



(b) Overtaking of a human obstacle.

Fig. A.7 TEB candidate paths and optimization sequence (a), and behaviour as a local planner in the presence of an obstacle on the global planner computed path (b).