UNIVERSITÀ DI TRENTO

University of Trento

Department of Information Engineering and Computer Science
ICT International Doctoral School

Socially aware robot navigation

Author:
Alessandro Antonucci

Advisor:
Prof. Luigi Palopoli
Co-Advisor:
Prof. Daniele Fontanelli

October, 2022

UNIVERSITY OF TRENTO

Abstract

ICT International Doctoral School
Department of Information Engineering and Computer Science

Socially aware robot navigation

by Alessandro ANTONUCCI

A growing number of applications involving autonomous mobile robots will require their navigation across environments in which spaces are shared with humans. In those situations, the robot's actions are socially acceptable if they reflect the behaviours that humans would generate in similar conditions. Therefore, the robot must perceive people in the environment and correctly react based on their actions and relevance to its mission. In order to give a push forward to human-robot interaction, the proposed research is focused on efficient robot motion algorithms, covering all the tasks needed in the whole process, such as obstacle detection, human motion tracking and prediction, socially aware navigation, etc. The final framework presented in this thesis is a robust and efficient solution enabling the robot to correctly understand the human intentions and consequently perform safe, legible, and socially compliant actions. The thesis retraces in its structure all the different steps of the framework through the presentation of the algorithms and models developed, and the experimental evaluations carried out both with simulations and on real robotic platforms, showing the performance obtained in real–time in complex scenarios, where the humans are present and play a prominent role in the robot decisions. The proposed implementations are all based on insightful combinations of traditional model-based techniques and machine learning algorithms, that are adequately fused to effectively solve the human-aware navigation. The specific synergy of the two methodology gives us greater flexibility and generalization than the navigation approaches proposed so far, while maintaining accuracy and reliability which are not always displayed by learning methods.

A Francesca Romana.

Contents

Chapter 1

Introduction

1.1 Problem statement

We are living a time when robots are no longer confined to industrial environments but are used in numerous applications, which require an unprecedented degree of autonomy. Modern robots have to adapt to humans, to understand their needs, to help humans carry out their activities. It is easy to imagine a near future in which the interaction between robots and humans will be so inbred that it will be executed naturally as one of the many actions that we perform every day. Ideally, the level of relation we aim for is perfectly illustrated by the rider-horse metaphor (Flemisch et al., 2003), in which the human developed an "innate" ability to use the robot's services, and the robot is able to grasp his/her intention without any explicit request.

Let us now focus on the specific problem of the navigation of an autonomous mobile robot across promiscuous areas, which are shared with human beings. Autonomous driving vehicles, flying robots, industrial robots, service robots are some use cases in which robots will interact with co-located humans (Cha et al., 2018). In industrial environments robots work side by side with the human operator. Safety becomes the major requirement and great care must be taken in the synergy with the workers, thus robots need to perceive the surroundings with as little uncertainty as possible and express their actions appropriately. Luckily, human workers have a great knowledge of the spaces and the robot's functionality. However, qualified training is necessary to be capable to drive these machines. The scenario in which service robots operate (some examples are depicted in Figure 1.1) is instead more dynamic, less structured, and often contains unfamiliar entities (Cha et al., 2018). Consequently, social conventions come into play. In cases like this, the actions of a robot are socially acceptable insofar as they reflect the behaviours that a human would generate in similar conditions.

The first cornerstone of a human-aware planner is a reliable predictive model of human motion. People are capable of moving in an expressive manner to communicate their intentions. However, this capacity depends on many factors. Sociological and cultural characteristics influence human reaction according to some tacit rules, such as deviating to the left or to the right, avoiding elderly people more carefully, etc. (Pettré et al., 2009). Behaviours also change according to psychological factors, for example people motion vary if they are in a hurry or they are simply wandering. However, choosing the right set of measurable and quantifiable physical parameters allows to obtain a prediction model that can be used under certain reliability constraints. For instance, pedestrians' walking direction provides great information about their intention (Farina et al., 2017b; Fotios et al., 2015). A further step forward over the human motion prediction is to consider that persons in a shared environment will cooperate,

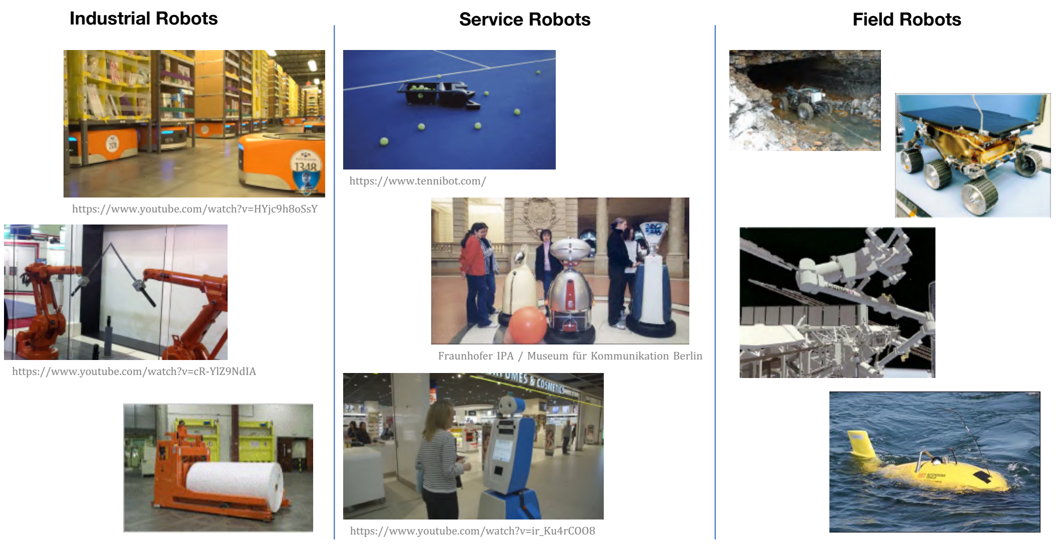

Industrial Robots
Service Robots
Field Robots
https://www.youtube.com/watch?v=HYjc9h8oSsY
https://www.youtube.com/watch?v=cR-YlZ9NdIA
https://www.tennibot.com/
Fraunhofer IPA / Museum für Kommunikation Berlin
https://www.youtube.com/watch?v=ir_Ku4rCOO8

Figure 1.1: Taxonomy for the autonomous mobile robots. Despite their high performance, dealing with industrial and service robots complicate the picture since the operational space can be shared with human beings.

and will help avoiding collisions and undesired behaviours. So in many situations the robot can make smaller evasion manoeuvrers than if it had to otherwise. This makes the robot to move more efficiently and more directly towards the goal. It is also known that moving people usually react to the actions of others, and cooperative strategies are used. Humans for example maintain spaces for themselves comparable to those they imagine the others would prefer, and this concept is supported by the Theory of Mind (Rios-Martinez et al., 2015; Wiltshire et al., 2017).

The second fundamental element for a robot planner is a precise and robust localization system with which the robot can localise itself in the environment and detects the position of obstacles. In addition, the robot needs to know the relative position of people nearby with whom it could interact. To this end, distance and vision-based sensors, as well as appropriate tracking algorithms, are required to detect and track humans. The major requirement of the tracking system is logically to run in real-time on board the robotic vehicle.

Looking at the recent literature, we note how both these two features are developed and implemented with machine learning techniques. This evolving trend is no coincidence, as Artificial intelligence (AI) research has indeed increased significantly in several research fields. The number of AI journal publications in 2020 is 5.4 times higher than it was in 2000, and only from 2019 to 2020 it grew by 34.5 % (Zhang et al., 2021). The technical improvement of the sensor equipment and, above all, the progressive increase in computing power have given a considerable boost to mobile robotics, giving to intelligent vehicles the capability to move on their own with greater autonomy and efficiency. The consequent step was the introduction of machine learning, that became extensively used in several operations of the robots: from the computer vision applications for human-robot interactions to independent localization and mapping, as from human motion prediction to socially compliant navigation. And also, numerous lines of research are being pursued on person identification and classification, action recognition, and human behaviour understanding. The massive application of learning techniques has certainly uplifted the performance of robots, yet it has also shown how many tasks cannot be undertaken with the mere training of a neural network. Just as machine learning is showing the

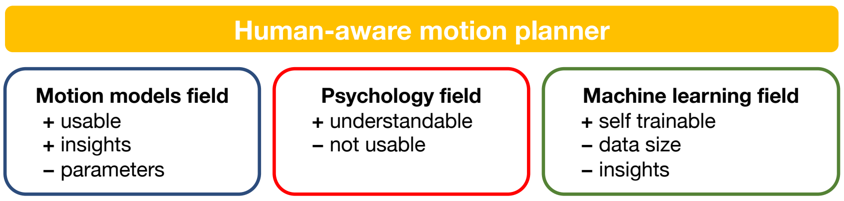


Figure 1.2: Key characteristics of different methodologies that can be employed in a human-aware motion planner. It is noticeable that relying on an individual method is not feasible, but the reciprocal weakness can be compensated by combining the techniques.

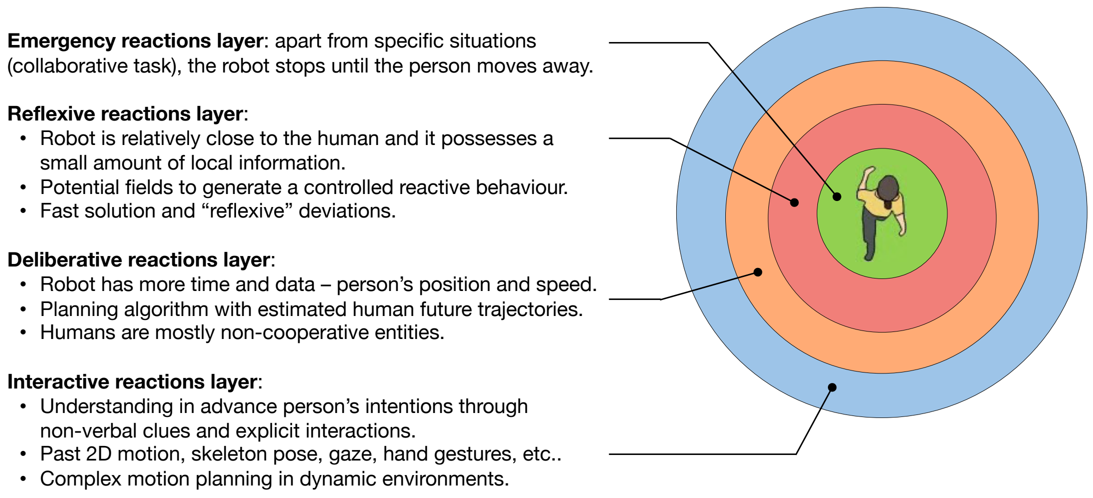
first difficulties in computer science, also in the most specific applications, such as mobile robotics, it is certainly a very powerful tool that must however be used with grain of salt. Artificial intelligence mostly suffers of brittleness (Alcorn et al., 2019), embedded biases (Bahari et al., 2022), bad uncertainty quantification (Abdar et al., 2021), and explainability (Xu et al., 2019). The latter problem is ranked as the third risk that industries and researchers consider relevant (Zhang et al., 2021).

The philosophy that passes through the works presented in the thesis succeeds this last consideration. The literature on mobile robotics contains innumerable contributions in which machine learning is abundantly used (with remarkable results), still it is not so trivial to transfer these approaches directly on the robots for a real-time application. Indeed, we note that as the complexity of the human behaviours to be modelled increases, the techniques become more and more sophisticated so that higher computational capability is needed. However, in some cases a simpler model might be sufficient. To give an example, neural networks are not really necessary to predict where a person who is walking straight will be in the next few seconds (Schöller et al., 2020).

Instead, to obtain the maximum yield it is necessary to correctly discern the strengths of each individual methodology, and moreover to understand how to compensate for their weaknesses with the combined use of different techniques. For instance, as depicted in Figure 1.2, motion predictors based on model representations are the best estimators when the observed dynamic is coherent with the model, but they are highly dependent on the choice of the parameters. Conversely, the machine learning can be helpful to represent complex dynamics, however, the dependency on data is somehow inevitable.

1.2 Research overview

The conducted research brought to the development of algorithms for different tasks of mobile robotics, devoted in particular for human perception and motion prediction, combining model-based and learning-based techniques, moving towards reliable and computationally acceptable solutions. The main objective was to investigate all the elements of a human-aware motion planner for the robot, where each element represent a different layer of a “cognitive” framework to manage the robot social interactions in shared spaces. The framework that we are depicting is closely inspired to the principle of proxemics (see Section 2.2). The structure of the framework (represented in Figure 1.3) is reasonable from the conceptual point of view, because it reflects the various manifestations of human actions (for example, instinctive or reasoned), moreover it tightly relates the different amount of information and the corresponding possible robot actions to a geometrical perspective, so that the behaviours of the robot can be socially compliant depending on the relative distance it has with the person. Finally, this design allows



Emergency reactions layer: apart from specific situations (collaborative task), the robot stops until the person moves away.

Reflexive reactions layer:
- Robot is relatively close to the human and it possesses a small amount of local information.
- Potential fields to generate a controlled reactive behaviour.
- Fast solution and "reflexive" deviations.

Deliberative reactions layer:
- Robot has more time and data – person's position and speed.
- Planning algorithm with estimated human future trajectories.
- Humans are mostly non-cooperative entities.

Interactive reactions layer:
- Understanding in advance person's intentions through non-verbal clues and explicit interactions.
- Past 2D motion, skeleton pose, gaze, hand gestures, etc..
- Complex motion planning in dynamic environments.

Figure 1.3: Cognitive framework used as guideline for the socially aware robot planning.

to divide the complete problem of human-robot interaction into nested modules that can be developed with a certain degree of independence. The innermost layer is mainly dedicated to collision avoidance strategies, given the short distances between human and robot. In the intermediate layer we have a passive interaction between human and robot, so we can use reactive methods to generate safe and socially admissible paths. The last layer can access to more information about the intentions of the person, and therefore we can actuate more complex operations in which robots and humans actively interact or collaborate to fulfil their task.

The research carried out mainly concerned the last two layers of the framework, where the robot has access to limited information on the human intentions (coming mostly from his/her past motion), and its able to proactively perform the appropriate action. By looking at the works presented in this thesis from a more technical side, they can be differentiated with respect to their scope in one of the three main steps of the general mobile robotic framework (depicted in Figure 1.4), which are:

- *Data acquisition*, related to the sensing system of the robot, from the localization to the people tracking, up to the person classification.
- *Human intention prediction*, that is the estimation of person's future actions and the recognition of his/her intentions.
- *Decision executing*: this last task is aimed at the actions that can be performed by the robot itself, which can interact with a person both passively (thus respecting the intentions of the latter but without provoking reactions) and active (i.e. triggering a reaction from the person by sending social signals about the actions taken).

This specific sequence of the tasks determined also how they will be proposed in the continuation of this thesis (see Section 1.3). The different topics are listed below.

1.2.1 Human positioning and tracking

The basis for giving an increased autonomy to robots that need to move in natural environments with humans is to enable the capability of such machines to understand the positions and the motions of people moving in their surroundings. The ideal sensor systems with which to equip a robot must provide

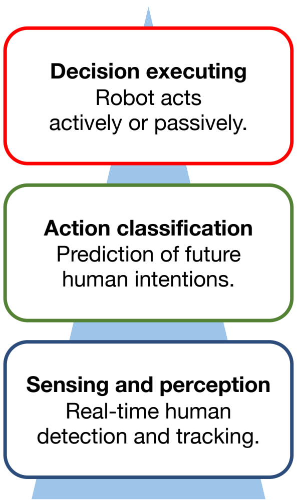


Figure 1.4: Tasks flow for a mobile robot. The process starts from the sensing and perception phase, continues through the action classification phase and reaches the decision executing phase.

for the most basic of requirements for the inner most layers of Figure 1.3 (i.e., performs the obstacles avoidance) to the more sophisticated ones such as human action recognition and tracking. Such systems must evidently be robust to failures but – to prove to be suitable for industrial and commercial applications – also have competitive computational speed and cost. Finally, given the high time variability of the environments in which they would work, we must ensure that most of the measurements will be performed by the robots themselves, from their relative perspective, rather than by an external infrastructure (as represented in Figure 1.5). Given these considerations, we investigates the effectiveness of a people tracking system with different sensors, namely, depth cameras, laser scanner, and millimeter-wave radar. The gathered data was coupled with an estimation algorithm to continuously track the human positions in the environment.

Experiments were conducted in indoor scenarios (our reference case study), to characterise and evaluate the performance of such a system and establish its suitability for an application on an actual service robot. The key outcome of the comparative evaluation is that a sensor system comprised of monocular camera and 2D laser scanner is satisfactory to accommodate the requirements stated above. The established instrument configuration it was then used for the related researches presented in this thesis. Moreover, we propose a mitigation strategy of the anomalies that usually affect the localization accuracy and robustness, such as delayed and spurious detections. It is worth noting that also in this step of the robotic framework (the lowest level of Figure 1.4), we adopted a solution that included classic estimation algorithms with more recent machine learning methods. A further finding is the idea of using the Bayesian filtering as a classifier, in which each mode represent a different motion behaviour. This clue was also adopted in the human motion prediction, were different target goals are inferred using a Bayesian estimator.

1.2.2 Human motion prediction

Numerous research results have applied machine learning techniques to classify the trajectories travelled by pedestrians and estimate their future position. The rationale comes from the complexity and characteristics of the problem. Each person has his/her unique way of walking, so a predictive model

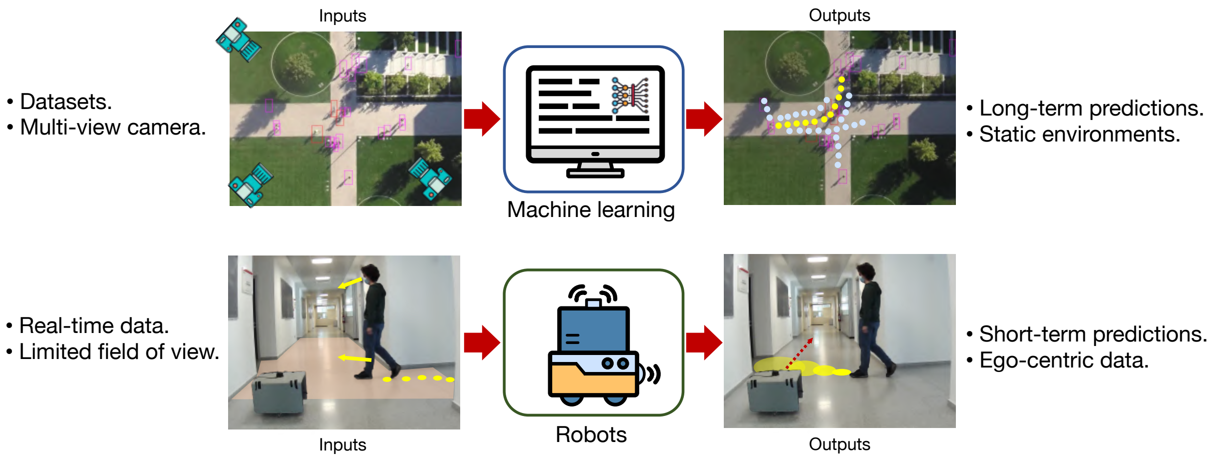


Figure 1.5: Differences between machine learning techniques applied for human motion prediction and motion predictors applied for robot navigation. It is highlighted how the two methodologies are based on different input data and produce specific outputs.

must adapt to all the possible behaviours, secondly there is a temporal correlation in the observable data (such as the succession of displacements), meaning that learning the past information can actually give insights on the future. However, the direct implementation of the neural network solutions on board the robot is not trivial. The robot control algorithms usually rely on ego-centric inputs and have serviceable outputs that are different from what traditional neural network approaches produce. As sketched in Figure 1.5, machine learning techniques for motion prediction commonly rely on well defined dataset, taken in specific scenarios with long-sided trajectories. The most common output data are long predictions of future movements, and the most travelled zones of the environments, which however are related to the specific dataset. If the robot can perceive the environment only from its point of view, it has a limited knowledge on the surroundings (see the shaded are in Figure 1.5), and its actions take place in much shorter time windows. It is also understandable that predictive methods based on specific dynamic models can obtain a future estimate when the choice of the model is consistent with the use case, thus their performance is better than any trained network, but the ability to adapt to new contexts is far less. Our idea is to take the best of the two approaches is a mixed solution, where the neural network embeds in its structure a dynamical model with which to represent the motion of the pedestrian, specifically the Social Force Model (SFM) by Helbing and Molnar, 1995. This choice allows us to delegate to the learning phase the actually unknown characteristics of the motion (that is, the model's parameters), and to keep the neural network with a simple structure with a drastically reduced number of learnable weights. The most important contribution is that the specific network increases the "explainability" of the training, as the behaviour can be interpreted in physical terms. As an additional result, we can obtain a good prediction accuracy even by using a small and synthetically generated training set. Finally, our solution can work even in scenarios radically different from those used in the training, and the type of the input and output data is consistent with the sensors used by the robot, thus we can guarantee the transfer learning on the real robotic platform.

1.2.3 Multi-robot navigation in human-shared environments

Among all the different interactions that occur between humans and robots while they both shared spaces, the most prominent is certainly the mutual collision avoidance. Following the ideal structure of the cognitive framework presented at the beginning of this section (Figure 1.3), we combined different approaches in a hierarchical framework consisting of global path planning, local path planning, and reactive strategies, to obtain safe and socially aware multi-robot navigation. Specifically, we identified five classes of requirements, namely: robustness and safety in navigation, socially aware motion planning, multi-agent coordination, responsiveness to dynamic environments, and computation efficiency. While we will present an overview of the complete framework and the result obtained through several tests and extensive experiments with a real robotic implementation, we will deepen the requirement related to the socially aware motion planning, as it is relevant to the researches contained in this thesis.

1.2.4 Teach-by-showing navigation

The capacity to determine a safe path is one of the most fundamental keys to broad usage of mobile robots in unstructured, human-populated, and possibly a-priori unknown environments. As a further investigation on human-robot interaction, beyond the collision avoidance, we developed a complete framework for human following and path reconstruction tasks. Indeed, existing methods for planning safe paths can be divided into two broad categories. The first type entrusts complete control of the robot to trained human users, who are expected to monitor the robot movements and determine their trajectories. In the second type of planners robots have to learn how to plan their own path and move independently. While some of these planners have shown interesting results, they can be unreliable, especially when a robot is navigating in complex environments that also contain humans or obstacles. These planners often require expensive hardware and sensors to produce excellent results.

One possible idea is to delegate the path planning routine to a human walking in front of the robot. The human operator must concentrate only on the path to take, naturally marking with her/his footsteps the trace to be followed. The algorithm does not require particularly expensive sensors or highly advanced software components. In our framework, the robot recognizes the human leader (or "path-finder") to then locate and track his/her movements. Then, a tailored sensor fusion algorithm based on a laser scanner and a depth camera, mounted on the robot chassis, allows the robot to robustly distinguish the path-finder from other people in its surroundings. Finally, we interpolate the tracked human positions with a continuous curvature pathway, ensuring path smoothness and safety in control, since the robot can stop in time before colliding with static obstacles and other people.

1.3 Structure of the thesis

Following the description of Section 1.2, the conducted research touched all the three main steps of the robotic paradigm. Since following this line of study, several scientific contributions have been published. While creating the structure of the thesis we have organized each chapter to present the results obtained in a particular task, starting from the perception side, followed by the prediction phase, and concluding with the control of the robot. More precisely, the thesis is organised as follows:

Reference	It appears mainly in...
Antonucci and Fontanelli, 2018	-
Antonucci et al., 2019a	Chapter 3
Antonucci et al., 2019c	Chapter 3
Antonucci et al., 2019b	Chapter 1
Antonucci et al., 2021a	Chapter 4
Boldrer et al., 2022	Chapter 5
Antonucci et al., 2021b*	Chapter 6
Shamsfakhr et al., 2022	-

*Submitted to IEEE T-RO

Table 1.1: Scientific contributions of the PhD.

- Chapter 3 is devoted to human detection and tracking algorithms. Different approaches using stereo and depth camera, laser scanner and radar technologies are presented and experimentally evaluated. This comparative study for the robust and efficient sensing setup is functional for the integration of our proposed mobile robot applications.

- In Chapter 4, we present our strategy for the human motion prediction with a combination of structured neural networks and model based inference. We will describe in detail the implemented algorithms and show the resulting experiments on a robotic platform.

- In Chapter 5 we show the implementation of the motion prediction module in a path planning solution for multi-robot navigation in dynamic environments shared with humans. The system was developed in collaboration with Manuel Boldrer, a PhD student working in the same lab, and implemented on multiple wheeled robots.

- Chapter 6 presents an efficient solution for accomplishing the people-following navigation task of a companion robot. The developed solution allows the robot to learn and follow the path traveled by the human operator even in crowded conditions.

These chapters are preceded by a review the current state of the art in Chapter 2, where we will point out approaches relevant for the tackled research. Finally, in Chapter 7 we will have a final discussion on the presented works.

This thesis contains various contributions in the field of mobile robotics, regarding different scopes and applications. Table 1.1 reports the list of publications done, as well as the chapters in which they are presented for those that are included in this work.

Chapter 2

Literature overview

In this chapter we want to relate the methods presented in this thesis with the current state of the art. We start with an overview on human motion prediction models in Section 2.1, then, in Section 2.3 we will go deeper into socially aware robot navigation contributions relevant to our proposed methods. Section 2.2 offers a discussion of the main characteristics that influence the interaction between mobile robots and people, that are worth considering when designing both model-based and learning-based algorithms. Finally, we will introduce the concept of the structured neural network and list the related papers in Section 2.4. The literature on human detection and tracking techniques is reported in detail in Section 3.1, while for the state of the art related to human following application we refer to Section 6.2. Moreover, each chapter of the thesis presents in its introductory part an overview of the state of the art, summarizing the contributions most inherent to the topic addressed.

2.1 Human motion prediction

A proper review of human motion models needs to start from crowd behavioural models, which is outside the scope of this thesis. Excellent surveys on this topic (listed in Table 2.1) are: Kok et al., 2016; Sakour and Hu, 2017; Haghani and Sarvi, 2018; Rudenko et al., 2019; Zitouni et al., 2019.

It is well established that, even if at an unconscious level, pedestrians are able to understand the time at which a collision is probably going to happen by means of neural processes involving brain and retina, and to modify their motion in order to avoid it (Basili et al., 2013; Cutting et al., 1995). Even if people always tend to minimize the effort in reaching their goal, performing an avoidance is more a compromise between safety and energy (Vassallo et al., 2017). Collision avoidance is a process that successfully takes place in many ways, even though pedestrians can substantially adjust walking speed, walking direction or both (Huber et al., 2014; Parisi et al., 2016). Furthermore, it has been observed that pedestrians share effort in avoidance manoeuvrers, contributing asymmetrically and taking on different roles (Olivier et al., 2013; Pettré et al., 2009). In order to obtain realistic and compliant navigation planners, it is appropriate to determine which psychological, personal or environmental factors influence the role assignment and the contribution that pedestrian put in collision prevention. Previous research results (Pettré et al., 2009; Olivier et al., 2013) stated that pedestrians are able estimate the Time-To-Collision (TTC) between them and the others from visual perception. Although there are still doubts about which behavioural factors influence the TTC, numerous models use it to characterize the motion adaptation (Moussaïd et al., 2011; Huber et al., 2014; Karamouzas et al., 2014). Moreover, in Huber et al., 2014 path adjustments were observed in interactions at different angles, whereas speed adjustments

Reference	Topic	
Kok et al., 2016	Crowd behaviour	Review of crowd behavior studies with the integration of the physics and biological attributes of the human motion.
Sakour and Hu, 2017	Crowd behaviour	Survey on crowd modeling and crowd simulation techniques.
Haghani and Sarvi, 2018	Crowd behaviour	Collection human crowd behaviour models with a focus on the three levels of decisions, namely strategic-level decision, tactical-level decision, and operational-level decision.
Rudenko et al., 2019	Human motion prediction	Survey of human trajectory prediction techniques, organized according to the model of motion and the input contextual cues.
Zitouni et al., 2019	Crowd behaviour	Review of socio-cognitive crowd behaviour models, from vision based data. The papers are divided in individualistic behaviour, group behaviour, social interaction behaviour and leader-follower behaviour.

Table 2.1: Summary of relevant surveys on human motion models.

were evident only in scenarios with crossing angles between $45°$ and $90°$. The influence of the looking behaviour on the crossing order was also evinced (Fotios et al., 2015; Croft and Panchuk, 2018; Olivier et al., 2013). There is still no unanimous idea of how the vision is really used, however it was observed that pedestrians use most of their vision without fixation on any object: essentially they look into space, presumably scanning their surroundings (Kitazawa and Fujiyama, 2010). The majority of the surveyed papers support the assumption of path adjustments as a preferred collision avoidance strategy in the presence of sufficient space (Prédhumeau et al., 2019). As for the most significant characteristic features, it can been noticed that speed, distance and time are essential and intrinsically connected (Pettré et al., 2009), while also orientation and gaze can play a decisive role (Belkada et al., 2021). In Olivier et al., 2013 collision avoidance between two persons walking along crossing paths was investigated, finding that the avoidance was mutually performed with a role-dependent behaviour (who crosses first and who gives way), and the walker giving way contributed more by adapting both the walking speed and path. Moreover, no inversion in the role of each walker was observed. Knorr et al., 2016 found experimentally that initial walking speed and heading was sufficient to correctly predict the crossing order. The person crossing first actually started the path adjustment and served a "symbolic cue" to the person giving way. They also observed that parameters such as gender, height and personality traits did not influenced pedestrian behaviour.

In the following, we present the most relevant research results on human motion prediction coming from surveys (Kok et al., 2016; Sakour and Hu, 2017; Haghani and Sarvi, 2018; Rudenko et al., 2019) and conventional academic search engines.

Model based. Reactive models are based on explicit dynamical motion models that follow Newton's laws of motion. In the famous Social Force Model (SFM) by Helbing and Molnar, 1995, agents are exposed to different repulsive (from obstacles) and attractive (towards goal) forces depending on their relative distances. The main advantages of this method are that its implementation is quite intuitive and it should work faster than alternative approaches. However, this model does not prevent the collisions between agents, thus numerous modifications have been proposed. In Pellegrini et al., 2009 all the agents evaluate the expected point of closest approach between the others, and use this information as the driving input for the steering manoeuvrers. Exploiting the IPS representation (see Section 2.2), Prédhumeau et al., 2019 modified the SFM with a dynamical personal space, depending on the crowd density. In their model, pedestrians perceive obstacles within a perception zone and a inner attention zone (corresponding to the IPS). The model called CP-SFM by Zanlungo et al., 2011 explicitly considers pedestrian collision avoidance by imposing the future repulsive forces that will occur at the collision point. Model-based motion prediction approaches mostly rely on heuristic rules and hand-crafted geometric relations, thus they are computationally efficient and able to describe complex interactions. However, it is unclear whether humans follow these schemes. Similarly, Karamouzas et al., 2009, apply the *principle of least effort* to model pedestrian interactions, according to which humans will try to avoid unnecessary detours and follow energy-efficient trajectories. In the model by Luo et al., 2018 agents use a gap seeking strategy, i.e. they steer towards an empty space in the crowd in order to minimize the collision avoidance effort, while the agents behind the gap seekers perform a follower strategy. In a cognitive approach based on heuristic rules proposed by Moussaïd et al., 2011, pedestrians behaviour is guided by visual information by assuming that they seek unobstructed walking direction, possibly without deviating too much from the direct path to destination. Moreover, the desired walking speed is influenced by the need to maintain a distance from the obstacle. Vision is also exploited by Park et al., 2013, since their collision prediction model is based on the angle between the agent moving direction and the gaze angle towards another agent.

Single dynamic models are not sufficient to describe complex scenarios with multiple people, which, on the other hand, are addressed with multi-modal methods. Prior to the ascent of the machine learning, the Interactive Multiple Model filter (IMM) was a widely used technique for multiple motions tracking and prediction (Mazor et al., 1998). Back in the early 2000s, Farmer et al., 2002 used an IMM filter for tracking the motion of the head and torso of a human car occupant. The algorithm fused a stationary, a human stationary, and a pre-crash brake models, while the model transition probability was determined a-priori from analysing videos of pre-crash braking events and then simulating these events with crash dummy. Schneider and Gavrila, 2013 combined several basic motion models (constant velocity, constant acceleration, and constant turn) in a IMM framework. The multiple model approach by Lee et al., 2016 fuses the uniform and the turn motion models embedded in a unscented Kalman filtering, but they conducted all the experiments in simulation. An adaptive IMM with 8 different constant velocity models (one for each possible direction of movement) was implemented by Burlet et al., 2006, using the information coming from a monocular camera. In Madrigal et al., 2014 the SFM is adapted according to four motion models, namely: going straight, finding one's way, walking around, and standing still. The transition from one model to another is managed by an IMM framework, weighted with Particle Filter (PF) algorithm. Schulz and Stiefelhagen, 2015 combined sevealt motion models as in Schneider and Gavrila, 2013, and controlled the model transitions with an intention recognition system based on the human dynamics (position and velocity) and awareness (head orientation). Jiang et al., 2011; Jiang and Huynh, 2017; Chen and Tang, 2018 employed the IMM to track multiple pedestrian from monocular videos. Both these methods use the HOG human detector trained using pedestrians full bodies from recorded datasets, so that they are not so suitable for detecting pedestrians from the point of view of the robot (as discussed in Section 1.2.2). Conversely,Ogawa et al., 2011 propose ans approach to detect pedestrians using in-vehicle Lidar mounted inside the vehicle cabin, but without the help of a second type of sensor (such as a camera). In fact, they had to deduct the effect of the vehicle's relative motion from the measurements before carrying out the correction with the IMM. Hashimoto et al., 2010 equipped a robot with two laser scanners, to identify and cluster the points belonging to the legs and waist of people, respectively, and tracked their motion with an IMM estimator including a constant velocity model, a sudden acceleration/deceleration model, and a stop model.

Learning based. Machine learning techniques have the advantages of improving over time, adapting to different environments, and making predictions with low online computational effort, because most of it happens offline. However, performance depends on proper selections of the environment and the observed features. Moreover, if an area becomes more crowded, the prediction reliability worsens compared to model-based approaches.

A common approach to manage multiple future trajectories is to generate different motion modes. For this reason, many recent learning approaches implement multiple predictions to describe mixed motion behaviours. In Gupta et al., 2018 a Generative Adversarial Network (GAN) based approach is exploited with a novel social pooling framework to predict multiple trajectories while learning social norms. A similar GAN based framework with a social attention mechanism is proposed by Sadeghian et al., 2019. Both these approaches utilizes pedestrians past trajectories and scene context information, but do not consider the agents destinations. Conversely, Mangalam et al., 2020 use Variational Autoencoder (VAE) based network to infer a distribution of waypoints and obtain a multi-modal trajectory prediction. Deo and Trivedi, 2020 reformulated maximum entropy IRL to jointly infer waypoints and agent trajectories on a 2D grid defined over the scene. Following the success of Recurrent Neural Network (RNN) models for sequence prediction tasks, Alahi et al., 2016 propose a Long-Short Term Memory (LSTM) model which can learn human movement and predict their future trajectories, by assigning one LSTM to each person in a scene. Since the simple use of one LSTM model per person does not capture the interactions between pedestrians, neighbouring LSTMs are connected in the social pooling layer to learn spatially proximal information between the agents. The model was extended by Kothari et al., 2021a with an LSTM-based model with an interaction module sharing velocities of nearby agents.

The rationale behind the approaches these last two contributions is that human motion is driven by their intent, more specifically by where a person will go or perform an action. The motion prediction approaches can be roughly clustered around two main themes (Lasota et al., 2017). The first is to estimate how humans move toward a target which is assumed to be known (the estimation, thus, is on the characteristics of the motion). The second is to extrapolate the position of the goal by using information on the environment, on the past trajectories and on possible clues readable from the body language. Goal-conditioned methods are indeed regarded as inverse planning or prediction by planning, where the algorithm, usually a neural network, learns the final intent or goal of the agent before predicting the full trajectory (Mangalam et al., 2020). Being closely related to the grounded psychological methodologies, the inclusion of the pedestrian destination choice in the forecast of humans' motion has a positive influence on the prediction performance (Kielar and Borrmann, 2016), but nevertheless, dynamic goal inference based on semantic of the environment is still an open issue (Rudenko et al., 2019). Most of the existing research contributions rely on a predefined set of waypoints, estimated from observed trajectory data. For example, goals can be placed retrieving the directions pedestrians mostly head to (Foka and Trahanias, 2010), observing the preferred stop position, or partitioning the environment with a Voronoi-based method (Kanda et al., 2009). Similarly, Ikeda et al., 2013 split the long-term human trajectory into a sequence of sub-goals from the analysis of recorded data. On the other hand, there has also been some research on real time goal-based motion prediction. In Wu et al., 2018 a heuristic method to automatically determine goal positions on a 2D semantic gridmap is proposed. Cell transitions are predicted through discrete Markov chains. In Luber et al., 2010 the SFM was extended with a virtual goal as the hypothetical position that a person would reach if s/he moved with constant velocity. Kothari et al., 2021b modelled the probability distribution of the future human positions with a discrete grid of possible sub-goals, and mixed it an LSTM-based neural network Alahi et al., 2016 to learn information from the past motion of the agents and their interactions. The technique illustrated by Ikeda et al., 2013 is used in Karunarathne et al., 2018 to obtain the preferred sub-goal within a shopping mall. Their algorithm estimates the candidate waypoint by weighing the sub-goals visibility and reachability, and let a service robot to move side-by-side to a person. More recently the Y-net model proposed by Mangalam et al., 2021 computes a distribution for the future trajectories, after the semantic segmentation of the input image, jointly inferring both the final goals and intermediate waypoints of agents in the scene. Their approach suggests the relevance of combining stochastic (that is, the uncertainty due to randomness) and epistemic factors (e.g. the effect of the long term goals).

Rehder et al., 2018 proposed a recurrent Mixture Density Network (RMDN) to learn a mixture of potential destinations, later used by a fully Convolutional Neural Networks (CNN) to predict the path planning. The uncertainty is modelled with a mixture of Gaussian-von-Mises distributions. Senanayake and Ramos, 2018 introduced probabilistic Directional Grid Maps (DGM) to fit a mixture of von-Mises distributions of motion directions attributed to each grid cell of a discretised environment. Katyal et al., 2020 built a static grid of waypoints and embedded this additional information in a GAN to infer multiple human movement predictions.

2.2 Motion and interaction features

Most of the applications presented in this thesis rely on the prediction of the pedestrians' future positions in a short time horizon. As in this thesis, most of the related works collected in this chapter are based on machine learning techniques. As we believe that the outcome of the learning depends among others on the correctly formulation of input used to feed the neural network, we begin our discussion by illustrating which are the most significant human motion features for the trajectory prediction task (a summary of them is depicted in Figure 2.1). Current position and velocity of the target agent remain the main attributes in prediction task (Rudenko et al., 2019). As pointed out by Schöller et al., 2020, long motion histories do not provide a notable improvement by neural networks and environmental priors potentially affect their generalization to unknown environments. Pedestrian's orientation can be a intentionality indicator towards a preferred destination (Ferrer et al., 2017), since human beings walk ahead most of the time, so that the walking behaviour can be represented with nonholonomic motion patterns (Farina et al., 2017b).

Social cues and non-verbal communication. Humans utilize a wide range of cues to express their intentions. The most explicit include body motion, sounds, and direct verbal communication (Saunderson and Nejat, 2019). Less overt are proxemics, eye gaze, and subtle gestures (Vinciarelli et al., 2009; Cha et al., 2018). The amount of information that can be obtained from non-verbal communication is very high, since the latter represents more than 60 percent of the communication between two people (Rios-Martinez et al., 2015). While referring also to mobile robots, a social cue (which is the actual

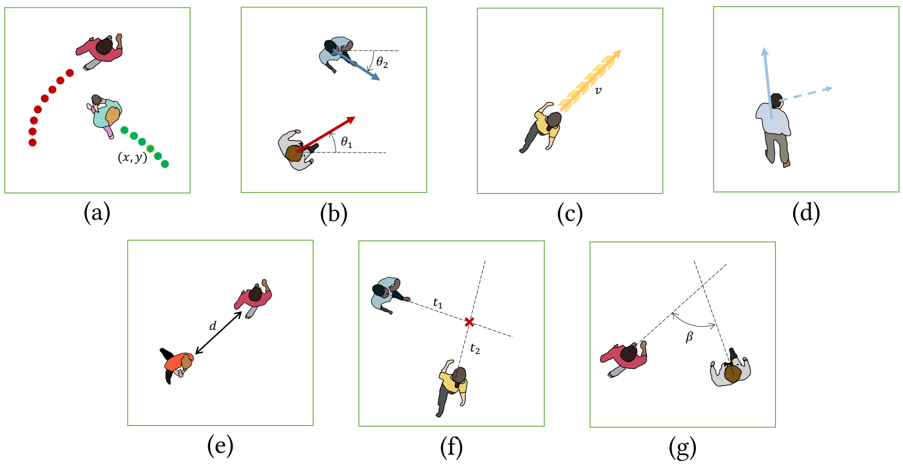


Figure 2.1: Representation of the most significant features to interpret and predict the human motion in the 2D space: (a) position in Cartesian coordinates, (b) heading angle, (c) linear velocity, (d) gaze direction. In the case of interaction between people, the significant measurable geometric characteristics are: (e) mutual distance, (f) time to collision, (g) relative bearing angle.

action) is a significant source of information used to understand the social signal (how the action is interpreted) (Fiore et al., 2013; Wiltshire et al., 2014). A human classified based dataset was used by Best et al., 2016 to cluster a small set of social cues and associate the most likely signal for each cluster. The two elements that constitute non-verbal communication are the action response and the information content (Cha et al., 2018). The first is the expectation that the observed human performs some action in response to the signal. This action acts as a type of feedback to the robot. The latter is the quality and type of information contained in a signal.

The use of social signals in human-robot interaction still suffers from complexity and scale. Dondrup et al., 2014 showed that some hesitation signals from the robot can be used as implicit feedback for humans to help them to infer motion intention of the robot. Moreover, since we are talking about non-humanoid robots, we have other non-verbal signaling modalities. Solutions can be to use directional lights or appropriate trajectory changes to communicate that an action has been taken. Other more explicit methods would be information screens or voice directions. Gestures are a form of non-verbal communication that include visible movements of the hands, arms, face, and other body parts to express an idea or meaning (Knapp et al., 2013). Among the broad category of gestures that researches has identified, interesting for mobile robotics are those that fall into the *metaphoric gestures*, i.e. the gestures that we use to depict concrete ideas or meanings, and the *deictic gestures*, employed for pointing to objects or areas in the environment. For example, in Che et al., 2020 the robot communicate implicitly its intention through its motion (slowing down and stopping), and explicitly through a wearable haptic interface worn by the human. In Watanabe et al., 2015, instead, the robot shows its navigational intentions with a projector emitting light on a common global coordinate frame. In their experimental evaluation, walking people found the intention communication intuitive and helpful for deciding what action to take. The authors propose an approach to show the navigational intentions of the robot with a projector emitting light on a common global coordinate frame. The information projected on the environment signals the future motion intentions of the in-group to the out-group resulting in smooth passing by interaction between the in-group and the out-group. In Narayanan et al., 2020 they trained a group convolution-based deep network to estimate human emotional states (i.e., happy, sad, angry,

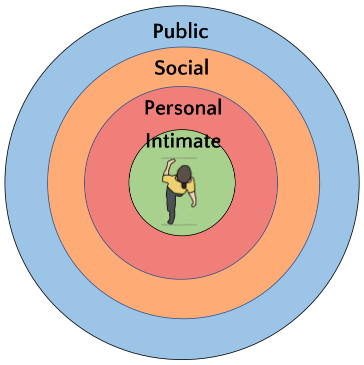


Figure 2.2: Proxemics interpersonal zones (Hall, 1969).

neutral) from 3D skeleton poses. The predicted emotions are in the robot navigation system to define a dynamic comfort space for the human. Influence of the environment is not considered. As there has been little work in this area, more research is necessary to understand how other communication modalities can achieve similar effects (Cha et al., 2018).

Proxemics. The anthropologist Edward C. Hall first introduced the concepts of *proxemics* in order to describe how humans use the interpersonal space and how they establish non-verbal communication channels tightly related to their interpersonal distance (Hall, 1969). Hall identified four circular zones (depicted in Figure 2.2) distinguished by the distance from the human body. Obviously, these distances are not rigid and vary according to age, culture, type of relation-ship and context (Table 2.2). Cultural differences play a crucial role in people behaviour (for instance, Hall's metrics were valid for US citizens), in fact some cultures avoid physical contact while others are more permissive. The

Zone	Size	Interaction
Intimate space	$0 - 45$ cm	Embracing, touching
Personal space	$45 - 120$ cm	Friends
Social space	$1.2 - 3.6$ m	Acquaintances and strangers
Public space	$> 3 - 6$ m	Public speaking

Table 2.2: Proxemics interpersonal zones (Hall, 1969).

original proxemics model describes only the explicit interactions between standing people, however its principle was adapted to depict how personal spaces are respected also in walking situation, and essentially four different shapes have been proposed, i.e., circular (Hall, 1969; Pradeep et al., 2016), egg-shaped (Hayduk, 1981; Amaoka et al., 2009; Lam et al., 2010; Nishitani et al., 2015; Toyoshima et al., 2018), ellipse-shaped (Helbing and Molnar, 1995; Ohki et al., 2010; Baig et al., 2014; Herrera et al., 2019) or asymmetric shapes (Rios-Martinez et al., 2015). The common finding is that people are more demanding about respecting their frontal space, while they pay less attention to what they have behind, thus personal space for walking pedestrian is widen on the front. Some related research results combined this mathematical formulation with learning-based methods. In Herrera et al., 2019, the robot starts with an elliptical social zone of no-collision of identified dimensions. Then, by collecting points-cloud depicting the presence of the human zone around the robot, the precise dimensions of the ellipse are inferred and adapted in real-time. Patompak et al., 2019 proposed to learn the personal area parameters

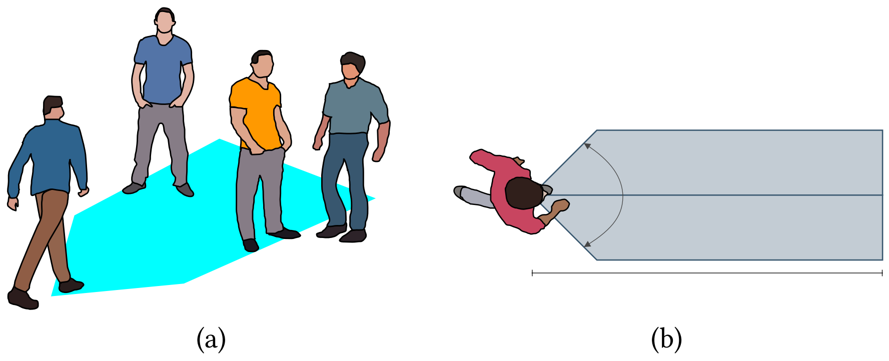


Figure 2.3: (a) IPS for a walking pedestrian according to Rios-Martinez et al., 2015. (b) IPS characteristic dimensions as described by Kitazawa and Fujiyama, 2010.

using a reinforcement learning algorithm. In Chik et al., 2019 a fully connected neural network is used to learn an adaptive proxemics costmap around humans, i.e. the human personal space boundaries that a robot should not cross, depending on pedestrians' relative linear position, angular position and velocity (with the lateral distance as the target output).

In the literature, comfort requirements are mostly understood as the distance that a robot has to keep from persons. Distance is not only useful for collision avoidance, but prevents the emotional discomfort humans may feel when a robot approaches them closer than usual. In this, proxemics plays a pivotal role in human social interactions for space negotiation Sebastian et al., 2017 and one of the major factors that influence social behaviours Vinciarelli et al., 2009, therefore it seems very versatile for human-robot interaction.

Information Process Space. Supporting the idea that personal space is dynamic (Herrera et al., 2019) and situation-dependent, in Kitazawa and Fujiyama, 2010 the Information Process Space (IPS) was introduced as the space within which all objects are considered as potential obstacles when a pedestrian is planning future trajectories. Their experiments showed that the IPS would have a cone shape rather than a semicircular one (see Figure 2.3). Furthermore, the subjects of their study paid no attention to the area with an angle of more than 45° from the walking direction. The IPS has sometimes been mentioned in literature (Park et al., 2013; Baig et al., 2014; Antonucci and Fontanelli, 2018; Prédhumeau et al., 2019; Narayanan et al., 2020), and occasionally it is expressed with other terminology, as in Moussaïd et al., 2011; Hu et al., 2013. Nevertheless, we consider it as an excellent way to model how humans manage their personal space while walking. Rios-Martinez et al., 2015 highlighted that the IPS could be strongly related to visual behaviour. However, pedestrian's visual behaviour is influenced by culture and age; moreover, head inclination can be misleading in inferring people intentions. We instead believe that it is strongly environment-related, deformable according to the available space. Other researchers considered a variable IPS. In Baig et al., 2014 the IPS is a circular sector that changes its aperture based on speed variations, ranging from 180° in high density situations to 90° during normal walking. A similar model is used in Park et al., 2013, where the sectional angle of the IPS is 90° when a pedestrian moves at its maximum walking speed and increases as the pedestrian loses his/her speed. When s/he stops, the angle of the IPS becomes 180°. In Hasan et al., 2019 the pedestrian's intention are modelled in a cone-shaped looking span, namely the Visual Field of Attention (VFOA). A summary of the cited contributions is reported in Table 2.3.

Reference	Shape	Dimensions
Kitazawa and Fujiyama, 2010	cone	$\theta = 45°$, $d = 4.5$ m
Moussaïd et al., 2011	semicircular	$\theta = 45° - 75°$, $d = 8 - 10$ m
Park et al., 2013; Baig et al., 2014	semicircular	$\theta = 45° - 90°$
Antonucci and Fontanelli, 2018	cone	$\theta = 45°$, $d = 2$ m
Prédhumeau et al., 2019	semicircular	$\theta = 110°$, $d = 7$ m
Hasan et al., 2019	cone	$\theta = 40°$, $d = 4$ m

Table 2.3: IPS shape and dimension in surveyed contributions.

In the case of motion of a single agent, the IPS maintains its effectiveness but loses meaning due to its definition. For consistency of terminology, we refer therefore to the Activity Space (AS), defined as the social space linked to actions performed by agents Rios-Martinez et al., 2015, that is, in the matter of walking pedestrians, the geometric space where a human being concentrate on and decide his/her future movements. The spatial information of the AS has sometimes been used as input feature for the learning-based motion predictors. In Antonini et al., 2006, agents plan their future positions within cells belonging to a semicircular space of dynamic length. Each alternative cell is defined by a certain speed and direction. In the paper by Pfeiffer et al., 2018, an obstacle aware LSTM neural network takes as input a 2D occupancy grid of the static obstacles and the information about nearby pedestrians using a 1D vector in polar space. In Section 4.5.1, we use a similar space representation (namely, the AoII) to determine the agent's future actions.

2.3 Socially aware robot navigation

To allow robots to interact correctly with people, they will also need to show appropriate social behaviour (Sebastian et al., 2017). In the study by Mutlu and Forlizzi, 2008, people were displeased when the robot did not respect social norms, as they would have expected. According to Rios-Martinez et al., 2015, a socially aware robot navigation is the strategy exhibited by a robot which identifies and follows social conventions (in terms of space management) in order to preserve a comfortable interaction with humans. The general idea of proxemics can be used to make robots choose an appropriate social distance for any interaction. For example, unless an explicit interaction is required, the robot should try to avoid passing very closely and entering into the intimate or personal space. Navigation with elderly generates problems, since the implicit signals deriving from the motion of the robot may be not correctly interpreted, or in general they can have difficulties to avoid the collision (Tamura et al., 2010). Pacchierotti et al., 2006 showed that even if there is no physical collision, to make a person feel safer an approaching robot should deviate at a proper distance (not less than 0.4 m), while Sardar et al., 2012 noticed that people show more compensatory behaviour when a robot invades their personal space than a human. Also, they sustained that people may judge robot functional intentions from the (social) behaviours the robot displays. The conclusion of Papenmeier et al., 2019 is that robots should avoid phases of decreasing velocity, such as during jerky movements, and should keep their orientation aligned with the motion direction in order to maximize human understanding of their intentions. As noted in Section 2.3, humans tend to adapt their trajectory collaboratively. Vassallo et al., 2017 investigated how humans modify their motion when in presence of robots. Their results shown that human-robot collision avoidance had similarities with the human–human avoidance, in terms of

Reference	Topic
Kruse et al., 2013	Collection of human-aware robot navigation approaches. Authors propose a general classification scheme grouping by features (comfortable, natural and social constraints) and by functionality (prediction, pose, path, behaviour and local planning).
Rios-Martinez et al., 2015	Comprehensive survey on proxemics theory and socially aware robot navigation.
Charalampous et al., 2017	Chronological trace of studies on robot navigation model, distinguishing the ones focused on metric mapping, semantic mapping, or social mapping.
Cheng et al., 2018	Enhancement of Kruse et al., 2013, with the inclusion of recent trend for deep learning. Related works are divided into: reactive based, predictive based, model based and learning based. The safety, comfort and energy cost evaluation metrics are highlighted.
Bonci et al., 2021	Survey on sensory equipment for human detection and action recognition in industrial environments.
Gao and Huang, 2021	Review of the evaluation protocols commonly used in socially aware robot navigation, highlighting pros and cons of different evaluation methods and metrics.
Möller et al., 2021	Recent survey on socially aware robot navigation, with a focus on approaches that involve the use of (depth) cameras.

Table 2.4: Summary of relevant surveys on socially aware robot navigation.

estimation of collision risk and anticipation. However, they also noticed that humans gave way to the robot, even though this choice was not optimal. This conservative strategy could be due to the lack of understanding of how the robot behaves or interacts with autonomous systems. Later, they found that the avoidance was better accomplished when the robot actively contributed to the interaction (Vassallo et al., 2018).

Different surveys have proposed a wide range of solutions to classify the human-aware navigation approaches. The research results considered for the literature review are listed in Table 2.4. In the following, we will cite the most relevant contributions, grouping them by the applied methodology. A comprehensive resume of the cited papers is reported in Table 2.5. As noted by Möller et al., 2021, assessing the effectiveness of a socially aware robot navigation method, unlike for instance goal-oriented robot navigation or trajectory prediction, is hard and possibly inaccurate, since the task performed by the robot is more complex, and the positive or negative effects it has on the humans sharing the space have to be taken into account (Saunderson and Nejat, 2019).

Model based. Predictive approaches are usually divided in model (or geometric) based and learning based. The former are based on assumptions on how people behave in general, while the latter exploit recorded data. In model-based approaches, the motion of people is generally represented with dynamic equations, and the future positions humans will reach are inferred with probabilistic reasoning. A lot of research results use simplified motion models, such as assuming that a person will walk straight at a constant speed (Ohki et al., 2010; Kruse et al., 2013), while recent approaches have started to regard the human motion as proactive or cooperative with respect to the robot actions. Back in the nineties, Tadokoro et al., 1995 predicted the human future motion using a stochastic process model. The probabilities of (positional) state transitions were heuristically determined depending on the environment. However, the robot's effect on human motion was purposely ignored. In Tamura et al., 2010 the SFM was used to predict the future trajectory for a pedestrian that can be aware of the robot (avoiding behaviour) or not aware (unavoiding behaviour). When the likelihood of the unavoiding behaviour is higher than the avoiding one, the robot has to avoid the collision by itself. The likelihood was measured with respect to manually classified patterns. Similarly, in Oli et al., 2013 each human being in the field of view of the robot is associated with an awareness flag (based on on visual clues, i.e. the gaze direction), which indicates whether s/he is aware or unaware of the presence of the robot. The paper by Ratsamee et al., 2015 extends the SFM by including a repulsive force due to face pose. The idea is that humans usually feel uncomfortable when they are observed, and look in the direction they will take when avoiding another person. Velocity Obstacle (VO) is a planner introduced by Fiorini and Shiller, 1998 which generates avoidance manoeuvres by selecting a legitimate velocity for the robot outside the collision cone, that is the space of velocities that would result in a collision with the moving obstacle. Since the VO does not consider the case where humans will participate in the avoidance, in Berg et al., 2008 the Reciprocal VO (RVO) was proposed. In Berg et al., 2011 the technique was further improved with the Optimal Reciprocal Collision Avoidance (ORCA), where all the agents proactively avoid each other, assuming that the same avoidance reasoning is respected. VO-based approaches are able to find an analytical solution that prevents collisions, however they assume perfect knowledge of the agents' future motion and generate rather homogeneous avoidance behaviours, thus results can be unrealistic. In Ferrer et al., 2017 an Extended-SFM was used to bring the robot closer to the reference pedestrian and make them walk side-by-side. The human future positions were predicted with a Naive Bayesian classifier. Then, in Ferrer and Sanfeliu, 2019 they introduced the Anticipative Kinodynamic Planning (AKP) to solve a multi-objective cost function that integrates humans future trajectories with respect to robot motion. Khambhaita and Alami, 2020 modelled the mutual avoidance by simultaneously optimizing both human and robot trajectories, mapping a least squares optimization problem into a graph representation. In Jin et al., 2020 the robot first assumes future cooperative behaviour from the human, then it switches between a reactive planner and a cooperative one depending on the confidence on the prior assumption. Human motion is predicted with the HSFM (Farina et al., 2017b) combined with collision prediction estimation (from Zanlungo et al., 2011). The motion planner in Johnson and Kuipers, 2018 uses a probabilistic model that incorporates social norms learned from observed data (e.g. the person remains to the right of the corridor). In Satake et al., 2012 a dataset of recorded human trajectories is used to build typical pedestrian patter in a shopping mall. A people approaching task is performed taking into account the person's awareness of the robot, obtained with geometrical relations between human and robot relative motion directions. In Bera et al., 2017 the human future trajectories are predicted by observing a set psychological cues mixed with proxemics theory. An Expectation Maximization process finds the most plausible motion model of the pedestrian, and the robot finds the less intrusive path. However, the pedestrian is intended as non-cooperative. In Lam et al., 2010 a navigation planner based on six different personal spaces (circle/egg-shaped with soft/hard contours) depending on passing priority rules is presented. First, the robot finds out a suitable free walking area, then it applies a potential-field approach to the corresponding personal area according to the rules (i.e. the higher priority will produce a stronger repulsive potential field). Mead and Matarić, 2017 accomplished the navigation of the robot with a reactive proxemics controller and a cost-based trajectory planner. The controller simply moves the robot to its goal pose based on the relative angles between the robot, the person, and the goal. The planner recognizes multimodal social signals (speech and gesture) produced by the human while the robot is moving to its goal. The human-aware path planner proposed in Cosgun et al., 2016 is divided into a static and dynamic component. The dynamic planner simulates the reaction of humans with the SFM and re-computes the path of the robot iteratively until the new one does not intersect the pedestrians' personal spaces. Two different social cues are used for corridor navigation in Lu and Smart, 2013, i.e., changing the path that the robot takes, and making or not eye contact with the human. However, their method assumes fixed human trajectories, thus does not take into account how humans might react to the behaviours of the robot as it approaches them. Herrera et al., 2016 modified the SFM through a fuzzy logic approach that incorporate social rules (that is, 14 different interactions) depending on the relative positions, orientations, distances and velocities between the robot and the human. Sebastian et al., 2017 used a Gaussian Mixture Model (GMM) to discriminate between different pedestrians behaviours and select the trajectory with highest social-appropriateness score. Summarising, model-based approaches do not need any training process and better reproduce humans behaviour, however they usually depend on a large number of parameters and suffer from representation incompleteness. Pure reactive planners are easy to implement and guarantee reliable collision avoidance. However, they are too short-sighted and lack of prediction ability.

Learning based. Since they are trained using real trajectories, learning-based methods better reproduce the human behaviour. But, conversely, the training phase needs a great amount of data and it performs well only for the chosen scenario. In Satake et al., 2009 a Support Vector Machine (SVM) was applied to classify 2 seconds of a pedestrian trajectory (using as features shape and velocity) into four behaviour classes: fast-walking, idle-walking, wandering, and stopping. Kuderer et al., 2012 proposed a Maximum Entropy learning method based on trajectory features such as time, acceleration, velocity, collision avoidance, topological variants. Their model yields a probability distribution over the trajectories of all agents involved in an interaction (humans and robots). The paper by Silva and Fraichard, 2017 is based on the observation that collision avoidance among humans is mutually solved. In case of two walkers having intersecting trajectories, each agent is expected to contribute a certain amount of *collision avoidance effort*, to solve the reciprocal avoidance task, which depends on factors like the crossing angle, time to collision and speed. The concept of the sub-goals conditioned navigation (see Section 2.1) in human motion models was also used as a navigation strategy for robots (Wang et al., 2008; Ye and Webb, 2009; Hong and Park, 2011). Deep Reinforcement Learning frameworks with set of social norms (Chen et al., 2017b) or collision avoidance policies Sathyamoorthy et al., 2020 are used to solve collision avoidance scenarios between humans and robots. In Che et al., 2020 the robot learns the best policy to avoid the human, then, it express its future intentions trough implicit and explicit communication, while Heiden et al., 2020b applied a RL solution fused with the concept of the Empowerment. Deep RL is used by Chen et al., 2019 to jointly model robot-human and human-human interactions, that subsequently are combined via a self-attention mechanism, to handle a variable number of pedestrian in the environment. The balanced trade-off between model-based and learning-based approaches is tackled by Graph Neural Networks Battaglia et al., 2018; Eiffert and Sukkarieh, 2019. For example, Manso et al., 2019 proposes a Graph Neural Network to model interactions (i.e., the edges of the graph) between the robot, humans and the environment (the nodes). The input features are the geometrical entities between the nodes. In contrast to IRL methods, Hamandi et al., 2019 adopted an imitation learning strategy to mimic humans navigation by eliminating the need of any explicit model. Similarly, in Gil and Sanfeliu, 2019 a SFM reward is embedded into an imitation learning approach. Convolution Neural Networks (CNN) are employed by Narayanan et al., 2020 to classify four different walking styles (related to as many emotions) after the skeleton gait estimation from the video stream taken by the robot. The navigation scheme combines this information with proxemic constraints to compute the socially aware path. The real-time application is burdened by time required to obtain the 3D skeletal tracking. A proxemic-costmap was developed in Chik et al., 2019 by learning a neural-network model using real human state data. The fully connected neural network takes as input the relative orientation, the relative velocity, and the relative distance between the agent and the robot, and learns the reciprocal lateral distance that humans prefer to maintain during interaction.

2.4 Structured neural network

The achievements reached by neural networks in recent years are indisputable. In domains such as image recognition, text analysis and generation, board games, etc. the networks used are very large (billions of parameters), and the size of the training datasets range from two to ten times the number of parameters. However, in the robotic field, it is often difficult, expensive, and sometimes dangerous to collect experimental data, as each device is different and often has unique characteristics. Conducting a thorough test campaign, thus, might be costly, time-consuming, or even impossible (Miller, 2019). Moreover, large neural networks necessitate sophisticated hardware and long inference times, both of which are unfeasible in mobile robots and other hardware domains. Another aspect to consider is the behavioural interpretability and predictability of the network output. In the areas mentioned above, if the network fails, it could lose a game or make a misinterpretation, but in robotics, an error could cause the breakage of the device or even worse. It is therefore clear that deep learning, in its current state, is clearly unsuitable for use in safety-critical contexts and scenarios without the essential precautions (Bahari et al., 2022). In robotics, to reduce the risk, the scientific community collects data through simulated environments, then transfers what has been learned in simulation to the real robot. However, the transfer is not painless. Although some approaches resolve the problem of retrieving the data, they do not guarantee predictability and stability. Moreover, such nets are not suitable for portable low-power hardware.

Artificial intelligence (AI) transparency, also known as *explainable* artificial intelligence (XAI), traces back outputs from AI algorithms to provide a way to understand what's happening in "human terms". XAI has recently gotten a lot of attention, in part due to the ratification of the General Data Protection Regulation law by the European Union, which guarantees the right to explanation for persons affected by AI decisions (Diallo et al., 2021). Users and human operators, on the other hand, have more faith in information systems that incorporate explainable AI.

Actually, the idea of including domain knowledge in neural networks is not new. Seminal works that described the idea in its basics were already published thirty years ago by Feldman et al., 1988 and Seidl and Lorenz, 1991. In Funahashi and Nakamura, 1993 we can find a theoretical background concerning the ability of neural networks to model dynamical system, which proves that an autonomous dynamical system can be approximated by a recurrent neural network to any degree of accuracy. In Chow and Li, 2000 this conclusion is extended to non-autonomous systems, i.e. subjected to an input. In Raissi et al., 2019 the concept of physics-informed neural networks is introduced, while Lu et al., 2018 showed that many existing deep neural networks can be interpreted as different numerical discretizations of differential equation (ODE). In these networks, despite being trained to comply with physical laws, the structure remains a fully connected neural network.

Reference	Input features	Modality	Interaction	Method
Tadokoro et al., 1995	position, grid map	M	avoidance	Stochastic process and genetic algorithm optimization
Fiorini and Shiller, 1998	position, velocity	M	avoidance	Velocity Obstacle (VO)
Satake et al., 2009	velocity, trajectory shape	L		Support Vector Machine classifier
Tamura et al., 2010	position, distance	M	avoidance	SFM controller
Lam et al., 2010	position, orientation, velocity	M	avoidance	Potential field
Berg et al., 2011	position, velocity	M	cooperative avoidance	Optimal Reciprocal Collision Avoidance (ORCA)
Kuderer et al., 2012	time, acceleration, velocity, collision avoidance, topological variants	L	cooperative avoidance	Maximum Entropy learning
Satake et al., 2012	angles between relative directions	M	approach	Expectation Maximization (EM)
Lu and Smart, 2013	position, eye gaze	M	avoidance	Dynamic costmap navigation
Oli et al., 2013	position, distance, eye gaze	M	avoidance	SFM controller
Ratsamee et al., 2015	position, distance, eye gaze	M	avoidance	SFM controller
Cosgun et al., 2016	position, angle, orientation	M	avoidance	A* search
Herrera et al., 2016	position, distance, orientation, velocity	M		SFM controller
Bera et al., 2017	position, velocity	M	avoidance	Expectation Maximization (EM)
Chen et al., 2017b	distance, orientation	L	cooperative avoidance	Deep RL
Mead and Matarić, 2017	position, relative angle	M	avoidance	Reactive proxemic controller
Sebastian et al., 2017	position, distance	M	avoidance	Gaussian Mixture Models (GMM)
Silva and Fraichard, 2017	crossing angle, crossing order, velocity, TTC	L	cooperative avoidance	ORCA and RL
Chen et al., 2019	position, orientation, velocity	L	avoidance	Deep RL
Chik et al., 2019	distance, orientation, velocity	L	avoidance	Fully connected NN
Eiffert and Sukkarieh, 2019	position, velocity	L	avoidance	RNN
Ferrer and Sanfeliu, 2019	distance, orientation	M	avoidance	Anticipative Kinodynamic Planning (AKP)
Gil and Sanfeliu, 2019	position, orientation, velocity	L	avoidance	Deep Imitation Learning
Hamandi et al., 2019	position, 2D scan	L	avoidance	Deep Imitation Learning
Manso et al., 2019	distance, orientation, relative angle	L	avoidance	Graph Convolutional Network (GCN)
Che et al., 2020	distance, velocity	L	cooperative avoidance	Maximum entropy IRL
Jin et al., 2020	position, orientation, velocity, TTC	M	cooperative avoidance	Extended-HSFM and MPC
Khambhaita and Alami, 2020	position, orientation, velocity, distance, TTC	M	cooperative avoidance	Timed Elastic Bands
Narayanan et al., 2020	image frame, 2D scan	L	avoidance	CNN
Sathyamoorthy et al., 2020	image frame, 2D scan	L	avoidance	Deep RL
Heiden et al., 2020b	position, velocity	L	avoidance	RL with Empowerment

MPC: model predictive control
ORCA: optimal reciprocal collision avoidance
NN: neural network
RNN: recurrent neural network
CNN: convolutional neural network
RL: reinforcement learning
IRL: inverse reinforcement learning
M: model-based; L: learning-based

Table 2.5: Features, characteristic parameters, and methods used in socially aware robot navigation approaches.

Reference	Network type	Training dataset
Clark, 1994	Highly structured	Synth
Evans et al., 2000	Med structured	Real
Garcia et al., 2007	Highly structured	Real
Kilic et al., 2011	Highly structured	Synth
Andre et al., 2013	Highly structured	Real + Synth
Broad et al., 2018	Med structured	Synth
Greydanus et al., 2019	Med structured	Synth
Qin et al., 2019	Med structured	Synth
Da Lio et al., 2020	Highly structured	Real
Heiden et al., 2020a	Med structured	Real
Günther et al., 2020	Med structure	Synth
Roehrl et al., 2020	Med structured	Real
Hochlehnert et al., 2021	Med structured	Synth
Masi et al., 2021	Highly structured	Synth

Table 2.6: Surveyed papers on structured neural network approaches.

Publications of structured neural networks can be found in different engineering areas. Summarized list is reported in Table 2.6. In Clark, 1994, Clark demonstrated that different non linear mass-spring-damper combinations (specifically, the Duffing oscillator and the Van der Pol oscillator) can be accurately modelled through simple feed forward neural network, where the learning task in used only for the parameters fitting. Some works have proposed hybrid framework with neural network and physical model. Broad et al., 2018 questioned about the possibility to learn the linearisation mapping of model of the state model of a general time-varying dynamical system, but the network was still a black box. Heiden et al., 2020a proposed a differentiable simulation engine for rigid-body dynamics where both the residual of the analytical model and the parameters of a neural network can be trained thought gradient-based optimization. Evans et al., 2000 used an hybrid Mixture Density Network (MDN) to obtain Cartesian wind vector components from satellite scatterometer data, that is by modelling conditional probability density functions. Their rationale was that a Gaussian mixture model with enough many kernels and a neural network with enough hidden units can closely approximate a desired conditional density. In Günther et al., 2020; Roehrl et al., 2020, they place side by side a standard model and the learnable control, that is a black box neural network that compensates for all the aspects that are difficult to model. Hoverer, this mixture of techniques cannot exploit features like the controller's training through the model. Qin et al., 2019 investigate the use of the ResNet in a one-step and multi-step method to approximate a dynamic system. The multi-step method exploits two approaches the recurrent ResNet and the recursive ResNet the first used for uniform time step, and the second for variable time step. The recurrent network are able estimate the dynamic up same seconds. Da Lio et al., 2020 explored the different performance between neural networks with and without pre-wired structure, recursive and non-recursive, for the longitudinal dynamics of a car with gears and two controls (brake and engine). Pre-wired structures of both convolutive and recurrent networks showed better interpretability of the training process, while the convolutive architecture was more accurate and robust. Hochlehnert et al., 2021 used structured networks for learning contact dynamics in data-limited regimes, thus by embedding in the neural network the Euler–Lagrange equations for non-deformable body impacts. The Lagrange approach on which is based their simulation results shown that the addition of an idealised touch feedback sensor positively affects the model's accuracy. Similarly, Greydanus et al., 2019 constrained the neural network to learn Hamiltonian formulations, thus ensuring energy conservation, but the structure of the networks was a fully connected, therefore black box. However, we notice that, in both cases, the network's properties are guaranteed by the training and not by the network's structure, which is instead considered in the approach presented in 4.4. In Kilic et al., 2011; Masi et al., 2021 the authors split the system into components, and each component is modelled with a fully connected neural network. An SNN and an Extended Kalman Filter (EKF) were compared by Andre et al., 2013 for the estimation of the internal parameters of a lithium-ion battery equivalent circuit model. Both the algorithms shown similar results, however the SNN achieved more computational speed and better memory usage during the offline inference of the battery state of health. Nevertheless, the EKF is an established technique easier to implement and with no need of a training phase. Structured networks were used by Garcia et al., 2007 for decoupling saturation-induced saliencies of the zero-sequence carrier-signal voltage of an AC machine. The performance were comparable with existing model-based methods. These last two approaches are not flexible and they require a very high level of knowledge of the physical system, comparable to that required for analytical modeling.

Chapter 3

Human positioning and tracking

We have stated that service robots are becoming increasingly pervasive in environments where human beings actually carry out daily duties. One of the main enablers of this revolutionary change is the capability of these robotic systems to interpret the human scene, detecting positions and predicting motions of the bystanders.

In this chapter, we will investigate the effectiveness of different people tracking systems. First, we investigate the use of a stereo camera pair using state-of-the-art image processing and classification algorithms. In order to increase the potential of this solution, we examine a different alternative based on a single monocular camera in combination with a laser scanner to reduce uncertainties, increase robustness and reduce the computation time. Finally, we evaluate the tracking performance of a last-generation millimeter-wave radar as a potentially cheaper and more robust alternative.

All the solutions proposed act as input to an estimation algorithm to continuously track the position of human beings moving in the surroundings. Our experiments carried out in laboratory conditions allowed us to characterise and evaluate the performance of the different solutions and revealed their level of suitability to the type of applications for actual service robots considered in this thesis. The final comments are stated at the end of the chapter.

3.1 Overview

The key research question now is if the task of identifying and tracking people in the workspace of the robot is an activity that can be executed with low cost hardware and low computational overhead. The specific purpose of this chapter is to assess the performance of people trackers that, based on cameras, laser scanners, or radar fixed on the robot chassis, estimate the presence of humans in the robot's surroundings.

The problem of detecting and tracking objects or human beings in a natural environment is not novel and many different solutions have been developed in the recent years for different purposes, ranging from healthcare applications (De Vito et al., 2014) to human beings counting (Vicente et al., 2009), from applications in sports (Nam et al., 2014) to object tracking (Doyle et al., 2013). A largely incomplete list of such applications includes people counting, security monitoring, and trajectory planning or optimization for service robots (Laoudias et al., 2018). Of particular relevance are systems that fuse information coming from different sensing sources for people tracking in natural environments (Colombo et al., 2014; Wang et al., 2012). The most commonly used in mobile robotics are namely image-based, laser-based, and radad-based sensors (Figure 3.1).

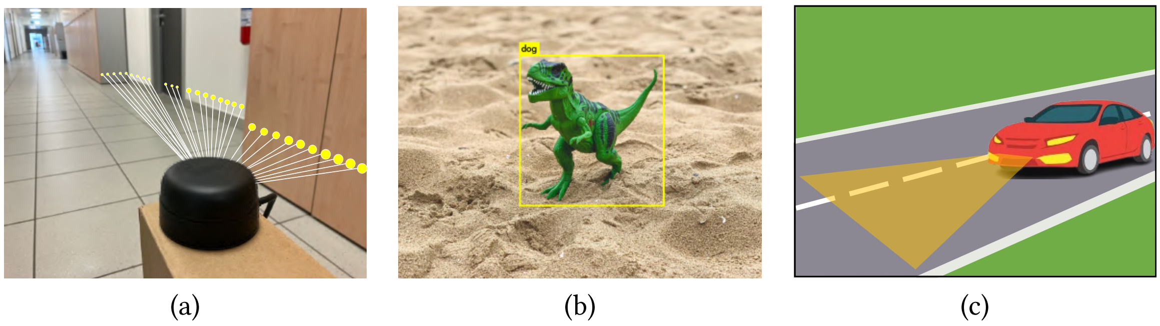


Figure 3.1: Sensors for mobile robotics: (a) laser-based, (b) camera-based, and (c) radar-based.

Image-based sensors (Figure 3.1-b), due to their flexibility and relatively high amount of data that can provide, are becoming quite popular and effective in many measurement applications (Shirmohammadi and Ferrero, 2014). Stereo or RGB-D cameras provide reliable depth information about the environment. In addition, different skeleton-based approaches for estimating the people's pose and motions from RGB images have been developed and are available as public domain software components (Presti and La Cascia, 2016). However, the simple use of visual information suffers form different limitations such as the sensitivity to lighting conditions, and the high computation times required to obtain depth and/or pose information (Laganière et al., 2006).

Laser-based sensors (Figure 3.1-a), on the other hand, are more accurate on larger ranges than cameras and are not affected by environmental nuisances, making them a perfect fit for indoor and outdoor operations (Shinohara et al., 1992; Demeyere et al., 2007). Furthermore, laser scanners are known to be less computation-hungry than vision-based approaches, and they also preserve people identity and privacy, since the information is achieved as a point cloud. On the other hand, distinguishing the human figure from a 2D slice of range data is a non-trivial task, which many authors have tackled seeking to exploit the geometric properties of the points that correspond to humans (see Arras et al., 2007 as a fundamental reference for this line of work).

A third alternative to monocular cameras and laser-based sensors is using a stereo or a RGB-D camera. These sensors provide a 3D map of the environment, while the detection of the human body can be performed on the same data provided by the cameras. A common problem between using 3D and monocular cameras is that the system is liable to illumination changes and to the presence of artefacts, which could endanger a correct recognition of the scene.

For this crucial robustness limitation, the camera system is usually paired with other types of sensors, for example with a sonar rig (Vadakkepat and Jing, 2006) or a range finder (Chavand et al., 1997). Particularly interesting is the combination of a 2D laser scanner and a monocular camera. The former can provide depth information, while the latter allows the system to detect and classify the different entities in the scene. The idea is not entirely new and in Table 3.1 we summarise the different solutions in the literature that fuse laser and camera data in order to track human targets. As shown in the table, all the cited papers use separate detection techniques for the LIght Detection and Ranging (LIDAR) and the camera, with the results fused in a post-processing phase.

Reference	LIDAR detection	Camera detection	Fusion method	Calibration
Monteiro et al., 2006	GMM and Majority Voting scheme classifiers	Haar-like features with AdaBoost classifier	Bayesian-based	Quadratic error minimisation mapping
Premebida et al., 2009	15 features as Arras et al., 2007 with GMMC classifier	HOG descriptor and COV features with FLDA classifier	Bayesian-based	Planar checkerboard and algebraic constraints, see Zhang and Pless, 2004
Gate et al., 2009	3 features likelihood classification	AdaBoost classifier	Bayesian-based	-
Tamas et al., 2010	3 leg features with GMM classifier	AdaBoost classifier	LIDAR r.s. to camera r.s. where targets overlap	Least square error minimisation mapping
Fotiadis et al., 2013 Garzon Oviedo et al., 2015	63 features with AdaBoost classifier	HOG descriptor with SVM classifier	Bayesian-based	As Zhang and Pless, 2004
Truong et al., 2015	Same as Arras et al., 2007	RBG-D-based human detection	Particle filter-based	-
Miller et al., 2016	Dynamic Means clustering	VeryFast pedestrian detector	LIDAR r.s. to camera r.s. with angular overlapping	-
Wunderlich et al., 2017	10 leg features histogram-based identification	Leg histogram-based identification	LIDAR r.s. to camera r.s. where targets overlap	Linear mapping
Aguirre and García-Silvente, 2019	3 leg features with Random Forest classifier	Mask-RCNN people detection	LIDAR r.s. to camera r.s. where targets overlap	As Kassir and Peynot, 2010; Zhang and Pless, 2004

Table 3.1: List of related results on 2D laser range and camera-based sensing system for pedestrian detection.

The last category of sensors falls in the class of Radio Frequency (RF) transceivers. Among the manifold wireless technologies recently adopted, radars based on phased-array transceivers at 60 GHz are gaining momentum. Since the solutions based on Received Signal Strength Intensity (RSSI) from reference nodes (e.g., Wi-Fi access points) suffer from major spurious fluctuations in both time and space, alternative techniques based on low-level Time-of-Flight (ToF) measurements of Ultra-Wide Band (UWB) or Chirp Spread Spectrum (CSS) signals have been proposed (Hur and Ahn, 2010; Rohrig and Telle, 2011; Loyez et al., 2014; De Angelis et al., 2016). UWB systems have been used for vital sign sensing as well as for near-field imaging applications since the early 2000s (Sheen et al., 2007). More recently UWB systems have been used to detect people trapped or behind walls (Loschonsky et al., 2009). This idea has led to the implementation of various kinds of millimeter-wave communication schemes and transceivers for indoor localization over the years (Ebelt et al., 2014). Moreover, millimeter-wave (mm-wave) RF signals can be used to implement radars with a range of a few meters. Radars (Figure 3.1-c) have a number of advantages compared with cameras, i.e.

- They are cheaper and need simpler signal processing algorithms;
- Target detection is more insensitive to obstructions;
- People's identity and privacy are preserved, as it is difficult to associate a cloud of points to a specific individual;
- Radar signals are not affected by changing illumination conditions: they can operate indoor and outdoor regardless of light variations or changeable weather conditions.

Even if the idea of implementing low-cost radars for people localization traces back to more than 10 years ago (Zhang et al., 2006), the solutions based on antenna arrays have evolved especially in the last few years as a result of the research activities on massive multiple-input multiple-output (MIMO) systems, particularly for 5G communications (Mendrzik et al., 2018; Lin et al., 2018). For instance, in (Guerra et al., 2017) a channel model for personal radar applications based on mm-wave antenna arrays is described and is used to scan the environment and reconstruct a map of it. Moreover, in (Guidi et al., 2016) the influence of different design parameters on mm-wave radar-based mapping performance is analyzed. Over the last few years, mm-wave RF signals have been used to localize and track a moving target in a variety of ways, e.g., by using Angle-of-Arrival measurements, by combining path loss with Angle-of-Arrival data or through Angle Differences-of-Arrival algorithms (Kanhere and Rappaport, 2018; Olivier et al., 2016). Since mm-wave radars operate at frequencies higher than 10 GHz (Park et al., 2014; Ng et al., 2017), arrays of small and closely spaced antennas can be easily implemented on the platform itself at a reduced cost and with a small footprint. In this case, the position of the target can be estimated from the Phase Differences of Arrival (PDoA) of the backscattered waveforms received by different antennas (Ebelt et al., 2014). This general approach has been also used to implement low-cost commercial off-the-shelf Systems-on-Chip (SoCs) embedding both RF circuitry and digital signal processing algorithms for position estimation.

The solutions adopted for localization and tracking take a different shape depending on the considered target type. When tracking a robot, it is possible to merge odometry and inertial sensors data with the information derived from visual landmarks or Radio Frequency Identification (RFID) tags with given coordinates in the considered reference frame (Nazemzadeh et al., 2017). When instead the entity

	Camera	Radar	LIDAR	Ultrasonic	Sensor fusion
Ambient light independent		✓	✓	✓	✓
All-weather operation		✓		✓	✓
Operating range	mid-far	near-far	mid-far	near	near-far
Angular resolution	high	medium	high	low	high
Cost	high	medium	high	low	medium
Data	classification, texture	motion measurement	2D/3D mapping	low-cost ranging	

Table 3.2: Comparison of sensor characteristics.

to be localized and tracked is a human, data fusion can still be a possibility using different sensing technologies. In general, the purpose of data fusion algorithms is to improve tracking accuracy, robustness and scalability, while reducing the cost and the complexity of the equipment. For instance, it is possible to use wearable inertial platforms to estimate the user's relative motion and RF transceivers (preferably based on standard, widely available Wi-Fi or Bluetooth modules) to measure the distance from fixed anchor nodes (Colombo et al., 2014). However, to increase the flexibility of robots, it would be preferred to track targets who are not required to carry any kind of electronic device. Visual, laser, and radar sensors share this characteristic. In conclusion, all the different sensors reported have unique strengths and ability to extract information from the surroundings, as listed in Table 3.2 borrowed from Gardill, 2019. The ultrasonic sensors share most of their features with the radar sensor, but with lower performance, therefore we have not considered them in our evaluation preferring the latter. Cameras offer the best for human classification and obstacle detection, while with radars and lidars it is possible to obtain precise spatial measurements of the entities in the neighbours of the robot. However, depth cameras and 3D lidars are the most expensive sensors, while radars are substantially cheaper. We can notice that only the sensor fusion is able to encompass the largest number of scenarios.

We evaluated the tracking performance of three different solutions. Since we are interested only on the tracking performance of the algorithm, we assume that the robot is standing in a fixed position with its sensors pointed toward the environment, and that it is localised in the environment within defined thresholds (Nazemzadeh et al., 2017; Magnago et al., 2018). First, we will assess the performance of two recently developed human tracking algorithms, namely YOLO, a fast and robust detector for objects in natural images (Redmon et al., 2016a), and OpenPose, which can perform 2D real-time multi-person pose estimation and segmentation, detecting the anatomical key-points for each of the detected human being (Cao et al., 2017). YOLO is a general classification framework, while OpenPose is specialised for human targets. The two methods are evaluated both in isolation and then fused together using a Kalman filter, using a depth camera (Section 3.2 for the technical description and 3.3 for the performance assessment and the experimental set-up). The second solution, presented in Section 3.4, employs YOLO on a monocular camera and in combination with a LIDAR, which is a novelty with respect to the solutions reported in Table 3.1. The advantages of this solution with respect to the use of 3D cameras are manyfold:

1. The use of YOLO reduces the computation time and enables a faster sampling rate;

2. The reduced computation time and the lower cost of the hardware components bring about a general cost reduction with respect to the use of a 3D camera;

3. The use of separate systems for depth estimation and classification improve the system robustness when one of the systems fails (e.g., if the camera is temporarily "blinded" we can still use the LIDAR sensor for some time assuming a certain degree of continuity in the scene).

The experimental evaluation is addressed in Section 3.5. A low-cost commercial mm-wave radar is finally analysed. Despite the promising usefulness for mobile robotics, the actual performance and the limitations of these platforms are still quite unclear. The information reported in the data sheets is usually partial or incomplete. In addition, a proper experimental characterization of these devices in dynamic conditions is hard to do and results are inaccurate or poorly presented. Section 3.6 summarizes the principle of operation of the radar-based platform and its main parameters of interest. The static and dynamic performance of the system under test are reported in Section 3.7.

3.2 Camera-based tracking

First, we want to assess the performance of a people tracker when YOLO and OpenPose are used with only visual information and for the service robotic application described in Section 3.1. The tracking system comprises two main components, i.e. the sensing system and the estimation algorithm. Notice that the main role of the estimation algorithm is to improve the performance of the sensing system and, whenever possible, increase the robustness of the solution by making it resilient to occasional sensor reading failures.

3.2.1 Stereo camera system

OpenPose (Cao et al., 2017) performs 2D realtime multi-person pose estimation from captured images and extracts the anatomic description, in terms of body, hands, face, and feet, for each of the detected person (Simon et al., 2017; Wei et al., 2016). Its architecture is made by a two-branch multi-stage Convolutional Neural Network: the first branch iteratively predicts the confidence map of body parts locations in the image (Wei et al., 2016), while the second iteratively predicts the Part Affinity Fields (PAFs), which is made of 2D vectors that connect body parts belonging to the same limb (Cao et al., 2017): the concatenation of these two components allows the estimates of the 2D coordinates of the body key-points (Figure 3.2-a).

YOLO is a fast and strong detector, in the form of a bounding box, for generic objects that can be found in a stream of images (Redmon et al., 2016a). The detection is performed by splitting the image into a cell grid and then by predicting the bounding boxes using cluster centroids references and anchor boxes (Redmon and Farhadi, 2017). Each box predicts the object class that it may contain using multi-labelled classification (Redmon and Farhadi, 2018a). YOLO returns a set of bounding boxes each containing 5 components, i.e. the (x, y) pixel coordinates of the centre of the box, its width and height, and the confidence, representing the similarity of height-width ratios between the predicted box and the ground truth box. YOLO is faster than most of the other state-of-the-art models, however, it becomes inaccurate with groups of objects or when the detected object has configurations different from the reference data. In addition, sometimes the boxes do not contain the object perfectly (see the

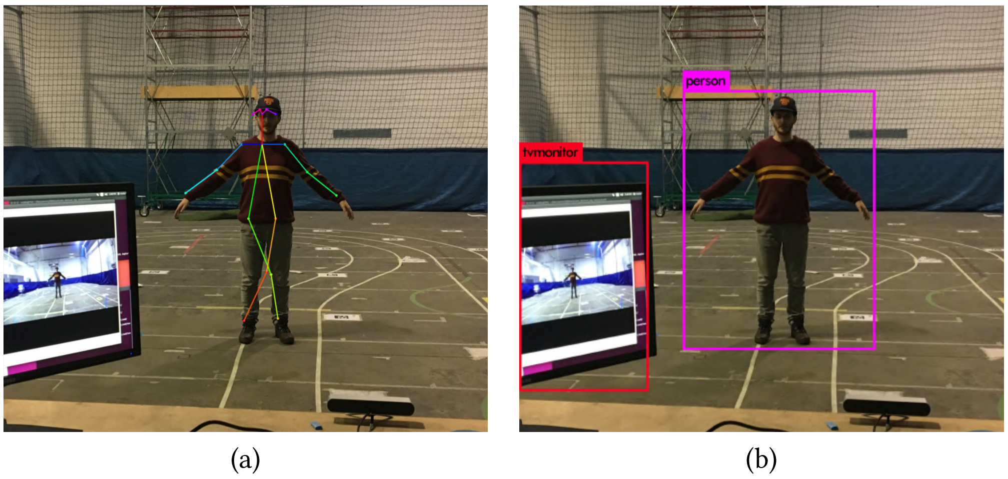


Figure 3.2: Examples of human detection in camera frame obtained with OpenPose (a) and YOLO (b). We notice that YOLO can detect more objects (e.g. the pc monitor in the frame), but the detection can be inaccurate (for instance, since the person has the arms raised, the spatial hindrance is badly represented. OpenPose, instead, provides a better detection of the human joints, even if some missdetections can occur, in fact in the example the detection of the legs is wrong.

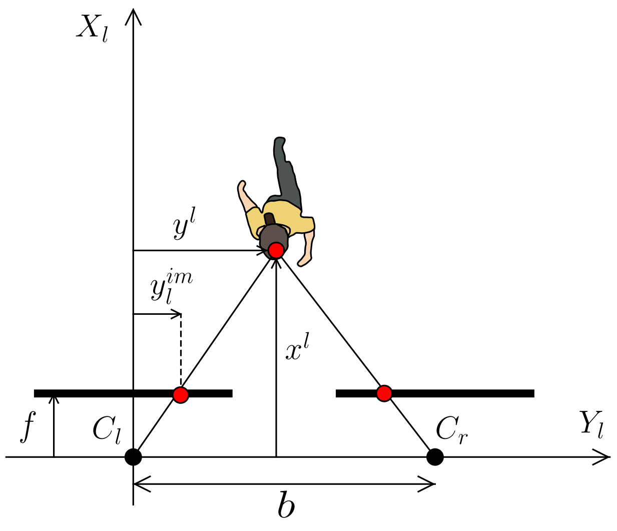


Figure 3.3: Stereo camera model.

example in Figure 3.2-b). While OpenPose determines the 3D object position using a mono camera or a stereo camera pair (see Figure 3.2-a), it demands more computing power than YOLO, which in turn needs the tracked object depth to perform 3D reconstruction (Redmon and Farhadi, 2018a).

By denoting with $\langle C_l \rangle = \{X_l, Y_l, Z_l\}$ and $\langle C_r \rangle = \{X_r, Y_r, Z_r\}$ the frames of a pin-hole parallel stereo camera system pair related to the right and left vision sensor, respectively (see Figure 3.3), the depth map can be obtained by calculating the difference between the horizontal coordinates of the pixels of an object image in the left and the right image (i.e., disparity), which is directly related to the distance x along X_r and X_l of the object from the image planes Mokrzycki, 1994 (see Figure 3.3). Since finding the corresponding pixels between the two cameras is a non-trivial task, several methods have been developed in the literature, ranging from pixel-based matching costs (such us, squared differences or truncated absolute differences), to area-based matching costs (e.g. SAD, SST, STAD). Beyond the accuracy of the specific method adopted, the task requires a proper computing power, which scales with the image dimension. To overcome this limit and let the system to work in real-time when using OpenPose, the key-points of the skeleton detected in both images are used as corresponding pairs, which are then used as sparse disparity maps. In particular, we took the left camera as a reference, so for each pair of points we measure the horizontal and vertical distances that the key-points of the right image would have related to the ones in the left image. Pairs of coincident points are discarded. To find the skeletons, the images are pre-filtered with a 2D Gaussian smoothing kernel for computation efficiency. The struct returned by OpenPose contains a set of key-points according to the 18-point model trained on the COCO dataset (Lin et al., 2014). The key-points are divided for each detected person and come with the image coordinates, a unique identifier and a confidence score in the range $[0, 1]$. On the basis of the stereo camera measure, the 3D coordinates (x_i^l, y_i^l, z_i^l) of the i-th key-point in $\langle C_l \rangle$, whose coordinates in the left image plane are denoted as (x_l^{im}, y_l^{im}), can be readily obtained as

$$\begin{aligned} x_i^l &= \frac{bf}{x_l^{im} - x_r^{im}} = \frac{bf}{d_i} \\ y_i^l &= \frac{x_l^{im} x_i^l}{f} \\ z_i^l &= \frac{y_l^{im} x_i^l}{f} \end{aligned} \tag{3.1}$$

where b is the distance between the centres of the two cameras (i.e. baseline, see Figure 3.3) and f is their focal length in pixels, supposed to be identical for the Y_l and Z_l coordinates (Hamzah and Ibrahim, 2016). Similarly, the equations for the right camera can be immediately obtained using (3.1). The human being whose skeleton has been detected will have an estimated position that is the weighted sum, where the weights are the OpenPose computed confidence, of all the reconstructed skeleton key-points 3D coordinates. As a consequence of the depicted algorithm, OpenPose reconstructs excellent but fairly slow (e.g., in post-processing on a quad-core 2.7GHz i7 computer it takes in the average of 500 ms per frame) human positions.

On the other hand, the YOLO library is needs only one image to detect objects in the scene, however it needs depth information. By using a stereo camera pair with built-in 3D reconstruction algorithms (precisely, the ZED camera in Figure 3.4 is used[1]), the 3D coordinates of these objects identified in the image is immediately available. More precisely, assuming to have a fixed camera and only one person in

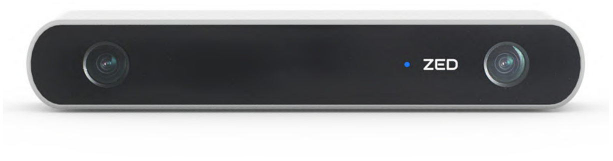


Figure 3.4: ZED camera[1].

the scene, we can start a stream of images from both cameras. YOLO, then, carries out a detection of the persons/objects in the images, and returns a set of bounding boxes which contain the detected objects. The 3D positions of the objects, given by the centre of their bounding boxes, are obtained by finding the points in the point cloud that are linked to a reference window of pixels nearby the bounding-box

[1]https://www.stereolabs.com/zed

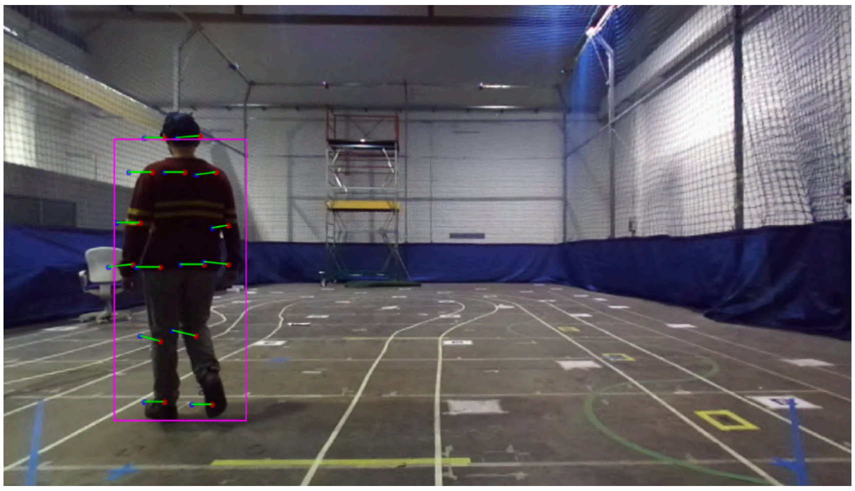


Figure 3.5: Human detected with OpenPose (blue and red points) and with YOLO (purple box).

centre. A graphical representation of the detected human body posture by OpenPose and YOLO are reported in Figure 3.5. Whenever the information relies on a monocular camera, we assume they are captured from the left camera only, i.e. expressed in $\langle C_l \rangle$.

3.2.2 Sensing system model

As described in Section 3.2.1, the sensing system relies on two different algorithms: OpenPose and YOLO, both using the ZED stereo camera images. From the perspective of the people tracking system, we will assume that each of the two solutions provides the position of a person in $\langle C_l \rangle$, which has been taken from reference. In particular, since the vertical position is not of interest while only the motion on the $X_l \times Y_l$ is of interest (see Figure 3.6), we will assume that both OpenPose and YOLO returns such positions as the centroid of the detected human. In particular, for OpenPose we have for the j-th detected person

$$\begin{bmatrix} x_j^l \\ y_j^l \end{bmatrix} = \frac{1}{n} \begin{bmatrix} \sum_{i=1}^{n} x_i^l \\ \sum_{i=1}^{n} y_i^l \end{bmatrix}, \tag{3.2}$$

where the summands are the skeleton detected points as in (3.1). Similarly, for YOLO we have an averaged function as in (3.2), where instead the points are given by the detected points in the box.

In both cases, we will assume that the measurements for the j−th person are given by

$$\begin{aligned} m_j^{op} &= \begin{bmatrix} x_j^l \\ y_j^l \end{bmatrix} + \varepsilon^{op}, \\ m_j^{yo} &= \begin{bmatrix} x_j^l \\ y_j^l \end{bmatrix} + \varepsilon^{yo}, \end{aligned} \tag{3.3}$$

where ε^{op} and ε^{yo} are the noises affecting the measurement process carried out by OpenPose and YOLO respectively and characterised in Section 3.3.

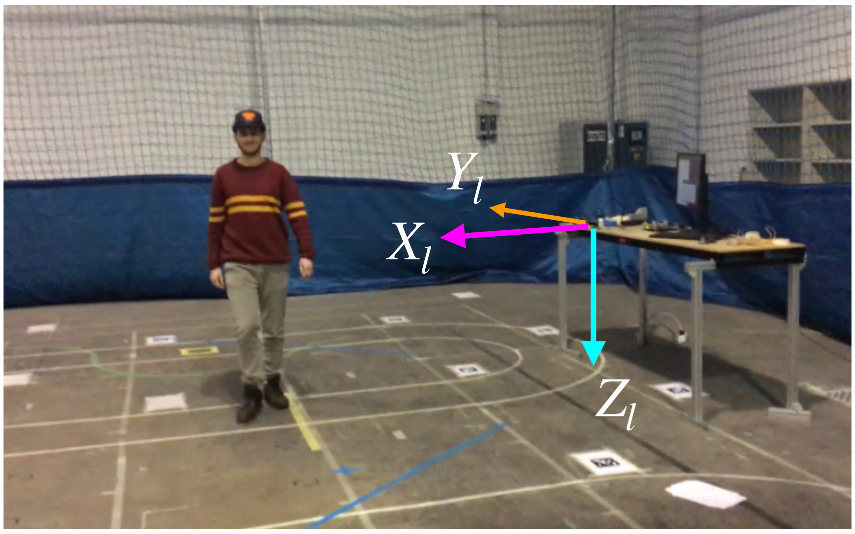


Figure 3.6: Camera reference system adopted in the stereo camera configuration.

3.2.3 Estimation algorithm

The estimation algorithm is built taking into account the fact that the person that is moving, once detected for the first time, follows an unimodal probability density function (pdf). In particular, to limit the computational cost and to follow well established literature results, we assume that this pdf is Gaussian. As a consequence, two Kalman filters are considered, whose difference relies on the motion model adopted. The first one is related to the well known constant velocity model (Ratsamee et al., 2013), a simplified version of the Social Force Model (SFM) (Helbing and Molnar, 1995). In such a case the human being is modelled as a material point moving on plane with constant velocity. Therefore, being $s(k) = [x(k), y(k), v_x(k), v_y(k)]^T$ the state at time k comprising the position and the velocity of the human on the plane of motion, we have for the Kalman Filter (KF)

$$\begin{aligned} s(k+1) &= \begin{bmatrix} 1 & 0 & \delta_t & 0 \\ 0 & 1 & 0 & \delta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ y(k) \\ v_x(k) \\ v_y(k) \end{bmatrix} + \begin{bmatrix} \frac{\delta_t^2}{2} & 0 \\ 0 & \frac{\delta_t^2}{2} \\ \delta_t & 0 \\ 0 & \delta_t \end{bmatrix} \begin{bmatrix} \nu_x(k) \\ \nu_y(k) \end{bmatrix} = \\ &= As(k) + B\nu(k), \end{aligned} \tag{3.4}$$

where $\nu(k)$ is the acceleration noise affecting the velocity variations that is supposed to be $\nu(k) \sim \mathcal{N}(0, N)$ (indeed no knowledge is available about the actual motion intention of the human, hence the random walk choice), with N being its covariance matrix.

The second model instead assumes that humans are moving according to a smooth dynamic, an idea based on direct observations (Arechavaleta et al., 2008). In this case, the motion model boils down to a unicycle dynamic (Farina et al., 2017a). Hence, by denoting with $q(k) = [x(k), y(k), \theta(k), v(k)]^T$ the state of the system at time k (where $\theta(k)$ is the orientation of the human and $v(k)$ its longitudinal velocity), we immediately have for the Extended Kalman Filter (EKF)

$$\begin{aligned} q(k+1) &= \begin{bmatrix} x(k) + \delta_t v_k \cos(\theta(k)) \\ y(k) + \delta_t v_k \sin(\theta(k)) \\ \theta(k) \\ v(k) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \delta_t & 0 \\ 0 & \delta_t \end{bmatrix} \begin{bmatrix} \eta_\theta(k) \\ \eta_v(k) \end{bmatrix} = \\ &= f(q(k)) + B\eta(k), \end{aligned} \tag{3.5}$$

where $\eta(k)$ represents the acceleration noise affecting the velocity variations and the perturbations of the system orientation $\theta(k)$, which are supposed to be defined as $\eta(k) \sim \mathcal{N}(0, E)$, with E being its covariance matrix.

In both cases, the initial state and the measurement update step are determined by the measure of the human being given by (3.3). Similarly, the filters estimation covariance matrices, $P_s(k)$ and $P_q(k)$ respectively, are initialised based on the covariance of ε^{op} or ε^{yo} in (3.3) (depending on the sensing system model adopted).

3.3 Tracking system results

For the experiments, we have adopted a C++ implementation of OpenPose based on Tensorflow and OpenCV[2]. Since our algorithm is executed in MATLAB, we have used the MATLAB API to call a Python module to construct an OpenPose class, pass the image as a NumPy array and get a struct with the pose positions whenever a person was detected in the image. Similarly, a C++ implementation of YOLO has been adapted for the available ZED stereo camera[3] adopted, which allows to execute the object detector in real time while performing image acquisition with the ZED camera.

The videos were recorded with the ZED camera and a Jetson TX1[4]. Since we needed time to run YOLO and to save the frames as well, the acquisition rate was about $3 \div 4$ fps. An OptiTrack Flex13 reference localization system[5] equipped with 14 cameras Prime 13 (with a resolution of 1280×1024 pixels and a frame rate of 120 Hz) was used to capture ground truth data of the participant and the ZED camera positions. The ZED camera was placed so that the image planes were orthogonal to the plane of motion (see Figure 3.6), and its 3D position related to the reference system origin was calibrated with ground truth data. We recorded several times different types of trajectories, i.e. linear paths with constant distance from the camera, diagonal and circular paths (see Figure 3.7). At first, the process covariance matrices N and E used in Section 3.2.3 are obtained computing the errors between the predicted state given by the Kalman filter different implementations and the ground truth. For the measurement covariances associated to ε^{op} and ε^{yo} in Section 3.2.1, we evaluate the error of OpenPose and YOLO on a large set of random paths. Figure 3.8 depicts the empirical Probability Mass Functions error along the X_l and Y_l for both the sensing systems adopted. Such results are used as covariance matrices for the joint Gaussian distributions hypothesised.

[2]https://github.com/ildoonet/tf-pose-estimation
[3]https://github.com/stereolabs/zed-yolo
[4]https://developer.nvidia.com/embedded/jetson-tx1-developer-kit/
[5]https://optitrack.com

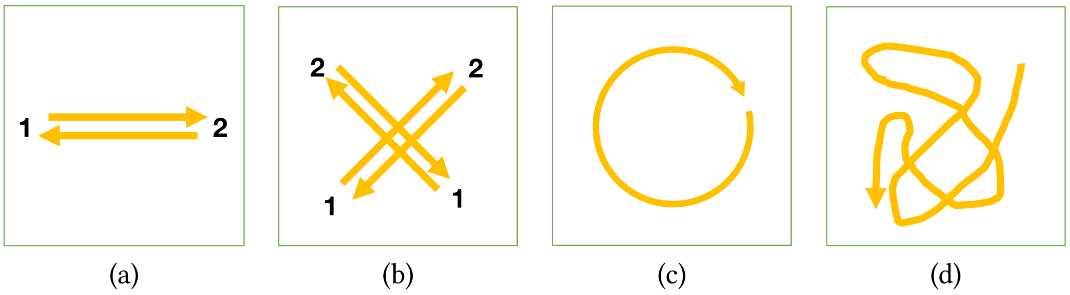


Figure 3.7: Types of trajectories for the experimental evaluation: (a) linear, (b) diagonal, (c) circular, and (d) random walks.

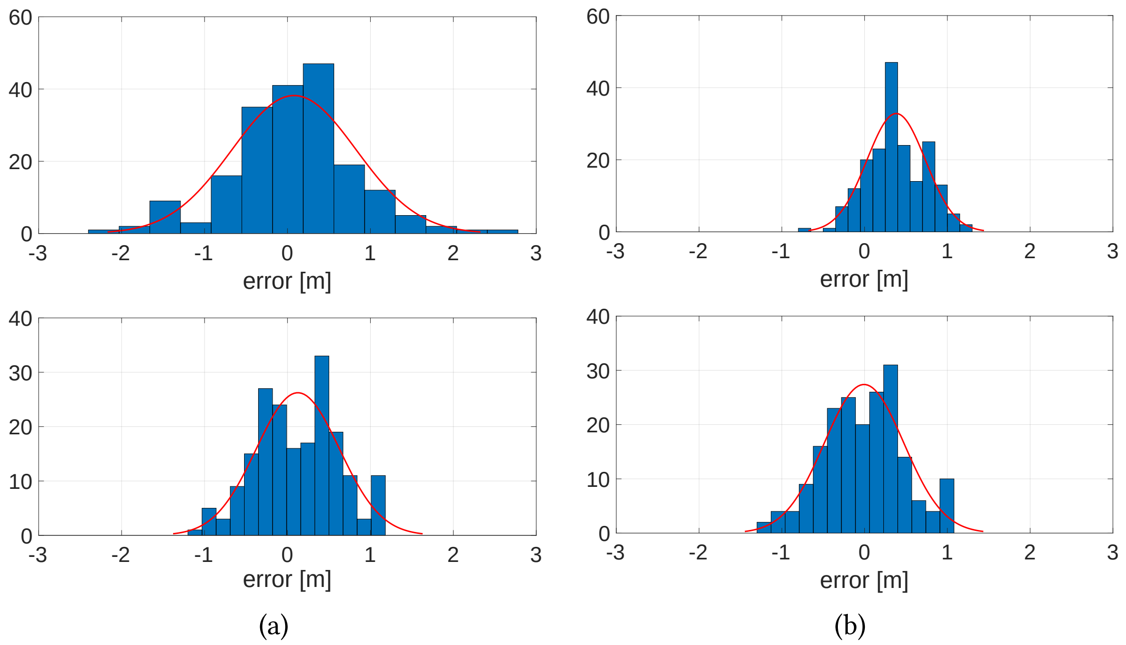


Figure 3.8: Error distribution along the X_l and Y_l axes for (a) Openpose and (b) YOLO.

Table 3.3 shows the experimental covariance matrices elements of the filtering error for the (x, y) positions with respect to the ground truth with both the Kalman filters designed. The term ρ represents the correlation coefficient between the two quantities. All the three different types of trajectories were tested. The best results are obtained using the EKF and the measurements given by YOLO or a mix of the two tracking algorithms. These results reflect the performance achievable with Openpose and YOLO. Indeed, OpenPose turns to be quite effective in detecting the human being skeleton, i.e. it is robust with respect to partial occlusions due to its well learned models (see the dashed line in Figure 3.10 for the X_l and Y_l coordinates). Nevertheless, when it is compared with the ground truth (dotted line in Figure 3.10), the detection accuracy is quite low with several outliers. An explanation of this phenomenon can be the different position of the body points between the two images, which leads to an incorrect averaged estimate. On the contrary, YOLO is not always capable of detecting the pose of the person, however when it succeed, the accuracy is relevant and there is an absence of outliers (solid line in Figure 3.10). The main reason behind this behaviour is the fact that YOLO uses the X_l estimated position given directly by the ZED camera through triangulation, which leads to a more accurate estimate when the person is detected.

		KF	EKF
Linear	OpenPose	$\sigma_x^2 = 0.3138, \sigma_y^2 = 0.2853, \rho = -0.4668$	$\sigma_x^2 = 0.1548, \sigma_y^2 = 0.2126, \rho = -0.1800$
	Yolo	$\sigma_x^2 = 0.0244, \sigma_y^2 = 0.1862, \rho = 0.2893$	$\sigma_x^2 = 0.0256, \sigma_y^2 = 0.1545, \rho = -0.3036$
	Mix	$\sigma_x^2 = 0.0578, \sigma_y^2 = 0.2217, \rho = -0.1760$	$\sigma_x^2 = 0.0717, \sigma_y^2 = 0.1943, \rho = -0.1873$
Diagonal	OpenPose	$\sigma_x^2 = 0.3740, \sigma_y^2 = 0.2586, \rho = -0.6719$	$\sigma_x^2 = 0.3105, \sigma_y^2 = 0.1934, \rho = -0.3903$
	Yolo	$\sigma_x^2 = 0.0957, \sigma_y^2 = 0.1683, \rho = -0.9170$	$\sigma_x^2 = 0.0897, \sigma_y^2 = 0.1400, \rho = -0.8209$
	Mix	$\sigma_x^2 = 0.1391, \sigma_y^2 = 0.2200, \rho = -0.7734$	$\sigma_x^2 = 0.1383, \sigma_y^2 = 0.1794, \rho = -0.6113$
Circle	OpenPose	$\sigma_x^2 = 1.2866, \sigma_y^2 = 0.2709, \rho = -0.4702$	$\sigma_x^2 = 0.4832, \sigma_y^2 = 0.2320, \rho = -0.1911$
	Yolo	$\sigma_x^2 = 0.1630, \sigma_y^2 = 0.1529, \rho = -0.2075$	$\sigma_x^2 = 0.1126, \sigma_y^2 = 0.1167, \rho = -0.1607$
	Mix	$\sigma_x^2 = 1.1887, \sigma_y^2 = 0.2346, \rho = -0.5110$	$\sigma_x^2 = 0.6968, \sigma_y^2 = 0.2034, \rho = -0.1458$

Table 3.3: Error covariance matrix data obtained for the KF and the EKF, using the different combination of measurements: OpenPose only, YOLO only and a combination thereof (denoted with *Mix*).

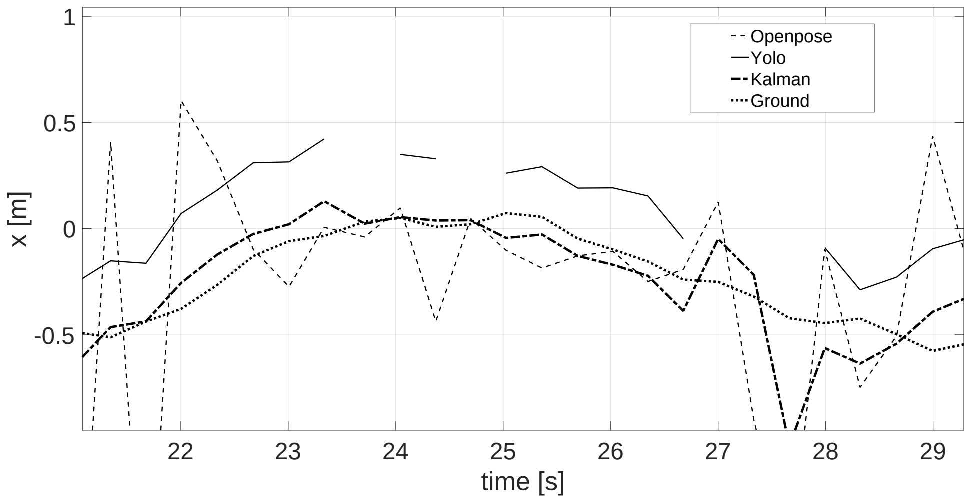
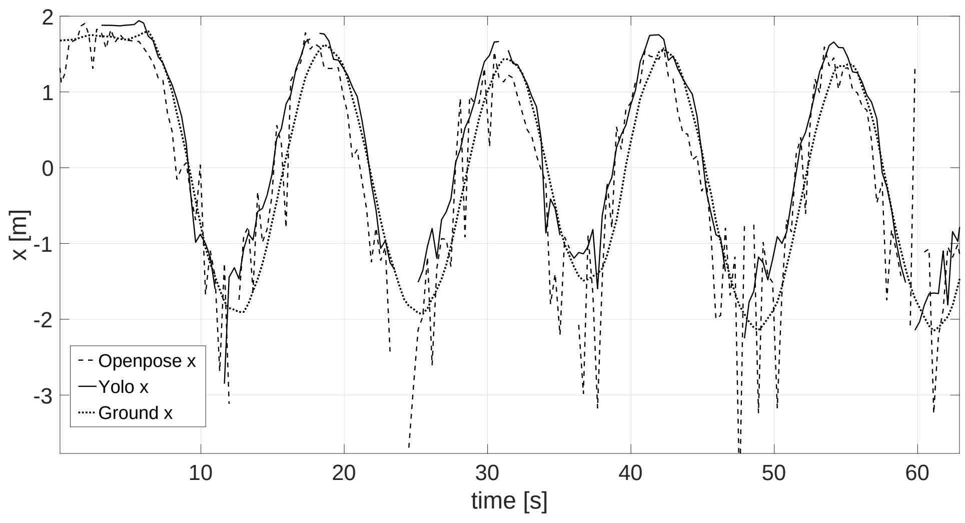
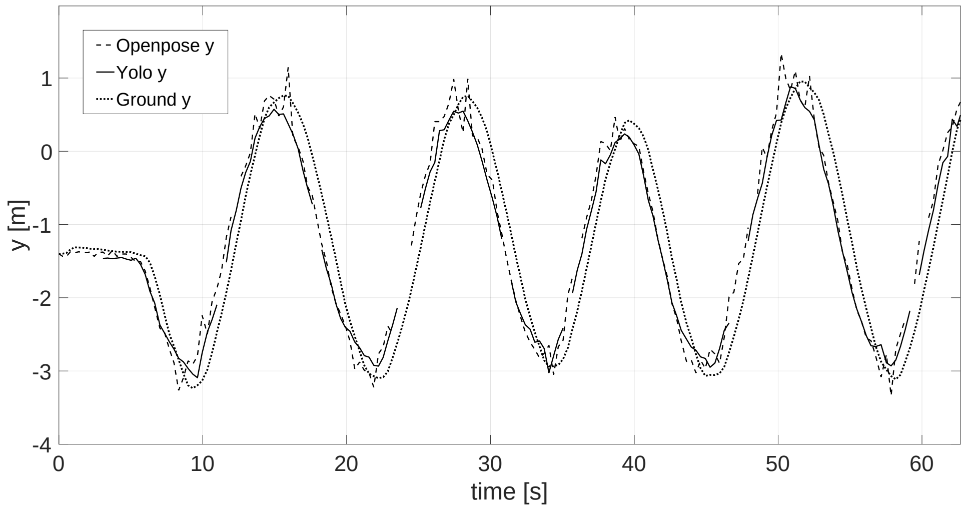


Figure 3.9: Filtering error when detection failures happen for YOLO (solid line) and when the OpenPose returns quite noisy data (dashed line): the EKF estimates (dash-dotted line) are quite close to the ground truth (dotted line).

As an example, Figure 3.11 shows the results of both the Kalman filters along a circular path using the different combinations of the sensing algorithms. The graphs report the time evolution of the distance d between the estimated point in the $X_l \times Y_l$ and the corresponding ground truth point. Notice that even though the results with the YOLO measurements (Figure 3.11-b) are better than those obtained with the OpenPose (Figure 3.11-a), YOLO suffers of intermittent detection of the human in the image space. To this end, Figure 3.9 reports the EKF estimates behaviour when some YOLO detection failures occur (it may be seen that the corresponding line is intermittent). Notice how the Kalman filter has the capability to build up a system that is more robust and more accurate by properly fusing the sensing data: indeed irrespective of the intermittent YOLO measurements (solid line) and of the quite noisy OpenPose measurements (dashed line), the EKF estimates (dash-dotted line) proves to close to the ground truth data (dotted line). Similar results can be obtained also for the KF case. In Figure 3.12 we reported a comparison between all the possible combination of techniques for a random and quite



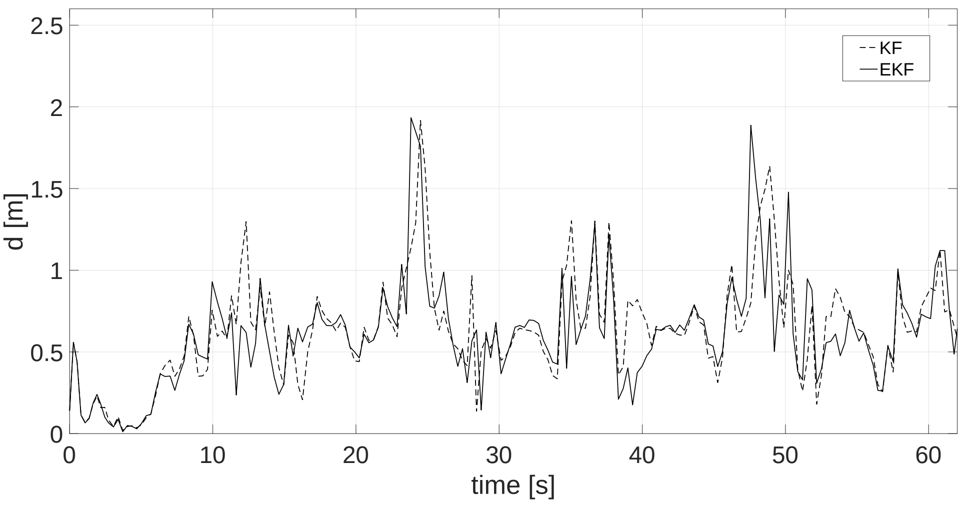
(a)



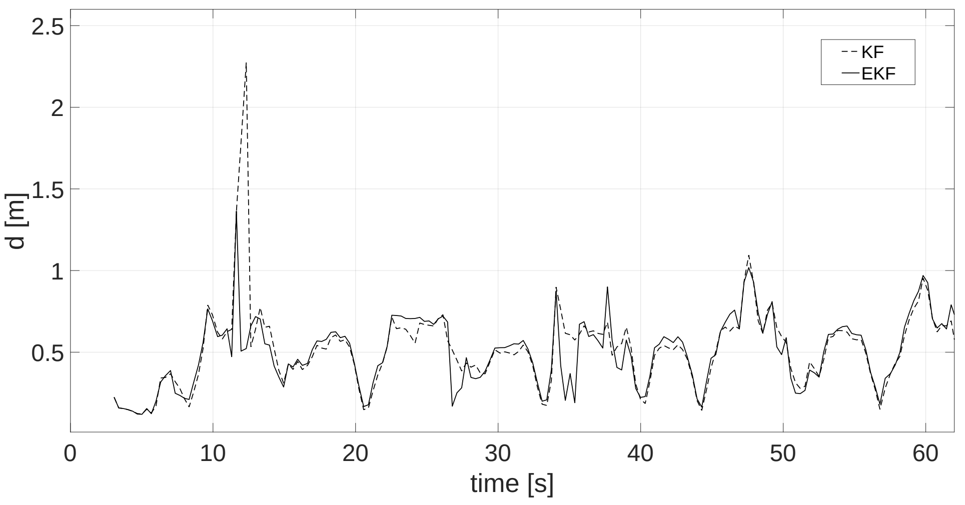
(b)

Figure 3.10: YOLO (solid line), OpenPose (dashed line) and ground truth (dotted line) comparison along the (a) x and (b) y coordinates when the person follows a circular path.

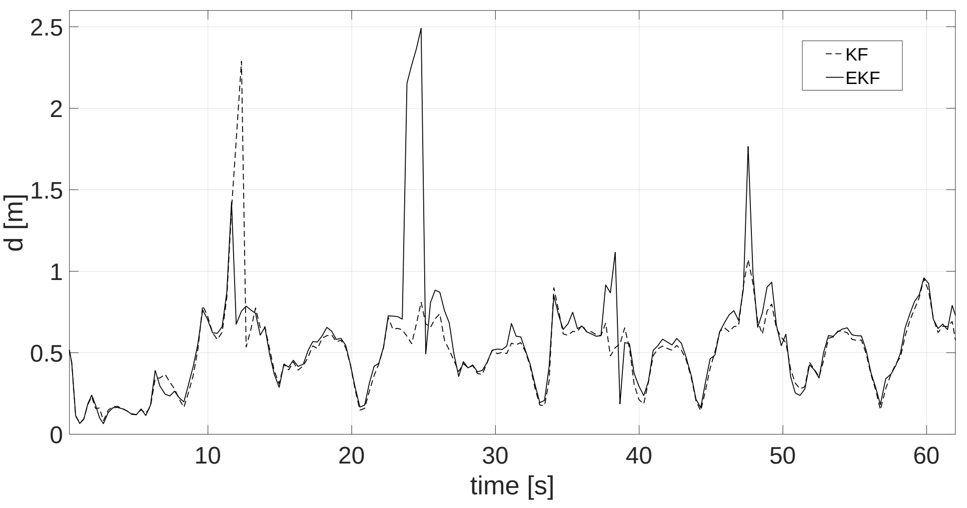
complex path (the ground truth is depicted with the blue lines). The tracking with OpenPose is quite noisy with both the KF and the EKF (Figure 3.12-a,b), with an increase of estimation error especially in the full scale areas of the sensor in the x and y directions. However, the detection in the image space is very robust, so very few missdetections occurred throughout the experiment. On the other hand, YOLO provides better measurements with less fluctuations with respect to the ground truth (Figure 3.12-c,d), and the Extended Kalman Filter (Figure 3.12-d) gets smoother trajectories when compared to KF (Figure 3.12-c). Unfortunately the number of failed detections is higher, as for example in the last part of the experiment (after about 110 seconds), where the measurements are completely absent. Finally, the combination of OpenPose and YOLO in the mixed approach Figure 3.12-e,f) allows to obtain precise estimates with a reduced number of missdetections, since the second detector compensates for the shortcomings of the first. However, in the mixed approach the computation time increases as we have to run two models together at a time.



(a)

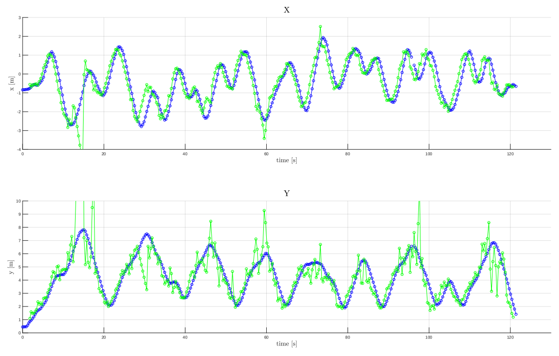


(b)

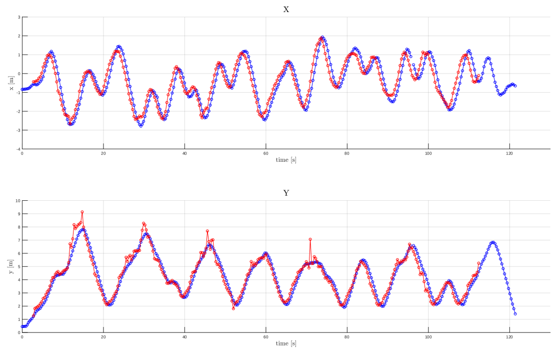


(c)

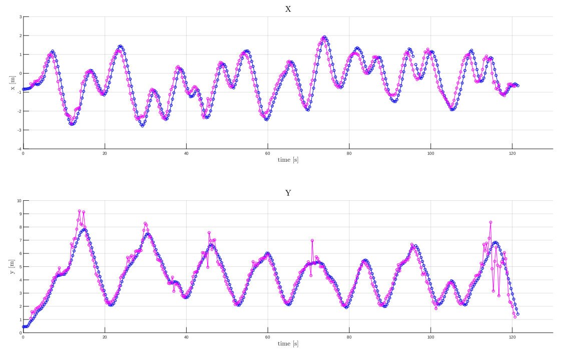
Figure 3.11: Distance errors for the KF (dashed line) and the EKF (solid line) in the circular path example using (a) only OpenPose, (b) only YOLO, (c) and both measurements.



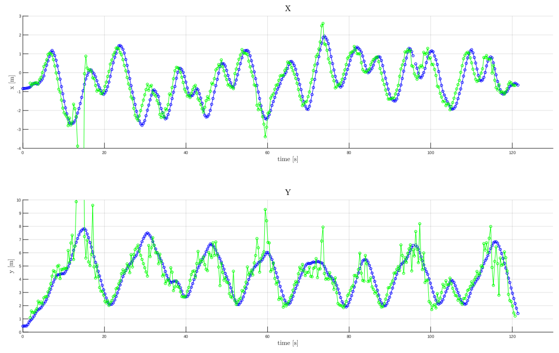
(a)



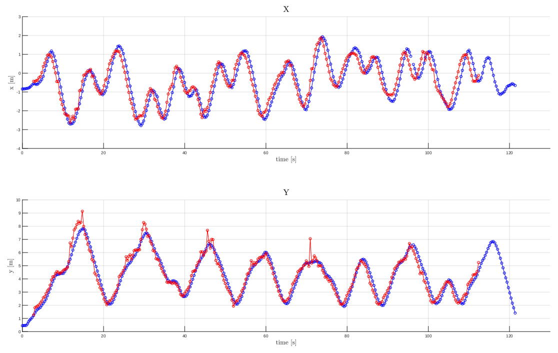
(c)



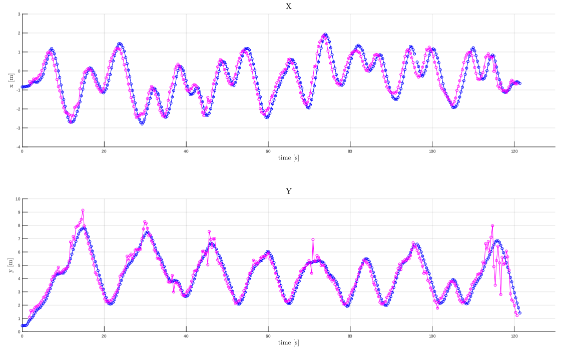
(e)



(b)



(d)



(f)

Figure 3.12: Comparison of KF (a) and EKF (b) estimates with OpenPose (green lines), KF (c) and EKF (d) estimates with Yolo (red lines), and KF (e) and EKF (f) estimates with the *mix* approach (magenta lines), with ground truth in blue, in the random path.

3.4 LIDAR-camera tracking

The second sensing system is based on the fusion of a LIDAR and a monocular camera (to save computation time), with YOLO working as the image processing algorithm. The laser scanner (the RPLidar A2[6] in Figure 3.13-a) employed has a view of 360° and maximum measuring distance up to 6 meters at typically 10 frames per second. The sensor and the camera were rigidly mounted on the same base, with parallel and aligned axes, previously calibrated using ground truth data. Hence, the mapping between points from the camera reference system $\langle C_l \rangle$ to the laser reference system $\langle L \rangle$ is obtained through a linear correspondence (see Figure 3.14).

3.4.1 LIDAR system

Each scan delivered by the laser scanner provide a sequence of n measurement points in the form of $\mathcal{P} = \{p_1, \dots, p_n\}$, represented in polar coordinates as $p_i = (r_i, \alpha_i)$, i.e. a range and an angle expressed in the system reference frame. Those points can be represented similarly in Cartesian coordinates, i.e. $p_i = (x_i, y_i) = (r_i \cos \alpha_i, r_i \sin \alpha_i)$. Figure 3.15 reports an actual scan from the available LIDAR, whose Cartesian coordinates are expressed in $\langle L \rangle$. At the time t_k scan, the measured points

Figure 3.13: (a) RPLidar A2 laser scanner[6] and (b) ELP camera[7].

are filtered and grouped into clusters based on the mutual Euclidean distances and on the richness, i.e. on a minimum number of sensed points for each cluster (see Algorithm 3). We get then a set of m_k clustered objects, each identified by the object centroid $o_j = (x_j, y_j)$. Given two sets of objects $\mathcal{O}_k = \{o_1(t_k), \dots, o_{m_k}(t_k)\}$ and $\mathcal{O}_{k+1} = \{o_1(t_{k+1}), \dots, o_{m_{k+1}}(t_{k+1})\}$, taken in two consecutive time instants t_k and t_{k+1} and possibly having $m_k \neq m_{k+1}$, we adopt a distance threshold to decide either for correspondence, i.e. the same object is tracked along the scans, or to allocate a new object. Due to the measurement noise of the LIDAR, we must also add a time threshold to introduce an inertia in updating objects, and filter possible spurious detections. Consequently, the tracking algorithm keeps track of the objects whose centroids were updated for T_a time instants, while it removes those whose centroid does not find correspondences for T_o scans (see Algorithm 1).

3.4.2 Sensing system model

Since the LIDAR is positioned parallel to the horizontal plane of the world reference system, the coordinates of $o_j(t_k)$ are comparable with the coordinates given by the bounding box of YOLO to retrieve the

[6]https://www.slamtec.com/en/Lidar/A2

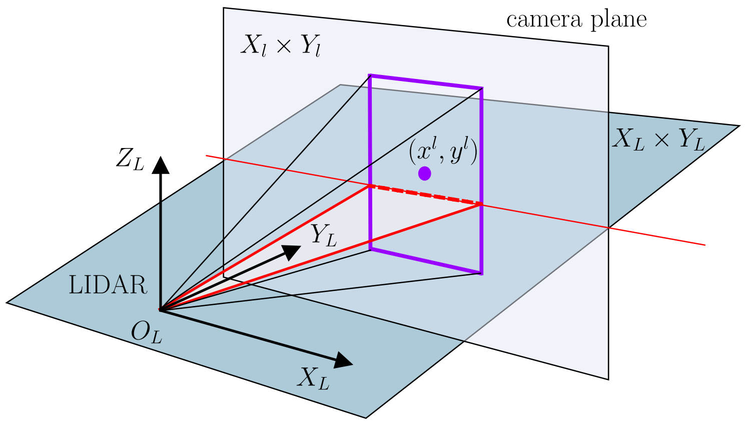


Figure 3.14: Schematic representation of the camera $\langle C_l \rangle$ and the LIDAR $\langle L \rangle$ reference systems. The YOLO bounding box (purple box) identifies a conical sector (red lines) in the LIDAR plane.

object depth. As detailed in Section 3.3, YOLO was found to be more robust and faster than OpenPose, although the bounding boxes dimensions are noisy and inconsistent. The minimum and maximum x-coordinates of the bounding box provided by YOLO in $\langle C_l \rangle$, mapped in $\langle L \rangle$, turn in a conical sector in the $X_L \times Y_L$ plane (see Figure 3.15).

With the hypothesis of uniquely associating a cone with an object, the nearest cluster within the conical sector is labeled as a person and its position $o_j(t_k) = (x_j(t_k), y_j(t_k))$ is tracked. Subsequently, if the object is appropriately updated over time, the tracking is based solely on the succeeding scans of the LIDAR, as opposed to many related works reported in Table 3.1, which also use the person's position coming from the visual information. When the object associated with the person is released from the tracking system due to failure in measurement detection or exit from the detection area, a new fusion between the information from the LIDAR and the camera is carried out. This way, the computing power is lowered down since no image processing is requested for tracking. Finally, we can assume that the measurements for the j-th person, once associated to a LIDAR object, are given by

$$m_j^{lc} = o_j = \begin{bmatrix} x_j^{lc} \\ y_j^{lc} \end{bmatrix} + \varepsilon^{lc}, \tag{3.6}$$

where ε^{lc} is the noise affecting the measurement process carried out by the LIDAR and characterised in Section 3.5.

3.4.3 Estimation algorithm

As for the camera-based sensing system, for the LIDAR-camera system as well two Kalman filters are adopted, in order to mitigate the noises ε^{lc} in (3.6) and make the estimation algorithm more robust. The first model adopted is the constant velocity model (3.4) adopted in Section 3.2.3, which is affected by the noise $\nu(k) \sim \mathcal{N}(0, N)$.

The second filter is instead a different formulation of the unicycle model (3.5), where the angular and linear velocities are both explicitly expressed as states. So, setting $q(k) = [x(k), y(k), \theta(k), v(k), \omega(k)]^T$ as the state at time k (where $\omega(k)$ is the angular velocity), we have the following Extended Kalman Filter

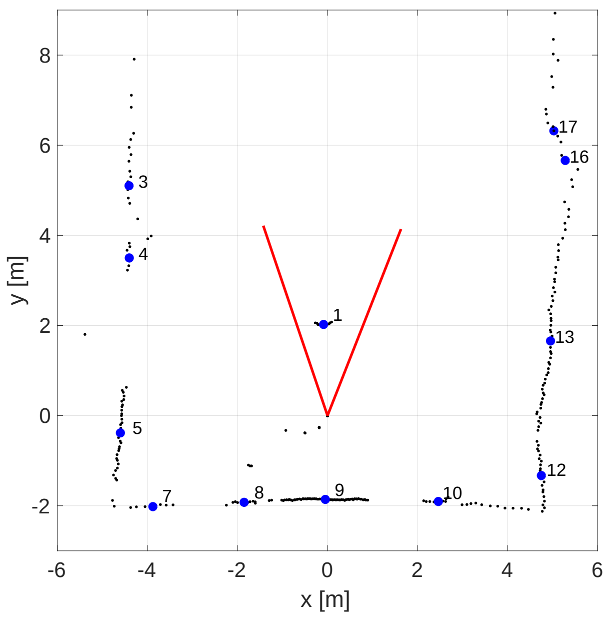


Figure 3.15: Laser scanned cloud points, with the object centroids (blue points) and the YOLO bounding box (the red lines projected as described in Figure 3.14) mapped in the LIDAR reference system $\langle L \rangle$.

(EKF) prediction model

$$
\begin{aligned}
q(k+1) &= \begin{bmatrix} x(k) + \delta_t v_k \cos(\theta(k)) \\ y(k) + \delta_t v_k \sin(\theta(k)) \\ \theta(k) + \delta_t \omega(k) \\ v(k) \\ \omega(k) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \delta_t & 0 \\ 0 & \delta_t \end{bmatrix} \begin{bmatrix} \eta_a(k) \\ \eta_\omega(k) \end{bmatrix} = \\
&= f(q(k)) + B\eta(k),
\end{aligned}
\tag{3.7}
$$

where $\eta(k)$ is the acceleration noise affecting the linear and the angular velocities that is assumed to be $\eta(k) \sim \mathcal{N}(0, E)$, with E being its covariance matrix. For this EKF formulation, the measurement functions are given in (3.6).

3.5 Tracking system results

The experimental set-up comprises an off-the-shelf USB Camera Module[7] (see Figure 3.13-b) for which we adapted the C++ implementation of YOLO used in Section 3.5 along with OpenCV. Likewise, a C++ class was written to call the LIDAR API. To further speed up the data acquisition, the camera frames are not saved, while only the LIDAR scans (as a vector of points) and YOLO detection structs are recovered.

The acquisition algorithm was executed on a Jetson TX2[8], with an acquisition rate of 10 Hz for both the LIDAR and YOLO. Ground truth data were obtained using the 3D tracking Optitrack system described in Section 3.3. The latter was placed so that the image were orthogonal to the plane of motion (see Figure 3.6), and its 3D position related to the reference system origin was calibrated with ground

[7]ELP-USBFHD01M-BL36 from http://www.elpcctv.com/

[8]https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-tx2/

Algorithm 1 Clusters assignment algorithm

```
Input: $\mathcal{O}_k = \{\mathbf{o}_1(t_k), \dots, \mathbf{o}_{m_k}(t_k)\}$, $\mathcal{O}^\star_{k+1} = \{\mathbf{o}^\star_1(t_{k+1}), \dots, \mathbf{o}^\star_{m_{k+1}}(t_{k+1})\}$, $\mathcal{H}_k = \{\text{hit}_1(t_k), \dots, \text{hit}_{m_k}(t_k)\}$,
    $\mathcal{T}_k = \{\text{track}_1(t_k), \dots, \text{track}_{m_k}(t_k)\}$
1: // $j = 1$, $\mathbf{c}_j \leftarrow \text{Append}(\mathbf{p}_1)$
2: $\text{updated}_j = False, \forall j = 1, \dots, m_k$ // Initialisation object update flag
3: for $i = 1$ to $m_{k+1}$ do
4:     for $j = 1$ to $m_k$ do
5:         if $||\mathbf{o}^\star_i(t_{k+1}) - \mathbf{o}_j(t_k)||_2 < d_{\text{min}}$ then
6:             $\mathbf{o}_j(t_k) \leftarrow \mathbf{o}^\star_i(t_{k+1})$ // Update cluster position
7:             $\text{updated}_j \leftarrow True$
8:         else
9:             $m_k \leftarrow m_k + 1$
10:            $\mathbf{o}_j(t_k) \leftarrow \text{Append}(\mathbf{o}^\star_i(t_{k+1}))$ // Allocate new cluster
11:            $\text{hit}_j = \text{Append}(0)$
12:            $\text{track}_j = \text{Append}(False)$
13:        end if
14:    end for
15: end for
16: for $j = 1$ to $m_k$ do
17:    if $\text{track}_j == True$ then
18:        if $\text{updated}_j == False$ then
19:            $\text{hit}_j \leftarrow \text{hit}_j - 1$
20:        else
21:            $\text{hit}_j \leftarrow 0$
22:        end if
23:        if $\text{hit}_j \leq T_o$ then
24:            for $i = j$ to $m_k - 1$ do
25:                $\mathbf{o}_i(t_k) \leftarrow \mathbf{o}_{i+1}(t_k)$ // Delete cluster
26:                $m_k \leftarrow m_k - 1$
27:            end for
28:        end if
29:    end if
30:    if $\text{track}_j == False$ then
31:        if $\text{updated}_j == True$ then
32:            $\text{hit}_j \leftarrow \text{hit}_j + 1$
33:        else
34:            $\text{hit}_j \leftarrow 0$
35:        end if
36:        if $\text{hit}_j \geq T_a$ then
37:            $\text{track}_j == True$ // Start tracking cluster
38:            $\text{hit}_j \leftarrow 0$
39:        end if
40:    end if
41: end for
42: $k \leftarrow k + 1$
Output: $\mathcal{O}_k = \{\mathbf{o}_1(t_k), \dots, \mathbf{o}_{m_k}(t_k)\}$
```

truth data. For the sake of comparison, we recorded the same types of trajectories used in Section 3.5, i.e. linear paths with constant distance from the camera, diagonal and circular paths. The process covariance matrices N and E affecting respectively the KF model (3.4) and the EKF model (3.7) are obtained computing the errors between the predicted state given by the filters and the ground truth on several testing trajectories. For the measurement covariances associated to ε^{lc} and affecting the LIDAR measured quantities in (3.6), we evaluate the error of LIDAR-camera system on a large set of random paths. Figure 3.16 depicts the empirical Probability Mass Functions error along the X_l and Y_l (recall

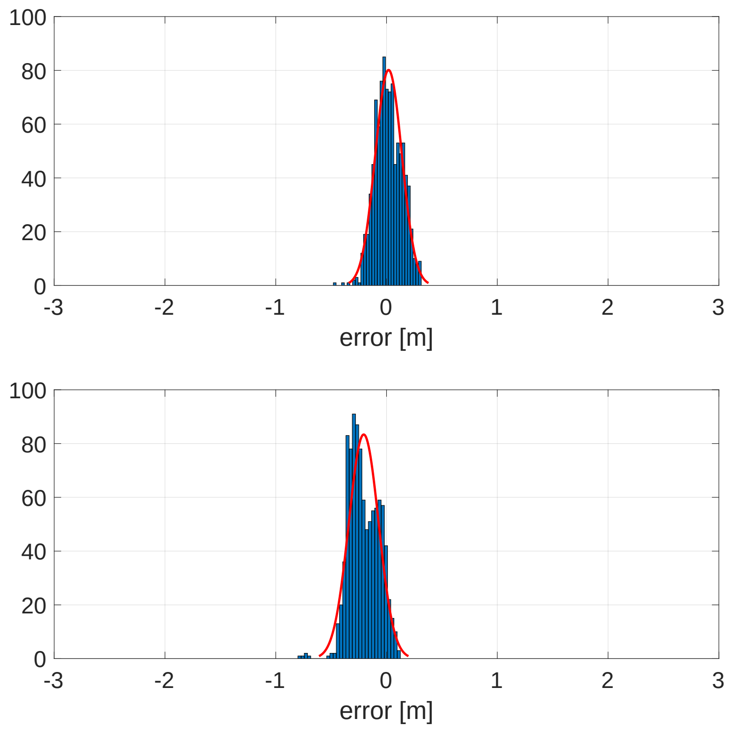


Figure 3.16: Error distribution along the X_l (top) and Y_l (bottom) axes for the LIDAR-camera system (histogram bars) and the relative Gaussian fit (red line).

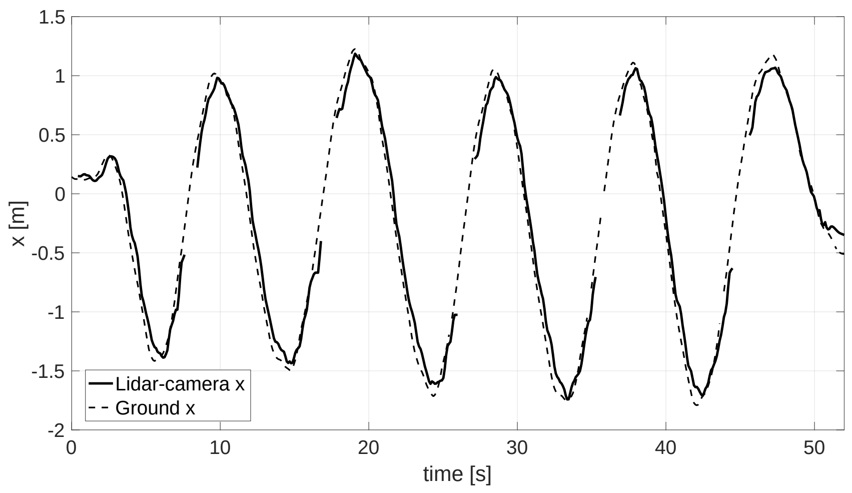
	KF	EKF
Linear	$\sigma_x^2 = 0.0259$, $\sigma_y^2 = 0.0211$, $\rho = 0.4981$	$\sigma_x^2 = 0.0327$, $\sigma_y^2 = 0.0207$, $\rho = 0.3584$
Diagonal	$\sigma_x^2 = 0.0174$, $\sigma_y^2 = 0.0414$, $\rho = -0.2013$	$\sigma_x^2 = 0.0192$, $\sigma_y^2 = 0.0448$, $\rho = -0.1711$
Circle	$\sigma_x^2 = 0.0502$, $\sigma_y^2 = 0.0391$, $\rho = -0.2838$	$\sigma_x^2 = 0.0404$, $\sigma_y^2 = 0.0360$, $\rho = -0.1315$

Table 3.4: Error covariance matrix data obtained for the KF and the EKF, using the LIDAR-camera measurements.

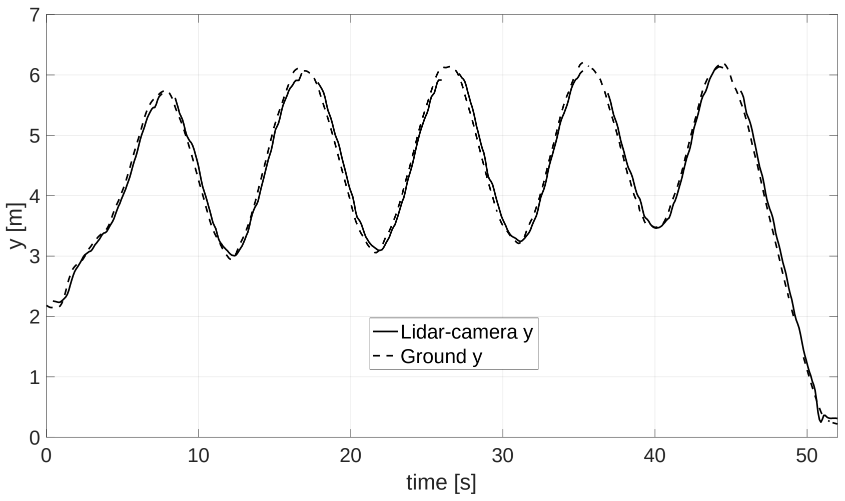
Figure 3.6) for the new sensing system. If compared with the results in Figure 3.8, it appears that the accuracy of the new system is greater than using OpenPose or YOLO. Notice that the error along Y_l has a slight bias, which is in the depth direction (see the reference $\langle L \rangle$ in Figure 3.15): the LIDAR tends to underestimate the actual person position. Such results are used as covariance matrices for the joint Gaussian distributions hypothesised.

Table 3.4 shows the new experimental covariance matrices elements of the filtering error for the (x, y) positions with respect to the ground truth with both the Kalman filters designed. It is evident how the filtering error is generally lower than any possible combination presented in Table 3.3. These results show how the LIDAR data, if properly processed and combined with the purely qualitative information of YOLO, are more accurate than the purely image-based tracking systems and with a reduced computation.

For what concerns the rate of missed measurements, this solution is definitely better than the solutions in Section 3.3, as it is evident from Figure 3.17 when a circular trajectory is considered as an example. In fact, notice how the number of missed data is definitely low, which allows us to state that the LIDAR-camera system is even more robust than a purely camera-based solution. In this respect, the



(a)



(b)

Figure 3.17: LIDAR-camera (solid line) and ground truth (dotted line) comparison along the (a) X_l and (b) Y_l axis when the person follows a circular path.

main tracking problems are the measurements at the boundaries of the detection area (clearly pointed out in Figure 3.17 recalling that the Y_l coordinates represent the depth) and the possible occlusions between two nearby objects. Other than that, the tracking is smooth and constant in all the space reachable by the sensor, and appears unrelated from the distance between the person from the sensing system. In other words, even at the furthest distance, the object associated with the person as described in Section 3.4.2 is made up of a sufficient number of measured points. From Figure 3.17 it can also be seen how the object is quickly recovered as soon as it returns within the maximum distance visible from the sensor.

Another important issue that comes out from Table 3.4 is that the two models behave slightly better than the other when the hypothesis are met: the KF works better for constant velocity paths, while the EKF for curved and smooth paths. As an example, Figure 3.18 shows the results of both the solutions along the circular path of Figure 3.17. The graph report the time evolution of the distance error d between the estimated points in the $X_l \times Y_l$ and the corresponding ground truth points. The peak errors are due to the loss of the LIDAR tracking when the human being reaches the sensing range boundary

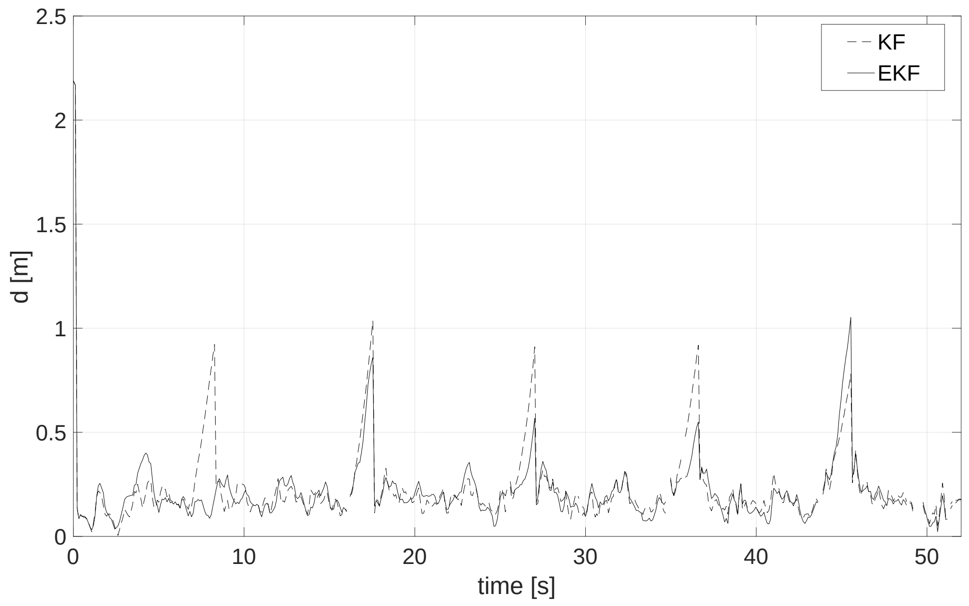


Figure 3.18: Distance errors for the KF (dashed line) and the EKF (solid line) in the circular path example using the LIDAR-camera measurements.

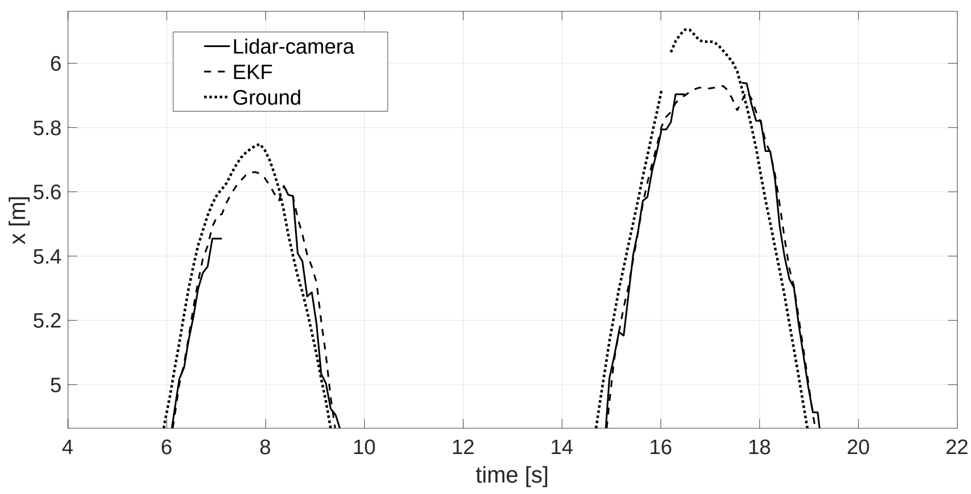


Figure 3.19: Zoomed filtering error when detection failures happen for the lidar-camera system (solid line): the EKF estimates (dashed line) are able to depict the nonlinear behaviour of the ground truth (dotted line).

(see Figure 3.17 and Figure 3.15). However, the EKF with the unicycle model (3.7) is able most of the time to keep track of the curve, while the KF with the constant velocity model (3.4) predicts a linear motion tangent to the curve (see the peaks in Figure 3.18). This behaviour is further emphasised in Figure 3.19: the EKF (dashed line) is able to keep track of the nonlinear smooth behaviour of the actual curved trajectory (dotted line).

3.6 Radar-based tracking

The radar-based platform is based on an evaluation module comprising the TI IWR6843 So[9] (see Figure 3.20), an array composed by three transmitting and four receiving antennas and other electronic components to connect the SoC with other external systems. The evaluation module is placed in piggyback on a custom motherboard designed by Tretec S.r.L., Trento. The motherboard is equipped with an STM ARM Cortex-H7 microcontroller running at 256 MHz, 2 MB of flash memory and 1 MB of SRAM memory. The system is powered by a IEEE 802.3af Power over Ethernet class 1 interface that can drain about 300 mA on average at 3.3 V. Along with the Ethernet link, I/O connectivity is provided by 1 mini-USB port (for debug purposes), 2 RS-485 and 1 RS-232 ports and 3 independent I2C links. A task scheduler and the lightweight TCP/IP stack (lwIP v.2.0.3) have been properly customized for the firmware provided. SoC configuration and radar data streaming rely on a TCP/IP Ethernet connection. The IP address can be assigned either statically or dynamically (if a DHCP server is available). Remote system configuration as well as data acquisition and monitoring is performed via a Graphic User Interface (GUI) implemented in Java. A brief description of the radar-based positioning algorithm and its main configuration parameters is reported in Table 3.5.

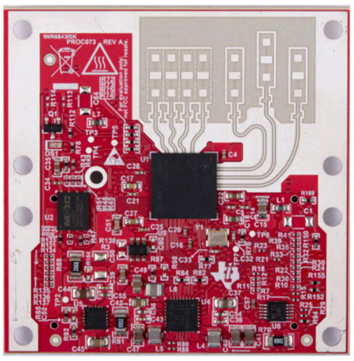


Figure 3.20: IWR6843 radar[9].

3.6.1 Algorithm and SoC overview

The IWR6843 SoC relies on a Frequency Modulated Continuous Wave (FMCW) radar able to estimate the distance, the angle and the velocity of a moving target. FMCW radars transmit repeated frames of chirp signals using an antenna array and receive the backscattered ones with another one. The received and transmitted waveforms are mixed to produce an intermediate frequency (IF) signal, whose initial phase is given by the difference between the phases of the transmitted and received chirps. The distance between the target obstacle and the transceiver is a nonlinear function of this IF signal. The velocity measurements are obtained by transmitting equally spaced chirps and then by comparing the phases of two reflected ones estimated through the Fast Fourier Transform (FFT). A second FFT, called Doppler-FFT, is performed in order to resolve ambiguity between two moving targets. Finally, the angle can be estimated from the phase changes associated with the spectral magnitude peaks resulting from the FFT or the Doppler-FFT.

[9]https://www.ti.com/product/IWR6843

Parameter	Short range configuration
Field of view	120 ° horizontal, 30 ° vertical
Maximum range	6.3 m
Range resolution	5.5 cm
Maximum velocity	23.6 Km/h
System power consumption	2 Watt

Table 3.5: Nominal radar characteristics from http://www.ti.com/tool/TIDEP-01000.

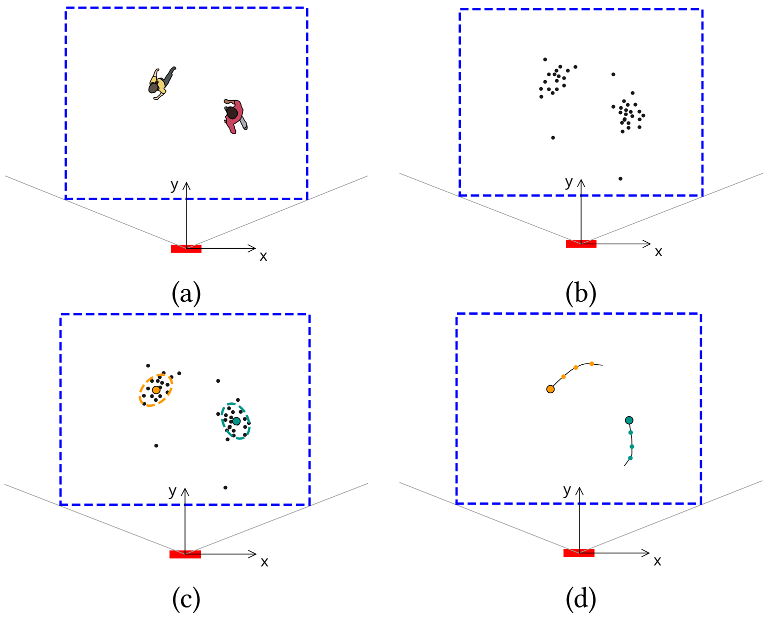


Figure 3.21: Radar tracking pipeline. In the represented example, two pedestrians are the radar's field of view (a). The reflected points (b) are gathered in clusters (c) and then tracked individually (d).

The IWR6843 SoC includes an ARM Cortex-R4F microcontroller (MCU) and a TI C674x Digital Signal Processor (DSP) acting as a co-processor for critical signal processing tasks. The people tracking algorithm (depicted in Figure 3.21) running on the ARM Cortex-R4F is called once per frame, gets the measurement data in polar coordinates (range, angle, Doppler frequency shift), and returns the tracked objects in a Cartesian space. Since any target has a finite size and several chirps are transmitted and received at the same time by the antenna array, usually the distance from a multitude of points is measured at the same time. Such points may refer to the same or to different objects. Therefore, the points should be grouped accordingly. The algorithm relies on both an allocation and an association step. The allocation step creates sets of points (to be assigned to distinct possible targets) that do not belong to any already existing set. The allocation step is guided by some configuration parameters, as explained more in detail in Section 3.6.2. The association step instead refers to the process of updating these sets, by keeping the previous points or adding the new ones that are closest to the corresponding centroid. An on-board extended Kalman filter (EKF) is used to track the position of the centroids resulting from the allocation and association steps. The pipeline of the tracking algorithm is outlined in Figure 3.22. At the beginning of the pipeline, the parts of the point cloud that fall outside the scene limits are ignored by the association and allocation steps. All the points that are allocated to a given target are first evaluated with the EKF predict step. Then, the association step assigns the measured

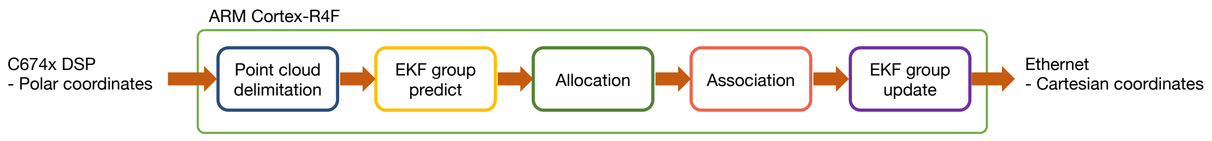


Figure 3.22: Flow chart of the tracking algorithm.

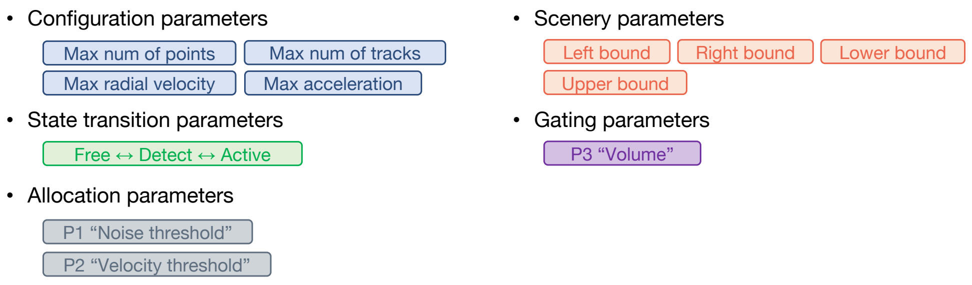


Figure 3.23: Radar parameters P_1, P_2, and P_3 were considered for the experimental evaluation.

points closest to a given centroid to the corresponding set, evaluating the set with the highest score. All the points that cannot be associated to any existing set are grouped into new sets, which are allocated to new possible targets. Finally, in the update step, the EKF estimates the positions of the targets on the basis of the set of associated and newly allocated points.

3.6.2 Configuration parameters

The behaviour of the detection and tracking algorithm embedded in the TI IWR6843 SoC depends on a variety of parameters. The most representative ones are listed in Figure 3.23. The *scenery* parameters, expressed in meters, define the limits of the detection area of the radar. Thus, all objects that lie outside such limits are not gathered into sets. It should be noted that these values are independent of the actual radar field of view, that could be smaller than the chosen area.

The *state transition* parameters determine the quickness with which each target in the allocation step starts or stops being tracked. In particular, in every frame a "hit" or a "miss" event can be associated to a given object, depending on whether a cluster of points is associated to a given object or not, and the numbers of consecutive hits or misses defines the change of state for the object. The *state transition* settings, used in the static and dynamic tests described in Section 3.7, are summarized below:

- A target is detected and starts being tracked as soon as the corresponding set of allocated points is steadily formed. This holds in both static and dynamic testing conditions;
- A target stops being tracked when the corresponding set of points disappears or stops moving for more than 50 or 500 algorithm iterations in dynamic or stationary conditions, respectively. Moreover, a target stops being tracked as soon as it exits the expected detection area.

Observe that, in the case of tests in stationary conditions, the threshold value adopted to halt tracking is 10 times larger than in the dynamic case, in order to prolong radar tracking ability as much as possible.

The *allocation* parameters guide the formation of the sets of points to be assigned to one or multiple targets. The most significant parameters of this kind will be labeled here and in the following as P_1 and P_2. The former one is an dimensionless threshold value used to discriminate possible spurious points (typically due to background noise) from those that are associated to a moving target. In practice, with a higher value of P_1 the number of sets tends to decrease. As a consequence, the probability of missing actual targets increases. On the contrary, if P_1 decreases, the probability of spurious (i.e., false) detections grows. Parameter P_2 is instead a velocity threshold (expressed in m/s) representing the minimum radial velocity associated with the centroid of the set allocated to the same target.

Finally, the so-called *gating* parameters establish how and to what extent some detected points can be associated to one of the existing sets. The main parameter of this type (labeled as P_3 in the following) determines the maximum volume of a set. In practice, the points that are farther than P_3 from the centroid of a given set are considered not to belong to the corresponding target.

3.7 Tracking system results

To test the performance of the radar-based platform, an experimental campaign was conducted in the laboratories of the University of Trento. The radar-based platform was fastened to a rigid support at about 2.30 m off the floor with a tilt angle of a few degrees (Figure 3.24). An OptiTrack Flex13

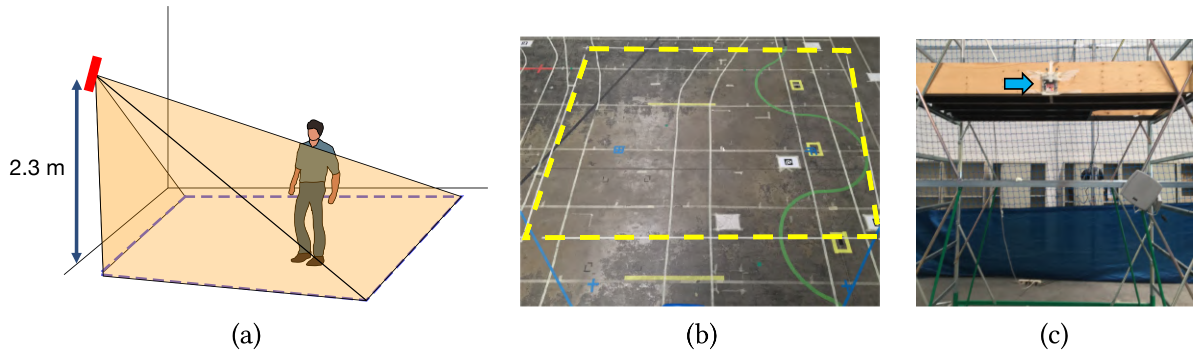


Figure 3.24: Experimental setup. The radar was placed on a rigid support (c), looking towards the ground plane (a). The dashed area in (b) depicts the actual detection area of the radar.

reference localization system (see Section 3.3) was used first to calibrate the radar-based platform in static conditions (namely, with a still target person) and then to measure the actual positions of one or two people moving along different paths. The trajectories estimated by the OptiTrack system can be regarded as the "ground truth", since the position of the ad-hoc reflective markers detected by OptiTrack can be measured with ± 1 mm accuracy. In the following subsections, first a description of the platform calibration procedure in static conditions and the related results are reported in Section 3.7.1. Then, in Section 3.7.2, the system behaviour under dynamic conditions is analyzed.

3.7.1 Calibration procedure and results in static conditions

The nominal area over which the radar is supposed to detect a target is a sector of a circle with a radius of 6 m and with aperture angle of about 118°. However, some preliminary tests showed that the actual detection area is instead quite irregular and smaller than expected. In particular, we found that:

- In the longitudinal direction, the detection range spans from 0.46 m ± 0.09 m to 5.55 m ± 0.12 m;
- In the transverse direction, the left and right detection ranges are 1.91 m ± 0.05 m and 3.04 m $\pm$ 0.36 m, respectively.

Possible setup-related uncertainty contributions that need to be properly compensated (or at least mitigated) for a better characterization of the platform are:

1. The angular and translational systematic offsets between the reference frame associated with the OptiTrack and the frame of the radar-based system;

2. The misalignment between the timestamps of the data collected by both positioning systems;

3. The intrinsic difference between the points of the body selected by either system to identify the position of the target.

Given that the area monitored by the OptiTrack system is about 10 m $\times$ 7 m (i.e., quite larger than the radar range), the position and the orientation of the TI IWR6843 SoC was measured in the OptiTrack reference frame by placing three reflective markers on it. Hence, a roto-translational transformation was used to align the coordinates of the selected points in the reference frames of both positioning systems.

The uncertainty contributions due to timestamp misalignment were minimized by using the same PC to collect both OptiTrack and radar-based position data. Indeed, if the same PC clock is used for data time-stamping there is no need to synchronize the positioning systems. Since both the OptiTrack cameras and the radar-based platform stream their data to the PC through dedicated Ethernet cables linked to the same Ethernet switch and with a similar length, the communication delays differ by less than 10 ms. Therefore, their impact on target positioning accuracy at common pedestrian speeds (i.e., lower than 3 m/s) is in the order of a few centimeters.

Finally, as far as the third uncertainty contribution is concerned, the (x, y) coordinates of the target estimated by the OptiTrack and the values returned by the radar-based platform at the same time are likely to refer to different points of the body. This is due to the fact that to make the reflective markers detected by the OptiTrack visible from any position, they are placed on hats worn by users. Thus, the location of a target for the OptiTrack coincides with the position of user's head. On the contrary, the position estimates returned by the radar-based system coincide with those of the centroid of the cloud of points clustered by the algorithm running in the TI IWR684 SoC. Such a centroid is generally located within the trunk, although some fluctuations may occur due to the changes in distance and relative orientation between the user and the radar-based platform. In conclusion, a space-dependent geometric offset (in the order of some tens of cm) exist between the target positions measured by the OptiTrack and by the radar system, respectively. These offsets can be estimated and compensated through a bivariate linear regression applied to the points collected when a target is still in 9 different positions of the room, as shown in Figure 3.25. The clusters of red points in Figure 3.25 represent indeed the planar positions of the target measured by the radar-based platform in 1 minute for $P_1 = 375$, $P_2 = 0$ and $P_3 = 5$ (namely when the radar is maximally sensitive to target motion) *after* applying the calibration procedure based on the bivariate linear regression described above. Observe that the post-calibration residual offsets between the mean values of the (x, y) coordinates measured in each position and the corresponding ground-truth values (highlighted by blue circle markers) are negligible, which confirms that the calibration procedure is statistically correct. The average standard uncertainty along both axes is about 5 cm with some fluctuations that depend on the relative orientation between target and radar, as well as on the fact that a person must slightly swing to trigger radar detection.

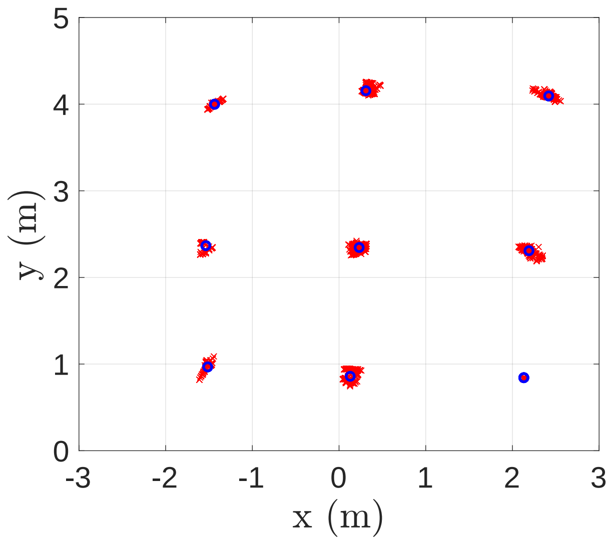
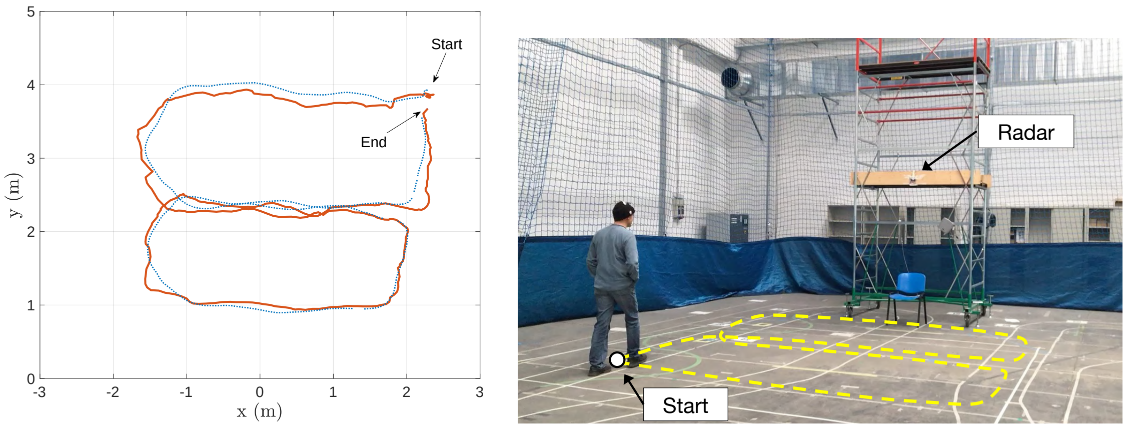


Figure 3.25: Planar coordinates measured by the radar-based platform (red dots) in 9 different positions after the calibration procedure is applied. The blue circle markers represent the actual target's position, i.e. measured by the OptiTrack system.



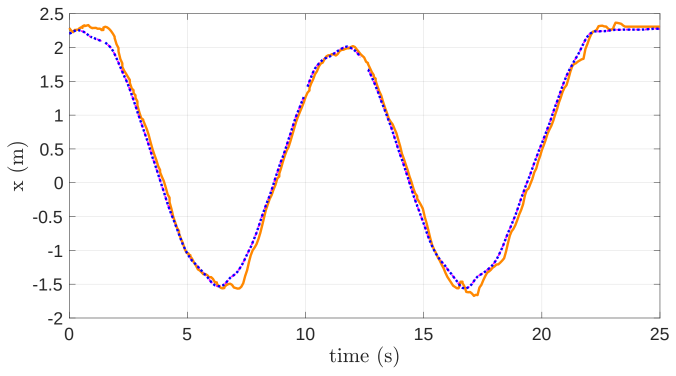
(a) (b)

Figure 3.26: (a) Example of an eight-shaped path. The solid and dotted lines refer to the trajectories estimated by the radar and by the OptiTrack system, respectively. (b) Snapshot of the experiment with the projection of the trajectory (yellow line).

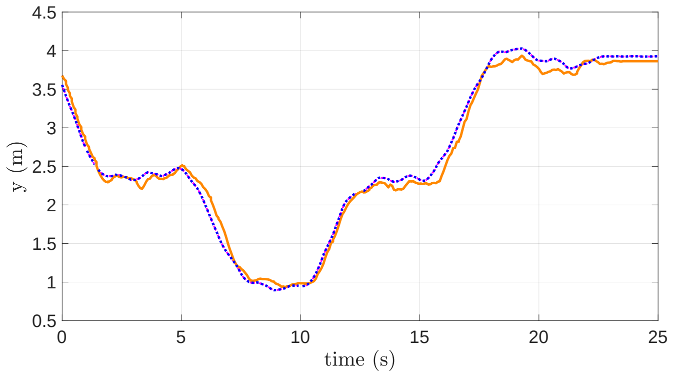
3.7.2 Results in dynamic conditions

After platform calibration, multiple experiments have been conducted in dynamic conditions, i.e., with one or two targets moving straight (both longitudinally and transversally with respect to detection area of the radar), along eight-shaped paths or just randomly. For each type of trajectory, repeated experiments have been performed for 26 different triples of P_1, P_2 and P_3 values. Such triples have been selected with a trial-and-error approach and refer to the configurations for which the radar is responsive and measurement results are qualitatively reasonable. Figure 3.26 displays an example of an eight-shaped trajectory estimated by the radar (solid line) for $P_1 = 500$, $P_2 = 0.5$ and $P_3 = 10$.

The same trajectory estimated by the OptiTrack system (dotted line) is also shown for the sake of comparison. Observe that the average Euclidean positioning error in this case is about 14 cm. In Figure 3.27 we report a more detailed view of the experiment for the x-axis and y-axis. We notice that



(a)



(b)

Figure 3.27: Ground truth data (dotted line), and radar measurements (solid line) associates with the $x-$axis (a) and $y-$axis (b) of the eight-shaped path of Figure 3.26.

the tracking of the radar (solid line) is very accurate, with slight deviations from the ground truth when the target is nearby the edges of the sensor area.

The bar diagram in Figure 3.28 shows the 99th percentiles of the relative positioning uncertainty $\gamma = \frac{d_r - d}{d}$ (where d_r and d are the distances of the target measured by the radar and by the OptiTrack system, respectively) as a function of the actual radial distance from the radar in three meaningful configurations. Observe that globally the relative uncertainty tends to decrease with distance. Therefore, the absolute positioning uncertainty tends to be quite independent of d, which is consistent with the qualitative result shown in Figure 3.26. However, by decreasing the value of P_1 and P_2 (which increases the sensitivity of the system, thus generating possible spurious detections) relative accuracy tends to improve.

Table 3.6 summarizes the results of the dynamic characterization of the radar-based platforms in different experiments. In particular, recalling that the *expanded uncertainty* is referred to as half of the width of the interval that is expected to encompass a given fraction of the values that can reasonably be attributed to a measurand (ISO/IEC Guide 98-3:2008, 2008), Table 3.6 reports the expanded uncertainties U_x and U_y (with coverage factor 2.7) associated with the measurement of the planar coordinates of the target when it moves over different kinds of paths and for several triples of P_1, P_2 and P_3 values. In practice, with the chosen coverage factor, the planar coordinates of a target lie in the intervals given by the (x, y) values measured by the radar-based platforms $\pm U_x$ and $\pm U_y$, respectively, with 99% confidence. Observe that in most configurations, the values of U_x and U_y range between 30 and 40 cm, with a few exceptions. Such results confirm that the radar-based localization is quite accurate. However,

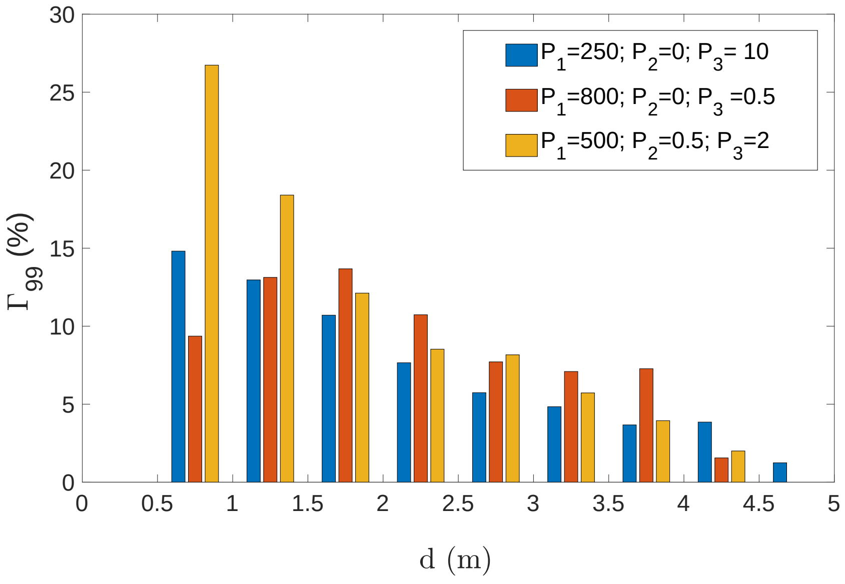


Figure 3.28: 99th percentiles of the relative positioning uncertainty as a function of the actual distance from the radar for three different configurations of parameters P_1, P_2 and P_3.

Conf.	Parameters			U_x [cm]	U_y [cm]	Q_S [%]
	P_1	P_2	P_3			
1	250	0	10	28	29	6.9
2	500	0	10	34	35	0
3	800	0	10	30	25	0
4	250	0.5	10	32	26	24
5	500	0.5	10	26	31	0
6	800	0.5	10	33	30	0
7	250	0.25	10	35	32	8.1
8	500	0.25	10	34	24	0
9	800	0.25	10	34	35	0
10	250	0	2	40	30	9.2
11	500	0	2	33	32	0
12	800	0	2	31	33	0
13	250	0.5	2	34	34	0
14	500	0.5	2	34	35	0
15	800	0.5	2	33	40	0
16	250	0.25	2	40	27	20.6
17	500	0.25	2	36	38	9.4
18	800	0.25	2	38	39	0
19	250	0	0.5	17	20	86.2
20	500	0	0.5	31	28	54.2
21	800	0	0.5	29	32	26.3
22	500	0.5	0.5	40	39	87.9
23	800	0.5	0.5	42	36	51.3
24	250	0.25	0.5	22	28	76.7
25	500	0.25	0.5	52	44	52.5
26	800	0.25	0.5	41	39	35.3

Table 3.6: Results of the dynamic characterization for different values of P_1, P_2 and P_3. U_x and U_y denote the expanded uncertainties (with coverage factor 2.7) associated with the measurement of the planar coordinates of the target when it moves over different paths. Q_S represents instead the frequency of spurious target detections.

the rightmost column of Table 3.6 shows that for some configurations, a quite high percentage Q_S of spurious detections actually appears. In other words, while the probability of missing a target moving within the chosen detection area is negligible, multiple targets can be detected even when just one person is actually within the radar range. The reason of such a behavior is unclear at the moment, but it is definitely due to the detection algorithm running in the IWR6843 SoC. In particular, the probability of spurious detections greatly depends on P_1 and P_3 values (indeed Q_S suddenly increases when P_1 or P_3 become excessively low) and it is also affected by the kind of path. For instance, the aggregated results obtained with different parameters show that the probability of spurious detections is quite high in the case of straight paths (more than 22%), while it is 10% or less when the target moves randomly. It is worth emphasizing that sometimes the target was not detected and tracked immediately. The percentage of these events is particularly relevant (about $11\ \%$) in the case of straight trajectories parallel to the $x-$axis, maybe because of the thinner shape of the sets of points clustered by the algorithm described in Section 3.6.1 when just a side of the target is observed. Notice that even if the radar-based platform is generally quite accurate, the risk of spurious detections sometimes can be very relevant. However, if P_1 and P_3 are set large enough, this risk becomes negligible.

3.7.3 Rejecting spurious measurements

From the analysis carried out in Section 3.7, it follows that for given particular values of the configuration parameters the radar occasionally provides spurious multiple target detections: besides the trajectory of the actual target, the algorithm returns additional points that are not related to any real object, but they are artifacts of the algorithm. Unfortunately, the system does not provide a direct access to the raw points generated by the SoC (at least using the firmware shipped with the device). For this reason, in case of multiple detections, we have developed an algorithm to create a ranking of different possible sets of points to decide which one is the most likely to refer to an actual target. The proposed algorithm is based on the combination of Kalman filtering and likelihood analysis.

First of all, from the paths estimated by the OptiTrack system, three different types of motion were identified: a) target standing still (H_0); b) target moving with a constant forward velocity (H_1); c) target moving with accelerated motion (H_2). For each one of these three kinds of motion, we developed a Kalman Filter (KF) whose model and the related parameters were set accordingly. In particular, the prediction step of each KF is given by

$$\begin{aligned} s^-_{i,k+1} &= A_i s_{i,k}, \\ \Sigma^-_{i,k+1} &= A_i \Sigma_{i,k} A_i^T + B_i Q B_i^T, \end{aligned} \tag{3.8}$$

where $i = 0, 1, 2$ denotes the H_0, H_1 and H_2 models, respectively, $s_{i,k}$ is the state of the $i-$th model at time step k and comprises the planar position, velocity and acceleration of the target within the chosen reference frame, whereas A_i is the system dynamic matrix of the $i-$th model. In particular, with reference to A_i, both target velocity and acceleration are zero for $i = 0$; the velocity is assumed to be constant and the acceleration is equal to zero for $i = 1$ and, finally, the acceleration is constant for $i = 2$. Moreover, $\Sigma_{i,k}$ is the covariance matrix of the estimation error associated with the $i-$th model at time step k, Q is the covariance matrix of the model uncertainties (assumed to be proportional to a perturbation in the acceleration space) and B_i is the uncertainties mapping matrix that enforce the constraints for the i–th model. Finally, the superscript $\cdot^-$ stands for the "predicted quantity".

In the update step of the KFs, the classic equations are used, i.e.

$$\begin{aligned}
e_{i,k+1} &= z_{k+1} - C s^-_{i,k+1}, \\
S_{i,k+1} &= C \Sigma^-_{i,k+1} C^T + R, \\
K_{i,k+1} &= \Sigma^-_{i,k+1} C^T S^{-1}_{i,k+1}, \\
s_{i,k+1} &= s^-_{i,k+1} - K_{i,k+1} e_{i,k+1}, \\
\Sigma_{i,k+1} &= \Sigma^-_{i,k+1} - K_{i,k+1} C \Sigma^-_{i,k+1},
\end{aligned} \tag{3.9}$$

where z_{k+1} are the radar measurements collected at time $k+1$, R is the covariance matrix of the measurement uncertainty contributions and C is the output matrix of the model (notice that for each model we have the same matrix C, since only the position is measured). Moreover, $e_{i,k+1}$ is the innovation vector and $S_{i,k+1}$ is its covariance.

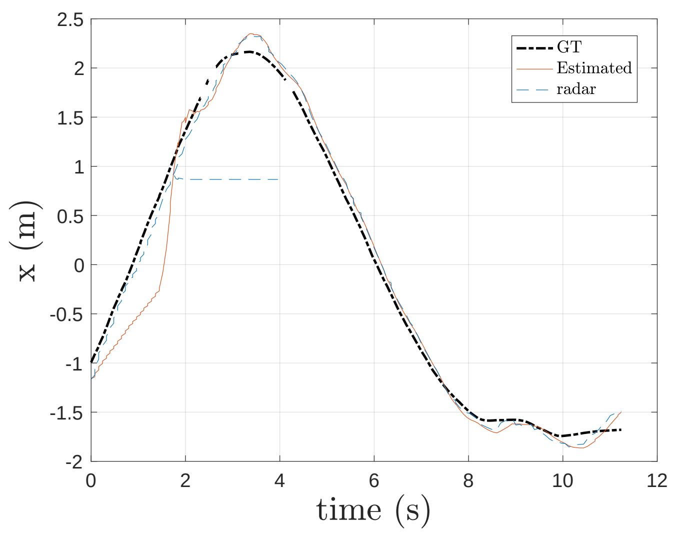
To select one of the possible three hypothesized motion models, i.e. H_0, H_1 and H_2, the Multiple Model Approach (MMA) presented in Y. Bar-Shalom, 2001 is used. This approach provides a stochastic criterion for model selection. For instance, the probability of choosing the H_0 model at time $k+1$, i.e. $\mu_{0,k+1}$, is given by the Bayes' theorem, i.e.,

$$\mu_{0,k+1} = \frac{\Pr\left[z_{k+1}|H_0\right] \mu_{0,k}}{\Pr\left[z_{k+1}\right]}, \tag{3.10}$$

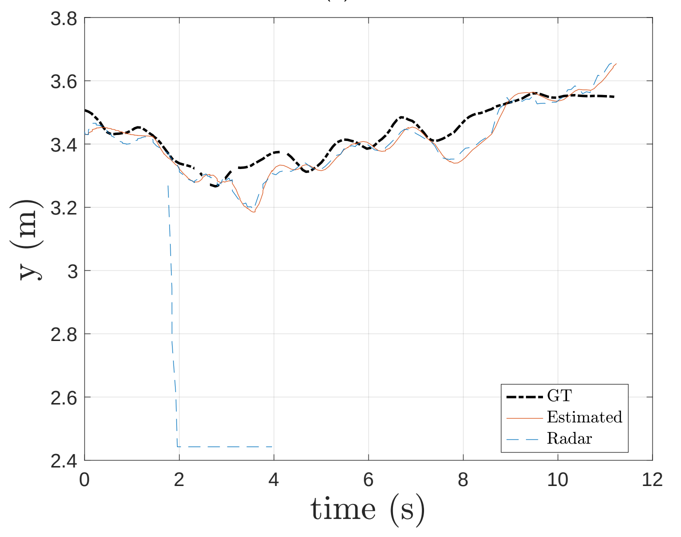
where $\Pr\left[z_{k+1}|H_0\right]$ is the likelihood of measuring z_{k+1} given the model H_0, which can be expressed as a Gaussian Probability Density Function (pdf) with mean $Cs^-_{0,k+1}$ and covariance matrix $S_{0,k+1}$, both reported in (3.9). Note that the estimates $s_{0,k+1}$, $s_{1,k+1}$ and $s_{2,k+1}$ returned by the three filters are computed independently. While this is sufficient to converge to the correct probability when the system does not change its dynamic over time, this cannot be ruled out in the problem at hand. To account for potential mode changes, a first-order generalized pseudo-Bayesian estimator is adopted (Y. Bar-Shalom, 2001). In this way, the estimates of each model are fused together in a single estimate before the prediction step associated with each model starts over. More precisely, the fused estimate and the corresponding covariance matrix are given by

$$\begin{aligned}
s_{k+1} &= \sum_{i=0}^{2} \mu_{i,k+1} s_{i,k+1}, \\
\Sigma_{k+1} &= \sum_{i=0}^{2} \mu_{i,k+1} \left(\Sigma_{i,k+1} + [s_{k+1} - s_{i,k+1}][s_{k+1} - s_{i,k+1}]^T \right).
\end{aligned} \tag{3.11}$$

Again, due to the peculiarity of the system considered, this approach has a bias towards model H_0. Indeed, since the sampling period of the radar measurements is quite short compared to the typical time constants of human motion, the sampled data will be very close in space and, consequently, the likelihood will be favorable to the model with the target standing still. To overcome this limitation, the fusion is performed only after some consecutive measures (e.g., 5) are collected. With this filter hacking entirely due to typical human motion dynamic versus sampling time of the sensor, the presented



(a)



(b)

Figure 3.29: Ground truth data (dash-dotted line), estimated trajectory (solid line) and radar measurements (dashed line) associates with the $x-$axis (a) and $y-$axis (b). Observe the spurious readings from about 2 s to 4 s.

approach is quite efficient in classifying the human dynamic with respect to models H_0, H_1 and H_2. To reject the spurious measurements, two further assumptions are considered: i) whenever multiple measurements are detected, the first-order generalized pseudo-Bayesian estimator is applied to the ensemble of the measured trajectories; ii) based on experimental evidence, we define as a spurious false detection, any trajectory that converges to H_0 while at least another one is classified either as H_1 or H_2. Using this simple heuristic criterion, most of spurious detections can be removed, as shown in Figure 3.29. Indeed, the estimated trajectory (solid line) follows the ground truth data (dash-dotted line) regardless of the spurious data between 2 s and 4 s. This behavior is due to the correct identification of spurious multiple detections, as evident from the probabilities of each model related to the two trajectories (the correct and the spurious one) shown in Figure 3.30.

Indeed, the top graph of Figure 3.30 displays the probability that the first radar trajectory is classified either as H_1 or H_2. Observe that such a probability is generally 1 except at the very beginning. On the

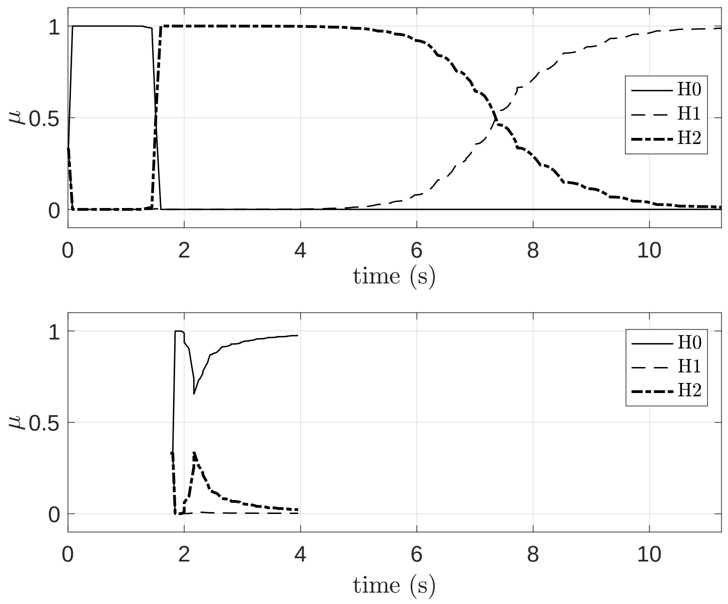


Figure 3.30: Probabilities of the three models H_0, H_1 and H_2 in the case of correct (top) and spurious (bottom) target detection. This probabilities corresponds to the coordinates shown in Figure 3.29.

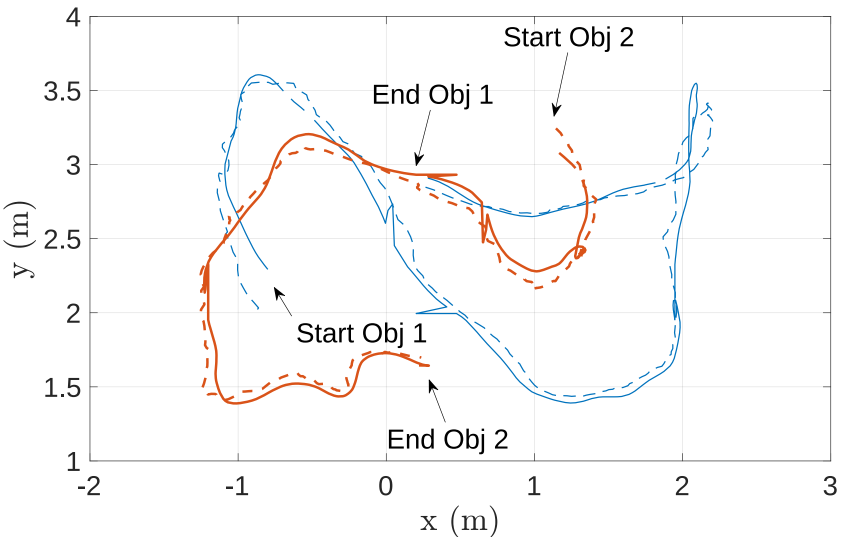


Figure 3.31: Ground truth data (solid lines) and radar measured trajectories (dashed lines) of two human beings (identified with thick or thin lines) moving randomly in the radar detection area.

contrary, when the spurious trajectory comes into play (just before 2 s), this is correctly classified as H_0 and removed accordingly. As a further example, in Figure 3.33 we report the results of the estimator in the case of two dynamic trajectories. We notice that the filter correctly recognizes when the person is standing still (the red points at the beginning and end of the trajectory), when he is travelling along linear paths (orange points), and lastly when he is turning (blue points).

When two targets are tracked, the radar performance are similar to those shown in Table 3.6. Also, the effectiveness of the algorithm developed to detect and to remove possible spurious targets is comparable to the single target case. For a qualitative analysis, Figure 3.31 reports an example with two

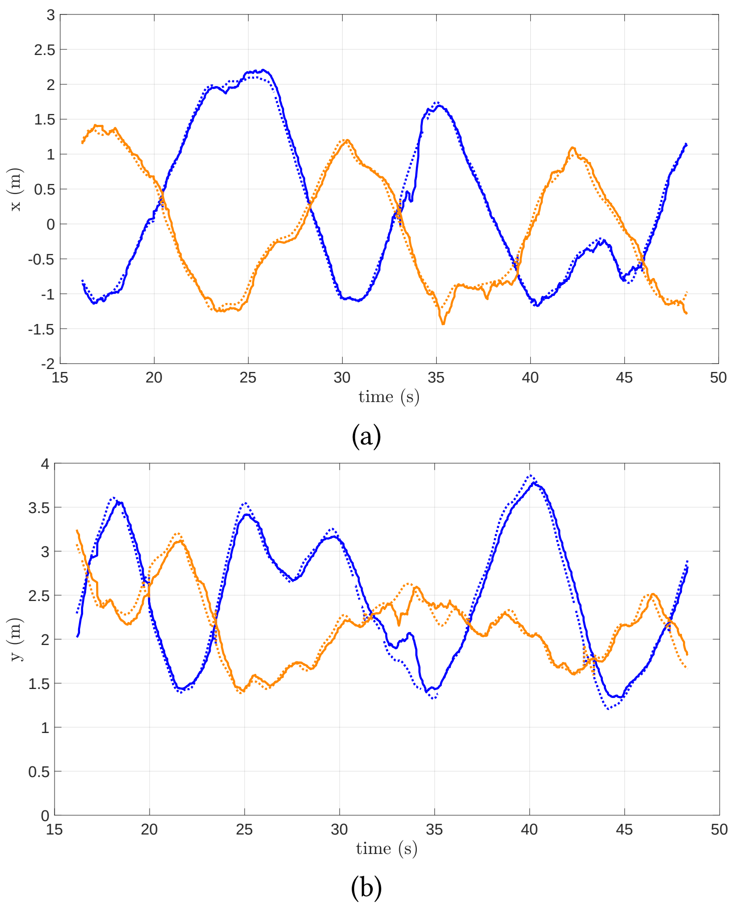


Figure 3.32: Tracking results on the $x-$axis (a) and $y-$axis (b) with two people performing random paths (see Figure 3.31). The ground truth data is depicted with the dotted lines, and the solid lines are the radar measurements.

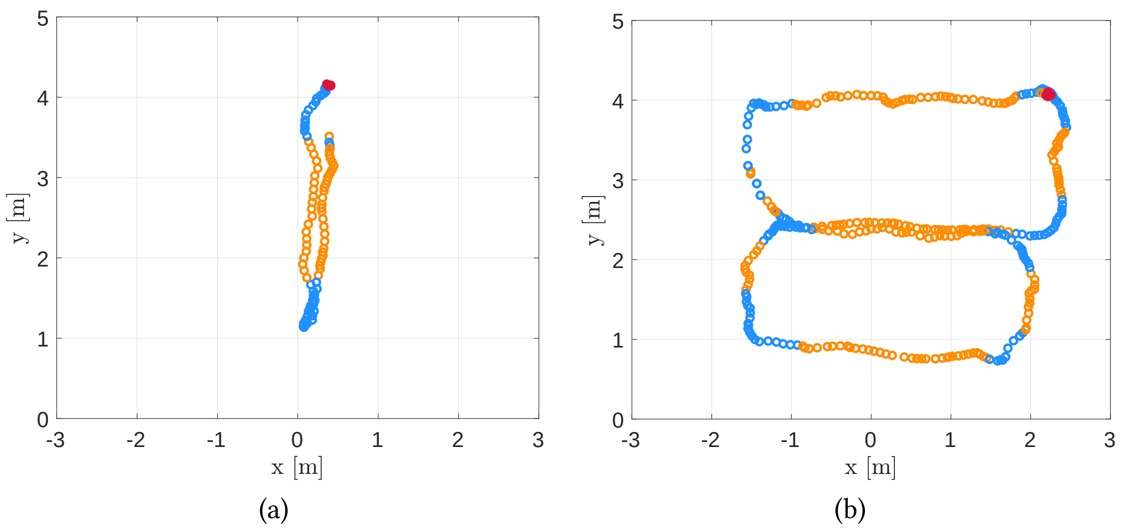


Figure 3.33: Tracking results with an agent performing a vertical (a) and an eight-shaped (b) path. The coloured dots represent the model with the highest probability for the corresponding sample, specifically, H_0 (red points), H_1 (orange points), and H_2 (blue points).

human beings moving randomly in the radar detection area. For both the tracked trajectories the average Euclidean error is about 20 cm, and, what is more, no miss detections occurred during the experiment. This is an interesting result from the point of view of the robotic applications. As reported in

Figure 3.32, the tracking algorithm is quite robust but still sensible to mutual occlusions between the targets. When the two humans cross their path (from 30 to 36 seconds in Figure 3.32), the measurement noise increases, even if the tracking remains correctly associated with the targets. While these results are encouraging, further experiments with an increasing number of targets are needed to evaluate the scalability of the proposed solution.

3.8 Comments

In this chapter we have presented the performance assessment analysis for people tracking system based on:

1. a full camera-based solution using the stereo ZED camera and using two people pose detectors, namely OpenPose and YOLO;
2. a LIDAR-camera combination using the YOLO detector;
3. a millimeter-wave radar with multi target clustering and tracking algorithm.

The final goal was to determine what are the characteristics of such solutions and if they are feasible for the problem of autonomous navigation of robots in populated environments. To make the system robust against occasional failures in the sensor readings and improve the quality of the detected persons, we also presented two Kalman-based solutions, whose main differences are rooted in two different human motion models reported in the literature. Several experiments have been conducted in our laboratory, which offers the possibility of having a quite effective ground truth for the actual human position. Results showed that a specific combination of components can be regarded as suitable and effective for the problem at hand.

In the first tracking system, we obtained the best results using the EKF and the measurements given by YOLO or a mix of the YOLO and OpenPose. OpenPose turned to be quite effective in detecting the human being skeleton, i.e. it is robust with respect to partial occlusions due to its well learned models. However, when it was compared with the ground truth, the detection accuracy was quite low with several outliers, which is due to the different positions of the body points between the two images, leading to an incorrect averaged estimate. On the contrary, YOLO was not always capable of detecting the pose of the person, however, when it succeed, the accuracy was relevant and there was an absence of outliers. The EKF and the KF both had the capability to build up a system that was more robust and more accurate by properly fusing the sensing data: indeed irrespective of the intermittent YOLO measurements and the quite noisy OpenPose measurements, the EKF estimates proved to be close to the ground truth data. In summary, the winner of our comparison is YOLO, whose low computational cost outweighs the supposed advantages of OpenPose.

The second tracking system was effective in demonstrating that very high levels of robustness and performance can be achieved with low computation times by using YOLO on a monocular camera and in combination with a LIDAR. Moreover, the new-designed model for the EKF solution was able to follow the smooth trajectory usually followed by human beings. Finally, the LIDAR-camera solution showed a reduced computational burden, which opens to implementation on cheap robotic hardware.

Finally, we tested a 60 GHz radar based on arrays of small and closely spaced antennas. The recent technological evolution of wireless technologies has brought to the development of these low power systems to currently be among the most advanced solutions for accurate RF indoor localization and tracking. The use of radars in the case of human localization and tracking has numerous and indisputable advantages, such as the relatively lightweight signal processing (e.g., compared to cameras), the preservation of the privacy of tracked subjects and the supposedly good level of accuracy. However, the literature on the tracking performance of these devices in dynamic conditions is largely unexplored, and we sought to fill this gap in our extensive experimental evaluation. The experimental results lead to multi-faceted conclusions. The radar system generally ensured planar positioning uncertainty (with 99% confidence level) in the order of 30-40 cm, which is acceptable for a large class of applications. However, the tracking algorithm running in the SoC was quite sensitive to the values of several parameters that, in some conditions, led to anomalies such as multiple spurious detections of the same user. A careful choice of the parameters can drastically reduce the frequency of these events. Moreover, it was possible to further mitigate this problem by adopting a Multiple Model Approach supported by a heuristic criterion. From this last finding, we established important considerations on the multi-goal hypothesis in another model contained in this thesis (as described in Chapter 4). In addition, the best sensors combination comprising a (depth) camera and a laser scanner was employed as the sensing system of the robotic platform used in the following works of the thesis (Chapters 4, 5, 6), that is the sensors where mounted on the robot, and the Kalman filter models where introduced in the object tracking and in the motion of the platform.

Summarising, the main outcomes of this chapter are the following:

- A new tracking system based on the LIDAR-camera combination; faster and cheaper than a sensing system that relies on stereo camera or radar sensors;
- A lightened and robust tracking algorithm;
- A refined filtering model adapted to the behaviour of people's trajectories;
- The empirical demonstration, as seen in the experiments in Section 3.7.3, that the Bayesian filtering can be used as a (behavioural) classifier.

This last consideration results directly from the experiments reported in Section 3.7.3, in which it was seen that different types of trajectory or motion of the tracked person could be identified by a filter with a model consistent with the observed motion. This clue was then taken up in the model for the trajectories prediction in Chapter 4, in which a Bayesian filter estimates a certain number of different predictions, each corresponding to a probable behaviour and a target goal.

The next research steps we are foreseen are threefold: first, the systems (especially the radar-based) will be tested with multiple targets, i.e. when occlusions may occur along with the tracking, to determine their scalability; second, the algorithms were tested in indoor laboratory environments, thus we plan to repeat the same experimental evaluation in actual working environments, both indoor and outdoor. Furthermore, since the tracking system on which the radar is based includes a background subtraction phase, its installation on a moving reference system such as a robot is not immediately applicable, and, hence, future works will be focused on mounting the radar on a real platform to properly investigate its suitability for robotics applications.

Chapter 4

Human motion prediction

When a robot travels across a human populated area, the paths it takes have to be in accordance with the motion of the bystanders. Thereby, the robot motion planner has to rely on an accurate prediction of how the humans are going to move in a time horizon of a few seconds. The input usable for this purpose can be of various kinds. In a foreseeable future, the robot could use the facial expression and the pose of the different parts of a person's body as predictors of her motion intent. But, perceiving and interpreting the body language is currently beyond the reach of the technology, at least for commonly used low cost service robots.

The overarching goal is to obtain reliable predictions to be used in a human-aware motion planning algorithm. Specifically, we want to predict where a person intends to go by simply looking at how s/he moved in the immediate past and at the shape of the environment. Our approach to predict human motion presented in this chapter is based on a neural network of a peculiar kind. Contrary to conventional deep neural networks, our network embeds in its structure the popular Social Force Model, a dynamic equation describing the motion in physical terms. Since the network is based on Newtonian principles, its parameters can be learned by experimental data on the existing device. Moreover, a known structure guarantees interpretability, performance, and a reduced number of parameters, permitting effective training even with few experimental data.

Our solution was developed in simulation and evaluated with a full set of experiments proving the validity of the approach on recorded data. We then transferred and implemented the algorithm into a wheeled robot and carried out real-world experiments with the robotic platform. The results are extensively reported in the continuation of the chapter. We will also describe in detail how we embedded the dynamic mode into the structure of a neural network, and we will present the multi-goal approach that completes our motion prediction framework.

4.1 Overview

The goal addressed in this chapter is to predict the motion of a human for several seconds ahead using a short segment of past observations. The philosophy that underlies our approach is that learning is a powerful tool, but, when applied to a complex task such as human motion prediction, it requires massive amounts of training data and the risk of overfitting the models is very concrete. With these considerations in mind, our approach restricts the application of learning to the sub-problems that are very difficult to tackle otherwise. In this class falls the estimation of the parameters of the SFM. Our idea is to use a neural network structured so that its connections reflect the dynamics of the SFM. In other words, we embed our prior knowledge into the neural network in the form of a model assuming that the latter, by a correct choice of parameters, closely approximates the dynamics of human motion. This way, the learning phase is concentrated on the aspects for which we actually lack any real knowledge: the SFM parameters and the virtual forces acting in the SFM.

The selection of the goals of the human motion, which is a key input for the SFM, deserves a special attention. As apparent from the survey reported in Section 2.4, this problem is way simpler than the estimation of the SFM parameters and force, and established techniques exist that are easy to implement and that can provide useful probabilistic information on the potential goals. In our work, we consider the following scenarios. The trivial case is when the possible goals are given or annotated on the map based on the analysis of the places of interest. On the opposite side of the spectrum, we consider a situation in which no prior information on the environment is available. In this case, we infer the position of the possible goals from the geometry of the activity space (Rios-Martinez et al., 2015), from the motion of the humans, and from the configuration of the static obstacles. Since for all of these scenarios, we can have multiple potential goals, we use a multi-goal approach based on the generation of different hypotheses and on the evaluation of the likelihood of the trajectories associated with each of them.

The advantages of the approach proposed in this chapter are manifold:

1. Wiring a model inside the neural network reduces the number of neurons by a significant amount (we estimate one or two orders of magnitude);

2. Using maximum likelihood evaluation of multiple hypotheses for the final goal of the target is a natural complement of the idea: the complexity is much lower than a monolithic neural network solution, the training is simplified and the probabilities associated with each possible destination can be used in *what-if* motion planning solutions;

3. As shown in our experiments, a relatively small number of synthetically generated samples is sufficient to generate accurate predictions, even for scenarios that are quite different from the ones considered in the training set, thus the solution is very practical for robotic applications that involve navigation across indoor unknown scenarios;

4. Because our neural network retains the model inside, its decisions can be explained in physical terms, which simplifies the interpretation of the results of the neural network and the explanation of its possible errors.

In the following, we will describe the structured neural network theory (Section 4.3) and our developed model (Section 4.4) with the multi-goal approach (Section 4.5). Sections 4.6 and 4.7 will present the experimental results.

4.2 Related work

Many physics-based models have been proposed to predict human motion in a social context. The most famous is the Social Force Model (SFM) (Helbing and Molnar, 1995). In the SFM a person is seen as a particle acted on by attractive forces (the goals) and repulsive forces (the obstacles). Since it is simple to implement and yet highly effective in the representation of the individual motion of human beings, it has been adopted by a number of research works. One peculiar feature is that the attractive and repulsive forces of the SFM can be generalised by geometric information only, while the resulting motion effect is a linear combination of these elements. The model however has known limitations. One of the most important is that modelling a person as a particle does not differentiate between motion patterns that are “natural” and others that are possible but not frequently taken (e.g., sideways motions). These issues can be addressed by leveraging a relatively high sampling rate and/or by integrating the preferential nonholonomic behaviour of the human motion into the model (Farina et al., 2017b; Arechavaleta et al., 2008).

An important problem to tackle in order to use the SFM is how to estimate its many parameters and in particular the intensity and the direction of the attractive and of the repulsive forces that animate the motion. A first possibility is to make heuristic “rule-of-the-thumb” choices, but this option is workable only in very specific conditions, e.g., the interaction between a robot and a human in free space (Colombo et al., 2013; Bevilacqua et al., 2018b). Another issue is how to estimate the target of walking pedestrians from the past motion (Rudenko et al., 2019). For example, in Luber et al., 2010, a virtual goal is chosen as the position that a person would reach if s/he moved with constant velocity, while in Ikeda et al., 2013 a set of trajectory sub-goals are estimated from the recorded data in a structured environment. Furthermore, Kretz et al., 2018 proposed a modified formulation of the SFM to calibrate the parameters with observable features from empirical data.

Neural Networks (NN) hold the promise to master the complexity of predicting the intention of humans in a relatively simple way. Similarly to our work, Hasan et al., 2019 modelled an area of visual attention and social interaction for the pedestrian, jointed to the head orientation, and used to strengthen the training of a Long Short Term Memory network (LSTM). A deep learning-based classifier is used in Ma et al., 2017 to learn behaviour patterns from visual cues is mixed with a game theory model encoding the SFM to forecast the interaction between multiple pedestrians. A common approach to manage multiple future trajectories has been to generate different motion modes. For this reason, newly learning approaches implement multiple predictions to describe mixed motion behaviours. In Gupta et al., 2018 a Generative Adversarial Network (GAN) based approach is exploited with a novel social pooling framework to predict multiple trajectories while learning social norms. A similar GAN based framework with a social attention mechanism is proposed by Sadeghian et al., 2019. Both these approaches utilize pedestrians’ past trajectories and scene context information, but do not consider the agents’ destinations. Conversely, a recent work by Mangalam et al., 2020 uses Variational Autoencoder (VAE) based network to infer a distribution of waypoints and obtain a multi-modal trajectory prediction, while Deo and Trivedi, 2020 reformulated maximum entropy IRL to jointly infer waypoints and human trajectories on a 2D grid defined over the scene.

In principle, a deep neural network (DNN) trained with a sufficient number of samples could learn the human motion patterns by discovering the underlying dynamic model on its own. However, the number of layers and of neurons required to manage the complexity of human behaviours can be very large and is anyway hard to predict. Equally difficult is to understand the number of samples that are needed to train a network of this complexity. Finally, the use of a DNN lacks a property of remarkable importance for many applications, i.e., explainability (see Section 2.4). When an autonomous system takes a decision it is important to understand why that specific choice has been made, to solve bugs or attribute legal responsibilities (Pasquale, 2017). The total absence of a prior model in a DNN makes explainability hard or even impossible to achieve. Another known limitation shared by NN approaches is their difficult training. The available annotated data sets are not many, and the parameters overfitting is an actual risk when data have strong similarities. As a result, as shown by Schöller et al., 2020, neural models can easily be outperformed by a simple constant velocity model in the case of linear trajectories.

Past attempts to use machine learning techniques in combination with physical models have applied gradient descend methods to learn the interaction forces (Wan et al., 2017), used linear regression and NN to predict the direction of motion (Zhang and Jia, 2020), used an evolutionary algorithm to optimise the SFM parameters from video segments (Johansson et al., 2007). Our approach is rather new and is inspired to the simulation of vehicle dynamics by Da Lio et al., 2020.

The motion of a person is driven by her intent, i.e., by where she intends to go or perform an action. Dynamic goal inference based on the semantics of the environment is still an open issue (Rudenko et al., 2019). Most existing works rely on a predefined set of goals, estimated from observed trajectory data. For example, goals can be deducted from the prevailing direction taken by the pedestrians, by noting the preferred locations where they stop, or by partitioning the environment with a Voronoi-based method (Kanda et al., 2009; Ikeda et al., 2013; Chik et al., 2019; Ferrer and Sanfeliu, 2019). On the other hand, a few papers have sought to identify the goals from real-time motion predictions. For instance, Wu et al., 2018 proposed a heuristic method to automatically determine goal positions on a 2D semantic grid map. Cell transitions are predicted through discrete Markov chains. In Karunarathne et al., 2018, Voronoi partitions (Ikeda et al., 2013) were used in order to obtain a good guess of the preferred sub-goal within a shopping mall. Their algorithm estimates the candidate goal by weighing the visibility and reachability of the sub-goals, and by using a service robot that moves side-by-side with the person. A recurrent Mixture Density Network (RMDN) was proposed by Rehder et al., 2018 to learn a mixture of potential destinations, which is fed into a Fully Convolutional Neural Network (CNN) to predict the human trajectories. Senanayake and Ramos, 2018 applied probabilistic Directional Grid Maps (DGM) in order to fit a mixture of von-Mises distributions of motion directions attributed to each grid cell of a discretised environment.

4.3 Structured neural network

In Section 2.4 we discussed the potential applications of the use of structured neural networks and analyzed various related works. In our case of interest, we are particularly interested in overcoming the following deficiencies:

- *Interpretability.* Network models are built as black boxes, with no dependence at all to the physical entities they are demanded to be trained. Adopting a pre-defined structure in the network will unlock a more understandable interpretation of its inference behaviour.
- *Lack of data.* Supervised learning is conditioned by the need to have a high number of labelled data. An advantage would be to exploit a limited dataset or, if the physical model is known, to synthetically generate data.
- *Limited computational capabilities.* The trade-off between computational power and speed is always a deal in robotic applications. Although dimensionality reduction of neural networks for

embedded systems is promisingly explored, rely on a lean network can simplify their implementation.

The key aspect of physical systems is to be casual, that is their present output is a function on past and current inputs, i.e., $y(t)$ depends only on $x(\tau)$, for $\tau \leq t$. The essence of such system is their memory, or, in terms of state-space representation, their state variables. The standard form of a discrete-time state space models is:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \mathcal{F}\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right) \\ \boldsymbol{y}_k &= \mathcal{H}\left(\boldsymbol{x}_k, \boldsymbol{u}_k\right), \end{aligned} \tag{4.1}$$

were $\boldsymbol{x}_k$ is the state of the system and $\boldsymbol{u}_k$ is the set of applied inputs to the system at time k. If $\mathcal{F}$ and $\mathcal{H}$ are known, the system (4.1) will constitute a white-box model; if those are unknown instead, the system (4.1) is said to be a black-box model (Da Lio et al., 2020). Actually, we are looking for gray-box representation using a neural network implementation. The state space representations can be translated in a Recurrent Neural Network (RNN), defined by the equations

$$\begin{aligned} h_i &= \sigma\left(W_{hh}h_{i-1} + W_h x x_i + b_h\right) \\ y_i &= W_{yh}h_i, \end{aligned} \tag{4.2}$$

by letting $\boldsymbol{x}_k$ be the network recurrent states. In addition, many physical systems can be approximated or behave like linear systems. In this case, the $\mathcal{F}$ and $\mathcal{H}$ of (4.1) are linear and the state space model takes the form

$$\begin{aligned} \boldsymbol{x}_{k+1} &= A\boldsymbol{x}_k + B\boldsymbol{u}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}_k + D\boldsymbol{u}_k, \end{aligned} \tag{4.3}$$

hence the linear subsystems can be implemented as linear layers of the neural network, which turns in a gray box since its weights can be interpreted. Let us introduce our approach with an explanatory example, using one of the simplest of dynamic models, namely the linear oscillator.

Linear oscillator example. The linear oscillator is a 1 Degree Of Freedom (DOF) system, composed

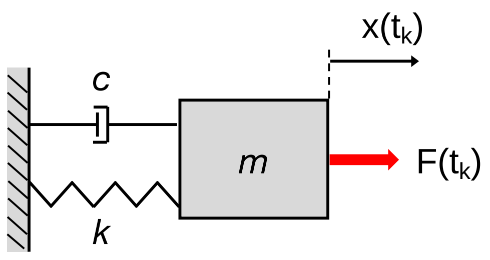


Figure 4.1: Linear oscillator with mass, spring, and damper.

by a mass m connected to a spring and a damper, being where the displacement $x(t)$ of the mass along the single direction constrained is the independent variable (see Figure 4.1).

The second order Ordinary Differential Equations (ODE) governing its motion is

$$m\ddot{x}(t) + c\dot{x}(t) + kx(t) = u(t), \tag{4.4}$$

Oscillator parameters	m: 1 kg k: 3 N/m c: 0.175 Ns/m	Simulation parameters	x_0: $[0; 1]$ m v_0: $[0; 0.5]$ m/s $\lvert F \rvert$: $[-3; 3]$ N

Table 4.1: Parameters for the linear oscillator simulation.

which can be discretized with the Euler method as

$$\begin{cases} x(t_{k+1}) = x(t_k) + \delta_t \dot{x}(t_k) \\ \dot{x}(t_{k+1}) = \dot{x}(t_k) + \dfrac{\delta_t}{m} \left(u(t_k) - c\dot{x}(t_k) - kx(t_k) \right), \end{cases} \tag{4.5}$$

where c is the damping coefficient, k is the stiffness coefficient, and $u(t_k) = F(t_k)$ is the input force to the system used to drive the states.

The linear oscillator presents linear relations between the state variables and their derivatives, i.e., they are multiplied by the constant model parameters or by the constant integration time δ_t. The neural network associated to the oscillator (represented in Figure 4.2) estimates the future position $x(t_{k+1})$ by taking as inputs the force $F(t_k)$ acting at time t_k and a window of n most recent samples of the past x positions $\Delta\mathbf{x}(t_k) = \left[x_{k-(n-1)}, \ldots, x_k\right]$, and combining them as

$$x(t_{k+1}) = w_x \Delta\mathbf{x}(t_k) + w_F F(t_k), \tag{4.6}$$

where $w_x \in \mathbb{R}^n$ and $w_F \in \mathbb{R}^1$ are the weight matrices. The number of learnable parameters for this network is $n + 1$.

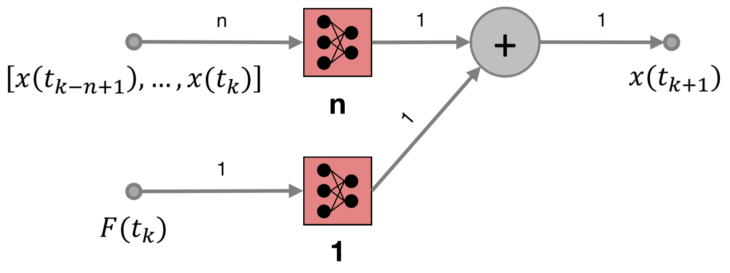


Figure 4.2: Neural network architecture for the linear oscillator system.

We created a set of synthetic data by running a number of simulations with an oscillator model implementation in MATLAB by choosing the combination of physical parameters as reported in Table 4.1. Specifically, we ran 100 simulations of 20 seconds with a sampling time of $\delta_t = 0.1$ s. Each realization has different random initial conditions in terms of independent variables, i.e., initial position x_0 and initial velocity v_0, and an input (randomly chosen between a step signal and a triangular signal) constrained to start and finish in within a random time limit shorter than the half of the total simulation time. The neural network model was coded and implemented in Keras and trained with the Adam optimiser with a learning rate of 0.005, batch size 128, and number of epochs 300. The training procedure took about only 11 s on a 2.7 GHz Intel Core i7 processor. The 70% of the synthetic data for each model was used as the training sets, while the remaining samples were used for validation. We selected $n = 20$ samples as observable data.

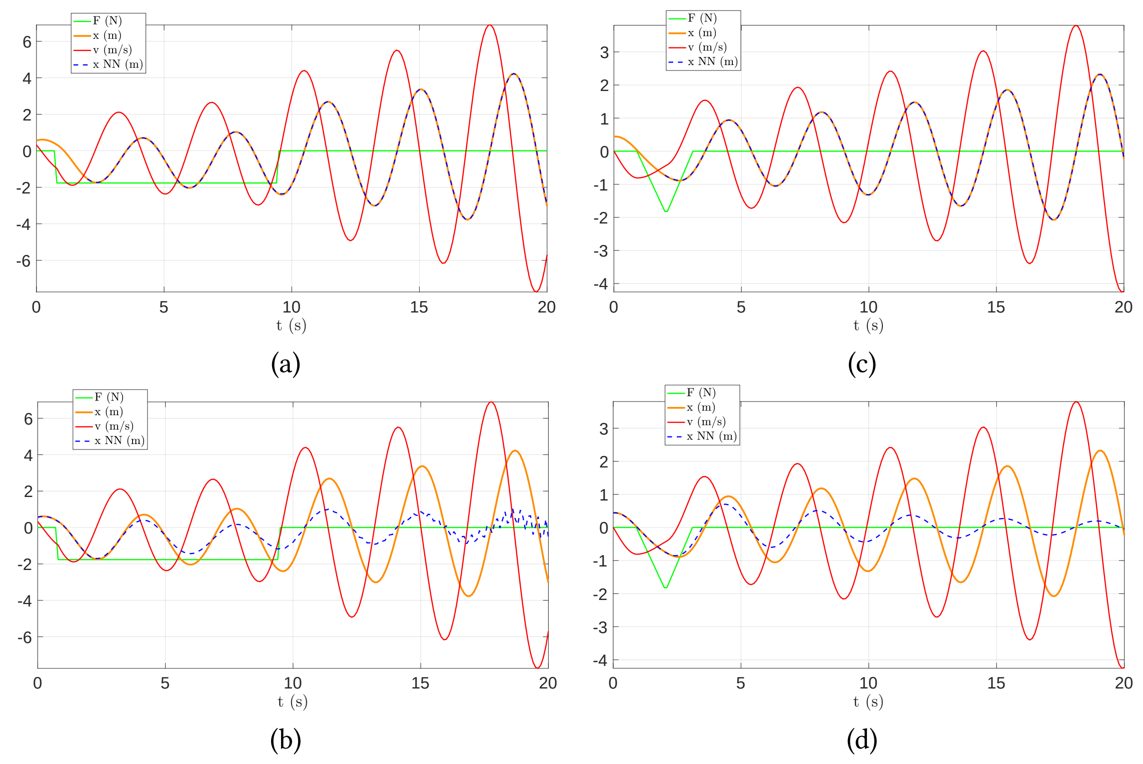


Figure 4.3: Position predictions (blue dashed lines) compared with the ground truth (orange lines) with a structured neural network with single-shot inference (a) and open loop inference (b). The same experiments is repeated in (c,d) after a fine-tuning of the whole network with a recurrent training. The step input is represented with the green lines, while the red lines depict the velocity.

We obtained a Root Mean Squared Error (RMSE) on the training set and on the validation set of 0.002872 m and 0.002695 m, respectively. Then, we tested the trained network on performing prediction on a different batch of data. The same simulator used for generating the training dataset was employed for creating new ground truth with the same set of parameters and inputs chosen for the dataset. In Figure 4.3-a we report the results of the linear position prediction on a generated example. For each step, we used the network to infer the future position given the n past samples and the input. After the first 2 s of observation, the prediction (blue dashed line) perfectly follows the ground truth position (orange line), even in the case of forced response under a step input (green line) and in the case of free response for the whole simulation time. We further compared the predicted positions and the actual ones with an “open loop” inference: the firsts n samples were collected and used to estimate the position at the next step, then we iteratively repeated the procedure by shifting the samples window by one step and predicting the subsequent positions without using the ground truth data but relying on the network predictions only. From the results depicted in Figure 4.3-b we can notice that the estimates of the neural networks have an underdamped response with increasing oscillations nearby the end of the simulation. Nevertheless, the prediction error is below 10% until about 2 seconds of prediction. We fine-tuned the neural network with a second training in a recurrent fashion, keeping the same hyperparameters, that is, we incorporated the model into a recursive cell, and we trained the resulting network on contiguous input and output windows of $p = 50$ steps. This way, while maintaining the same number of neurons, we forced the neural network to learn the temporal relationships between

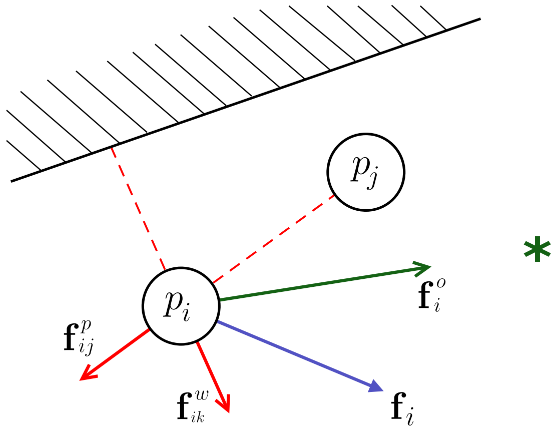


Figure 4.4: Social Force Model representation. The i-th pedestrian is pushed towards the goal by an attractive force $\mathbf{f}_i^o$, while its repulsed by the nearby pedestrian ($\mathbf{f}_{ij}^p$) and from the static obstacle ($\mathbf{f}_{ik}^w$), so that the total social force actin is $\mathbf{f}_i$.

input and output, and making it clear that the network has an internal state, which influences the output. The training procedure took about 56 s, and we obtained RMSE of 0.002932 m and 0.002731 m on the training set and on the validation set, respectively. Still, the single-shot prediction was excellently comparable with the ground truth (Figure 4.3-c), while the open loop prediction did not present any spurious oscillations, as can be seen in Figure 4.3-d.

4.4 Model

Human motion predictions are generated using an embedding of the SFM into a structured neural network, i.e., a neural network organised such that the neurons process the input signals according to (4.7). Two separate branches are designed to estimate the SFM forces in two different scenarios. In the case of the *open environment* scenario, the agent moves freely towards its goal, so it is subject only to the force term in (4.8). In the second scenario, the pedestrian moves across a space cluttered with obstacles and is affected by the repulsive force (4.9). We will henceforth refer to this scenario as *structured environment*. Consequently, each network branch models effects of different nature, i.e. attractive or repulsive forces, which are summed up at the end to produce the resulting force.

4.4.1 Social Force Model

Firstly, we briefly review necessary background material on the Social Force Model. In the SFM (Helbing and Molnar, 1995), the i-th pedestrian is modelled as a particle with mass m_i and radius r_i (see Figure 4.4). Its position is denoted as $\mathbf{p}_i = [x_i, y_i]^T$ and is expressed in the frame $\langle F \rangle = \{X_f, Y_f\}$. The human moves towards his/her target at a certain desired walking speed with magnitude v_i^d and following a second order dynamic. At the same time, the motion is perturbed by the environment, e.g. fixed obstacles, walls, furniture, etc., and other agents in the environment. Omitting the subscript i for readability, the total force $\mathbf{f}$ that acts on the i-th pedestrian is given by $\mathbf{f} = \mathbf{f}^o + \mathbf{f}^e$, i.e.

$$m\dot{\mathbf{v}} = \mathbf{f}^o + \sum_{j(\neq i)} \mathbf{f}_j^p + \sum_k \mathbf{f}_k^w, \tag{4.7}$$

where $\mathbf{v} = \dot{\mathbf{p}}$. Moreover, the attractive force $\mathbf{f}^o$ is defined as

$$\mathbf{f}^o = \frac{m}{\tau}\left[v^d(t)\mathbf{e}^d(t) - \mathbf{v}(t)\right], \tag{4.8}$$

where the characteristic time $\tau > 0$ parameter determines the rate of change of the velocity vector, while $\mathbf{e}^d$ is the unit vector pointing towards the goal. The force exerted by the static obstacle k on the i-th pedestrian is given by

$$\mathbf{f}_k^w = Ae^{(r-d_k)/B}\mathbf{n}_k + k_1 g\left(r - d_k\right)\mathbf{n}_k - k_2 g\left(r - d_k\right)\left(\mathbf{v}\cdot\mathbf{t}_k\right)\mathbf{t}_k, \tag{4.9}$$

i.e. it is the sum of a repulsive component, a compression force and a sliding friction force. We denote by $d_k = ||\mathbf{p} - \mathbf{p}_k||$ the distance between the pedestrian centre of mass and the coordinates of the obstacle closest point, so that

$$\mathbf{n}_w = (\mathbf{p} - \mathbf{p}_k)/d_k \quad \text{and}$$
$$\mathbf{t}_k = [-\mathbf{n}_k(2), \mathbf{n}_k(1)]^T$$

are the distance unit vector and its tangential direction, respectively. The function $g(x) = \max\{0, x\}$ models the fact that both the compression and the sliding friction forces exist only if the pedestrian touches the obstacle (i.e., $d_k > r$). A, B, k_1 and k_2 are the model parameters. Similarly, the force exerted by pedestrian j-th on i-th pedestrian is

$$\mathbf{f}_j^p = A_p e^{(2r-d_j)/B_p}\mathbf{n}_j + k_1 g\left(2r - d_j\right)\mathbf{n}_j - k_2 g\left(2r - d_j\right)\Delta v_j \mathbf{t}_j, \tag{4.10}$$

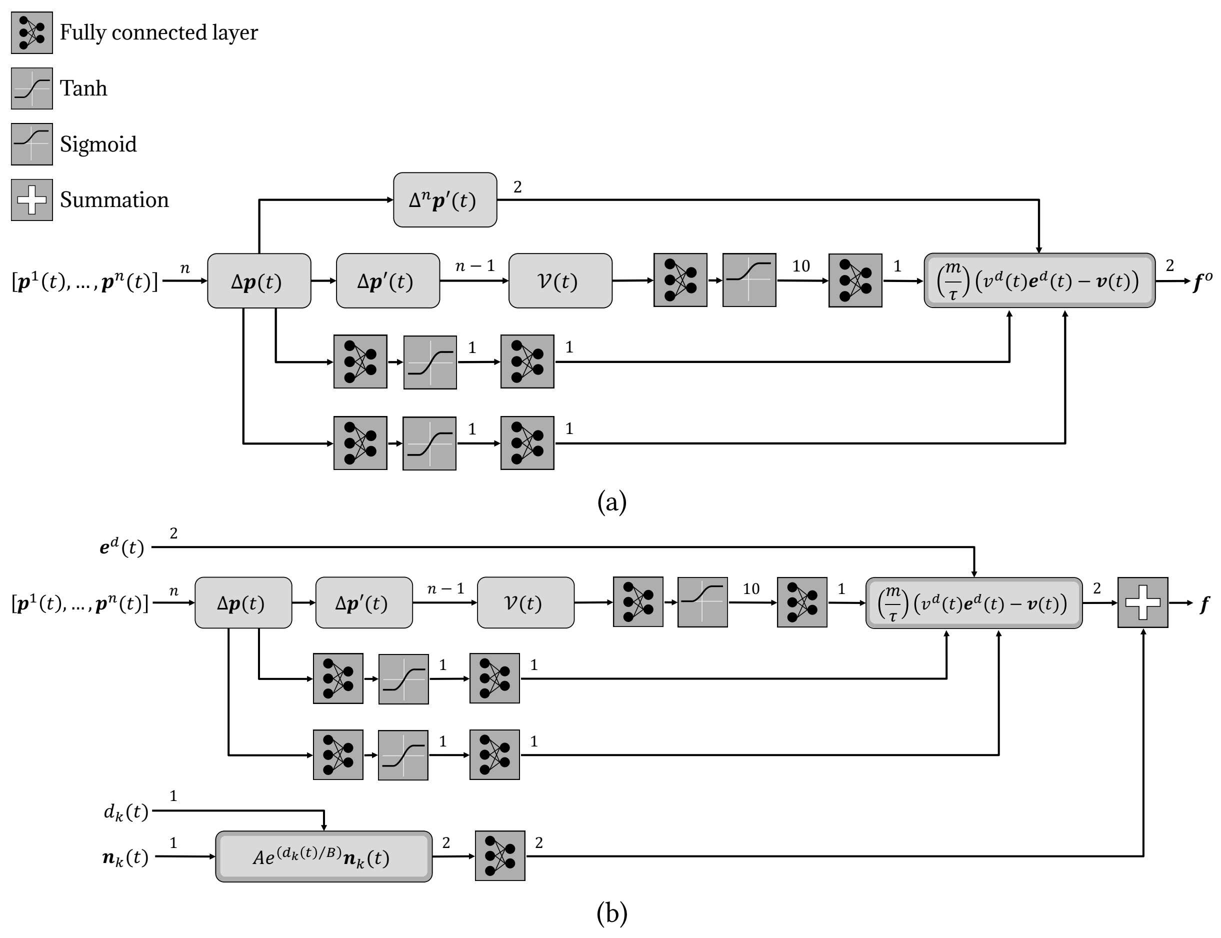
where $d_j = ||\mathbf{p}-\mathbf{p}_j||$ is the distance between the two pedestrians' centres of mass, $\Delta v_j = \left(\mathbf{v} - \mathbf{v}_j\right)^T \mathbf{t}_j$ is the tangential velocities difference, and $\mathbf{n}_j$, $\mathbf{t}_j$ are the distance unit vector pointing from pedestrian j to pedestrian i and its tangential direction, respectively. In the following we will neglect the interaction forces $\mathbf{f}_j^p$ with the j-th pedestrian in (4.7) for the time being.

4.4.2 Open environment

While freely moving towards the desired goal, the pedestrian is only affected by the force $\mathbf{f}^o$ in (4.8). Hence, the first branch of the neural network (named *Net1*) is used to predict the two force components f_x^o, f_y^o. The network inputs are the n more recent samples of the past $\mathbf{p}$ coordinates of the pedestrian (from the current time t) obtained with a sampling period δ_t. In order to avoid spatial biases, the coordinates are shifted with respect to the first sample of the window, so as to generate the input vector $\Delta\mathbf{p}(t) \in \mathbb{R}^{2\times n}$ as in the following equation

$$\begin{aligned}\Delta\mathbf{p}(t) = &[\mathbf{p}(t-(n-1)\delta_t), \mathbf{p}(t-(n-2)\delta_t), \ldots, \mathbf{p}(t)] + \\ &- \mathbf{p}(t-(n-1)\delta_t)\mathbf{1}_n,\end{aligned} \tag{4.11}$$

where $\mathbf{1}_n$ is an n-dimensional column vector with all ones. The first part of *Net1* consists of two hidden layers with no biases and with only one fully connected output neuron that learns the instantaneous velocity v_x, v_y on the X_f and Y_f axis, respectively. These two layers are followed by a $\tanh(\cdot)$ activation function. Another neuron with no bias follows each layer in order to facilitate the convergence of the estimates to their actual range. Moreover, the most recent relative motion measurement $\Delta^n\mathbf{p}(t) =$



Fully connected layer
Tanh
Sigmoid
Summation
$[\boldsymbol{p}^1(t),\dots,\boldsymbol{p}^n(t)]$
$\Delta^n\boldsymbol{p}'(t)$
$\Delta\boldsymbol{p}(t)$
$\Delta\boldsymbol{p}'(t)$
$\mathcal{V}(t)$
$\left(\frac{m}{\tau}\right)(v^d(t)\boldsymbol{e}^d(t)-\boldsymbol{v}(t))$
$\boldsymbol{f}^o$
(a)
$\boldsymbol{e}^d(t)$
$d_k(t)$
$\boldsymbol{n}_k(t)$
$Ae^{(d_k(t)/B)}\boldsymbol{n}_k(t)$
$\boldsymbol{f}$
(b)

Figure 4.5: Schemes of the *Net1* (a) and the *Net2* (b). The numbers on the connections between the layers represent their output size.

$\mathbf{p}(t) - \mathbf{p}(t-1)$ is used to estimate the components of the normalised unit vector $\mathbf{e}^d$ pointing to the goal. Finally, the vector of the desired velocity v^d is derived from the velocity magnitudes $\mathcal{V}(t) \in \mathbb{R}^n$ estimated by the following

$$\begin{aligned} \Delta\mathbf{p}'(t) &= \left[\Delta^2\mathbf{p},\dots,\Delta^n\mathbf{p}\right] - \left[\Delta^1\mathbf{p},\dots,\Delta^{n-1}\mathbf{p}\right], \\ \mathcal{V}(t) &= \left[||\Delta^1\mathbf{p}'(t)||,\dots,||\Delta^{n-1}\mathbf{p}'(t)||\right]. \end{aligned} \tag{4.12}$$

All the estimates pass through a Lambda layer where they are combined and weighted according to the m and τ parameters in (4.8). The *Net1* output is then the estimate of $\mathbf{f}^o = \left[f^o_x, f^o_y\right]^T$. The force input (4.8) is translated in the form of a structured neural network by the following

$$\mathbf{f}^o = \overbrace{\text{sig}\left(\mathcal{V}(t)W_v\right)w_{vs}}^{\frac{mv^d}{\tau}} \overbrace{\frac{\Delta^n\mathbf{p}(t)}{||\Delta^n\mathbf{p}(t)||}}^{\mathbf{e}^d(t)} - \overbrace{\tanh\left(\Delta\mathbf{p}(t)W_\mathbf{v}\right)\odot w_{\mathbf{v}s}}^{\frac{\mathbf{v}(t)m}{\tau}}, \tag{4.13}$$

where $W_v \in \mathbb{R}^{10\times n-1}$, $w_{vs} \in \mathbb{R}^{10}$, $W_\mathbf{v} \in \mathbb{R}^{2\times 20}$, $w_{\mathbf{v}s} \in \mathbb{R}^{1\times 2}$, are the weight matrices and $\odot$ is the Hadamard product. The sig$(\cdot)$ sigmoid activation function is used to keep a positive sign for the v^d estimate, which then are rescaled by the weight matrix w_{vs}. The number of learnable parameters for *Net1* (represented in Figure 4.5-a) is 123, where 100 of them are entirely devoted to the desired velocity estimates.

4.4.3 Structured environment

For the environment with obstacles, the second branch *Net2* has two parallel sub-branches to predict $\mathbf{f}^o = \left[f_x^o, f_y^o\right]^T$ (described above) and $\mathbf{f}_k^w = \left[f_x^w, f_y^w\right]^T$ (reported in (4.9)) components of the force, respectively. While in the case of open environment the direction of the motion offers a good clue on the goal of the pedestrian, this is not the case in presence of obstacles. In fact, a given direction of motion can be chosen either because it leads to the goal or because it is a good way to avoid the obstacles (Rudenko et al., 2020). For this reason, we directly provide $\mathbf{e}^d$ as input to the sub-branch that estimates the attractive force $\mathbf{f}^o$. Our strategy for choosing the goal position (and, hence, $\mathbf{e}^d$) is described later in Section 4.5. In order to reduce the learning complexity, we neglected the compression and the sliding friction forces, both in the SFM simulations and in the neural network, since they only play a role during contacts (which should be avoided by design of the planned path). The second sub-branch of the network then comprises a Lambda layer (followed by two single neuron layers with no bias) that takes as inputs the distance d_k and the components of the unit vector $\mathbf{n}_k$ at time t. The inputs are combined in an exponential form as in (4.9), where the only two learnable weights reflect the A, B parameters of the SFM. The formulation of the total force $\mathbf{f}$ in the structured neural network form, as shown in Figure 4.5-b, is then given by

$$\begin{aligned}\mathbf{f} =& \operatorname{sig}\left(\mathcal{V}(t)W_v\right) w_{vs}\mathbf{e}^d(t) - \tanh\left(\Delta\mathbf{p}(t)W_\mathbf{v}\right) \odot w_{\mathbf{v}s} + \\ &+ \left(w_A e^{d_k(t)/w_B}\mathbf{n}_k(t)\right) \odot w_{\mathbf{f}s},\end{aligned} \tag{4.14}$$

where again $w_A, w_B \in \mathbb{R}^1$, and $w_{\mathbf{f}s} \in \mathbb{R}^{1\times 2}$ are the new 4 learning weights.

Since the predictions are quantifiable values, both networks were trained using a mean squared error loss on the predicted forces as in the following equation

$$\mathcal{L}_\mathbf{f} = \operatorname{MSE}\left(\mathbf{f}_i, \tilde{\mathbf{f}}_i\right) = \frac{1}{N}\sum_{i=0}^{N}\left(\mathbf{f}_i - \tilde{\mathbf{f}}_i\right)^2, \tag{4.15}$$

where N is the total number of samples in the training set and $\tilde{\mathbf{f}}$ is the ground truth future force of the agent. Compared to other common metrics, the MSE is easy to interpret, although the squaring part of the function magnifies the outliers.

4.5 Multi-goal prediction

The branch *Net2* described in the previous Section 4.4.3 takes as input the position of the goal. Our strategy is a multi-goal inference consisting of two steps. The first step is to formulate several hypotheses on the goal and carry out a prediction based on *Net2* for each. The second step is to perform a likelihood analysis based on the Multiple Model Approach (MMA) presented in Bar-Shalom et al., 2004. We apply both the first-order generalised pseudo-Bayesian (GPB1) estimator and the Interactive Multiple Model (IMM) estimator to each predicted trajectory and select the one with the highest confidence, i.e., the most probable goal.

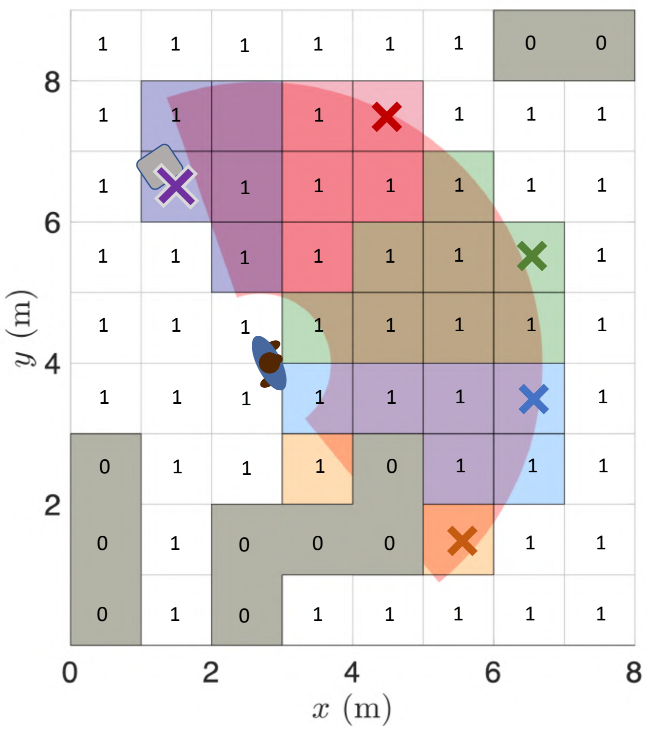


Figure 4.6: Pedestrian's AoII and the cells grouped according to (4.16). The cells are labelled with 0 if occupied by an obstacle, 1 otherwise. The coloured crosses represent the selected goals. The AoII is estimated first removing the obstacles from the disk sector (shaded area), then it is split into different sub-sectors and for each of them we select a candidate either on a point of interest (purple cross) or via geometric considerations.

4.5.1 Selection of the goals

As the person moves across the environment, s/he chooses the next goal within an area defined as Area of Immediate Interest (AoII). Roughly speaking, the AoII is the region of the space that contains all the entities that are likely to influence a human's motion at the current time and in the near future. The AoII contains the possible goals and is estimated in a different ways depending on our prior knowledge of the environment.

Under the observation that humans move preferably headways (Arechavaleta et al., 2008) and from the related works on the personal space commented in Section 2.2, it is reasonable to expect that the motion is attracted by an entity inside a disk sector with aperture lower than 180°. Our experiments suggested that a good choice for the aperture is $\theta = 160°$, which incidentally is an approximation of the human field of view, including binocular and peripheral vision (Harrington, January, 1981). The radius of the sector is related to the distance that can be travelled in the interval time corresponding to the prediction horizon.

Considering that the possible goals cannot be within an obstacle, we can estimate the AoII by intersecting the disk sector with an occupancy grid derived from the map (see the shaded area in Figure 4.6). In formal terms the grid map, composed of N cells, is denoted as $\mathcal{C} = \{c_1, \ldots, c_N\}$, whose elements are labelled with 0 if occupied by an obstacle, 1 otherwise (see Algorithm 2). The AoII at time t is denoted by $\mathbf{c}(t) \subseteq \mathcal{C}$ and is defined as:

$$\mathbf{c}(t) = \{\forall c \in \mathcal{C} : [\text{Disk}(t) \cap c] \neq \emptyset \wedge c = 1\}, \qquad (4.16)$$

where Disk(t) is the disk sector described above centred in the position of the human.

The AoII is segmented in $K-1$ equally spaced radial cones with aperture ϑ, each of them representing a different area of interest. With an odd number of cones, we have one cone associated with the human's decision to walk forward, while the others represent potential turns. Similarly to (4.16), we define the k-th (for $k = 1, \dots, K-1$) cone cells as $\mathbf{c}_k(t) \subset \mathbf{c}(t)$. For each $\mathbf{c}_k(t)$ we select a goal $g_k(t)$ according to the following logic. If a known point of interest falls within $\mathbf{c}_k(t)$ we set $g_k(t)$ equal to the point of interest, otherwise $g_k(t)$ is given the cell centroid maximising the Manhattan distance to the pedestrian occupied cell ($c_{\mathbf{p}}(t)$) and minimising the angular displacement from the k-th cone bisecting direction ($\psi_k(t)$), i.e.

$$g_k(t) = \arg\left(\max_{c\in\mathbf{c}_k(t)} ||c - c_{\mathbf{p}}(t)|| \min_{c\in\mathbf{c}_k(t)} \angle\overline{c\, c_{\mathbf{p}}(t)} - \psi_k(t)\right).$$

In cases where the the map of the environment is not available or is incomplete, the occupancy grid could not reveal the position of some obstacles (i.e., we could have some cells labelled as 1, whereas they should be labelled as 0). As a consequence, we could select some candidate goals in unrealistic or impossible positions. This is not a critical problem because the estimation technique described below will quickly deplete the likelihood of fictitious goals. Still, the use of additional information enables us to purge unrealistic candidates and, hence, leads to a better performance. For instance, the robot's sensors could reveal the presence of some of the obstacles despite their limited visibility of the scene allowing us to correctly label as 0 at least some of the occupied cells.

To model the a-priori probability of selecting one of the goals we adopt the von-Mises distribution, that is a Gaussian pdf with mean $\phi(t)$ (the pedestrian actual heading), variance σ^2 and normalised in $[-\pi, \pi]$, denoted with

$$\mathcal{N}_{VM}(\phi(t), 1/\sigma^2) = \exp(\kappa \cos(\vartheta_k - \phi(t)))/(2\pi I_o(\kappa)),$$

where κ is the concentration parameter, and $I_o(\kappa)$ is the modified Bessel function. In practice, the confidence of the k-th goal belonging to the cone with aperture ϑ_k is given by

$$\Pr\left[g_k(t)|\mathbf{p}^t\right] = \int_{\vartheta_k} \mathcal{N}_{VM}(\nu; \phi(t), 1/\sigma^2) d\nu.$$

All the goals described above represent the human's intention to continue walking, preferably forward. To model the intentions to stop, or eventually make a "U turn", we add one goal $g_K(t)$ placed in the current pedestrian position $\mathbf{p}^t$. To set the confidence of $g_K(t)$, we consider an heuristic value equal to half the $\min_{k=1,\dots,K-1} \Pr\left[g_k(t)|\mathbf{p}^t\right]$.

While the agent moves in the environment, the goal positions are updated in order to be compliant with the current pedestrian location. Specifically, knowing the set of goals at $t-1$, we first compute the AoII geometry at t obtaining a new set of goals, then we solve the proper association between $g_k(t-1)$ and $g_k(t)$ via the Munkres assignment algorithm (Munkres, 1957).

Algorithm 2 Goal selection algorithm

```
Input: CAD map, $N$, $K$, GridSize, $\mathbf{p}(t)$, $\phi(t)$
 1: $\mathcal{C} \leftarrow$ CreateGridMap (CAD map, $N$, GridSize) // Cell map $\{c_1, \dots, c_N\}$
 2: Disk$(t) \leftarrow$ RotoTranslate $(\mathbf{p}(t), \phi(t))$
 3: $j = 1, \mathbf{c}(t) = \{\emptyset\}$
 4: for $i = 1$ to $N$ do
 5:     if [Disk$(t) \cap c_i] \neq \emptyset$ and $c_i == 1$ then
 6:         $\mathbf{c}(t) \leftarrow$ Append$(c_i)$ // Add cell $c_i \subset \mathcal{C}$ to the AoII set $\mathbf{c}(t)$
 7:         $j \leftarrow j + 1$
 8:     end if
 9: end for
10: for $k = 1$ to $K - 1$ do
11:     $\mathbf{c}_k(t) = \{\}$
12:     for $i = 1$ to $j$ do
13:         if $\vartheta_k < \angle\overline{\mathbf{c}_i(t)\, c_{\mathbf{p}}(t)} < \vartheta_{k+1}$ then
14:             $\mathbf{c}_k(t) \leftarrow$ Append$(\mathbf{c}_i(t))$ // Add $\mathbf{c}_i(t)$ to the cells of $k$-th cone
15:         end if
16:     end for
17:     if PointOfInterest $\subset \mathbf{c}_k(t)$ then
18:         $\mathbf{g}_k(t) \leftarrow$ PointOfInterest // Cell goal
19:     else
20:         $\mathbf{g}_k(t) \leftarrow \arg\left(\max_{c\in\mathbf{c}_k(t)} ||c - c_{\mathbf{p}}(t)||_2 \min_{c\in\mathbf{c}_k(t)} \angle\overline{c\, c_{\mathbf{p}}(t)} - \psi_k(t)\right)$ // Cell goal
21:     end if
22: end for
23: $\mathbf{g}_K(t) \leftarrow \mathbf{p}(t)$ // Stop-goal
Output: $g_k(t), k = 1, \dots, K$
```

4.5.2 Likelihood computation

For each possible goal we compute a *Net2* prediction and execute a Kalman Filter (KF) iteration. The prediction step for the k-th goal is given by

$$\begin{aligned} \bar{\mathbf{s}}_k^{t+1} &= \mathcal{F}(\bar{\mathbf{s}}_k^t, \mathbf{f}), \\ \Sigma_k^{t+1} &= A\Sigma_k^t A^T + BQB^T, \end{aligned} \tag{4.17}$$

where $\bar{\mathbf{s}}_k^t = \left[\bar{\mathbf{p}}_k^t, \bar{\mathbf{v}}_k^t\right]^T$ is the state at time t, $\mathcal{F}(\cdot)$ represents the second-order dynamic model (4.7), and $A = \frac{\partial\mathcal{F}(\bar{\mathbf{s}}_k^t,\mathbf{f})}{\partial\bar{\mathbf{s}}_k^t}$ is the linearised system dynamic matrix. Moreover, Σ_k^t is the covariance matrix of the estimation error associated with the k−th goal state, $B = \frac{\partial\mathcal{F}(\bar{\mathbf{s}}_k^t,\mathbf{f})}{\partial\mathbf{f}}$ is the force linearised input vector, having additive uncertainties with covariance matrix Q (that we assume to be proportional to a perturbation in the acceleration space). In the update step, we use the observations $\mathbf{p}^{t+1}$ to evaluate the innovation vector $\boldsymbol{\varepsilon}_k^{t+1}$, its covariance S_k^{t+1}, and update the state covariance, i.e.

$$\begin{aligned} \boldsymbol{\varepsilon}_k^{t+1} &= \mathbf{p}^{t+1} - H\bar{\mathbf{s}}_k^{t+1}, \\ S_k^{t+1} &= H\Sigma_k^{t+1}H^T + R, \\ K_k^{t+1} &= \Sigma_{i,k+1}^{-}H^T(S_k^{t+1})^{-1}, \\ \Sigma_k^{t+1} &= \Sigma_k^{t+1} - K_k^{t+1}H\Sigma_k^{t+1}, \end{aligned} \tag{4.18}$$

where R is the covariance matrix of the measurement uncertainty and $H = [I, 0]^T$ is the output matrix. The computation of the likelihood can be carried out through two different approaches: GBP1 and IMM (Bar-Shalom et al., 2004). The former is easy to set up but has a good performance only when the choice of the goal is relatively consistent in time (i.e, the human does not change her/his mind on where to go). The latter caters for possible changes in the goal through an homogeneous Markov Chain (MC). The higher level of generality of IMM requires some additional effort in the calibration of the transition probabilities in the MC.

When using GPB1, the probability of $g_k(t+1)$, dubbed μ_k^{t+1}, can be estimated using the following Bayesian rule:

$$\mu_k^{t+1} = \Lambda_k^{t+1} \mu_k^t / \Pr\left[\mathbf{p}^{t+1}\right], \tag{4.19}$$

where $\Lambda_k^{t+1} = \Pr\left[\mathbf{p}^{t+1} | g_k(t+1)\right]$ is the likelihood of the observed position $\mathbf{p}^{t+1}$ for the goal $g_k(t+1)$, which can be modelled as a bivariate Gaussian pdf with mean ε_k^{t+1} and covariance S_k^{t+1}. Unlike the standard GPB1 approach, where the state estimates for each model (i.e. goal) are fused together, we let each KF run independently, therefore the updated states are not needed.

In the more general case in which we use the IMM approach, we combine the goal probabilities μ_k^{t+1} in (4.19) with the transition probabilities p_{kj} of dynamically changing the goal, i.e. $p_{kj} = \Pr\left[g_k(t+1) | g_j(t)\right]$. The goal switching is assumed to be a homogeneous Markov process, so that the transition probabilities are known and time-invariant. Hence, in the IMM, (4.19) is substituted with the mixing probabilities

$$\mu_k^{t+1} = \frac{\Lambda_k^{t+1}}{\sum\limits_{k=1}^{K} \Lambda_k^{t+1} \sum\limits_{j=1}^{K} p_{kj} \mu_j^t} \sum_{j=1}^{K} p_{kj} \mu_j^t. \tag{4.20}$$

The Markov chain transition probabilities are defined as a stochastic matrix. The persistence of each mode is equal to a probability α, while the transition towards each other mode is an even distribution of $1-\alpha$, i.e.

$$p_{kj} = \alpha I + \frac{1-\alpha}{K-1}(\mathbf{1} - I), \tag{4.21}$$

where $\mathbf{1}$ is a matrix with all ones and proper dimensions.

As the agent moves in the environment, we eventually repeat the *Net2* predictions based on the new measured human position, and the updated goals. This occurrence is triggered by one of the following events: *i*) the average among all the innovations (i.e., the norm of ε_k^t in (4.18)) is greater than an upper limit ε_M; *ii*) the Euclidean distance between the observation $\mathbf{p}^t$ and one goal $g_k(t)$ is lower than δ_M (i.e. the pedestrian is close to the k-th goal). When any of these conditions befalls and we reinitialise $g_k(t)$ to $g_k^\star(t)$, its first confidence $\mu_k^{t,\star}$ is obtained using μ_k^t as a prior, i.e.

$$\mu_k^{t,\star} \leftarrow \lambda \mu_k^t + (1-\lambda) \Pr\left[g_k^\star(t) | \mathbf{p}^t\right], \tag{4.22}$$

where $\lambda \in [0,1]$ is a weighting parameter. The rationale is that the probabilities reached before the reinitialisation can hold as a starting guess, thus we mix them with the a-priori $\Pr\left[g_k^\star(t) | \mathbf{p}^t\right]$.

4.6 Training and experimental validation

In this section we show a number of experimental results reported on well known datasets and on other data sets of our making. We first describe the training phase, which was totally conducted using synthetic data, and then we will discuss the performance of our predictor in different operating conditions.

4.6.1 Training details

Both *Net1* and *Net2* were implemented in Keras and trained with the Adam optimiser with a learning rate of 0.005, batch size 128, and number of epochs 300 using a 2.7 GHz Intel Core i7 processor. We created two synthetic sets of trajectories generated by the SFM in an open space (for *Net1*) and in a structured environments (for *Net2*). For the structured case, the environment we assumed consisted of two intersecting corridors (as in Figure 4.9-a). In the first set, we ran 800 simulations of 20 seconds with a sampling time of $\delta_t = 0.1$ s. The initial positions were randomly chosen within a range between 8 and 10 m from the final goal, and, for each simulation, a random set of parameters for the SFM model in (4.8) were taken from the intervals $[50, 90]$ kg for m, $[0.5, 0.9]$ s for τ and $[0.5, 3]$ m/s^2 for v^d. For the second set, we ran 1200 simulations where the agent moves through the corridors intersection, starting from one of the four possible waypoint areas (see Figure 4.9-a) and reaching another one. We set the parameters in (4.9) to $A = 1000$, $B = 0.08$, according to Farina et al., 2017b. The window of the motion observations was empirically set to $n = 10$ steps, which provides a good trade-off between learning speed and network prediction accuracy without over-fitting. This is consistent with the fact that the networks mostly depend on the most recent data, and that a longer motion observation does not significantly improve the prediction accuracy (Schöller et al., 2020).

The 70% of each synthetic dataset was used as the training sets, while the remaining samples were used for validation. Notice that, in order to avoid possible correlations between training and validation, the samples randomisation was done after dividing the two sets. In Table 4.2 we report the hyperparameters and the implementation characteristics of our proposed networks *Net 1* and *Net 2* and other comparative models used in the experiments in Sections 4.6.3 and 4.6.4.

4.6.2 Validation on simulated data

After the training, we performed a first validation step on a different batch of simulation data. The difference between this batch and the one used for training lies in the geometric configuration of the simulated environment (which was no longer a simple intersection of two corridors).

SFM forces generation. For this evaluation, we first generated the entire trajectory. For each step, the network was used to infer the force sample. In Figure 4.7 we report the results of the force predictions of the *Net2* network (similar results are obtained for *Net1*). Figure 4.7-a depicts the trajectories generated by the SFM, while Figure 4.7-b the comparison between the real and the network predicted force components $f_x = f_x^o + f_x^w$ and $f_y = f_y^o + f_y^w$ described in (4.8) and (4.9). Despite a slight underestimation of the forces in the occurrence of the horizontal collisions with the walls (see the peaks in Figure 4.7-b), the prediction remains consistent even if the configuration of the obstacles was very different from the one used in training (see Figure 4.7-a). We can legitimately conclude that the performance of *Net2* has not been negatively affected by the environmental bias introduced during the

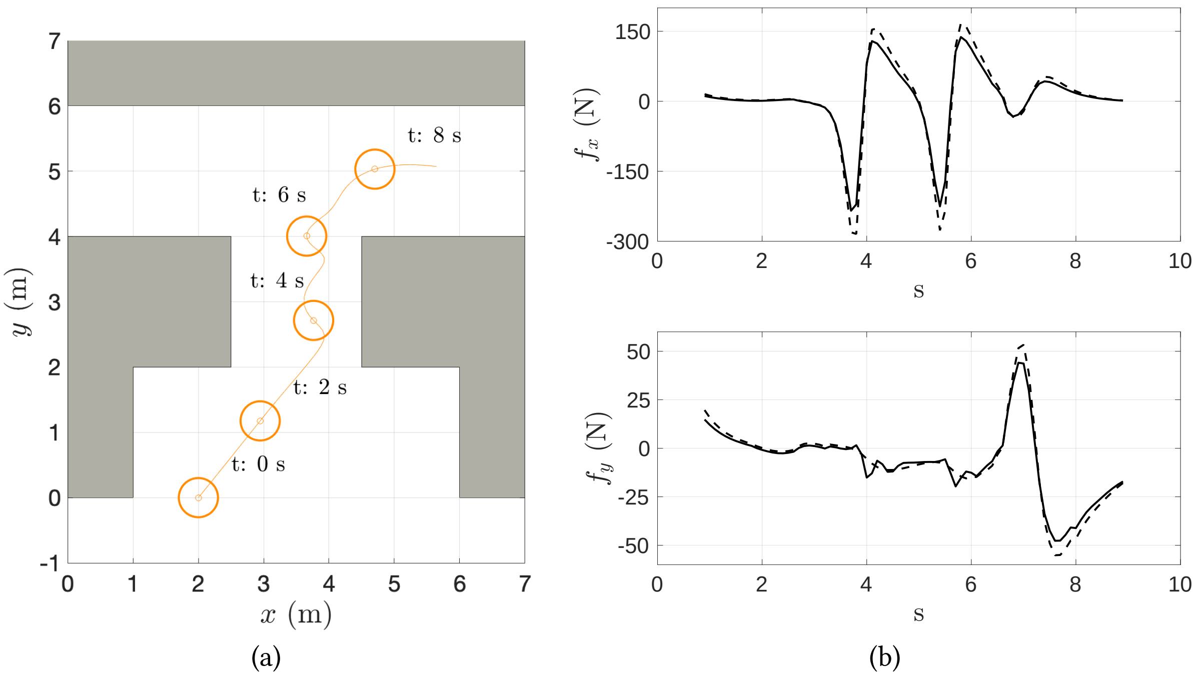
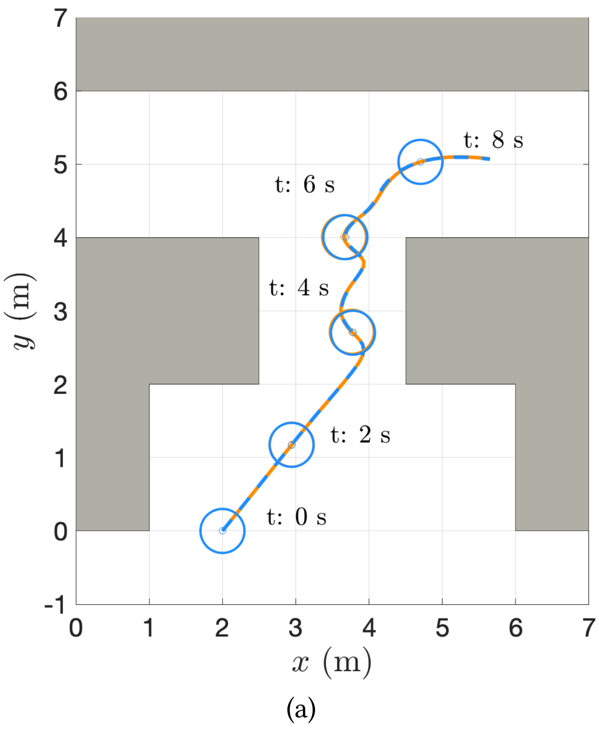


Figure 4.7: Validation example. (a) SFM simulated trajectories and (b) *Net2* forces predictions (solid) compared with the SFM forces (dashed).

training phase. The structure of our NN and the abstract modelling of the environment guides the learning process toward a correct understanding of the "physics" of the human motion rather than of specific behaviours, and makes the learning results usable across a wide gamut of possible environments. This fact is further substantiated by the inspection of the learned weights. In our example, the weights of the second branch $w_A w_{\mathbf{fs}} = 3.4 \cdot 10^3$ and $w_B = 0.0834$ are pretty close to the A and B parameters, respectively, used in the synthetic dataset. Likewise, if we combine the weights of the first branch, as in (4.14), we obtain an estimated mass of 55.6 kg, which is correctly positive and of the same order of magnitude of the chosen m. Since the NN internal weights correspond to physical quantities, unexpected values (e.g., negative masses, exaggerated velocities) are used to reveal either the absence of a real convergence or that the system has been over-fitted. This "interpretability" of the results is the most important trait of the future generation of explainable AI systems (Barredo Arrieta et al., 2020).

Long term prediction. Given that the forces are correctly estimated, we fed those quantities into the SFM (4.7) to generate long term motion predictions. To this end, we first collected one second of measurements from a trajectory corresponding to a window of $n = 10$ samples. These measurements were then used to estimate of the SFM force, and, hence, the position at the next step. The whole procedure was iteratively repeated shifting each time ahead by one sample the window. This way, after 10 prediction steps, the computation was made "in open loop", relying solely on the SFM predictions. The comparison between the predicted trajectory and the actual one is shown in Figure 4.8 for two different scenarios. In both scenario, the actual trajectories (solid lines) are well replicated by the predicted human motions (dashed lines) even for a long horizon (7 or 9 seconds, respectively).

The case of uncertain goals. In the example described above, our objective was to validate the NN based prediction of the human motion. For this reason, we assumed that the goal of the human



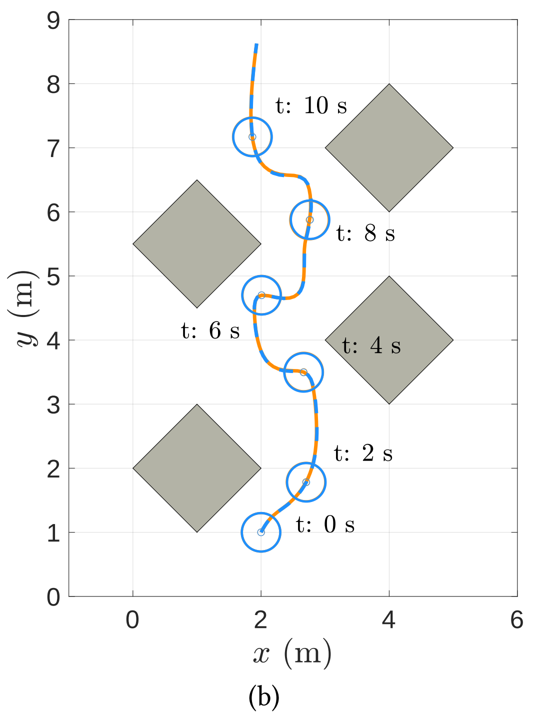
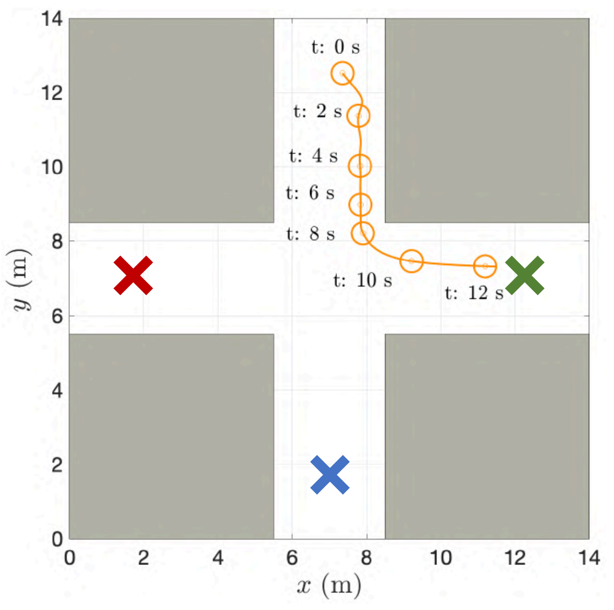


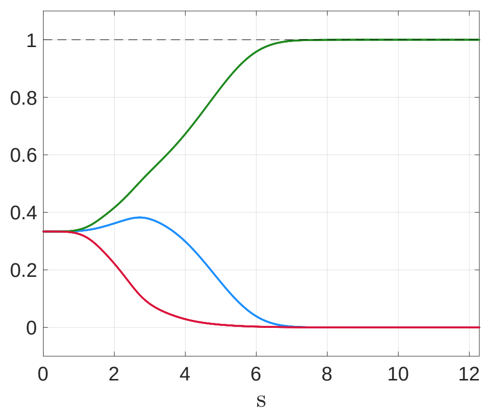
Figure 4.8: Comparison of actual (solid) and predicted (dashed) trajectories using learned *Net2* forces with $n = 10$ steps with a prediction horizon of (a) 7 and (b) 9 s in a structured environment.

target was known. Now, we move to the general case of unknown goal in order to validate the entire approach.

Let us consider again the intersecting corridors scenario. Following the method described in Section 4.5, we identify four goals, which correspond indeed to the areas chosen for the network training. The pedestrian starts from one of the areas (from the uppermost one in the experiment in Figure 4.9-a), and reaches one of the other three. Therefore, we generate three different hypotheses for the trajectory predictions with the *Net2* network (one per area, respectively) choosing as goal the area centroids. As shown in Fig 4.9-b, after about 2 s, the GBP1 classifier is able to find the correct goal, while in the following seconds the confidence towards the simulated trajectory increases. The evaluation of *Net2* with the multi-goal strategy is then further proved on real human trajectories collected in a structured environment. In particular, we record the data in two different portions of a hallway with multiple exits in our department at the University of Trento. Data were collected using a LiDAR, which was placed about 90 cm from the ground at the centre of the two scenes, in order to entirely see every side of the corridors (see Figure 4.10-a). The sensed data were used to both extract the walls information (that is, the static points between subsequent frames) and the pedestrian observations. Our acquisition algorithm was used to extract points belonging to the person's waist, and clustered them into a single planar position. In the first recorded set, depicted in Figure 4.10-a, the person could go to three different goals, i.e. one directly to the left, one to the right and one right at the end of the hallway (see the captured inlet image). In Figure 4.11-a we report the classification result for the trajectory of the *blue* goal with the GPB1 estimator. In this experiment, we manually give to the model the prior knowledge of all the goal locations, i.e. the beginning of the corridors. As reported in Figure 4.11-b, the first goal on the left of the person is discarded after about 4 s, while the confidence of the two remaining goals



(a)



(b)

Figure 4.9: (a) The agent moves from the uppermost area to the green exit on the right. The coloured crosses represent all the possible goals of the scenario. (b) Probabilities of the three goals estimated by the GPB1 method.

remains almost the same, until the correct goal is found after about 6 s, before the pedestrian oversteps the next exit. Notice that the oscillating trajectory of the orange sample is due to the SFM dynamic with respect to static obstacles.

4.6.3 Net2 performance evaluation

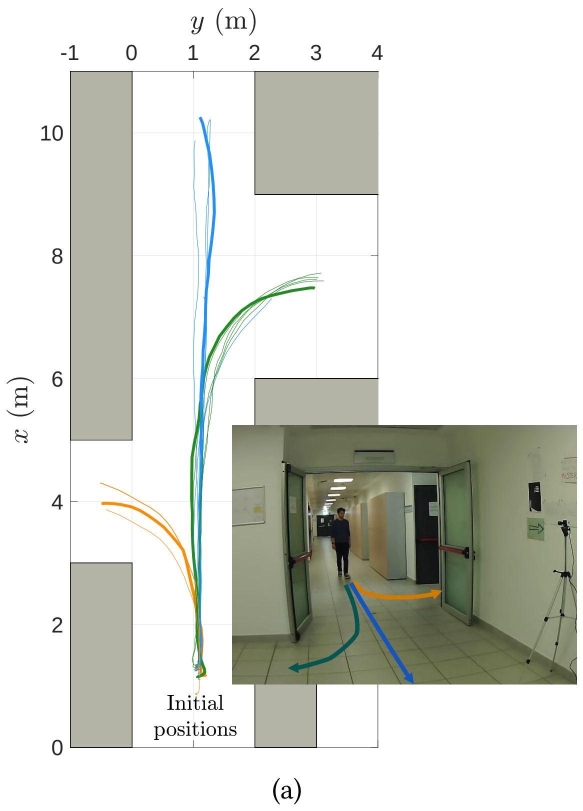
To experimentally validate the performance of the multi-goal strategy with the IMM estimator, we used the second set of actually recorded trajectories in Figure 4.10-b, where the person could go towards different exits, located at different walking directions. Wall shapes were restored by looking at the static measures of the LiDAR, while here we use no prior information on the points of interest positions. As customary in the literature from this kind of problems (Alahi et al., 2016; Rudenko et al., 2019; Schöller et al., 2020), we compute the errors using the Mean Euclidean Distance (MDE_K) and the Final Displacement Error (FDE_K) to determine the approach performance, i.e.

$$\begin{aligned} \text{MDE}_K &= \min_{i\in\{1,\dots,K\}} \frac{1}{n_f}\Sigma_{j=t+1}^{t+n_f+1} \left\|\hat{\mathbf{p}}^j - \mathbf{p}_i^j\right\|_2, \\ \text{FDE}_K &= \min_{i\in\{1,\dots,K\}} \left\|\hat{\mathbf{p}}^{t+n_f+1} - \mathbf{p}_i^{t+n_f+1}\right\|_2, \end{aligned} \tag{4.23}$$

where n_f is the number of sample of the predicted trajectory. In particular, due to the presence of multiple forecasted future trajectories, we select the best predicted trajectory among the K samples. In the literature, the observation window is usually set to 8 time-steps (that is, 3.2 s according to the data acquisition frame rate of the datasets), while the predictions span the successive 4.8 s. In our model (**SFM-NN**), we observe only for 1 s (n=10 steps) of the real-world trajectories and predict for 4.8 s for a fair comparison. Our strategy was compared with other two orthogonal approaches. We implemented

	SFM-NN (*Net1*)	SFM-NN (*Net2*)	FC 1/2/3	FF	LSTM
Prediction					
Number of parameters	123	127	22,080 / 12,380 / 9,560	2694	201,240
Inference speed (Keras version)	0.1484 s	0.1823 s	0.0012 s / 0.0015 s / 0.0680 s	-	-
Inference speed (C++ version)	60.7050 μs	70.6534 μs	65.4340 μs / 41.0731 μs / 1478.07 μ	-	-
Training					
Learning rate	0.005	0.005	0.005	0.0004	0.0004
Epochs	300	300	300	35	35
Batch size	128	128	128	64	64
Optimizer	Adam	Adam	Adam	Adam	Adam

Table 4.2: Comparison of implementation details of our proposed neural networks and related models used in the experiments reported in Sections 4.6.3 and 4.6.4. Specifically, we compare our models *Net1* and *Net2* (**SFM-NN**s), three different Fully Connected neural networks (**FC**s), the Feed Forward neural network (**FF**) and the LSTM network (**LSTM**) from Schöller et al., 2020. The speed of execution was measured by running 10000 times over a prediction window of 4.8 s: (a) the Keras implementation of each neural network (in seconds), (b) the C++ implementation of the same networks (in microseconds).



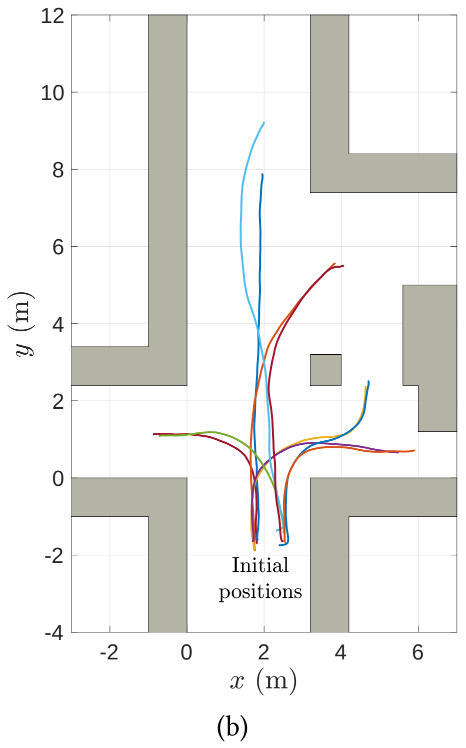


Figure 4.10: Experimental trajectories in a hallway. (a) First set of collected trajectories: bold lines are the ones used for the multi-goal classifier evaluation and a picture of the experimental set-up. (b) Second set of collected trajectories.

the Constant Velocity model (**CV**) as proposed by Schöller et al., 2020, where the multimodality is introduced by predicting $K = 20$ samples with a random noise on the walking direction, i.e. $\mathcal{N}(0, \sigma_a^2)$, with $\sigma_a = 25°$. Moreover, we trained different Fully Connected neural networks with the same hyper parameters of *Net2* with the synthetic dataset. For each network we tested several structures in order to find the best fitting. The **FC1** is made of two hidden layers with 100 and 80 neurons, both followed by ReLU activation functions. It takes as input all the past motions and has two output layers with 48 neurons each to return all the prediction steps. The **FC2** receives as inputs also explicit scene information (the same way as *Net2* does), has two hidden layers with 50 and 30 neurons, and predicts for the entire window. The **FC3** is similar to the **FC2**, however as well as our proposed approach, it predicts only the position at $t + 1$ and recursively uses the past predictions as input for the new inference. In Table 4.3 we report the prediction errors for all the evaluated models. The proposed **SFM-NN** masters the additional complexity of the scene by using *Net2*, while the influence of the environment is implicitly learned by the **FC1** and explicitly encoded in the **FC2** and **FC3**. Nonetheless, our approach outperforms all the other approaches w.r.t. both MDE and FDE and even if we drew just $K = 10$ samples. Therefore, even if the information of the environment are considered, the classical NN approaches are strongly scenario-dependent and poorly generalise when trained with synthetic datasets. Unlike other multimodal approaches, the advantage of our model is to have the a-priori confidence estimated over the different modes as depicted in Section 4.5. As shown in Table 4.2, the **FC**s obtained the fastest execution times on the Keras implementation, in particular the **FC1** and **FC2** since both are designed to infer the whole prediction window with a single execution. The peculiar structure of our networks

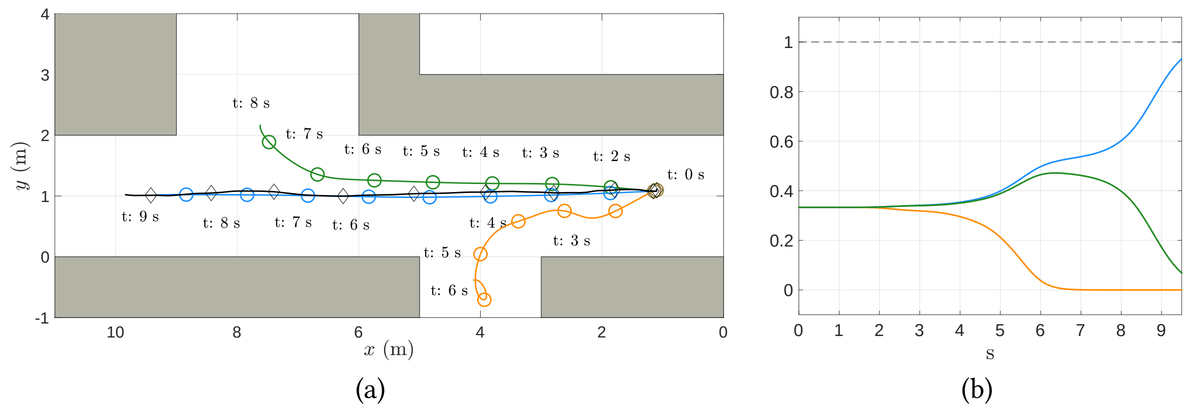


Figure 4.11: (a) Trajectory predictions compared with the ground truth (black line). (b) Probabilities of the trajectory predictions while the pedestrian moves towards the blue goal, as estimated by the GPB1 method.

	K	MDE_K	FDE_K		MDE	FDE
CV	20	0.47	1.28	CV-s	1.65	2.73
FC1	1	0.96	1.84	SFM-NN-s	**0.74**	**1.49**
FC2	1	0.96	1.97			
FC3	1	0.65	1.43	SFM-NN cl	**0.16**	**0.16**
SFM-NN	10	**0.38**	**0.68**			

Table 4.3: Comparison of prediction errors with our model (**SFM-NN**), Constant Velocity (**CV**) model and Fully Connected (**FCs**) networks, on collected trajectories. The suffix "-s" depicts the models with only the most confident estimate. The metrics are reported in meters.

makes them slightly slower in their Keras implementation, however, this is not a significant disadvantage as it is used for offline prediction. In the C++ implementation (introduced in Section 4.7.1), the inference times are suitable for real-time robotic applications.

To increase the fairness, we also endowed the **CV** model with an IMM estimator to obtain a-priori confidence as well. For comparison, we select for both the algorithms the most probable trajectory, i.e. the one with the highest confidence, and named them **CV-s** and **SFM-NN-s**, respectively. This way, in the prediction phase both models were reduced to single-modal approaches. The comparative results reported in Table 4.3 further confirm the effectiveness of our solution. For a visual representation of the behaviour of the different approaches in Table 4.3, we offer in Figure 4.12 a qualitative representation of the different predicted trajectories. The **CV** model (Figure 4.12-a) reached a good average performance. However, walls information and trajectory curvature changing are not obviously predictable based on the motion history. The **FC1** (Figure 4.12-b) properly predicts linear trajectories, while it was not able to catch the turning behaviour nor the environmental effects. Instead, even if only the most probable trajectory is predicted (i.e., **SFM-NN-s**), our model (Figure 4.12-c) correctly predicted all the walking directions and well approximated the pedestrian behaviours.

Finally, in the last column of Table 4.3 we reported the performance of the IMM estimator in closed loop, that is with the continuous update of the goal probabilities with respect to the observations, i.e. we let the *Net2* behave as a filter when new observations come. These experimental results are of paramount relevance for an application of the solution to real-time pedestrian motion estimation of

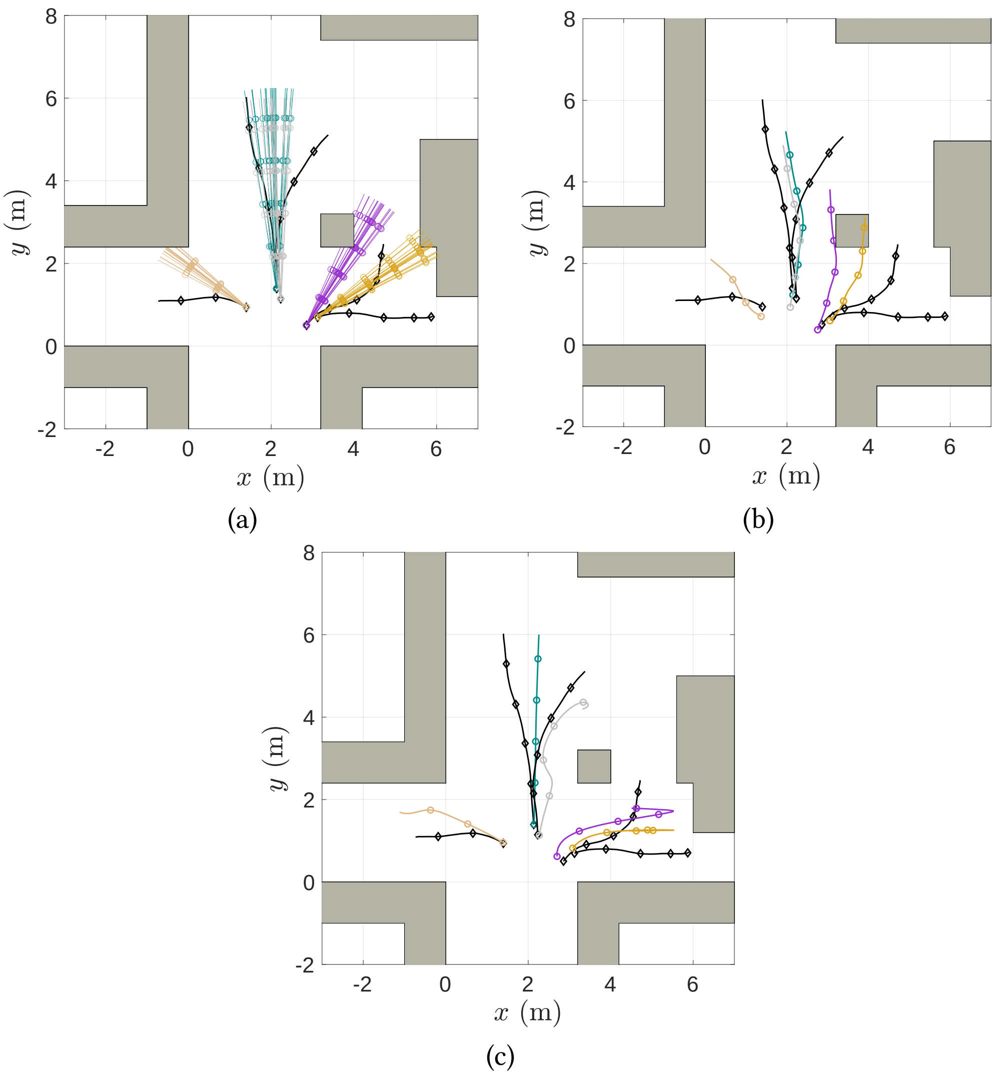


Figure 4.12: Predicted trajectories with **CV** (a), **FC1** (b) and **SFM-NN-s** (c). The ground truth is depicted in black for the five sampled trajectories, while colour codes are adopted for the corresponding predicted trajectories from the different approaches.

service robots. Notice that in this case, the performance are definitely increased and the two metrics, MDE and FDE, are obviously the same, i.e., we come up with just one prediction.

4.6.4 Net1 performance evaluation

To further substantiate the analysis, we experimentally validate the performance of the *Net1* network and make a comparison with other methods in the literature, we used two widely known human motion datasets: the ETH (Pellegrini et al., 2009) dataset (with the scene *Hotel* and *ETH*) and UCY (Lerner et al., 2007) dataset (with scene *UCY*, *Zara1* and *Zara2*). These datasets comprise real world human trajectories in open scenarios, where the influence of static obstacles is mostly negligible, thus the embedded SFM model plays a major role. The performance were evaluated using the MDE_K and the FDE_K defined in (4.23), where $K{=}1$ since we are considering for *Net1* just one model. Unlike the leave-one-out approach used in Alahi et al., 2016; Rudenko et al., 2019, we used the synthetically learned *Net1* to validate the prediction accuracy over real-world data, and we use the same observation and prediction

Dataset	Metric	CV	CA	FF	LSTM	SFM	SFM-NN
Hotel	MDE	0.27	0.95	1.59	**0.15**	0.69	0.34
	FDE	0.51	2.41	3.12	**0.33**	1.63	0.75
ETH	MDE	**0.58**	1.35	0.67	0.60	0.89	0.61
	FDE	**1.15**	3.29	1.32	1.31	2.12	1.48
UCY	MDE	0.46	0.79	0.69	0.52	0.89	**0.42**
	FDE	1.02	2.03	1.38	1.25	2.12	**1.02**
Zara1	MDE	0.34	0.59	0.39	0.43	0.61	**0.31**
	FDE	**0.76**	1.50	0.81	0.93	1.43	0.78
Zara2	MDE	0.31	0.50	0.38	0.51	0.84	**0.27**
	FDE	0.69	1.30	0.77	1.09	1.96	**0.67**
AVG	MDE	0.39	0.84	0.74	0.44	0.79	**0.39**
	FDE	**0.83**	2.11	1.48	0.98	1.85	0.94

Table 4.4: Prediction errors (average errors in AGV rows) with our model (**SMF-NN**) and other state-of-the-art models on the real-world datasets. The metrics are reported in meters.

window as in Section 4.6.3, In Table 4.4 we show the prediction errors comparison with the Constant Velocity (**CV**), the Constant Accelerated (**CA**) models and the Feed Forward (**FF**) neural network, all implemented by Schöller et al., 2020. Moreover, we report the comparison with the LSTM network (**LSTM**) by Alahi et al., 2016, and with the standard isolated SFM model (**SFM**) with its parameters optimised with the leave-one-out approach on the real dataset. Among the methods that can be adopted as a baseline, we therefore chose single-prediction models which, like the *Net1*, did not make use of additional information, such as the goal target.

At a first look, our model still shows good prediction performance in each dataset, with an average MDE of about 40 cm with the exception of the *ETH* scenario. The performance decrease in the case of the final displacements measured by FDE: this result is mainly due to the strong non-linearity of the real-world trajectories, which are not easily followed by *Net1* in open loop. However, the results are instead remarkable for the following reasons: 1. compared with the optimally trained **SFM**, the results are radically better due to the presence of the NN that is able to add flexibility to the predictions coming from the context; 2. notwithstanding the worsened training conditions of the proposed approach, the reduced observation interval and the absence of obstacles (which are the main novelty of the proposed approach, as aforementioned), our results provides similar performance than the other methods in the literature; 3. the proposed method, despite having similar performance as state-of-the-art approaches (e.g., the **CV**) in open spaces, has the potential to further model other effects, such as human to human interactions, which will be the subject of future investigations.

These experimental results are, to the best of the Authors' knowledge, the first actual evidence of a reliable human motion prediction carried out in real-time by a moving service robot in a natural environment that deals with random human beings in an unknown scenario. We want to remark here that this is actually possible if the knowledge gained by the NN is abstracted by the wired model adopted.

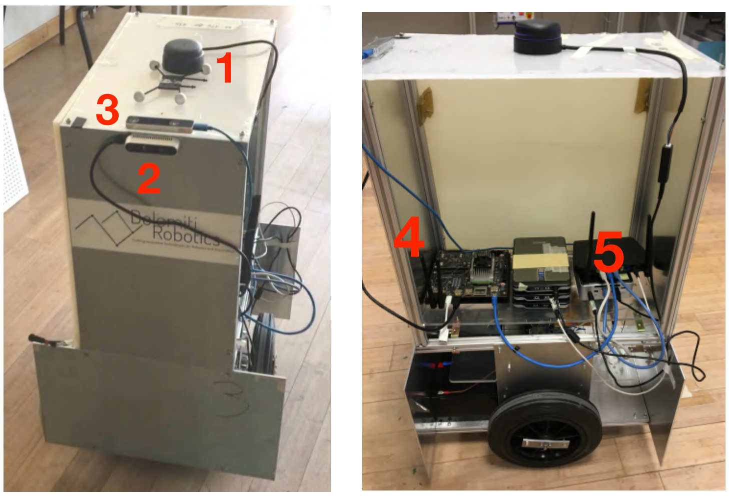


Figure 4.13: (a) Robot sensing system setup, consisting of LIDAR sensor[10] (1) and RealSense D435[11] (2) for the detection and tracking, and RealSense T265[12] (3) for the visual odometry. (b) Computing boards on the robot: Jetson TX2[8] (4) and Intel® NUC (5).

4.7 Robotic implementation and experimental results

In this section we show a set of experiments conducted using a real robot. We first describe the platform's components and the software framework, and then we will discuss the performance of the proposed solution.

4.7.1 Robotic platform

In order to create a suitable robotic platform for the experiments, we equipped a two-wheeled unicycle robot with a sensing system composed of a 2D LIDAR and an RGB-D camera (see Figure 4.13). The RGB-D sensor is used for human detection and tracking, while the LIDAR data are used to improve the tracking performance, to localise the robot in the environment and to measure the distances to the surrounding obstacles. The LIDAR (an RPLidar A3[10]) has a view angle of 360°, a maximum measuring distance up to 40 meters, and is typically operated at 20 revolutions per second. The RGB-D camera adopted is an Intel® RealSense™ D435[11], working in an ideal range spanning from 0.5 to 3 m. In addition, rotational encoders on each rear wheel and a visual inertial camera facing upwards (an Intel® RealSense™ T265[12]) provides odometry data.

The solution proposed in Sections 4.4 and 4.5 was prototyped in C++ and is comprised of two modules (see Figure 4.14): *detection and tracking* and *motion prediction*. The algorithm was executed on a Jetson TX2[8] and on a Intel® NUC, both embedded on the robot.

The *detection and tracking* module is used to: 1. provide real-time information on the presence of fixed obstacles and walls, 2. detect and track the human target. The human tracking algorithm is derived from the solutions presented in Section 3.5, while the sensors fusion is described in detail in Section 6.3.3. We first identify and track the person in the image space of the camera, and then we merge

[10] https://www.slamtec.com/en/Lidar/A3
[11] https://www.intelrealsense.com/depth-camera-d435/
[12] https://www.intelrealsense.com/tracking-camera-t265/

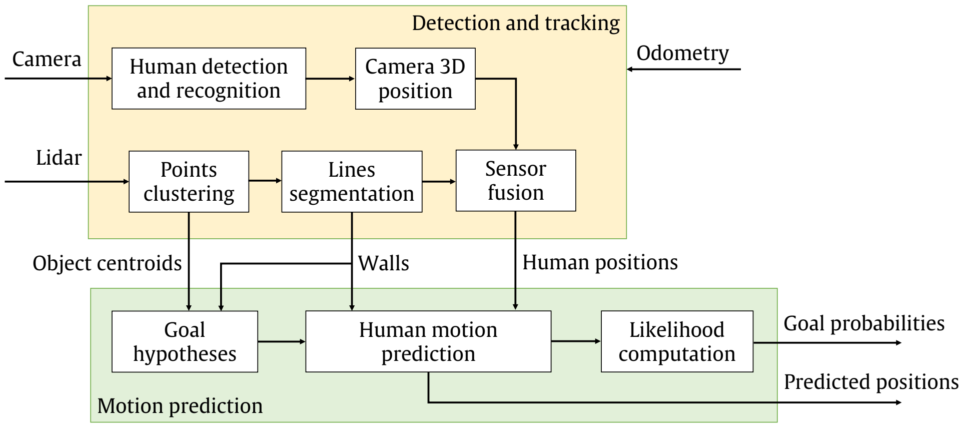


Figure 4.14: Scheme of the robot framework. The detection and tracking module is described in Section 6.3.3, while the motion prediction module includes the pipeline of Section 4.4.

the camera data and the LIDAR readings in order to identify the 2D position of the human target even in presence of occlusions and/or of other moving entities. The origin of the fixed reference frame $\langle F \rangle$ can be freely defined in the available map or w.r.t. the robot's initial position. The subsequent positions of the robot $\mathbf{p}_r = [x_r, y_r, \phi_r]^T$ are then retrieved with a standard localisation algorithm (Nazemzadeh et al., 2017). The assumed sensing configuration allows us to bring all the measurements taken by the robot in its relative reference frame $\langle R \rangle$ back to the fixed frame $\langle F \rangle$ by subtracting the relative robot motion from the measures. The pedestrian's 2D positions are retrieved by the robot with a sensor fusion approach (see Section 6.3.3). As regards the detection of obstacles walls and fixed obstacles, we use the sequence of measurements from the LIDAR and apply the following steps to reconstruct the information of interest: 1. we group the points obtained from the LIDAR into clusters, 2. we filter out spurious points through the Ramer-Douglas–Peucker algorithm (Douglas and Peucker, 1973), 3. we interpolate the remaining points and generate a 2D evaluation of walls and obstacles boundaries. We observe that this information is key to the computation of the repulsive forces in the SFM-based prediction.

The *motion prediction* module implements the whole set of algorithmic solutions described in Section 4.4 to predict the motion of the human target: generation of the goal hypotheses, neural network based implementation of the SFM and likelihood computation. After the network predictions, the future human positions are obtained with the double integration of the SFM model (4.7), with a discretization time δ_t, that is imposed by the sensor with the lowest sampling frequency and is comparable with the sampling time used in simulation. The training of the neural network was performed offline and the weights produced by the training were transferred into the C++ implementation. The network inputs (human and wall positions) are the same described in Section 4.4.3, which are a commonplace in mobile robotics applications. Hence, the embedding of the SFM formulation in the neural network design allows us to seamlessly perform predictions both in real and simulated environments.

4.7.2 Experiments with mobile robot

The experimental evaluation of *Net2* with the multi-goal strategy is carried out through actual experiments in our department at the University of Trento. In particular, we let our robot moves and encounters people in a portion of a hallway with multiple exits. In this setting, the robot navigates in the corridors while it detects, tracks, and predicts the motion of a nearby human. The robot has no prior knowledge on the environment, nor the points of interest of the human.

Below we present qualitative results of the prediction module. All the robot collected measurements, i.e. the robot path, the human trajectory and the obstacle data, are reported in the global reference frame. As shown in Figure 4.15, while the robot detects a person which is moving along the corridor in the same robot direction, the initial goal guesses are quite equiprobable (recall that the probability is measured by μ_k^t in (4.20)), but already catch the trend of a forward motion (Figure 4.15-a with reference to the right colour code describing probabilities), while, after some additional samples of the human positions are collected, the confidence of the goal ahead is increased (Figure 4.15-b) and propagated on new goals after the update described in Section 4.5 in (4.22) (Figure 4.15-c). In the bottom row of Figure 4.15, instead, we show how a bended human path is predicted, so that the initial guess on the goal ahead (Figure 4.15-d) is correctly transferred to the goal on the side (Figure 4.15-e) in light of the observed trajectory and the corresponding prediction of a lateral motion. Finally, the best confidence switches to the stop-goal as soon as the person decides to remain on the spot (Figure 4.15-f). These experimental results are, to the best of our knowledge, the first actual evidence of a reliable human motion prediction carried out in real-time by a moving service robot in a natural environment that deals with random human beings in an unknown scenario. We want to remark here that this is actually possible if the knowledge gained by the neural network is abstracted by the wired model adopted.

4.8 Comments

In this chapter, we have shown a novel technique for predicting human motion embedding the environmental contextual information. Our idea is to bridge the gap between model-based and learning-based approaches in order to retain the advantages of both. Indeed, the proposed strategy is based on the combination of a neural network with a famous physics inspired dynamic model, the SFM. In the combination, each of the two approaches emphasises its own strengths and compensates for the weakness of the other. Specifically, the SFM brings a structure to the neural network, reducing its complexity and the number of samples needed for the training. Furthermore, neural network predictions become explainable and physically interpretable. On the other hand, the neural network expresses its full power in terms of flexibility, and of its ability to learn the complex parameter set of the SFM, which would be very difficult to estimate in real-time by conventional means for the strong non linearities of the model. Our simulations and experiments reveal the full potential of the marriage between the two worlds of physics inspired models and neural networks.

Some important points remain open and can attract our efforts in the near future. First, the neural network designed can be improved by embedding different models which are potentially more realistic than the SFM, for example the the HSFM (Farina et al., 2017b) or the PHSFM (Antonucci and Fontanelli, 2018). Moreover, we want to test a new architecture end-to-end where the inputs are the observed trajectories and the outputs are the predicted ones, so as not to depend on ground truth data and to evaluate the network training on datasets of collected pedestrians data. Finally, we plan to extend the input of the model to include interaction with other humans and to account for unspoken signs (e.g., pose, eye gaze, facial expression). We refer to the next chapter for an example of a motion planning algorithm that is designed to make the best use of the proposed structured network.

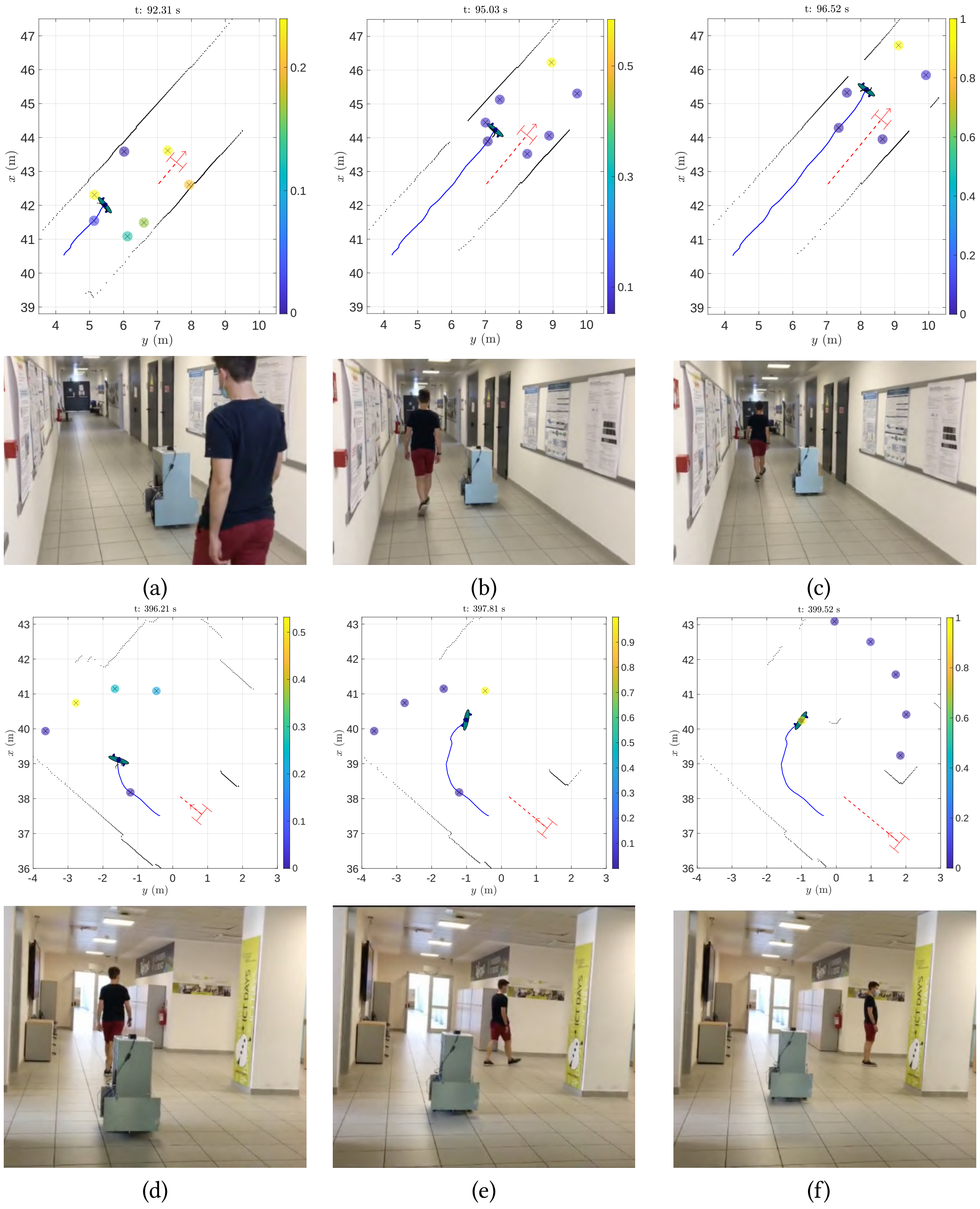


Figure 4.15: Different snapshots of the experimental results when a moving person (with blue solid trajectory) is tracked by a robot (red dashed trajectory). Both, a human showing a forward motion (a-c) and a turning and stopping motion (d-f) are reported. The black dots are the LIDAR data collected by the robot, while the coloured circles represent the predicted human goals at the snapshot time, each with a confidence μ_k^t depicted with the colour mapping on the side.

Chapter 5

Navigation in human-shared environments

In our definition of human-shared environments, we include all those cases in which robots have to navigate in partial or complete autonomy, finding themselves in a certain proximity to people sharing a common space with them, and coming across various kinds of obstacles. Possible contexts are both industrial workspaces, such as warehouse hallways with racks, and general non-industrial applications (co-bots, delivery robots, agricultural robots, etc.). Regardless of the type of application, usually the planning algorithms of the robots focus on optimal goal-seeking and/or efficient collision avoidance, but they do not explicitly consider the social aspect of the interaction with humans.

Unless the specific application requires collaboration between humans and robots, the main requirement we want to obtain is that robots are of minimal disturbance to people. The collision avoidance, in order to be effective and interpretable, must take place in an anticipatory and smooth way, so that it is not misunderstood as "improvised" or not controlled by the robot, which could make people uncomfortable. Secondly, if the robot moves in a reactive and non-anticipatory manner, or is not cohesive with the planned path, undesired phenomena such as the "freezing robot problem" (Trautman and Krause, 2010) can occur, with increasingly pronounced effects as the number of people and robots present increases. We want to address the problem of planning and controlling the navigation of a group of robots, avoiding the people present, coordinating their movements, and minimizing stops while keeping safety as an essential constraint. Specifically, our solution is structured in three different nested layers of increasing level of abstraction and information used, organized following the principle of the cognitive framework inspired by the proxemics theory as mentioned in the introduction of this thesis. Given the particular modularity of the framework, we will present its overall architecture and deepen the description of the modules personally developed. Then, we will show the experimental results of our navigation strategy implemented on three mobile robots with humans moving in the neighbourhoods.

5.1 Overview

The fundamental talent for mobile robots is their ability to navigate their way through (un)known environments in complete autonomy. However, we envisage a scenario in which robots will be so complementary to humans that future robotic applications will have to meet demands with a higher level of expectations. Specifically, we can identify five classes of requirements (Figure 5.2).

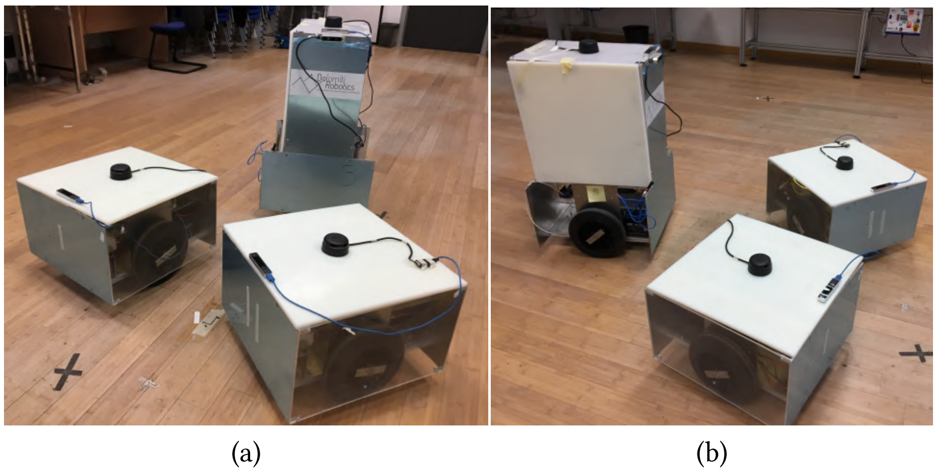


Figure 5.1: The three mobile robots used for experimental campaign.

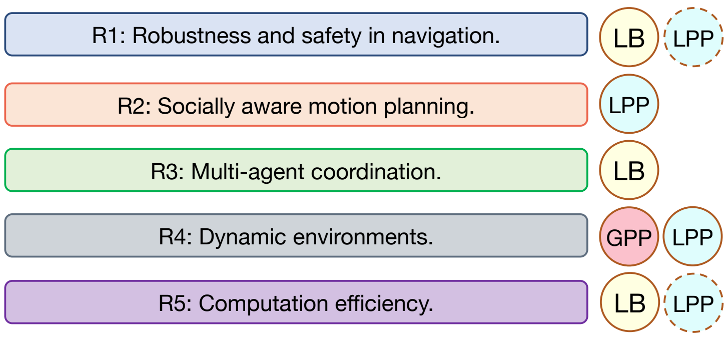

R1: Robustness and safety in navigation.
LB
LPP
R2: Socially aware motion planning.
LPP
R3: Multi-agent coordination.
LB
R4: Dynamic environments.
GPP
LPP
R5: Computation efficiency.
LB
LPP

Figure 5.2: Requirements for multi-robot navigation in human-shared environment and framework layers (GPP, LPP, LB) where the requirements are satisfied (full circle) or partially satisfied (dashed circle).

Robustness and safety in navigation (R1). An essential requirement for the navigation algorithm is to ensure safety for the humans and the absence of damages to the robots and to other assets. Since we cannot impose a desired velocity on people, collisions with the robot cannot be ruled out for arbitrary velocities whatever is the algorithm adopted. Hence, by safe navigation we mean:

- collision will be avoided with static obstacles and other robots;
- no collision will be generated from the robot control choices when a human is encountered.

That is, we want to guarantee safe navigation as long as the humans do not induce on purpose a collision by running into the robot, which is a reasonable condition in the considered scenarios.
Socially aware motion planning (R2). A robot traveling nearby humans can be perceived as an unfamiliar and intimidating presence. This feeling is exacerbated if the robot follows weird motion patterns (e.g., sharp turns) or if the trajectory penetrates into the humans' private space (Section 2.2). If the robot follows instead smooth trajectories and anticipates the human, its presence could become more acceptable. We will accomplish this task by exploiting the motion predictions of each human in the scene following the results of Chapter 4.
Multi-agent coordination (R3). As the motion planning complexity increases according to the number of robots involved, we aim for an appropriate coordination protocol that is scalable and computationally efficient.

Dynamic environments (R4). For the evaluation reported in the experimental section, we will follow the assumption that the robots plan their navigation using prior knowledge on the environment. This is a reasonable condition since we can imagine tasks where for example CAD maps (for indoor applications) or street maps (for outdoor applications) are given. However, our foreseen planning framework should efficiently react to unexpected conditions as soon as they are detected by on-board sensors, and guarantee the robot's convergence to its goal.
Computation efficiency (R5). A computational efficiency real–time controller must be flanked with an high speed frequent data collection from the environment. This requirement goes hand-in-hand with the use of lean hardware motivated by cost and energy consumption considerations, and benefits from the evaluations made in Chapter 3.

Each of these requirements is difficult to address in its own right, and the interweaving between different requirements introduces additional complexity. As no silver bullet method exists to cope with the difficulties of the problem outlined above (see Section 5.2), the most obvious solution is to adopt a combination of methods. Our solution is a hierarchical architecture (depicted in Figure 5.4), in which each layer receives information and takes decisions that are propagated to the layers below.

The uppermost layer is a *Global Path Planner* (GPP), which works over the entire mission space exploiting an abstract knowledge of the environment. Moving down in the hierarchy, we introduce new decision layers, which use more detailed information on the environment but take decisions with a narrower spatio-temporal horizon. In particular, the second layer exploits the on-board sensors to detect the nearby objects and to identify and track each human in the sensing range. This information is passed together with the planned path into the *Local Path Planner* (LPP), which accounts for the presence of obstacles and embeds a prediction-based component to determine the future evolution of the position of the humans. The lowermost layer consists of a distributed *Lloyd-based controller* (LB) that intervenes on the planned path by reacting to the information provided by low-range sensors. This layer also takes care of the multi-agent coordination exploiting the communication network.

Our proposed architecture allows each layer to have a certain degree of independence, meaning that the properties enforced at one layer are not invalidated by the layers below, and the overall process is not stalled by inconsistent decisions taken from different levels. Remarkably, the interaction between the *global path planner*, the *local path planner* and the *Lloyd-based controller* satisfies all the requirements R1–R5, as schematized in Figure 5.2. Indeed, we are able to obtain: high levels of safety and robustness thanks to the high frequency corrections implemented by LB (R1); socially aware trajectories thanks to the LPP, which accounts for the prediction of human motions (R2); successful multi-agent coordination handled by LB (R3); an effective management of dynamic scenarios and progress towards the final destination thanks to the integration of the global planning GPP and the local planning LPP (R4); computational efficiency since the intensive task of multi-agent coordination is moved from planning layer to the innermost level of the framework, and the LPP and LB perform at high frequency (R5).

After a comparison with the related literature on robot navigation in human-shared environments similar to our approach in Section 5.2, we will describe the complete framework in Section 5.3. Notice that the GPP and the LB layers are not specific contributions of this thesis. Since they are part and parcel of the framework architecture, we will briefly report their functioning, while we will deepen the description of the *local path planner*. Then, Section 5.4 will presents the experimental results with real multiple robots and people, followed by final considerations discussed in Section 5.5.

5.2 Related work

Despite the intensive research activity of the last few years, it is still missing a comprehensive navigation system that satisfies all the requirements stated in Section 5.1. A more extensive review of the related work on socially aware robot navigation is presented in Section 2.3. Here, we report the most relevant contributions to our use case and comment on their relationship to our solution. In a recent survey, Cheng et al., 2018 propose a taxonomy of the papers that populated the recent literature on robot navigation in human-shared environments. They identify three families of methods, which are useful to refer to also in this context: *reactive methods*, *predictive planners*, and *learning-based methods*.

Reactive methods Reactive methods show their potential in managing the presence of unexpected static and dynamic obstacles. The most popular reactive methods are based on Velocity Obstacles (VO) (Fiorini and Shiller, 1998; Alonso-Mora et al., 2013; Boldrer et al., 2020b), and force fields (Olfati-Saber, 2006; Boldrer et al., 2020c). Both are suitable for use in multi-agent navigations. Claes and Tuyls, 2018 propose a method based on the dynamic window approach (DWA) (Fox et al., 1997) and VO, which appears as an effective solution to support the navigation even in the absence of external sensors (e.g. tracking cameras, UHF-RFID tags).

Reactive methods based on VO or on force fields share a good degree of computational efficiency (R5) and, in some cases, can be adapted to operate in a multi-agent framework (R3). Moreover, these methods are designed to react quickly and to preserve safety also in a dynamic environment (R1). However, generally speaking, reactive solutions are not necessarily effective in dealing with humans. For instance, VO-based methods require a reliable estimate of the velocity of the obstacle. When this condition is not satisfied, it could lead to collisions (failing R1). Furthermore, the decision mechanism of reactive methods is “short-sighted” giving rise to two possible problems: *i.* deadlocking configurations may occur in an uncertain environment (failing R4), and *ii.* the generated paths may encroach into the humans’ private space (failing R2).

Predictive planners To fruitfully take into account the presence of humans is essentially necessary to rely on *predictive methods*, i.e., algorithms that exploit model-based predictions of the human motion. A number of different methods have been surveyed and compared by Khambhaita and Alami, 2017. Bevilacqua et al., 2018b propose a probabilistic reactive planner for the avoidance of a single pedestrian that moves according to the Headed Social Force Model (HSFM) (Farina et al., 2017b), while Ziebart et al., 2009 present a navigation strategy that maximises the robot efficiency in reaching the goal and at the same time minimises the robot-human hindrance relying on a reliable prediction model for the human motion. Bajcsy et al., 2019 propose a safe navigation method for multi-drones in human-shared environments. A maximum-entropy model was employed to predict the human motion, while a sequential trajectory planning managed the collisions with other robots. Mavrogiannis et al., 2018; Mavrogiannis et al., 2019 design a social momentum planning framework for socially aware multi-agent navigation, reporting good performance evidence in crowded situations. Similarly, Knepper and Rus, 2012 proposed a model-predictive, sampling-based planner.

The application of predictive models allows to generate human-friendly trajectories. As long as the prediction is supported by well-fitted models and precise measurements, the result is accurate and the planned trajectory is safe (R1), smooth, and socially aware (R2). Since these methods use relatively long planning horizons, typically deadlock situations are unlikely to happen (R4). However, predictive planners are computationally intensive and cannot be executed with high frequencies (failing R5, as depicted in Figure 5.2). This also depends on the fact that a minimum is required to collect enough input data to make a prediction. As a result, if the measurements are affected by noise and the resulting prediction is imprecise, the humans' safety trajectories cannot be robustly guaranteed (R1). A final point is that predictive planners usually do not take into explicit consideration the presence of other robots. Some adaptations are possible, e.g., assigning fixed priorities to the agent to access shared areas and assuming a massive exchange of information (Bajcsy et al., 2019), but they do not exploit the full range of options in robot collaboration (R3).

Learning-based methods A comprehensive review of recent developments in machine learning approaches for robot navigation is reported in Section 2.3. Some contributions have evident connection with the framework presented in this chapter. Back in 2012, Luber et al., 2012 used learning approaches to synthesise a socially aware navigation strategy. Fan et al., 2020 proposed a multi-robot collision avoidance method with Deep Reinforcement Learning, showing the effectiveness of the algorithm through simulations and experiments, while Han et al., 2020 present very promising results from the experimental point of view. Other recent works used learning-based techniques for single robot navigation in human-shared environments (Güldenring et al., 2020; Liu et al., 2020; Nishimura and Yonetani, 2020; Kretzschmar et al., 2016; Kim and Pineau, 2016).

Some issues that hinder the application of learning-based methods to navigation: *i.* in order to obtain a sufficient level of safety (R1) and social awareness (R2), it is possible to train the robots algorithm by using a massive amount of examples derived from a specific operational scenario, which, however, could easily drive the system to overfitting with the inability to generalise its behaviours to dynamic environments (failing R4); *ii.* even if the system is well trained it is very hard to provide any type of safety guarantees (failing R1) or to explain the failures; *iii.* the management of multi-agent scenarios (R3) introduces an additional complexity: the robot needs to learn how to avoid obstacles and how to move in a socially aware fashion, but also how to coordinate its behaviours with other agents. Finally, the computational effort (R5) is acceptable for navigation purposes.

5.3 Framework architecture

We consider a number N of autonomous agents. Albeit the results reported in this chapter are of general validity, we will make explicit reference to the robots available in our laboratories, i.e., unicycle-like vehicles with kinematic model

$$\begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix} = \begin{bmatrix} v_i \cos\theta_i \\ v_i \sin\theta_i \\ \omega_i \end{bmatrix}, \tag{5.1}$$

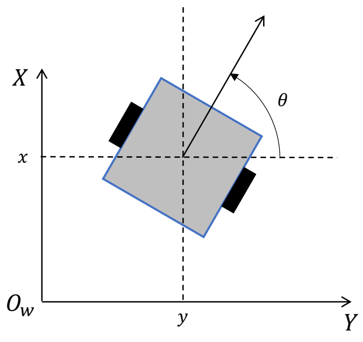


Figure 5.3: Robot representation and adopted reference frame.

where $\mathbf{p}_i(t) = [x_i, y_i]^T$ is the mid point of the robot rear axle at time t and θ_i is the yaw angle (see Figure 5.3). The linear velocity v_i and the angular velocity ω_i are the control inputs.

Given the initial positions $\mathcal{S} = \{\mathbf{p}_1(0), \mathbf{p}_2(0), \ldots, \mathbf{p}_N(0)\}$, we want to drive each robot into its final goal position $\mathcal{E} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N\}$. The trajectories are required to satisfy the following properties:

1. Safety (R1): let $\mathbf{q}$ be any point belonging to an obstacle or a human. Then if $\|\mathbf{q} - \mathbf{p}_i(t)\| = \delta_i$, therefore the obstacle is at a minimum distance δ_i, we must ensure that the robot will not approach it further, by trying to increase or at least maintain that distance, $\langle \mathbf{p}_i - \mathbf{q}, v_i \rangle \leq 0$.

2. Socially aware paths (R2): the compliance with social rules is multidimensional. Let $\mathbf{p}_h(t)$ represent the position of a generic human. A first parameter is the minimum distance between robot and humans, $d_{i,h}^{\text{min}} = \min_t \|\mathbf{p}_i(t) - \mathbf{p}_h(t)\|$. The second parameter is the interaction time between them: let $T_{i,h}$ be the set of times such that $\forall t \in T_{i,h}$, $\|\mathbf{p}_i(t) - \mathbf{p}_h(t)\| \leq \delta_{\text{th}}$ where δ_{th} is a threshold value. We define as the interaction time $t_{i,h}$, the sum of the Lebesgue measures of all the intervals contained in $T_{i,h}$. Strictly speaking, we intend as a socially aware trajectory the one that keeps larger values of $d_{i,h}^{\text{min}}$ and smaller values of $t_{i,h}$ with respect to a non-socially aware trajectory, i.e., the robots have to make pedestrians deviate as little as possible from their desired path. In addition, the trajectory should be smooth and predictable: this can be measured through the average robot trajectory curvature $\bar{\kappa}_i$, which will be detailed in Section 5.3.2. Our definition of social awareness does not clearly cover all the aspects related to the social awareness of the robot navigation. However, it can be regarded as a first step in the right direction as it captures some essential features that make a trajectory socially aware: 1. following smooth trajectories (which are more predictable by the humans with respect to sudden manoeuvres), 2. staying clear from the humans intimate space (hence, the person is not intimidated by the robot and feels safer), 3. reducing the interaction time (so as to evaluate the effectiveness of the human motion predictions).

3. Efficiency and convergence (R4): the path length l_i and the mission time t_i should be both as small as possible.

4. Scalable multi-agent coordination (R3+R5): we aim at a coordination protocol between multiple agents such that: *i.* the safety property (specified in point 1)) is satisfied; *ii.* the robots avoid collisions among them; *iii.* efficiency and convergence detailed in point 3) are significantly better than those that would be obtained by accounting for individual non-cooperating behaviours.

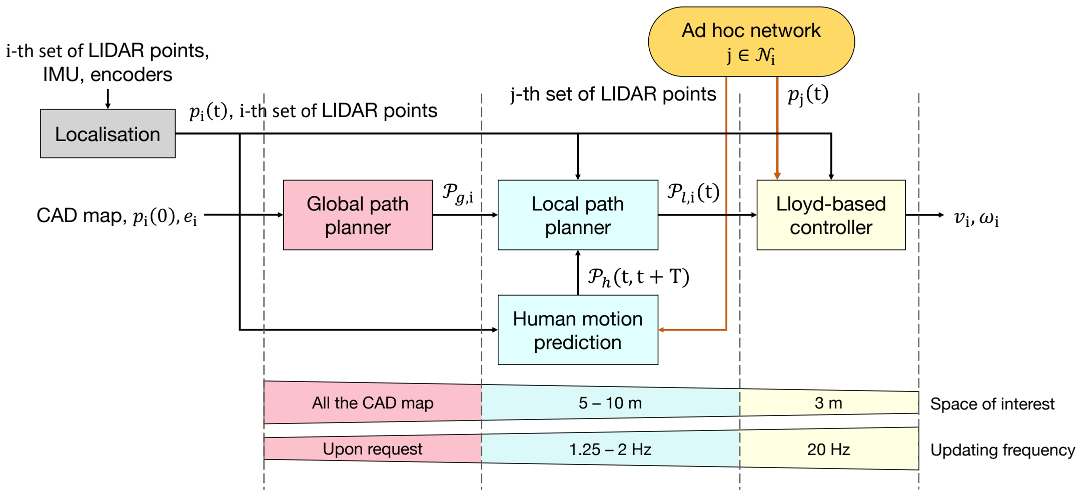


Figure 5.4: Schematic representation of the framework for the i–th agent.

Moreover, the coordination protocol should be scalable in the number of robots and it should be able to deal with high updating frequencies (i.e., the robots have to be able to compute updated control inputs in a limited amount of time).

Our navigation strategy is based on the combination of three nested layers: 1. *global path planner* (GPP), 2. *local path planner* (LPP), and 3. *Lloyd-based approach* (LB). Figure 5.4 depicts the overall solution approach for the i-th agent. The GPP generates the global path $\mathcal{P}_{g,i}$, given the CAD map of the mission space, the starting robot position $\mathbf{p}_i(0)$ and its goal position $\mathbf{e}_i$. The global path $\mathcal{P}_{g,i}$ is then passed to the LPP which plans for the local path $\mathcal{P}_{l,i}(t)$, which is synthesised every T_p seconds and requires as input the vehicle position, i.e., the output of the localisation module $\mathbf{p}_i(t)$, and the humans predicted paths $\mathcal{P}_h(t, t + n_f\delta_t)$ within n_f time steps δ_t (which is dictated by the sampling time of the available sensors). The localisation module takes as inputs the i–th sensor readings that are LIDAR sensed points, the encoders measurements, and the IMU data. The human motion prediction block uses the LIDAR sensed points (see Figure 5.4) and it fuses the i–th and all the $j \in \mathcal{N}_i$ agents LIDAR points, where $\mathcal{N}_i$ indicates the set of neighbours of the i–th agent. Once $\mathcal{P}_{l,i}(t)$ is computed, it is passed to the Lloyd-based controller along with the agent position $\mathbf{p}_i(t)$, the neighbours positions $\mathbf{p}_j(t)$, $j \in \mathcal{N}_i$, and the i-th LIDAR points. Every T_c seconds, the control inputs v_i and ω_i are computed. The ad hoc network module depicted in Figure 5.4 implements neighbouring inter-agent communication to share positions and environmental data.

As it is depicted in Figure 5.4, the portion of the environment used by each module decreases passing through GPP, LPP and LB, hence the updating frequency (inversely proportional to the computational burden) grows. Finally, since each robot is equipped with just a LIDAR sensor, the navigation task becomes challenging since the sensor readings are noisy and obstructions may lead to unreliable detections. To make this task more robust, we allow the robots to exchange information on the humans' positions and velocities and fuse these information by using the well known DeGroot model (DeGroot, 1974), thus obtaining a consistent human motion prediction. This is important mainly for two reasons: *i.* the plan generated by the robots that share the information of humans' positions and velocities permits the robots to reach consensus on the future human positions, hence to produce consistent, shared manoeuvres *ii.* it permits to share the human prediction also to the agents that do not directly perceive the pedestrian presence.

5.3.1 Global path planner

The role of the *global path planner* (GPP) is to provide a feasible path towards the goal for each robot. This component is executed at the beginning of the mission or upon request. The component relies on a global map that should be available at the beginning of the mission (e.g., derived from the CAD drawings). Our solution for the global path planning module is based on the Fast Marching Square algorithm (FM^2), which is an extension of the classical Fast Marching Method (FMM). Even if this method is not a specific contribution of this thesis, we report below a few explanatory notes for the sake of completeness. FMM is an efficient numerical algorithm for solving Eikonal equation (Osher and Sethian, 1988; Valero-Gomez et al., 2013), which describes how a wavefront propagates. By using FMM we are able to determine a potential of "slowness" for the robot, i.e. a potential map that gives the robot's admissible velocity at each position. FM^2 is an extension of FMM, introduced by Garrido et al., 2009. The algorithm operates in two phases. During the first phase, FMM is applied, propagating a different wave from each of the obstacle cells. As a result of this phase, it is possible to generate a cost map that accounts for the slowness for the robot generated by each obstacle through the wavefront. During the second phase, FMM is applied once again propagating backwards a wave from the goal. The propagation is affected by the cost map generated during the first phase. The result is the shortest trajectory (in time) which links in a smooth way the start and the goal cells, avoiding obstacles.

Even though our solution is agnostic to the choice of the global planner, we found good reasons for the choice of FM^2. In the first place, the algorithm is computationally efficient: even on large binary cost maps (e.g., the size of a university department), the plan is produced in a few hundreds of milliseconds. Secondly, it produces trajectories that are smoother with respect to other planning algorithms based on discrete cost maps, and it allows us to penalise paths passing very close to an obstacle, choosing alternative routes whenever possible; Finally, it ensures convergence to the global minimum, identifying the unique solution that connect the start cell to the goal cell. In Figure 5.5 we show an example of the output path $\mathcal{P}_{g,i}$ produced by the FM^2 algorithm, for a single agent.

5.3.2 Local path planner

The local path planner is needed in order to account for obstacles (both static and dynamic) that are not present in the map used by the GPP. The LPP consists of two interacting modules: the *local position-based path planner* (LPos) and the *local prediction-based path planner* (LPred). In simple terms, the LPos has the responsibility to avoid static obstacles identifying a sufficiently large safe corridor (the "navigable corridor"), whilst the LPred uses predictive models for human motion in order to safely avoid the humans in the scene.

The local position-based path planner

The GPP produces a global path consisting of a sequence of segments. Every T_p, the local position-based path planner comes to play. It computes the local goal position, determined by selecting a point along the global path $\mathcal{P}_{g,i}$, which is at a look ahead distance L_i from the current robot position $p_i(t)$. This

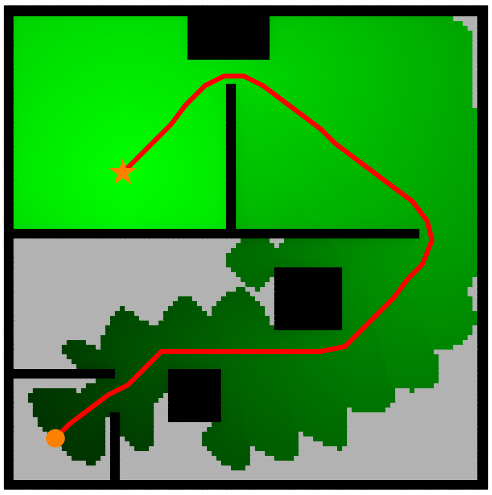


Figure 5.5: Example of global path $\mathcal{P}_{g,i}$ planned by the FM^2 algorithm. Black cells represent obstacles, grey cells represent cells that have not been explored, while green cells having different shade show the times of arrival from the goal (orange star). The resulting path from the starting position (orange circle) to the goal is shown in red.

module uses both the map information and the onboard sensor readings to identify a feasible corridor connecting the robot's current position to the next local goal. The corridor defines a portion of the environment where it is possible to define a collision-free path. Once a corridor is available, a safe path should be synthesised.

The computation of the feasible corridor is performed in two steps. In the first step, "raw" path is generated by employing FM^2 (the same algorithm used to generate the global path, and described in Section 5.3.1). In the second step, a number of way-lines are grown around the raw path. The way-lines are generated by sampling waypoints along the raw path (more densely in close proximity to changes of directions, to provide more granularity and more degrees of freedom during the following interpolation phase), and casting rays from each of these waypoints, along both directions, perpendicular to the raw path until the maximum corridor width has been reached, or an obstacle is hit (as an example, see the thin blue lines in Figure 5.6).

Once the navigation corridor has been constructed, each way-line is sampled extracting a number of discrete points w_p. For each point we select a finite number of fixed orientations w_o. The next step is then to find a sequence of motion primitives, each one crossing a way-line with an assigned orientation, such that continuity and inclination is preserved across each intersection and that the total final cost is minimised. Let us identify with $\mathcal{X}_i$ the sequence of cells of the cost map crossed by the path $\mathcal{P}_{l,i}(t)$. The cost function H_i adopted to determine the optimal trajectory is based on a convex combination of the travel time (that depends on the length of the path and on the proximity to obstacles) and the integral of the path curvature squared (to drive the search towards smooth paths and to avoid sharp turns unless necessary), i.e.

$$H_i = \lambda \sum_{\chi \in \mathcal{X}_i} \frac{1}{\alpha_i(\chi)} + (1-\lambda) \int_0^{l_i} \kappa_i(s)^2 ds \tag{5.2}$$

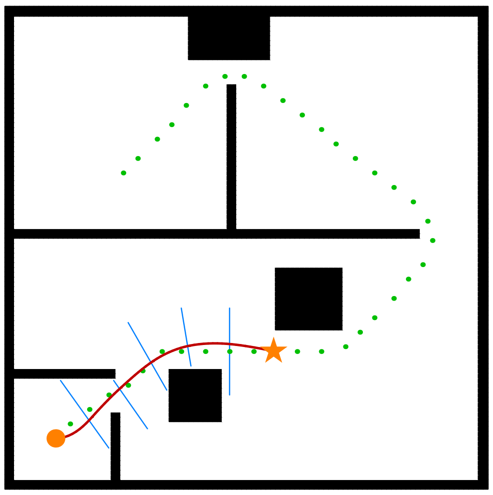


Figure 5.6: Example of local path $\mathcal{P}_{l,i}(t)$ by using smooth spline interpolation. The robot position is indicated with orange circle. Black cells represent obstacles, as green points we have the output coming from the global path planner $\mathcal{P}_{g,i}$. The resulting path from the robot position (orange circle) to the goal is shown in red, the way lines are depicted in blue.

where $\alpha_i(\chi)$ is the "slowness", computed using the FMM, associated to the cell χ with size dx, $\lambda \in [0, 1]$ is a weighting parameter and κ_i is the path curvature defined as

$$\kappa_i(s) = \frac{\|\dot{x}_p(s)\ddot{y}_p(s) - \dot{y}_p(s)\ddot{x}_p(s)\|}{(\dot{x}_p(s)^2 + \dot{y}_p(s)^2)^{\frac{3}{2}}}$$

where $[x_p(s), y_p(s)]^T \in \mathcal{P}_{l,i}(t), \forall s \in [0, l_i]$, and where s is the curvilinear abscissa of the path.

We selected clothoids as motion primitives to connect pairs of consecutive waypoints, as they present several advantages (Bertolazzi and Frego, 2015), that are: the direct parametrisation with arc-length; the linearity of the curvature that enhance the path smoothness; the simple analytical analysis that allows us to manipulate these curves with quite efficient libraries (Bertolazzi et al., 2018). The search for the optimal solution comes from an Iterative Dynamic Programming approach. We took inspiration from the solution developed by Frego et al., 2020 to solve the Multipoint Markov-Dubins problem. The optimal solution can thus be determined using the Bellman's "principle of optimality" testing backward each point and each orientation. The algorithm has quadratic complexity in the product of the number of sampled point w_p and of sampled orientations w_o for each way-line. The complexity can be reduced by an iterative approach. At the first iteration, the algorithm is executed on a relatively low number of sampled points. Once a first suboptimal solution is identified, the next iteration extracts additional samples in a neighbourhood of the intersection of the solution with the way-lines and the algorithm is executed again to refine the solution. We can then execute a number σ of additional iterations, each time considering samples on an increasingly narrow search space around the current solution. In Figure 5.6 we depict an example of the optimal path $\mathcal{P}_{l,i}(t)$ computed by the i–th agent through LPos.

The local prediction-based path planner

As discussed above, the LPred module improves the path produced by the LPos by integrating the predicted motion of each human for n_f samples ahead using the past n_p measurements. The strategy

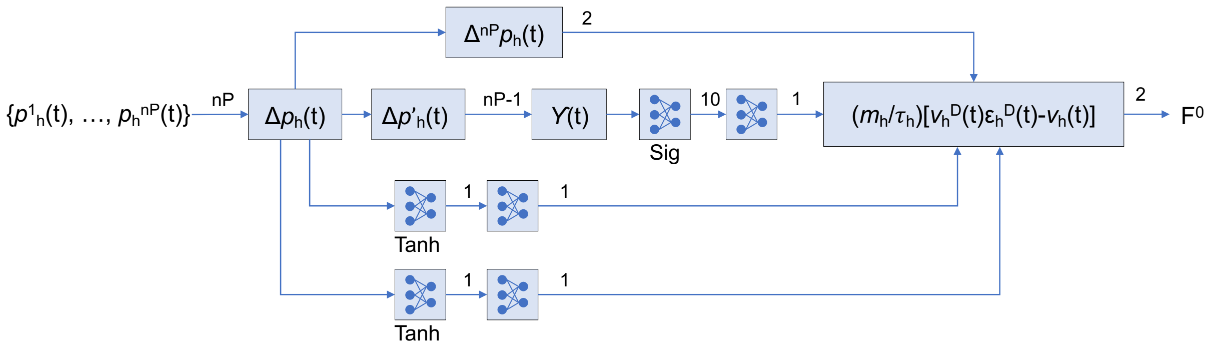


Figure 5.7: Scheme of the proposed structured neural network used in the *LPred*. The drawn blue squares represent the fully connected layers, with their activation function (if any) and their output size.

to predict the human motion take inspiration from the structured neural network model presented in Section 4.4.

Indeed, in our model the h-th human, with mass m_h, moves towards his/her target at a certain desired walking speed with magnitude ν_h^D and following a second order dynamic. We identify his/her measured position as $\mathbf{p}_h(t) = [x_h, y_h]^T$. Hence, the human moves driven by an attractive force $\mathbf{F}^o = \left[f_x^o, f_y^o\right]^T$ defined as

$$\mathbf{F}^o = \frac{m_h}{\tau_h}\left[\nu_h^D(t)\boldsymbol{\varepsilon}_h^D(t) - \mathbf{f}_h(t)\right], \tag{5.3}$$

where the characteristic time $\tau_h > 0$ parameter determines the rate of change of the velocity vector and $\boldsymbol{\varepsilon}_h^D$ is the unit vector pointing towards the goal, i.e., the desired heading. In compliance with the interaction that takes place between the robot and the human expected for the requirement R2, we make here customary assumption that the motion of the human is not perturbed by the walls or static obstacles and by the robots. This assumption has been made to generalise the human motion predictions, avoiding any environmental bias related to the training scenario. What is more, considering the human trajectory agnostic to the robot presence is a conservative choice, as it predicts for the worst-case condition of non-cooperative humans. As a result, the level of safety achieved by the solution is higher fulfilment of the requirement R1. We observe that our layered scheme allows as to adopt a simplified human model, as obstacles and unforeseen pedestrian movements can be efficiently handled by the lower layers of our framework, as detailed in Section 5.4.

Following the model in Section 4.4, we use a structured neural network to predict the human motions, where the neurons are organised according to the structure of the SFM model in (5.3). Specifically, the neural network predicts the force $\mathbf{F}^o$, given the n_P more recent measurements of the past human positions from the current time t (see the scheme reported in Figure 5.7). We consider the vector of the coordinates $\Delta\mathbf{p}_h(t) \in \mathbb{R}^{2\times n_P}$ shifted with respect to the first sample of the window (i.e., relative displacements), so that

$$\begin{aligned}\Delta\mathbf{p}_h(t) = &\left[\mathbf{p}_h(t-(n_P-1)\delta_t), \mathbf{p}_h(t-(n_P-2)\delta_t), \cdots, \mathbf{p}_h(t)\right] + \\ &-\mathbf{p}_h(t-(n_P-1)\delta_t)\mathbf{1}_{n_P}^T,\end{aligned} \tag{5.4}$$

where $\mathbf{1}_{n_P}$ is a column vector with n_P ones, to reduce spatial bias in the network training. The neural network inputs are processed by two hidden layers with no biases and with only one fully connected output neuron. The latter is devoted to learn the instantaneous velocity $\mathbf{v}_h = \dot{\mathbf{p}}_h = [v_x, v_y]^T$ and is followed by $\tanh(\cdot)$ activation functions. Another neuron without bias follows each layer to recover the normalisation of the estimates to their actual range. To estimate the components of the normalised goal-directed unit vector $\boldsymbol{\varepsilon}_h^D$, we use the most recent relative motion measurement $\Delta^{n_P} p_h(t)$, i.e., the n_P-th element of $\Delta \mathbf{p}_h(t)$ in (5.4). Finally, the vector of normalised velocities magnitudes $\Upsilon(t) \in \mathbb{R}^{n_P - 1}$ derived as

$$\Upsilon(t) = \left[\|\Delta^1 \mathbf{p}_h'(t)\|, \dots, \|\Delta^{n_P-1} \mathbf{p}_h'(t)\|\right]^T,$$

where

$$\begin{aligned}\Delta \mathbf{p}_h'(t) = \left[\Delta^2 \mathbf{p}_h(t), \dots, \Delta^{n_P} \mathbf{p}_h(t)\right]^T + \\ - \left[0, \dots, \Delta^{n_P-1} \mathbf{p}_h(t)\right]^T,\end{aligned}$$

is used to estimate the desired speed ν_h^D, through two fully connected neurons with a sigmoid activation function necessary to keep a positive sign for the ν_h^D estimate. All the estimates are passed to a Lambda layer where they are combined and weighted according to the m_h and τ_h parameters in (5.3), thus generating the final output that is the estimate of $\mathbf{F}^o$.

The future human positions are then propagated for n_f samples forward using a constant accelerated motion inspired by the SFM, i.e.

$$\mathbf{x}_h(t) = \begin{bmatrix} \mathbf{p}_h(t) \\ \mathbf{v}_h(t) \end{bmatrix}, \quad \dot{\mathbf{x}}_h(t) = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \mathbf{x}_h(t) + \begin{bmatrix} 0 \\ I \end{bmatrix} \mathbf{F}^o, \tag{5.5}$$

thus generating the human predicted path $\mathcal{P}_h(t, t + n_f \delta_t)$.

The future positions of the humans $\mathcal{P}_h(t, t + n_f \delta_t)$ are then merged with the LPos path modifying the cost function in (5.2) as follows

$$\tilde{H}_i = \lambda \sum_{\chi \in \mathcal{X}_i} \frac{1}{\tilde{\alpha}_i(\chi)} + (1 - \lambda) \int_0^{l_i} \kappa_i(s)^2 ds, \tag{5.6}$$

where the slowness map $\tilde{\alpha}_i(\chi)$ includes also the probability of collision with dynamic obstacles, which is obtained by considering $\mathcal{P}_h(t, t+n_f\delta_t)$ and the future robot positions in the time interval $[t, t+n_f\delta_t]$, computed assuming the actual forward velocity v_i along the planned path $\mathcal{P}_{l,i}(t)$. In Figure 5.8, we show a scenario where the robot detects the human's position, estimates its velocity and predicts its future position. Accordingly, the robot computes the possible points where the collision may occur and plans a feasible route, minimising the cost function (5.6).

5.3.3 Lloyd-based control

As aforementioned, the planning layers only cover requirements R2, R4 and satisfy R5, while a nominal safety (partial cover of R1) is guaranteed in prediction. The full coverage of R1 (safety) and R3 (multi-agent coordination) is delegated to the Llyod-based controller, that we will briefly describe in the in the following. The LB is the innermost layer of our framework, and its mostly based on the formulation presented by Boldrer et al., 2020a for robot navigation.

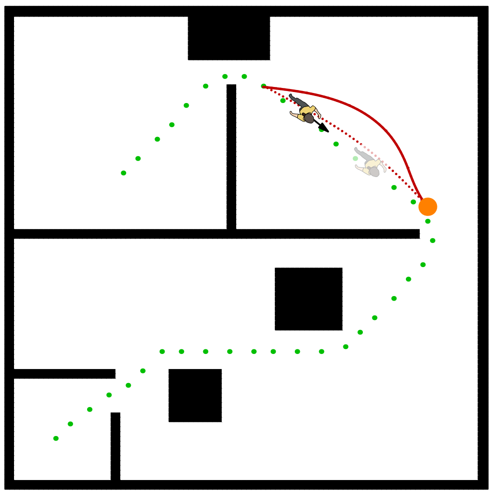


Figure 5.8: Example of local path $\mathcal{P}_{l,i}(t)$ planned by using the human motion prediction information. Black cells represent obstacles, as green points we depict the output coming from the global path planner $\mathcal{P}_{g,i}$. The red dotted line would have been the local path planned without the presence of the humans. The red solid line is the local path $\mathcal{P}_{l,i}(t)$ obtained by using the predictive module i.e. $\mathcal{P}_h(t, t + n_f\delta_t)$. The robot indeed estimate the human velocity (black arrow) and the zone of collision (faded human), then it plan a safe path accordingly.

Let us define the robots' mission space $\mathcal{Q} \subset \mathbb{R}^2$, and let us indicate $\mathbf{q} \in \mathcal{Q}$ as a generic point position. For the i-th robot, the associated Voronoi cell is defined as

$$\mathcal{V}_i(\mathbf{p}) = \{\mathbf{q} \in \mathcal{Q} \mid \|\mathbf{q} - \mathbf{p}_i\| \leq \|\mathbf{q} - \mathbf{p}_j\|\}, \forall j \in \mathcal{N}_i, \tag{5.7}$$

where $\mathbf{p} = [\mathbf{p}_1, \dots, \mathbf{p}_N]^T$ is the vector of all the robots positions and $\mathcal{N}_i$ is the neighbour set of robot i (i.e., the set of robots that can communicate with the i-th robot). Consider the coverage cost function

$$J_{\text{cov}}(\mathbf{p}, \mathcal{V}) = \sum_{i=1}^{N} \int_{\mathcal{V}_i} \|\mathbf{q} - \mathbf{p}_i\|^2 \varphi(\mathbf{q}) d\mathbf{q}, \tag{5.8}$$

where $\varphi : \mathcal{Q} \rightarrow \mathbb{R}_+$ is a weighting function $\forall \mathbf{q} \in \mathcal{Q}$. The LB policy is to make agents follow the gradient descent by imposing

$$\dot{\mathbf{p}}_i = -k_p \left(\mathbf{p}_i - C_{\mathcal{V}_i}\right), \tag{5.9}$$

where $k_p > 0$ is a tuning parameter and $C_{\mathcal{V}_i} = \int_{\mathcal{V}_i} \mathbf{q}\varphi(\mathbf{q})d\mathbf{q} / \int_{\mathcal{V}_i} \varphi(\mathbf{q})d\mathbf{q}$ is the centroid position computed over the i-th Voronoi cell (Cortes et al., 2004). In the control law (5.9), each agent converges asymptotically to its Voronoi centroid position. In our framework, (5.9) is adapted to multi-agent navigation, taking into account both static and dynamic obstacles, which modify the geometry of the Voronoi cell.

In the case of *static obstacles*, we modify the geometry of the Voronoi cell used for the computation of the centroid. Consider the following Voronoi-Visible partition

$$\mathcal{W}_i = \{\mathcal{V}_i \cap \mathcal{S}_i\}, \tag{5.10}$$

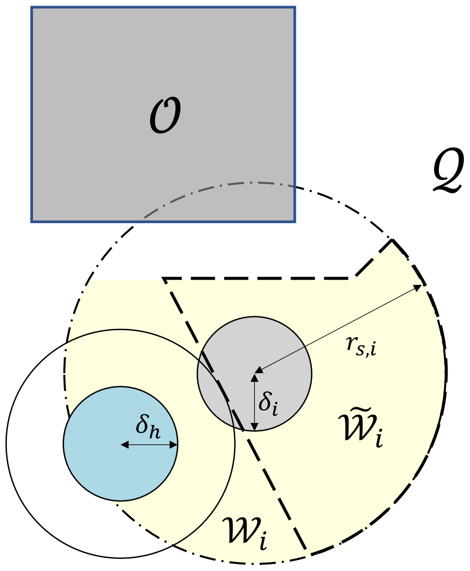


Figure 5.9: Voronoi cell geometry for robot navigation. The i-th robot with radius δ_i (gray circle) has an Voronoi-Visible partition $\mathcal{W}_i$ (yellow area), which is a function of the obstacles $\mathcal{O}$ (inflated by δ_i) and its maximum sensing range $r_{s,i}$ as in (5.10). The presence of the h-th dynamic obstacle of radius δ_h (blue circle) modify the cell in $\tilde{\mathcal{W}}_i$ (dashed area) according to (5.14).
.

where $\mathcal{S}_i$ indicates the i−th agent's Visibility set

$$\mathcal{S}_i = \left\{ \mathbf{q} \in \mathcal{Q} \mid \mathbf{p}_i - \zeta\left(\mathbf{p}_i - \mathbf{q}\right) \notin \tilde{\mathcal{O}}_i \right\} \cap \left\{ \mathbf{q} \mid \|\mathbf{q} - \mathbf{p}_i\| \leq r_{s,i} \right\}, \tag{5.11}$$

$\forall \zeta \in [0, 1]$, the set $\tilde{\mathcal{O}}_i \subset \mathbb{R}^2$ is the inflated obstacle space considered to account for the i−th agent's encumbrance, and $r_{s,i} \in [0, R_{s,i})$ is a tuning parameter that can assume values between zero and the maximum sensing range $R_{s,i}$ (see Figure 5.9 for a graphical representation). Since the Voronoi-Visible cell does not maintain a convex shape, in some pathological cases the centroid may not belong to the cell itself. Hence, the centroid position has to be modified as

$$C_{\mathcal{W}_i} = \arg \min_{q \in \mathcal{W}_i} \left\| \mathbf{q} - \bar{C}_{\mathcal{W}_i} \right\|, \tag{5.12}$$

where $\bar{C}_{\mathcal{W}_i}$ is the real, possibly unfeasible, centroid position. The Voronoi cell is modified also in presence of *dynamic obstacles*, i.e., humans and other robots. First, we assume that the encumbrances of the i-th agent and the h-th human can be approximated with circles centred in $\mathbf{p}_i$ and $\mathbf{p}_h$ of radius δ_i and δ_h respectively. Then we modify the partition as follows

$$\tilde{\mathcal{V}}_i = \begin{cases} \left\{ \mathbf{q} \in \mathcal{Q} \mid \|\mathbf{q} - \mathbf{p}_i\| \leq \|\mathbf{q} - \mathbf{p}_l\| \right\}, \text{ if } \Delta_{il} \leq \frac{\|\mathbf{p}_i - \mathbf{p}_l\|}{2} \\ \left\{ \mathbf{q} \in \mathcal{Q} \mid \|\mathbf{q} - \mathbf{p}_i\| \leq \|\mathbf{q} - \tilde{\mathbf{p}}_l\| \right\}, \text{ otherwise }, \end{cases} \tag{5.13}$$

where $l = \{1, \ldots, N\} \cup \{h_1, \ldots, h_{N_h}\}$ (i.e., the set of all the dynamic obstacles, robots and humans), $\Delta_{il} = \delta_i + \delta_l$ and $\tilde{\mathbf{p}}_l = \mathbf{p}_l + 2\left(\Delta_{il} - \frac{\|\mathbf{p}_i - \mathbf{p}_l\|}{2}\right) \frac{\mathbf{p}_i - \mathbf{p}_l}{\|\mathbf{p}_i - \mathbf{p}_l\|}$. In Figure 5.9, we also depict an example of Voronoi-Visible (5.10) partition modified in order to manage for dynamic obstacles (5.13), i.e.

$$\tilde{\mathcal{W}}_i = \{\tilde{\mathcal{V}}_i \cap \mathcal{S}_i\}. \tag{5.14}$$

As discussed below, moving the robot towards the centroid of the cell as modified by (5.14) and (5.12) guarantees the avoidance of both static and dynamic obstacles.

The LB control is designed by discretising the planned path $\mathcal{P}_{l,i}(t)$ in a set of waypoints $\mathcal{WP}_i = \{\text{wp}_i^1, \text{wp}_i^2, ..., \text{wp}_i^m\}$. If wp_i^k is the waypoint closest to the agent position $\mathbf{p}_i$, we integrate the information coming from the LPP by selecting as "active" waypoint $\text{wp}_i^{k+\iota_i}$, which lies in the intersection between the planned path $\mathcal{P}_{l,i}$ and the robot's Visibility set $\mathcal{S}_i$. The updated cost function for a generic cell partition $\mathcal{F}_i$ (which can be either (5.7), (5.10), (5.13) or (5.14)) we want to minimise is:

$$J(\mathbf{p}, \mathcal{F}) = \sum_{i=1}^{n} \int_{\mathcal{F}_i} \|\mathbf{q} - \mathbf{p}_i\|^2 \varphi_i(\mathcal{P}_{l,i}, \mathbf{p}_i, \mathbf{q}) d\mathbf{q}, \tag{5.15}$$

where the weighting factor $\varphi_i(\mathcal{P}_{l,i}, \mathbf{p}_i, \mathbf{q})$ is a kernel function fully parameterised by the active waypoint:

$$\varphi_i\left(\mathcal{P}_{l,i}, \mathbf{p}_i, \mathbf{q}\right) = \eta \mathrm{e}^{\left(-\left\|\mathrm{q}-\mathrm{wp}_i^{k+\iota_i}\right\|/\rho_i\right)} \tag{5.16}$$

The spread factor ρ_i quantifies how closely we aim for the goal. If ρ_i is small the centroid position $C_{\mathcal{F}_i}$ tends to converge towards the active waypoint position $\text{wp}_i^{k+\iota_i}$. On the other hand, a large value for ρ_i means that the relevance of the waypoint reduces and the geometric feature of the cell becomes prominent. The spread factor ρ_i is adjusted adaptively to explicitly consider the presence of obstacles, other robots, and humans (Boldrer et al., 2020c).

Due to the nonholonomic constraints of the model (5.1), the velocity $\dot{p}_i$ computed as in (5.9) cannot be applied directly. However, this problem can be addressed by applying the control logic proposed in Boldrer et al., 2020c. To this end, with reference to the generic cell partition $\mathcal{F}_i$, we define the desired heading as

$$h_i^D = \frac{C_{\mathcal{F}_i} - p_i}{\|C_{\mathcal{F}_i} - p_i\|},$$

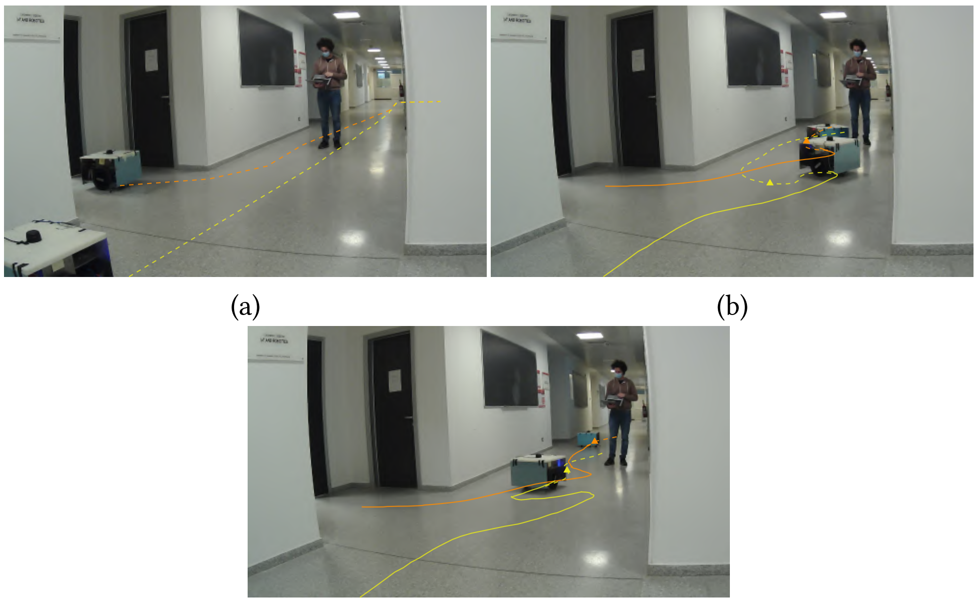
while the angular velocity and the linear velocity are computed respectively as follow,

$$\begin{aligned} \omega_i &= -\kappa_\omega \left(1 - \langle h_i^D, h_i \rangle\right)^\gamma \operatorname{sign}\left(h_{i,x}^D h_{i,y} - h_{i,y}^D h_{i,x}\right), \\ \dot{v}_i &= \begin{cases} k_a \left(v_i^D - v_i\right), & \text{if } \langle h_i^D, h_i \rangle \geq \cos\psi \wedge \beta_i \geq \beta_{i,\min} \\ -k_b v_i, & \text{otherwise,} \end{cases} \end{aligned} \tag{5.17}$$

where $\gamma \in (0, \frac{1}{2})$, k_ω, k_a, k_b are positive tuning parameters, v_i^D is the desired forward velocity, $h_i = [\cos\theta_i, \sin\theta_i]^T$ is the i–th robot heading, while $\psi \in (-\pi/2, \pi/2)$ is a design parameter.

5.4 Experimental results

In this section, we report an experimental comparison between the performance of the proposed framework sketched in Figure 5.4 with respect to the execution of subsets of functionalities, thus empirically proving the superior performance of the approach. The different performance will be compared both quantitatively and qualitatively with reference to the requirements R1–R5. The experiments were conducted using three mobile robotic platforms (see Figure 5.1) developed in our laboratories and equipped with an RPLidar A2[6], IMU and encoders, as well as with a Z83 II Intel Atom computing platform. The

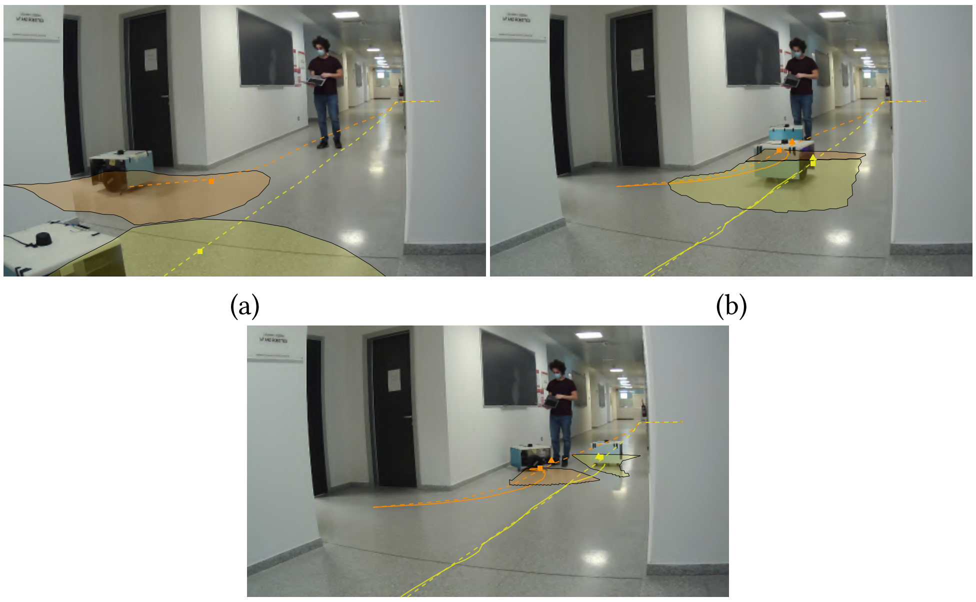


(a) (b) (c)

Figure 5.10: Three different instants of time with the robot positions, past trajectories (solid lines), planned paths (dashed lines) and the active waypoints $wp^{k+\iota_i}$ (triangles). In (a) we show the global planned path $\mathcal{P}_{g,i}$ (dashed lines), while in figures (b) and (c) the local position-based planned paths $\mathcal{P}_{l,i}(t)$ for each robot. In this case we tested the local position-based path planner. The approach works well when has to avoid obstacles with low speed. On the other side the avoidance of other robots results in non-efficient trajectories.

robots in the experiments moved with a desired velocity of $v_i^D = 50$ cm/s and relied on a Montecarlo Localisation module with an uncertainty in position below 14 cm and in orientation below one degree (Saarinen et al., 2013). For all the robots, we set the parameters as follows. The LB controller parameters are, the robots' radius $\delta_i = 0.35$ m, $r_{s,i} = 1.5$ m in (5.11), $\iota = 3$ in (5.16), $\kappa_\omega = 2$, $\gamma = 0.4$, $k_b = 8$, $k_a = 5$ and $\psi = \frac{\pi}{7}$ rad in (5.17). The parameters relative to the predictive path planner are the number of past human position measurements $n_P = 10$ used to predict $n_f = 30$ future human positions, $m_h = 100$ kg in (5.3), $\delta_t = 100$ ms in (5.4), the way-line sampled points $w_p = 7$ and the number of fixed orientations $w_o = 5$, $\lambda = 0.3$ in (5.2), the map discretisation $dx = 0.05$ m and the path planner look ahead distance $L_i = 3$ m. Particular attention should be paid to the control parameters that are related to the safety conditions. As we mentioned in Section 5.3.3, we are able to guarantee the safety requirement if the velocity of the robot can be directly imposed. By considering our unicycle like robots, we should set the safety critical parameters in order to have sufficiently quick reactions to the variations of the centroid position. These parameters, together with the other control parameters, are empirically defined based on the experimental evidences and returns the remarkable performance metrics adopted in our evaluations, as reported in the rest of this section.

For the training of the neural network of the LPred, we follow the path we have described in Section 4.6, where we have described in detail all the parameters relating to the training procedure.



(a) (b) (c)

Figure 5.11: Three different instant of time where we depict the robot positions, the past trajectories (solid lines), the global planned path (dashed lines) $\mathcal{P}_{g,i}$, the Voronoi cells associated to the robots $\tilde{\mathcal{W}}_i$, the active waypoints $wp^{k+\iota_i}$ (triangles) and the centroids positions $C_{\tilde{\mathcal{W}}_i}$ (squares). In this case we tested the Lloyd based approach. The robots are able to navigate together in an excellent way, however in some cases the progress of them can be easily blocked, leading to deadlock.

5.4.1 Navigation strategies comparison

Local position-based path planner (LPos). We start by showing the performance of the system when it relies only on the GPP and on the LPos, thus using only a planning mechanism to avoid unmapped static obstacles. The control inputs are computed by (5.17) and imposing the centroid position to be consistently equal to the active waypoint $wp_i^{k+\iota_i}$. This way, we use the same controller excluding the effects of the Lloyd-based algorithm by forcing $C_{\mathcal{F}_i} = wp_i^{k+\iota_i}$. As shown in Figure 5.10, the robots plan and follow a path treating the humans and the other robots as static obstacles, i.e., the planner uses as input the position of all the obstacles and neglects their velocity. Since the LPos activation period allowed by our computing platform is in the range $T_p = 330 - 800$ ms (failing R5), this solution cannot manage dynamic obstacles with an acceptable level of safety (failing R1). Moreover, as clearly visible in Figure 5.10, the robot behind (in yellow) continuously replans in order to avoid collisions with the robot ahead: the absence of coordination between the agents (R3) generates unnatural manoeuvres (failing R2 and R4).

Lloyd-based approach (LB). We now consider a scenario in which the path generated by the GPP is directly executed by a Llyod-based controller (i.e., we are excluding a refinement step by the LPos layer), which is activated every $T_c = 50$ ms. In Figure 5.11, we show that multiple robots are managed by the multi-agent behaviour implicit in the LB controller (R3). However, in contrast to what we have seen for the application of LPos only, the use of a lightweight algorithm (R5) enables frequent updates, making the trajectory safe (R1 and acceptable also in terms of multi-agent coordination R3). On the other hand, as shown in Figure 5.11-c, because the short sight nature of the algorithm the progress can be hindered by the presence of a non-collaborative human along the way. The absence of an adequate

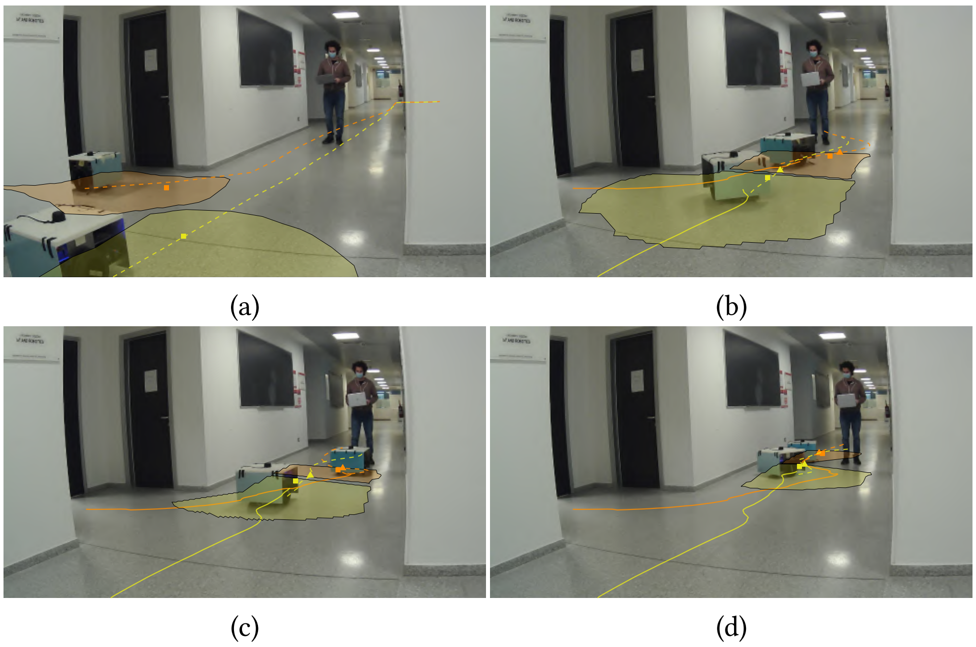


Figure 5.12: Four different instant of time where we depict the robot positions, the associated voronoi cells $\tilde{\mathcal{W}}_i$, the centroids positions $C_{\tilde{\mathcal{W}}_i}$ (squares), the active waypoints $wp^{k+\iota_i}$ (triangles) and the planned paths, in (a) the global planned paths $\mathcal{P}_{g,i}$, while in (b),(c) and (d) the local position-based planned paths $\mathcal{P}_{l,i}(t)$. We adopt the combination of the Lloyd-based and the position-based planned paths, by benefiting from the advantages and eliminating the disadvantages associated with approaches taken individually.

re-planning that travels around the obstacle can even generate deadlock conditions (failing R2 and R4).

Combination of LPos and LB. The combination of LPos and LB can compensate for the deficiencies of the two methods in isolation presented previously and produce a reliable solution. In Figure 5.12 we show how the robots avoid successfully a human being, with a correct management of the multi-robot navigation. The LPos operates on a time horizon dictated by T_p and helps to avoid deadlock solutions, while the LB controller short-term corrections every T_c ensure safety and robots coordination.

5.4.2 Human motion prediction

In the experiments proposed above in Section 5.4.1, we restricted to the case of almost static humans. If the humans move at non-negligible speed, the application of LPos+LB can deliver a bad performance or produce socially unacceptable paths (failing R2). This is illustrated in this section, where we compare the use of LPos+LB and LPP+LB (where LPP comprises both LPos and LPred) in two common case studies: the *passing* case (i.e., human moving on a collision course parallel to the robot and with opposite orientation) and the *crossing* case (i.e., human crossing the trajectory of the robot). In Figure 5.13 we show the results of the passing case.

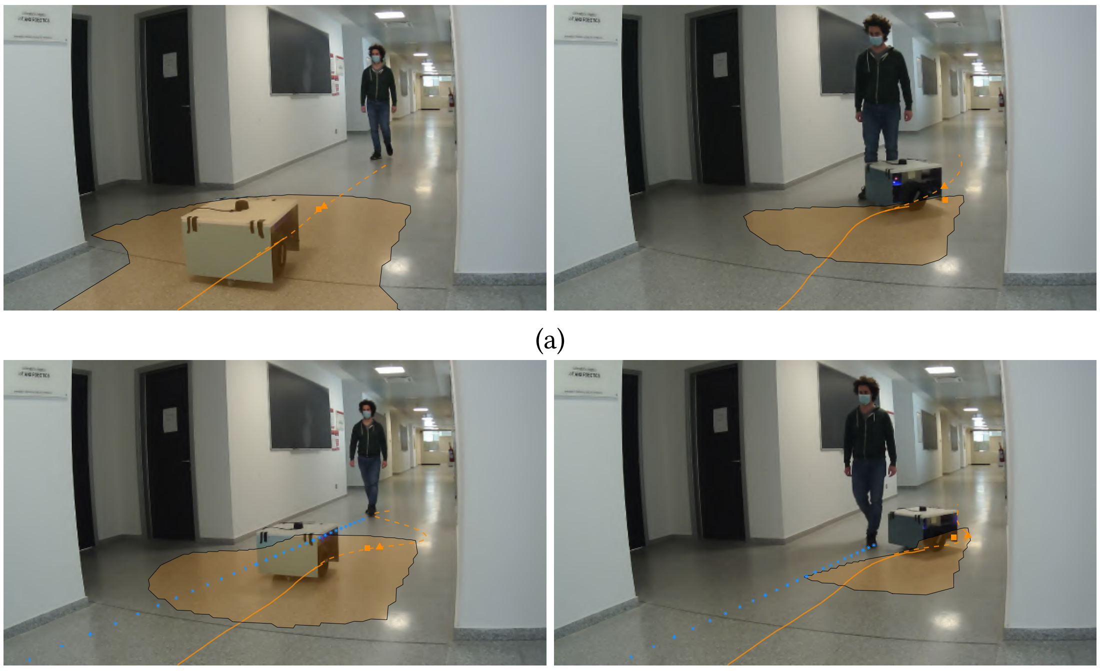


Figure 5.13: In (a) we show the combined approach without human motion prediction, in (b) we pass the human motion prediction (blue asterisks) to the path planner. The planner use the prediction to anticipate the manoeuvre (left hand side), it results in a motion that stays clearer from the private space of the human being, and do not force him to slow down or deviate from his initially desired path.

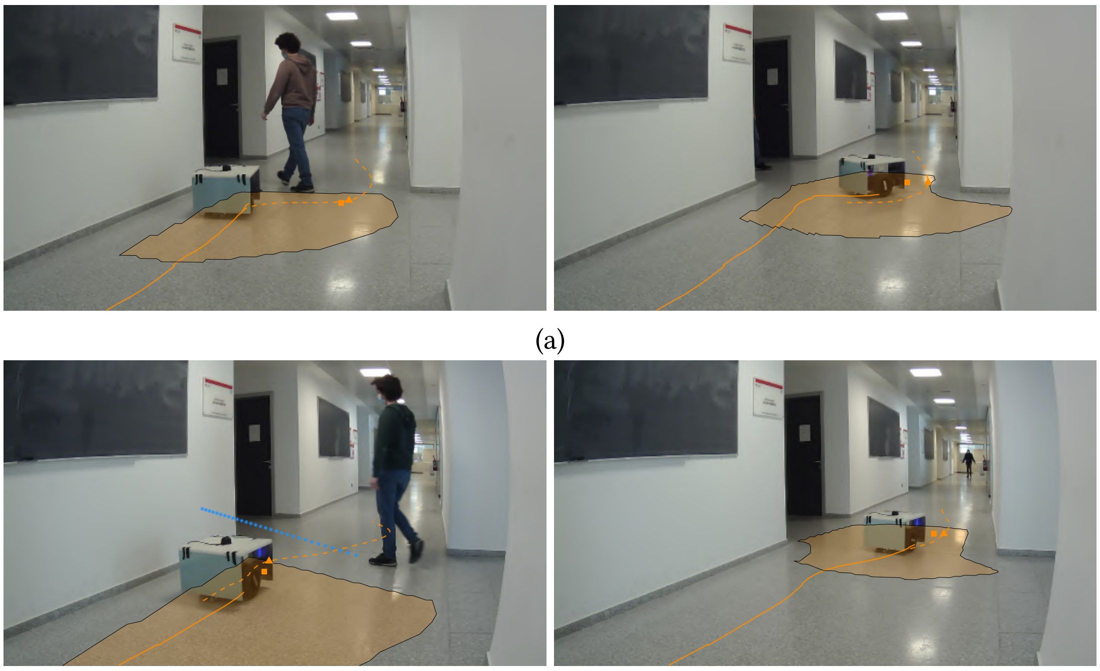


Figure 5.14: In (a) the crossing case where the robot do not use human predictions, in (b) the same situation with human motion prediction (blue asterisks). It can be notice how in (b) the robot understand earlier to pass behind the human being with respect to (a), this results in 1. maintaining a greater distance from the human and 2. deviate less from the global path planner (i.e. diminish the time to reach the goal).

	v_h (m/s)	Metric	Prediction	No Prediction
Passing	0.2	t_g (s)	20.38	20.87
		$d_{i,h}^{\min}$ (m)	0.5	0.52
		$t_{i,h}^{\max}$ (s)	7.03	7.27
	0.4	t_g (s)	21.83	×
		$d_{i,h}^{\min}$ (m)	0.46	×
		$t_{i,h}^{\max}$ (s)	5.60	×
Crossing	0.4	t_g (s)	13.00	14.91
		$d_{i,h}^{\min}$ (m)	0.88	0.45
		$t_{i,h}^{\max}$ (s)	5.15	6.38
	0.6	t_g (s)	12.56	14.20
		$d_{i,h}^{\min}$ (m)	1.06	0.74
		$t_{i,h}^{\max}$ (s)	4.10	7.10

Table 5.1: Quantitative results with two sets of experiments, i.e. *passing* case and *crossing* case. We report as metrics (i) the time t_g to reach the target, (ii) the minimum distance $d_{i,h}^{\min}$ between the robot and the human, (iii) the time interval $t_{i,h}^{\max}$ of the interactions (when distances are between 0 and 2 m).

Specifically, in Figure 5.13-a we show the result of the LPos+LB solution, whereas in Figure 5.13-b we show the results of using human motion prediction (LPP+LB). The predicted human positions are marked with blue asterisks in the picture. The dashed line denotes the planned path. As apparent from Figure 5.13-a, the absence of prediction on the motion of the human defers the avoidance manoeuvre forcing the human to slow down and stop until the robot passes by. On the contrary, the predictive planner LPred anticipates the avoidance manoeuvre allowing the human to continue his walk undisturbed.

The crossing case is depicted in Figure 5.14. Also in this case, Figure 5.14-a shows the result of the application of LPos+LB while Figure 5.14-b shows the application of human motion prediction (LPP+LB). As evident form the figure, the use of the human prediction allows the robot to start the avoidance manoeuvre (passing behind the person) much earlier, using the free space and minimising the deviation from the global path. On the contrary, when using LPos+LB the robot tries to pass in front of the human until it is forced to slow down and deviate for safety reasons when it comes in the close proximity of the human. In both the cases of Figure 5.14, the manoeuvre of the robot is safer (it stays clear of the human) and delivers better performance. This is best shown taking a look at some quantitative performance indices: the time to reach the goal position t_g, the minimum distance achieved between the robot and the human being position $d_{i,h}^{\min}$ and the maximum time interval of interaction $t_{i,h}^{\max}$ (when the human-robot distance is between 0 and 2 m). For the sake of repeatability, for this comparison we simulated the motion of the human using a robot moving at constant speed. We replicated the experiment both for the passing and for the crossing case multiple times at different velocities v_h of the dynamic obstacle, keeping a constant desired velocity v^D for the robot. The results are reported in Table 5.1. Notice that for the passing case with no prediction, i.e., LPos + LB, the collision was not avoided for the case of $v_h = 0.4$ m/s, hence no results can be shown. The predictive algorithm works better in all the cases. By increasing the velocity of the dynamic obstacle v_h, the benefits are even more evident.

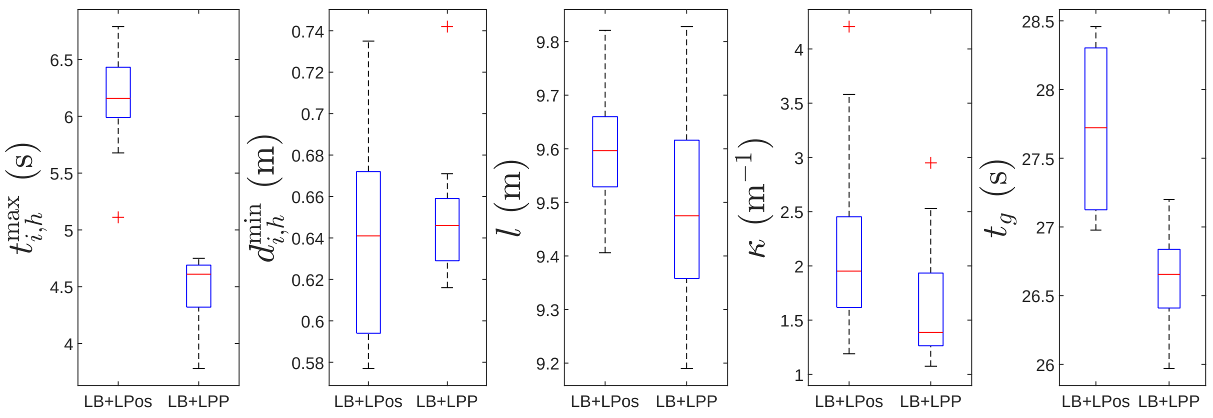


Figure 5.15: Box plots for the chosen metrics computed on data collected from an experiment with two robots and one pedestrian, repeated 10 times by using LB+LPos and LB+LPP.

In the previously presented results, we demonstrated the superiority of the LB+LPP algorithm compared to LB, LPos or LB+LPos. To provide a metric of the robustness and the repeatability, Figure 5.15 reports some statistics of the results obtained from 20 separate experiments expressed in terms of the described metrics. In particular, we made repeated experiments comprising two agents and one human being that goes straight ahead in a passing by condition. The human behaves "nicely": s/he does not induce a collision on purpose and, if the risk materialises, s/he just stops.

Since using the LPos or LB in isolation may lead to collisions and/or unnatural behaviours (LPos) or to deadlock conditions (LB) (as shown in Section 5.4.1), we report only the experimental data obtained by using LB+LPos and LB+LPP. Figure 5.15 clearly shows the repeatability and the robustness of LB+LPP, which leads the robots to stay clear from the pedestrian private space ($d_{i,h}^{\min}$) and to reach the goal position in a finite time (t_g), maintaining consistent values for path curvature (κ), path length (l) and interaction time with the pedestrian ($t_{i,h}^{\max}$). Instead, the LB+LPos has consistently worse performance since it does not predict the human intentions. Thus, it does not anticipate the avoidance manoeuvre and, as a consequence, the human has to temporarily stop and let the robot pass by (the reader is referred to the multimedia material for further evidence). This fact is captured by the time of interaction $t_{i,h}^{\max}$ and also by the time to reach the goal t_g, which perform significantly better in the LB+LPP case. Moreover, as showed in the experiments of Table 5.1, in the case of LB+LPos, if the pedestrian maintains constant speed a collision would occur. By using LB+LPP, instead, the robots avoids the human without forcing her/him to change the forward velocity.

In Figure 5.16 we show a more complex experiment with 3 agents and multiple humans in an hallway. The robots in this case share the estimated humans positions with the neighbours, making robots aware of the presence of pedestrians even if they have the view obstructed.

In Figure 5.17 we show the time evolution of the distance between the centroid position $C_{\tilde{\mathcal{W}}_i}$ and the active waypoint $wp_i^{k+\iota_i}$ for the agent in front, i.e. Ag. 2, and the agent in the middle position, i.e. Ag. 3. This value quantifies the "action" of the Lloyd-based approach at each time instant. As evident from the figure, Ag. 3 has larger values of $\|C_{\tilde{\mathcal{W}}_i} - wp_i^{k+\iota_i}\|$ because of the presence of Ag. 2 in front, while the latter presents peaks of that value in the interval of time when it interacts with humans. We report also the minimum distance $d_{ij} = \|p_i - p_j\|$ between the agents during the experiment: $d_{\min,12} = 1.18$ m, $d_{\min,13} = 0.93$ m, $d_{\min,23} = 1.33$ m, which clearly show that the safety constraints

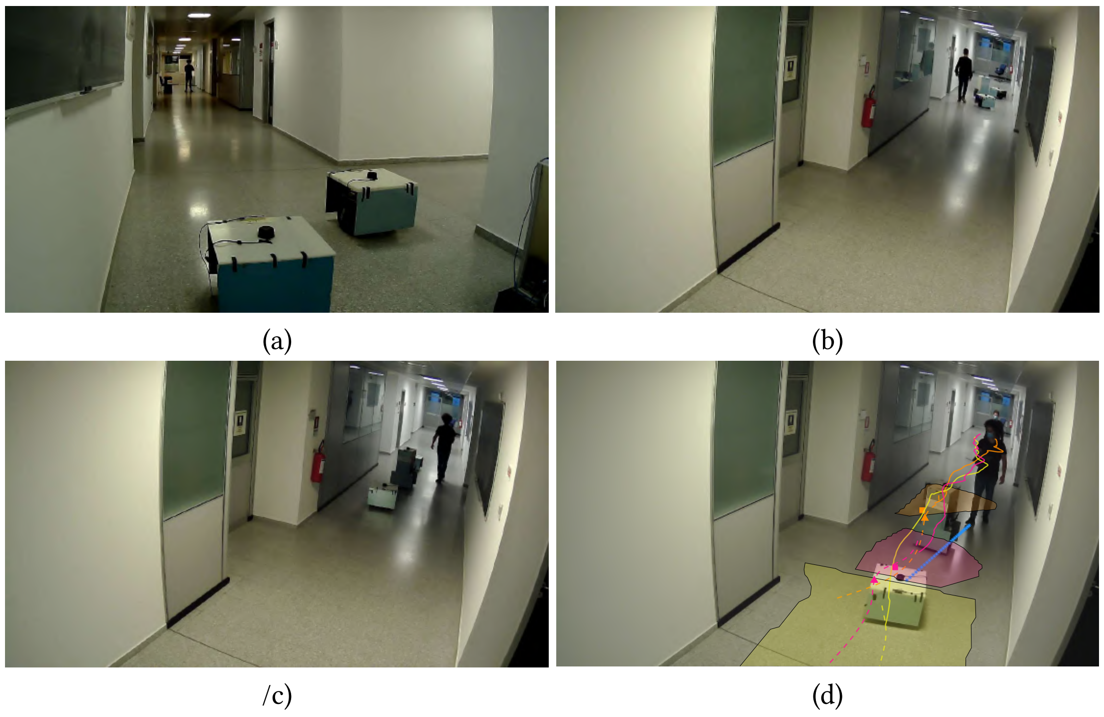


Figure 5.16: Four snapshots representing different instants of time of an experiment with three robots. The corresponding times instant are $t_a = 0$ s, $t_b = 21.5$ s, $t_c = 34.5$ s, $t_d = 49$ s. In (d) we depict the past trajectories (solid lines), the $\mathcal{W}_i$ cell for each robot, the centroid position $C_{\mathcal{W}_i}$, the active waypoint position $wp^{k+\iota_i}$, the planned paths by using LPP (dashed lines) and the human motion predictions (lines with asterisks).

$\Delta_{ij} = 0.7$ m, is abundantly respected. Finally, in Figure 5.18 we report the average updating frequency for the LPP as a function of the number of humans in the robot range. Since the local path planner have a space of interest less than 10 m, which is much larger with respect to the area of attention in humans in shared spaces (as discussed in Sections 4.5.1 and 2.2), the number of perceived pedestrians is usually quite limited. In Figure 5.18 we consider a maximum of 8 pedestrians. As expected, the LPP updating frequency decreases with the number of humans perceived, remaining, however, in the target frequency interval $1.25 - 3.0$ Hz specified in Figure 5.4 for an effective real–time implementation.

5.5 Comments

In this chapter, we have presented a novel solution for multi-agent navigation in human-shared environments. Our idea derived from the cognitive framework introduced in Section 1.2, and then its implementation has been realized by nesting layers of different tasks, namely a global path planner, a predictive path planner, and a reactive approach, which were combined to obtain a safe and socially aware multi-agent navigation. We tested our approach on real robotic platforms in challenging natural environments and situations, where the proposed layered scheme outperformed different possible approaches. Each layer is based on different types and qualities of information, and focuses on a time horizon with a specific frequency update chosen according to the appropriate action that the robot should perform. With the specific modularity of the levels, which are organized according to a vertical architecture, we give a certain degree of independence to each task, which can superimpose its

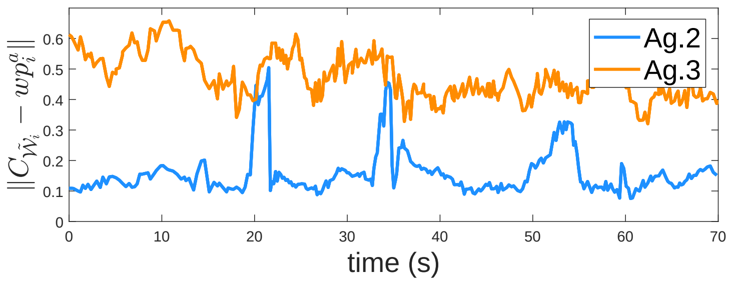


Figure 5.17: Quantitative results for the experiments depicted in Figure 5.16. In blue the distances between the active waypoint wp_2^a and the centroid position $C_{\tilde{\mathcal{W}}_2}$, in orange the same quantity for Ag. 3.

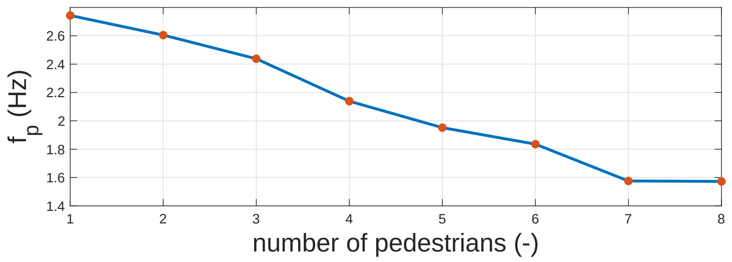


Figure 5.18: Updating frequency (on average) of the path planner LPP $f_p = 1/T_p$ as a function of the number of pedestrians perceived in the 10 meters range.

effects in the resulting robot action however without hindering (or being hampered by) another layer. In addition, the organization of our solution allows us to plan new developments on each level without changing the framework architecture.

A crucial aspect that must be guaranteed in any navigation algorithm for shared spaces is safety, i.e. robots must navigate without colliding with static and dynamic obstacles. Our approach to safety is based on redundant solutions. The first level of safety is guaranteed by the LPred planner, which accounts both for the static obstacles and for the predicted motion of the humans. The second level is enforced by the LB controller which accommodates for possible prediction errors, the presence of other agents, and for obstacles not detected at planning time.

Indeed, as a future work direction, we are seeking to integrate more sensor (e.g., a visual-based tracker for humans, as in Section 3.2) in order to improve the accuracy of the prediction and hence increases the robustness of the generation of local plans. In the same direction goes an improvement of the human prediction model, where the integration of more sophisticated models for human motion, such as the HFSM (Farina et al., 2017b) into the structured neural network, or the multi-goal approach presented in Section 4.5, could further improve the quality of the prediction. Finally, an interesting research avenue is the computationally efficient integration of multi-agent behaviours within the LPP. In order to preserve safety, and fulfill all the requirements chosen for the socially aware navigation, the multi-agent coordination is delegated solely to the LB layer. Improved performance could be obtained by bringing the information coming from neighbors to the local path planner level.

Chapter 6

Teach-by-showing navigation

One of the most important barriers towards a widespread use of mobile robots in unstructured and human populated work environments is the ability to plan a safe path. Existing approaches to plan safe paths can be roughly divided in two broad categories. In the first type, the robot's control is fully entrusted to trained human users, who are responsible for monitoring robot movements and determining their trajectories. The second type of planners is the one that train robots to plan their own pathways and move independently. With novel solutions we can achieve promising results, as shown in the previous chapter. But still, navigating in complex environments presents several problems mainly due to localization procedures and the interpretation of more complicated social norms than simple proxemics. However, we can reverse the paradigm. Our idea is to delegate the navigation activity to a human operator that walks in front of the robot marking with her/his footsteps the path to be followed. Thus, the person must concentrate only on the path to take, while the robot for its part is able to memorize the path travelled and reuse it in future missions.

This approach greatly simplifies the path planning task, thus it does not require particularly expensive sensors or highly advanced software components. Nevertheless, its implementation requires a high degree of robustness in locating the specific person to be followed (the *path-finder*), since the high accuracy of the distance of the entities around the robot ensures its safety, as the robot can stop in time before colliding with static obstacles and other people. We propose a three phases approach to fulfil this goal: 1. identification and tracking of the person in the image space; 2. sensor fusion between camera data and laser sensors; 3. point interpolation with continuous curvature paths. The approach, described in detail in what follows, was extensively validated with experimental results on our mobile robot.

6.1 Overview

When an autonomous mobile robot of remarkable size and mass navigates the treacherous waters of unstructured and human-populated environments, safety concerns and regulation constraints take centre stage and become a barrier for the adoption of this technology. To mitigate this problem, we advocate a mixed approach. When the mobile robot travels across a known safe or segregated area, it can move in full autonomy, whilst whenever it enters a shared or dangerous area where, in case, environmental reliable information lacks (e.g., absence of an a-priori map or in a highly dynamic environment), the responsibility of the most critical decisions (i.e., motion planning) is shifted to a human operator.

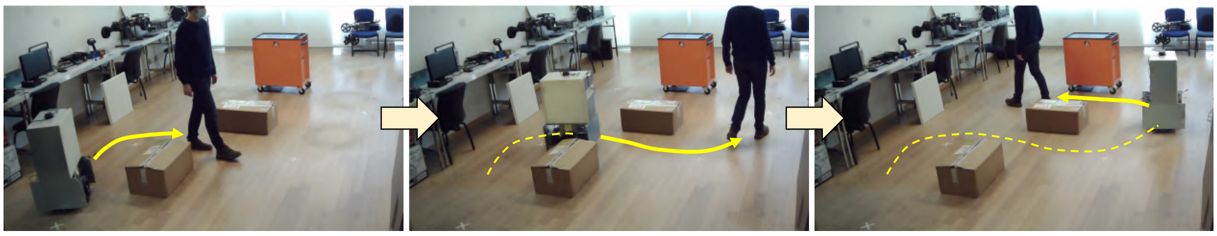


Figure 6.1: Different snapshots representing a series of time instants where the robot follows the human operator across an environment with various static obstacles.

Our reference scenario can be described as follows. The mobile robot starts its mission with a person standing in front. The robot looks at the person with its visual devices, extracts a number of important features and elects her/him as a *path-finder*. Then starts the second phase: the person walks to the destination, with the robot tracking and following her/him moving along the path marked by her/his footsteps (see Figure 6.1). After the path-finder reaches the destination, the path is memorised and can be used for future missions. Observe that this is not a standard leader-follower application in which the robot is allowed to sway sideways as far as it keeps a specified distance from its leader (Lam et al., 2010). In our case, the human is a path-finder and the robot follows exactly her/his virtual footprints. The advantages are manyfold. From the robot's perspective, the human acts as an external module for the motion planning task, simplifying the complexity of the software components and of the sensing systems, while enabling the motion in a-priori unknown environments. From the perspective of the operator, s/he is in condition to drive a complex and heavy robot without any skill other than being able to walk. The robot operates semi-autonomously, i.e. it does not interfere with the pathfinder choices nor does it modify the path. However, it is allowed to stop when an obstacle materialises below a safety distance.

Therefore, our system is required to comply with two requirements:

Q1: The robot shall follow the path-finder even if s/he falls outside of the visual cone of the camera: the path has to be reconstructed and exactly followed even after sharp turns. This marks a remarkable departure from standard visual servoing approaches, which require the human to constantly remain within the robot's field of view.

Q2: The robot shall not collide with humans and obstacles. Although the path-finder is assumed to follow a safe path, the robot has to react to the unpredicted changes typical of a dynamic environment.

Our processing and execution pipeline has three phases:

1. Identification of the path-finder within the front camera image frames;

2. Fusion of the visual information with the one coming from other sensors;

3. Reconstruction of a smooth and feasible path from the time series of the path-finder's positions to be followed by a controlled motion along the path.

The first phase is troublesome because the path-finder position is extracted from a noisy source, in which an ambiguous classification of the different subjects is quite frequent. Our solution is to split the first phase into three sub-phases. The first one detects the objects of interest within the image using a state-of-the art Convolutional Neural Networks (CNN) detector. The second sub-phase recognises the path-finder between the objects detected in the image. The feature identification is kick-started during

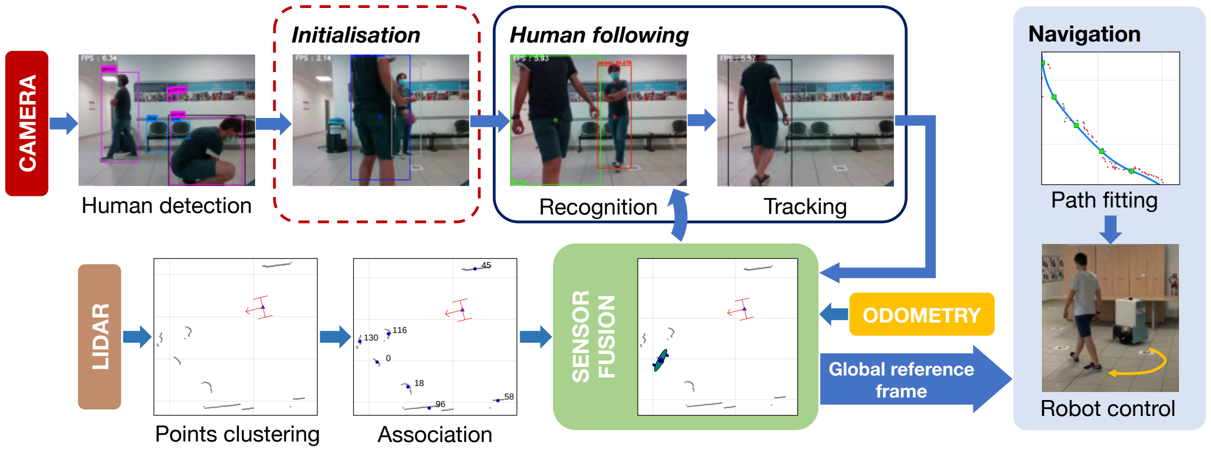


Figure 6.2: Flow diagram of our framework. The system starts with the initialisation procedure, collecting visual features of the path-finder. Then, in the path-finder following phase, the recognition module retrieves the new path-finder's position (see Section 6.3.2). The sensor fusion module fuses the camera tracking with the data from the LIDAR sensor, and redirects back the information to the recognition module (see Section 6.3.3). Finally, the set of the path-finder's positions over time are forwarded to the path reconstruction module (see Section 6.4.1) and the control module (see Section 6.4.2).

the starting phase and is continuously refined during the system operation. The recognition properly said is performed by a K-Nearest Neighbour (KNN) classifier. The third sub-phase consists of a tracking module, which ensures continuity in the estimated positions of the target across different frames. In the second phase, the image information is fused with the measurements of a LIDAR sensor to reconstruct the correct location of the target and its headway distance from the robot. The specific sensors setup is a direct consequence of the considerations reported in Section 3.5. The third phase processes the time series of the estimated position of the path-finder, refining the path and guiding the navigation. This step uses clothoid curves to interpolate the points, which produces a path with continuous curvature and easy to follow for a robot. Finally, the control module follows the estimated path and enforces the necessary safety policies.

The idea outlined above can be seen as an original and modern application of the teach-by-showing approach to mobile robots moving in a complex scenario. This is classified in the recent literature (Islam et al., 2019) as a very relevant and largely open problem and is the key methodological contribution made in this chapter. As a result, planning in environments that are a-priori unknown to the robot becomes feasible, which is a remarkable novelty for the field. Other two contributions have a more technical nature and descend from the complexity of our safety and reliability requirements. The first of them is the combination of tracking filter and neural network to estimate and follow the path-finder's position, which allows us to follow the path-finder even when s/he falls outside of the camera's visual cone. The second one is the idea to feedback the fused estimate into the recognition module and to exploit a trained neural network using its last layer to classify the person's feature set. This solution significantly improves the system's ability to distinguish between persons with similar features and resolve misclassifications due to illumination changes and partial occlusions (see Figure 6.2).

The chapter is organised as follows. In Section 6.2, we summarise the most important existing results that we used as a reference for this work. In Section 6.3, we present our general architecture and provide details on the perception components, while in Section 6.4, we show our solution for path reconstruction and the control strategy for following the path. The experiments supporting the validity of the approach are described in Section 6.5. Finally, we give our comments in Section 6.6.

6.2 Related work

People following is a complex activity requiring a combination of perception, planning, control, and interaction strategies. Following a specific person rather than any person adds more to the complexity of the problem and is largely classified as an open problem (Honig et al., 2018). The main issue is that in a complex scenario many people can look similar if they do not wear specific markers. Most of the methods developed in the last decade and surveyed by Islam et al., 2019, claim a good performance in detection and tracking of humans, while less of one half apply online learning or perform person re-identification, and even fewer do both. Target re-identification and recovery were handled first with probabilistic models (e.g. Kalman filters), features-based techniques, and more recently with appearance-based deep networks, but these methods were not investigated further. Human-following applications require instead the combination of sophisticated learning approaches, model-based filtering and path interpolation.

6.2.1 People following

The combination of detection, tracking, and recognition was proposed by Jiang et al., 2018 using Speeded Up Robust Features (SURF). Chen et al., 2017a employed an adaptive boosting (AdaBoost) together with a stereo camera to real-time track a person, where the depth information is used to reinforce the classifier. Their approach can deal with appearance changes, people with similar clothes, and complete occlusions, but follows a classic visual-servoing approach: the robot control module is programmed to keep the target always within the camera frame, which is a remarkable difference with respect to our approach. Similarly, Wang et al., 2018 combined a monocular camera with an ultrasonic sensor to fuse range information with Kernelised Correlation Filters (KCF) based visual tracking. Their system has been tested in the case of visual interferences such as occlusions and illumination changes, however, due to the nature of the sensors employed, the human must remain in the camera view, and there is no specific strategy if the human's appearance changes. An implementation of RGB-D camera, laser scanner, and EKF is used Nikdel et al., 2018 for their following-ahead mobile platform. Their framework likewise assumes that the human will often be outside the camera view, so the laser data and a nonholonomic human motion model are used to recover missing image data. Nevertheless, the presence of multiple humans undermines the tracking performance, which is instead one of the positive features of our solutions.

6.2.2 Background material on vision-based techniques

Object detection. Object detection is in our framework the first element of the processing pipeline. For this component, we sought a good compromise between classification accuracy and achievable frame rate. The available methods range from object detection and segmentation methods (Liu et al., 2016; Redmon et al., 2016b; Girshick et al., 2014), to specific solutions for human pose detection such as OpenPose (Cao et al., 2019). YOLO (Redmon et al., 2016b) is a very effective solution based on a single CNN; its main known disadvantage materialises when two classes have similar probabilities or the shape of the element is not perfect and the algorithm could produce different bounding boxes for the same object (see also Section 3.1). Alternative solutions such as SSD (Liu et al., 2016) apply correction techniques to overcome the limitation of the approach (Neubeck and Van Gool, 2006). SSD is also based on a single CNN to produce bounding boxes, but internally performs Non-Maximum Suppression (NMS) to remove unnecessary detection. Moreover, while the architecture of YOLO is designed as a compact block, SSD is instead modular, divided into convolution layers of different scales combined at the end.

People recognition. People recognition in computer vision is difficult in its own right. An additional level of complexity for robotics applications is introduced by the fact that the camera used for image acquisition is mobile. Traditional offline algorithms like Support Vector Machines (SVM) (Hearst et al., 1998) are known to react quickly to classification queries, but are not a good fit for our scenario, because we lack a prior knowledge on who is going to be the path-finder and we need to be robust against possible changes in her/his appearance. Methods based on key feature point matching (Pun et al., 2015) are known to be robust and are widely used to find small patterns in complex images, but in our tests the PRID450 (Person Re-IDentification) dataset (Roth et al., 2014) showed a high number of errors for low-res images and for deformable shapes such as humans clothes (see Section 6.5.1). Our final solution was based on the use of a K-Nearest Neighbours (KNN) classifier, which is an efficient training-free classification method albeit it requires the knowledge of representative points for the classification. For this information we used the last layer of a CNN, which gets trained with the different views of the path-finder. The idea of using a CNN classifier to extract the feature set was presented by Ristani and Tomasi, 2018, who proposed a solution to match detections from multiple cameras. The classifiers evaluated for comparison in this work are the Deep Neural Networks (DNNs) based GoogLeNet (Szegedy et al., 2015) and ResNet (He et al., 2016). ResNet architecture is made of convolution blocks stacked one after the other, with an additional identity connection that preserves the input image through several layers of the network. GoogLeNet introduced the so-called inception module, which parallelises three different convolution filters and a max-pooling filter.

Person tracking. For person tracking, we could select from a large variety of approaches for the tracking of general objects (the fact that our object of interest is a person does not make a big difference in this case). Specifically, we considered: the Multiple Instance Learning (MIL) tracker (Babenko et al., 2010), the Kernelised Correlation Filters (KCF) tracker (Henriques et al., 2012), the Median Flow tracker (Kalal et al., 2010), the Channel and Spatial Reliability Tracker (CSRT) (Lukezic et al., 2017), the Minimum Output Sum of Squared Error (MOSSE) tracker (Bolme et al., 2010), the Generic Object Tracking Using Regression Networks (GOTURN) tracker (Held et al., 2016), and the Tracking-Learning-Detection (TLD) (Kalal et al., 2011). The MIL tracker (Babenko et al., 2010) is trained online during the execution of the tracking by generating negative samples from bounding boxes that do not overlap the correct one and by creating multiple instances around the correct sample for classification improvement. The KCF tracker (Henriques et al., 2012) is an extension of the MIL tracker which relies on Fast Fourier Transformations to increase accuracy and speed, but its weakness stands in full occlusions. The Median Flow tracker (Kalal et al., 2010) is a reliable method that locates the subject according to its trajectory, thus using an estimation of its motion model, however, despite its being robust, it suffers with high deformable subjects such as animals or humans. The CSRT (Lukezic et al., 2017) uses

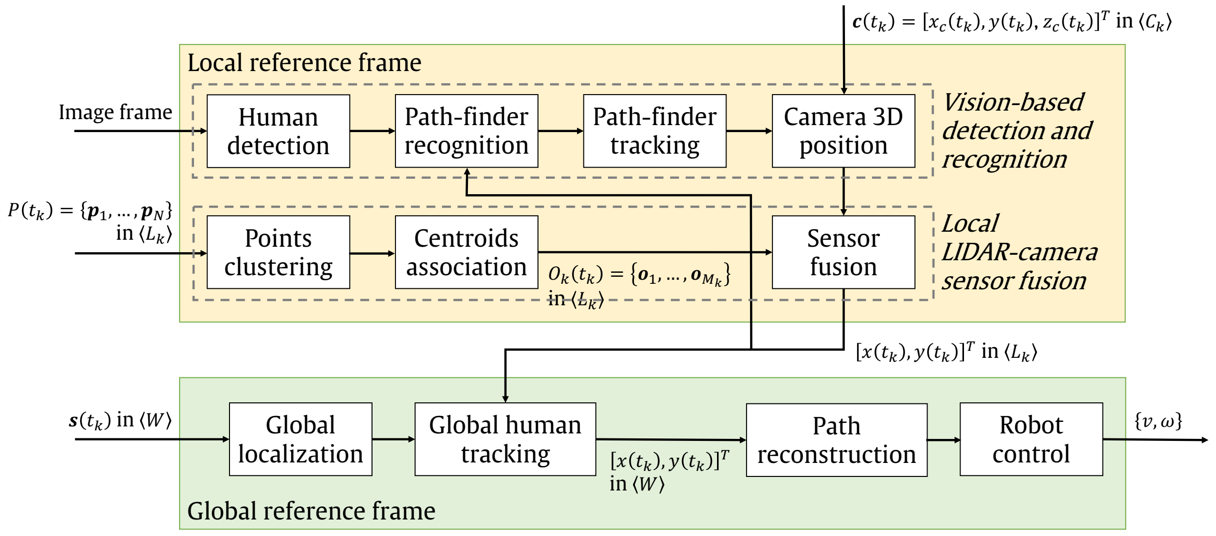

$\boldsymbol{c}(t_k) = [x_c(t_k), y(t_k), z_c(t_k)]^T$ in $\langle C_k \rangle$
Local reference frame
Image frame
Human detection
Path-finder recognition
Path-finder tracking
Camera 3D position
Vision-based detection and recognition
$P(t_k) = \{\boldsymbol{p}_1, \dots, \boldsymbol{p}_N\}$ in $\langle L_k \rangle$
Points clustering
Centroids association
$O_k(t_k) = \{\boldsymbol{o}_1, \dots, \boldsymbol{o}_{M_k}\}$ in $\langle L_k \rangle$
Sensor fusion
Local LIDAR-camera sensor fusion
$[x(t_k), y(t_k)]^T$ in $\langle L_k \rangle$
$\boldsymbol{s}(t_k)$ in $\langle W \rangle$
Global localization
Global human tracking
$[x(t_k), y(t_k)]^T$ in $\langle W \rangle$
Path reconstruction
Robot control
$\{v, \omega\}$
Global reference frame

Figure 6.3: Overall scheme of the algorithm.

a high number of cross correlation filters in order to reach a very high accuracy, compensated by a low frame-per-second (FPS) rate. The authors of the MOSSE tracker (Bolme et al., 2010), based on the MOSSE correlation filter, state robustness against variations in lighting, scale and non-rigid deformations, moreover, in our experiments it showed extremely fast computations, i.e. high FPS rate (see Section 6.5.1). The GOTURN tracker (Held et al., 2016) is based on an offline trained CNN, hence can perform at a very high FPS rate. However, since it takes one frame at a time and always compares it to the previous one, this algorithm suffers total occlusions. Differently from all the other methods, the TLD (Kalal et al., 2011) is able to overcome long-time total occlusion and to offer a long-term tracking, which is paid by a low FPS rate and a huge quantity of false-positive predictions.

6.2.3 Sensor fusion

Our application requires 3D reconstruction of the human pose. The combination of stereo and RGB-D sensors with skeleton-based approaches proves very useful for this purpose and it is significantly simplified by the availability of public domain software components, as seen in Section 3.2. However, the simple use of visual information has known limitations, such as the sensitivity to lighting conditions and the high computation times. Laser-based sensors, on the other hand, are relatively reliable on a long range and are less computation hungry than vision based approaches. However, recognising a specific person from a slice of a 2D point cloud is hopeless (recall Section 3.4). For this reason, moving in a direction frequently taken in robotics (Zhen et al., 2019; Wolcott and Eustice, 2014; Nguyen et al., 2021), and following the results of the experiments in Section 3.5, we apply a combination of cameras and LIDARs. The use of separate systems for depth estimation and classification improves the robustness of the tracking system when one of the sensors fails: if the human falls outside the camera's field of view, we keep using the LIDAR sensor for tracking, whose reliability changes according to the adopted human motion model.

6.3 Tracking the human path-finder

Before going into the details of the algorithm we use to track the human, we succinctly describe the available sensing system and the model of the platform. The reference model for the robot (in Figure 6.4-a) is the unicycle, whose kinematics can be described in discrete−time by the following Zero-Order-Hold model:

$$\mathbf{s}(t_{k+1}) = \begin{bmatrix} x_r(t_k) + \cos(\varphi_r(t_k))(t_{k+1} - t_k)v_r(t_k) \\ y_r(t_k) + \sin(\varphi_r(t_k))(t_{k+1} - t_k)v_r(t_k) \\ \varphi_r(t_k) + (t_{k+1} - t_k)\omega_r(t_k)f \end{bmatrix} \tag{6.1}$$

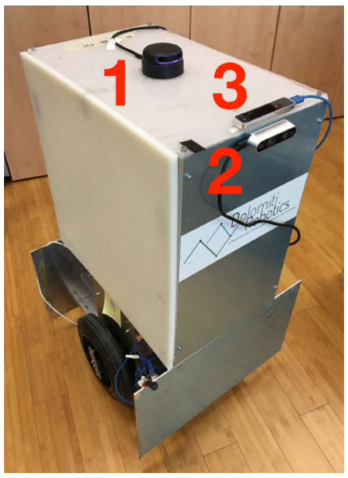
where $\mathbf{s}(t_k) = [x_r(t_k), y_r(t_k), \varphi_r(t_k)]^T$ is the state of the robot, the Cartesian coordinates $(x_r(t_k), y_r(t_k))$ refer to the mid-point of the rear wheels axle in the $X_w \times Y_w$ plane expressed in the $\langle W \rangle = \{X_w, Y_w, Z_w\}$ world reference frame, $\varphi_r(t_k)$ the longitudinal direction of the vehicle with respect to the X_w axis, $v_r(t_k)$ and $\omega_r(t_k)$ the longitudinal and angular velocities, respectively, and t_k the reference time instant, which is usually chosen as an integer multiple of a fixed sampling time. Importantly, the proposed framework would be applicable to different robot dynamics; however, as explained next, the unicycle structure is particularly convenient for the class of applications we address.

Without loss of generality, we assume here that the choice of the sampling time $\delta_t = t_{k+1} - t_k$ is imposed by the sensor with the lowest sampling frequency. The assumed sensing configuration is based on the presence of rotation encoders on each of the rear wheels or any other sensing system able to provide ego-motion informations. For the perception of the surroundings, the sensing system comprises a LIDAR and an RGB-D camera, as the configuration described in Section 4.7.1 following the experimental results of Section 3.5. The LIDAR[10] data are used to both track humans around the vehicle and to localise the vehicle inside the environment. The RGB-D camera[11] is primarily used for the human detection and tracking, indeed whose produced data are used in the vision-based detection and recognition system described in Section 6.3.2.

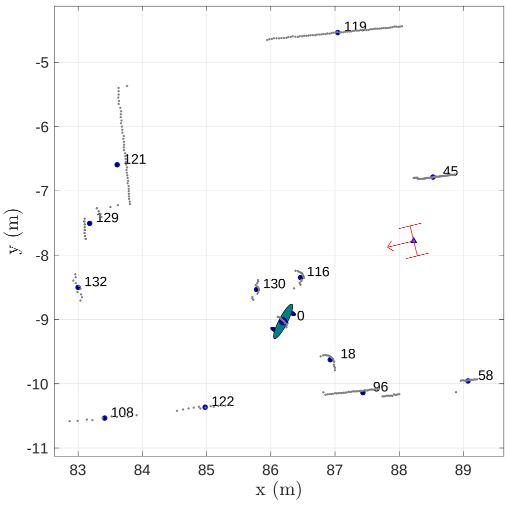
The LIDAR and the camera are rigidly mounted on the top of the robot chassis (see Figure 6.4-a) and return the measurements at time t_k in the LIDAR $\langle L_k \rangle$ and camera $\langle C_k \rangle$ reference frames, respectively, which are bot rigidly linked to the robot (i.e., they operate with a local coordinates reference system). The transformation matrix ${}^{L}T_{C}$ between the two frames is estimated during an initial calibration phase.

6.3.1 Solution overview

The proposed scheme is sketched in Figure 6.3. A first group of processing activities operates in the local frame, where it seeks to detect and track the path-finder. Such activities are based on two distinct and converging flows of information. The first flow (Vision-based detection and recognition) comes from the RGB-D camera and allows us to identify and track the path-finder position within the image space. The second comes from the LIDAR sensor and looks for the same information from a different source with three different purposes. The first purpose is to increase the robustness of the vision based tracking by injecting the LIDAR data into the recognition activity. The second is to improve the accuracy of the estimation by fusing depth and visual information. The third is to allow the system to track the path-finder even when s/he falls out of the RGB-D sensor visual cone. A second group of processing activities takes place in the global reference frame. The main objective is to transform the time sequence of the



(a)



(b)

Figure 6.4: (a) Robot sensing system setup, consisting of LIDAR sensor[10] (1), RealSense D435[11] (2) for the path-finder detection and tracking, and RealSense T265[12] (3) for the visual odometry. (b) Representation of 2D data measured by the robot (red), with the raw laser scanned cloud points (thin grey dots) and their corresponding object centroids (thick blue points) along with the IDs (number in the figure), expressed in the LIDAR reference system $\langle L_k \rangle$ at time t_k. The human position (green shape with ID 0) can be retrieved with the sensor fusion.

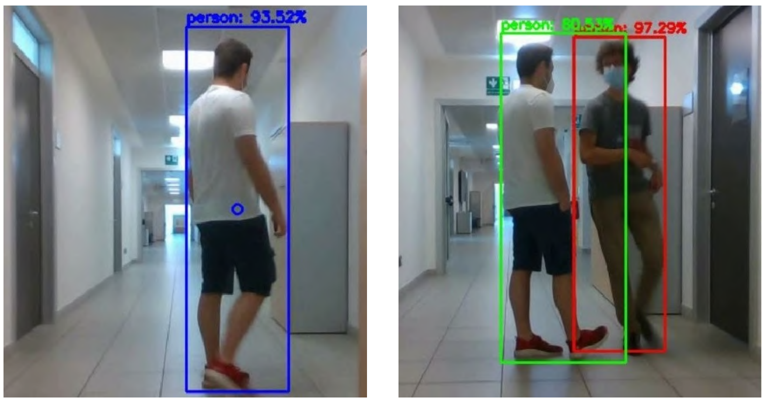
path-finder's positions into a set of smooth geometric motion primitives in order to have them followed by the robot.

6.3.2 Vision-based detection and recognition

The vision based algorithm takes the lion share in our tracking solution, but, as we mentioned above and as explained in Section 6.3.3, the tracking performance and robustness is significantly improved by the integration of the LIDAR data. In the next paragraphs we will first describe the different components and then discuss how they interoperate in an integrated pipeline.

Human detection. The first activity of the detection and recognition algorithm is to localise D people inside an image frame. To this end, we have chosen the latest version of YOLO available, YOLOv3 (Redmon and Farhadi, 2018b), and a lighter implementation of SDD, namely MobileNet (Howard et al., 2017) (designed to execute on low power devices). The detection module identifies the objects in view through the smallest bounding box that contains the required element. In the starting phase, the person associated with the largest bounding box is recognised as the path-finder.

Path-finder recognition. This module is used to understand which of the humans found in the frame corresponds to the path-finder, thus also enabling a coherent connection between the detector (based on a KNN classifier) and the tracker. The KNN classifier uses the vector points generated by two DNN image classifiers: ResNet50 (He et al., 2016), which produces a representation point in 2048 dimensions, and GoogLeNet (Szegedy et al., 2015), which produces a representation point in 1024 dimensions. If the path-finder is contained in the list of the D people detected, the information is passed to the image tracker, otherwise the procedure loops back to the detection phase. As previously mentioned, this phase also exploits the data coming from the sensor fusion phase in order to improve the tracking performance. This important feature is described in detail below. If the estimation error of the global



(a) (b)

Figure 6.5: (a) Initialisation phase: the detected path-finder is depicted with a blue rectangle. (b) Human following phase: the path-finder is correctly recognised (green rectangle), while another person is a negative sample (red rectangle).

human tracking (introduced in Section 6.3.4) exceeds the desired path-finder tracking uncertainty due to repetitive sensor or detection failures, the system reaches a faulty condition, the robot stops and the process restarts from scratch.

Path-finder tracking. This module is periodically executed to track the path-finder location. The pathfinder is sought with a number of frames chosen as a fixed parameter m in order to avoid the problems associated with long-term sequences. We implemented for the tracker the method that best fitted our requirements, i.e. KCF, CSRT and MOSSE, and we finally adopted the KCF in our experiments, since it offers a good compromise between robustness and speed. We emphasise that if a single detection fails or the path-finder is not found, the tracker cannot be started.

The vision pipeline as a whole. The system operates in two phases: *initialisation* and *human following*. The initialisation phase is needed because our vision processing pipeline leverages a learning-based classification of the human pathfinder, which in turn requires the knowledge of her/his features. During this phase, which lasts for Δ_t seconds, the robot collects a series of bounding boxes used to create the set of positive representative points into the N-dimensional space of the KNN. Simultaneously, a negative sample is randomly picked up from a database and it is also given to the KNN to balance the number of positive and negative samples. The negative samples come from our customised version of the Market1501 dataset (Zheng et al., 2015). An example of the initialisation phase is shown in Figure 6.5-a.

When the system switches to the *human following* phase, it carries out a first detection step. Then, it uses a KNN classifier in order to distinguish between positive detections and negative ones. In order to make the classification robust, the system uses the information from the path-finder position estimated at the previous time step (see Section 6.3.3). If the Euclidean distance between the last estimated position and the 3D position measured by the camera is lower than a threshold d_f, the detected position is considered as valid and the new set of visual features is added to the positive dataset. Detections without the described distance correspondence can be added to the negative dataset. This simple feedback mechanism significantly improves the system reliability and its resilience to wrong classifications or changes in the path-finder's appearance. Finally, the path-finder tracking module is executed, and the

Algorithm 3 Laser scanner points clustering algorithm

```
Input: $\mathbf{p}_i = (x_i, y_i)$, $i = 1, \dots, N$ // Get measurement points
 1: $j = 1$, $\mathbf{c}_j \leftarrow$ Append($\mathbf{p}_1$) // Initialisation first object
 2: for $i = 2$ to $N$ do
 3:     if $||\mathbf{p}_i - \mathbf{p}_{i-1}||_2 < d_c$ then
 4:         $\mathbf{c}_j \leftarrow$ Append($\mathbf{p}_i$) // Add $\mathbf{p}_i$ to the set $\mathbf{c}_j$
 5:     else
 6:         if $\dim(\mathbf{c}_j) > m_c$ then
 7:             $\mathbf{o}_j = \sum_{k=1}^{\dim(\mathbf{c}_j)} \mathbf{c}_j(k)/\dim(\mathbf{c}_j)$ // Object centroid
 8:         end if
 9:         $j \leftarrow j + 1$, $\mathbf{c}_j \leftarrow$ Append($\mathbf{p}_i$) // Start new object
10:     end if
11: end for
Output: $\mathcal{O} = \{\mathbf{o}_1(t_k), \dots, \mathbf{o}_j(t_k)\}$
```

resulting information is passed to the fusion module. After the successive m frames, the detection and recognition stages are re-executed.

This processing workflow and the different feedback cycles it is based on delivers a highly performant image processing and an improved robot localisation accuracy, as shown by the experimental data in Section 6.5.

6.3.3 Local LIDAR-camera sensor fusion

The information coming from the vision-based algorithm are combined with the LIDAR in the local reference frame $\langle L_k \rangle$ in order to make the procedure more robust, as aforementioned. Moreover, the path-finder can be tracked for some time also when s/he evades the vision cone of the RGB-D camera just relying on the LIDAR information.

LIDAR clustering. The sensor reading delivered by the laser scanner at time t_k provides a sequence of N_k measurement points in the form of $\mathcal{P}_k = \{\mathbf{p}_1, \dots, \mathbf{p}_{N_k}\}$, represented in polar coordinates as $\mathbf{p}_i = (r_i, \alpha_i)$, i.e. the range r_i and the angle α_i expressed in the planar LIDAR reference frame $\langle L_k \rangle$ (see Figure 6.4-b for an example of an actual scan). At time t_k, the measured points are filtered and grouped into M_k clusters based on the mutual Euclidean distances and on the richness, i.e. on a minimum number of sensed points for each cluster, each identified by the object centroid $\mathbf{o}_j(t_k) = [x_j(t_k), y_j(t_k), 0]^T$, $j = 1, \dots, M_k$ (see the ID numbers in Figure 6.4-b and the Algorithm 3). Given two sets of objects $\mathcal{O}_k = \{\mathbf{o}_1(t_k), \dots, \mathbf{o}_{M_k}(t_k)\}$ and $\mathcal{O}_{k+1}$, taken in two consecutive time instants t_k and t_{k+1} and possibly having $M_k \neq M_{k+1}$, we adopt the Munkres assignment algorithm (Munkres, 1957), which gives us more robust matches than obtainable from the Algorithm 1, to decide either if the two objects are actually the same or if a new object has been detected. To this end, since between t_k and t_{k+1} the robot moves for δ_t seconds according to the model (6.1), we can update its position $\mathbf{s}(t_{k+1})$ in $\langle L_{k+1} \rangle$ either by using ego-motion data or, if available, the global localisation module. After the motion, $\mathcal{O}_k$ previously expressed in $\langle L_k \rangle$, is projected in new local frame $\langle L_{k+1} \rangle$. The presence of LIDAR measurement noise imposes the use of a forgetting factor, hence the algorithm removes the object whose centroid does not find correspondences for T_{fd} time instants. This way we can disambiguate the different entities in the robot's surroundings, increase robustness to partial occlusions and exploit indeed a prior-based tracking (since each cluster has its own unique ID, as depicted in Figure 6.4-b).

Algorithm 4 Sensor fusion algorithm

```
Input: $\mathcal{O}(t_{k-1})$ in $\langle L_{k-1}\rangle$, $\mathcal{O}'(t_k)$ in $\langle L_k\rangle$, $\mathbf{c}_i(t_k)$, $\mathbf{s}(t_k)$
 1: $\mathcal{O}(t_{k-1})$ in $\langle L_k\rangle \leftarrow$ RotoTranslate $(\mathcal{O}(t_{k-1}), \mathbf{s}(t_k))$
 2: $\mathcal{O}(t_k) \leftarrow$ Munkres $(\mathcal{O}(t_{k-1}), \mathcal{O}'(t_k))$
 3: if State == $Init$ then
 4:     $\mathbf{d}_j = ||\mathbf{o}_j(t_k) - \mathbf{c}_i(t_k)||_2, \forall j = 1, \dots, M_k$
 5:     $j^* = \arg\min(\mathbf{d})$
 6:     if $d_{j^*} < d_\text{p}$ then
 7:         $pm_{\mathbf{o}_j^*} \leftarrow pm_{\mathbf{o}_j^*} + 1$ // Number of matches
 8:     end if
 9:     if $\exists\, \mathbf{o}_j \in \mathcal{O} \mid pm_{\mathbf{o}_j} > m_\text{p}$ then
10:         $\mathbf{o}_j \leftarrow \mathbf{o}_j^\star$
11:         $\mathbf{o}_k \leftarrow \mathbf{o}_j^\text{ob}, \forall k \neq j$
12:         State $\leftarrow Track$
13:     end if
14: end if
15: if State == $Track$ then
16:     $d = ||\mathbf{o}_j^\star(t_k) - \mathbf{c}_i(t_k)||_2$
17:     if $d > d_\text{p}$ then
18:         $fm_{\mathbf{o}_j^*} \leftarrow fm_{\mathbf{o}_j^*} + 1$ // Number of mismatches
19:     end if
20:     if $fm_{\mathbf{o}_j^\star} > m_\text{r}$ then
21:         $\mathbf{o}_j \leftarrow \mathbf{o}_j^\text{un}, \forall j = 1, \dots, M_k$
22:         State $\leftarrow Init$
23:     end if
24: end if
Output: $(x(t_k), y(t_k))$ in $\langle L_k\rangle$
```

Camera 3D position. The tracking module described in Section 6.3.2 returns a bounding box $[x, y, w, h]$ in the image frame, containing the (x, y) pixel coordinates of the top-left corner of the box, its width w and height h, which is then converted in the $\langle C\rangle = \{X_c, Y_c, Z_c\}$ pin-hole camera reference system. Notice that the depth information along the Z_c axis is retrieved via the RealSense™API. As a consequence, the centroid of the i-th bounding box $\mathbf{c}_i(t_k) = [x_c(t_k), y_c(t_k), z_c(t_k)]^T$ can be expressed in the camera reference system $\langle C_k\rangle$ at time t_k.

Sensor fusion. Given the set of objects $\mathcal{O}_k$ and the centroid(s) of the bounding box(es) $\mathbf{c}_i(t_k)$, taken at the same time instant t_k, we adopt a spatio-temporal correspondence algorithm with the two sets of measurements to decide if the tracked object is the same or a new one has entered into the scene. Our algorithm is implemented as a finite state machine, comprising the *Init* and *Track* states (see Algorithm 4). The rationale is the following: in the *Init* state, we look for a correct match between the j-th clustered object $\mathbf{o}_j(t_k)$ and the i-th bounding box centroid $\mathbf{c}_i(t_k)$, which occurs when their Euclidean distance in the local LIDAR frame $\langle L_k\rangle$ is below a threshold d_p (this is obtained in $\langle L_0\rangle$ at the end of the *initialisation phase*, where the path-finder stands in front of the robot for the initial bootstrap). When the correct match is found with the same j-th object $\mathbf{o}_j$ for m_p time instants, the j-th object is "promoted" as PATH-FINDER $\mathbf{o}_j^\star(t_k)$, while all the other objects are labelled as OBSTACLE, whereupon the state changes to *Track*. Notice that this procedure reduces at the same time the computation times and the probability of mismatch, while making the algorithm robust to sensor failures (either the bounding box or the LIDAR cluster are sufficient for recognition). In the *Track* state, at time t_{k+1}, the match is evaluated for the $\mathbf{o}_i^\star(t_{k+1})$ only, as all the possible new objects in $\mathcal{O}_{k+1}$, by default labelled as UNKNOWN, become OBSTACLE objects. When a mismatch between the $\mathbf{o}_j^\star(t_{k+1})$ and $c_i(t_{k+1})$ occurs for m_r time instants, the track is rejected, i.e. all objects become UNKNOWN again, and the algorithm switches back to the *Init* state. Possible mismatch events happen if the Euclidean distance in $\langle L_{k+1}\rangle$ is higher than d_p or the object $\mathbf{o}_j^\star(t_{k+1})$ itself is removed during the clustering association. Finally, the Cartesian coordinates of the path-finder $(x(t_{k+1}), y(t_{k+1}))$ in the local frame $\langle L_{k+1}\rangle$ are propagated back to the people recognition module (see Section 6.3.2) to strengthen the human tracking, since such information form a prior for the next path-finder detection.

6.3.4 Global human tracking

Since the human is used as a path-finder for future executions of the path, her/his position should be estimated in the global reference frame $\langle W\rangle$. To this end, we first need to estimate the robot position $s(t_k)$ in $\langle W\rangle$. This is accomplished fusing together the encoder readings, the visual odometry and the LIDAR points $p_i(t_k)$ with an a-priori map of the environment (if available) or solving a SLAM problem. The $s(t_k)$ robot position and the path-finder local measurements in $\langle L_k\rangle$ are used to obtain the Cartesian coordinates $(x(t_k), y(t_k))$ of the path-finder in $\langle W\rangle$.

To track the human in $\langle W\rangle$, an estimation algorithm is needed, whose main role is to further improve the accuracy of the reconstructed path and to further increase the robustness to occasional sensor failures. In order to limit the computational cost, we make the assumption that the path-finder moves with a velocity following a Gaussian probability density function. This random walk hypothesis is quite standard in the literature and derives from the lack of knowledge about the actual human motion intentions. What is more, following the observation that humans actually move with a smooth dynamic (Arechavaleta et al., 2008), the motion model can be approximated by a unicycle dynamic (Farina et al., 2017b). Hence, we explicitly express the angular and linear velocities as states, and by denoting with $\bar{\mathbf{h}}(t_k) = [x(t_k), y(t_k), \theta(t_k), v(t_k), \omega(t_k)]^T$ the state at time t_k (where $v(t_k)$ and $\omega(t_k)$ are the forward and angular velocities, respectively), we have the following model

$$\begin{aligned}\bar{\mathbf{h}}(t_{k+1}) &= \begin{bmatrix} x(t_k) + \delta_t v(t_k)\cos(\theta(t_k)) \\ y(t_k) + \delta_t v(t_k)\sin(\theta(t_k)) \\ \theta(t_k) + \delta_t \omega(t_k) \\ v(t_k) \\ \omega(t_k) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \delta_t & 0 \\ 0 & \delta_t \end{bmatrix} \begin{bmatrix} \eta_a(t_k) \\ \eta_\omega(t_k) \end{bmatrix} = \\ &= f(\bar{\mathbf{h}}(t_k)) + B\eta(t_k),\end{aligned} \tag{6.2}$$

where $\eta(t_k)$ is the acceleration noise affecting the linear and the angular velocities that is assumed to be $\eta(t_k) \sim \mathcal{N}(0, E)$, with E being its covariance matrix, and white (as customary).

6.4 Navigation

The aims of the navigation module are twofold: reconstructing the path followed by the path-finder in a form that can be followed by the robot, controlling the motion in order for the robot to follow the path with a good accuracy (small deviations are inevitable but they should be kept in check).

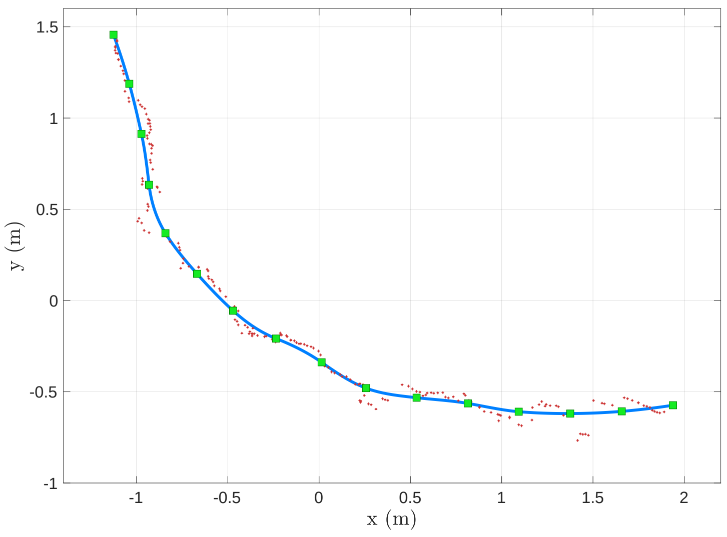


Figure 6.6: Example of path fitting and reconstruction. The red stars represent the input data. The green squares are the fitted waypoints, sampled at a uniform distance along the path. The blue solid line is the reconstructed, smoothed path, to be followed by the robot.

6.4.1 Path reconstruction

As shown in the scheme in Figure 6.3, the path reconstruction module continuously receives new information on the current position of the path-finder from the perception module. This way, it creates a dataset composed of a time series of 2D path-finder positions, which are updated in real–time. The module executes a local path fitting of the estimated path-finder trajectory. An example execution of the process is shown in Figure 6.6. The path is reconstructed using the following steps.

1. Once a new path-finder position is received, it is compared with the previous one, and, if the Euclidean distance is greater than a small threshold value, it is recorded into the dataset. This action is necessary to handle the scenario where the path-finder stops for a long time, in order to avoid an unnecessary growth of the dataset.

2. When the new position qualifies for its inclusion into the dataset, the x and y components of the data points are fitted using a classical smoothing algorithm, i.e. the LOESS (Locally Estimated Scatterplot Smoothing) (Cleveland, 1979).

3. The fitted data points are then connected by a polyline, and a number of waypoints are sampled at a uniform curvilinear distance (corresponding to the green squares of Figure 6.6).

4. The waypoints are connected by a G2 clothoid spline (corresponding to the solid blue line of Figure 6.6), using the algorithms and techniques discussed in Bertolazzi and Frego, 2018b; Bertolazzi and Frego, 2018a, and for which an efficient C++ implementation is available (Bertolazzi et al., 2018).

The choice of the clothoid comes from the observation that humans tend to follow the unicycle-like dynamics (Farina et al., 2017b) given in (6.2), which naturally generates clothoid curves. What is more, due to the continuity of the curves and of their curvature, clothoids have been proved to be effective to mimic a human path by a robotic agent (Bevilacqua et al., 2018a).

6.4.2 Robot control

When a path is reconstructed, following the steps described above, the controller module takes the responsibility to execute a safe navigation of the robot following as closely as possible the prescribed path. For this work, we employed the path following algorithm described in Andreetto et al., 2017, which is velocity-independent and avoids the singularities presented by other common algorithms when the vehicle has to stop and the velocity is set to zero. The velocity of the robot is chosen by our controller based both on the distance from the end of the path (corresponding to the path-finder position), with the aim of following the path-finder at a constant (curvilinear) distance, on the current path curvature (the vehicle is slowed down when traveling a sharp curve), and on the past robot velocities (to limit the maximum allowed accelerations). Furthermore, the control module implements a safety policy whereby, when an obstacle is encountered along the path, the robot first slows down and then stops if the path remains occluded (notice that, since the path-finder have already travelled that area, the obstacle is necessarily dynamic, e.g. a human being, hence it is expected to pass by in a short time).

6.5 Experimental results

In this section, we first present our experimental evaluation to decide the most effective combination of solutions for vision-based detection and recognition. Then, we will report and discuss the performance of our system as a whole using a real robotic platform.

6.5.1 Vision module

We have organised the analysis of the vision module along three directions of prominent relevance for the application at hand: detection, recognition and tracking.

Detection. The comparison among the different possibilities aims to rate the computational efficiency (measured in FPS) and the algorithm precision (measured as mean Average Precision - mAP) from the data reported in the previously cited (Liu et al., 2016; Redmon et al., 2016b; He et al., 2017; Redmon and Farhadi, 2018b; Cao et al., 2019), and shown in Table 6.1. By observing the data, it appears that the single-stage algorithms (SSD and YOLO) are much faster with respect to the two-stage methods (R-CNN): they execute around 5 to 10 times faster than R-CNN. Instead, the precision of the three detectors is almost the same. Also from our experimental evaluation, SSD and YOLO perform better than R-CNN and OpenPose both in computational efficiency and algorithm precision. To visually show an example, we have applied all these algorithms on the same picture. From the results presented in Figure 6.7 follows that all the solutions recognise the five people in the foreground, but with a different level of performance. Mask R-CNN is the most accurate algorithm in our example, and is able to recognise four people with a precision of 99% and the last one with 92%. YOLO detects even a sixth person in the background that none of the others have recognised. SSD has troubles with the player on the left, detected with a confidence percentage of only 32%. Our final choice fell on SSD, since it implements the CNN with a relatively small number of parameters: this ensures low execution time at the price of a slight detection inaccuracy (which is however compensated by the LIDAR data fusion, as discussed in Section 6.3).

	YOLO				R-CNN			
	v1	v2	SSD	R-CNN	Fast	Faster	Mask	OpenPose
FPS	45	67	46	0.05	0.5	7	7	10
mAP	66	**76**	**74**	53	70	**73**	-	-

Table 6.1: Comparison of mAP and FPS for the object detector algorithms (Liu et al., 2016; Redmon et al., 2016b; He et al., 2017; Redmon and Farhadi, 2018b; Cao et al., 2019). In bold the mAP of the three state of the art detectors. The data are based on the Pascal VOC 2007 dataset (Everingham et al., 2007)

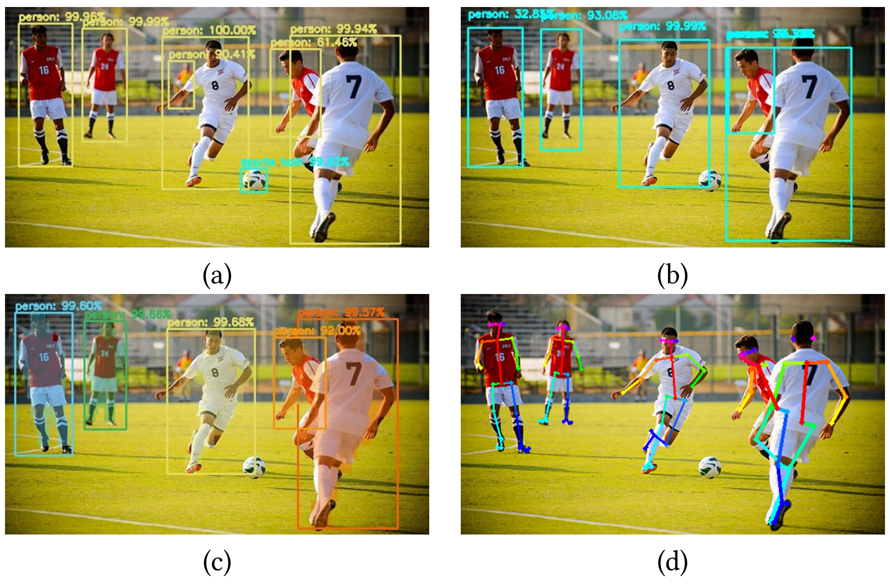


Figure 6.7: An example of application of YOLO (a), SSD (b), Mask R-CNN (c) and OpenPose (d) applied on the same image.

Recognition. The first tests for the recognition module considered the feature point matching on the PRID450 (Person Re-IDentification) dataset (Roth et al., 2014), that contains thousands of cropped images of people walking outdoors. The dataset is constructed with multiple shots of the same person in different moments and perspectives. The results reported in Figure 6.8 show the main problems of this techniques with human shapes. Humans present a highly deformable-body, with a surface (clothes) that continuously changes its aspect. However, the key points matching is designed for a pattern that is repeated often and clearly, as a consequence, the matching performance is not reliable at all. For example, even with the same subject with almost the same position (bottom-left couple of images in Figure 6.8), the key points matching fails with most of the points. The exception is the bottom-right image that has a perfect matching, but the two pictures examined are exactly the same.

Instead, the proposed recognition module based on KNN has been also tested on the Market1501 dataset (Zheng et al., 2015) in conjunction with either ResNet50 or GoogLeNet. Similarly to the PRID450 dataset, the Market1501 contains sets of images of hundreds of people captured from different perspectives and in different moments. As an example, we selected two small datasets with 11 and 2 people respectively, with and 9 images per person, then we “trained” the KNN (i.e., we stored the data) with the representative points extracted from the images, and retrieved the most similar people. In Figure 6.9, we report an example of the queries computed on the KNN classifier for the two chosen detectors. In the test elaborated with the first dataset, the classifier produced approximatively 50% of correct responses

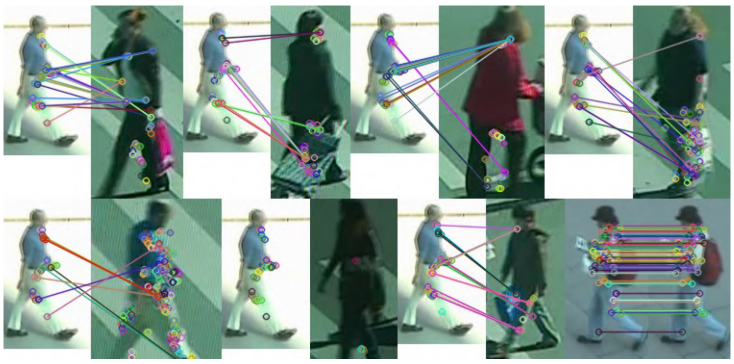


Figure 6.8: Some example of matching samples with the key points matching on the PRID450 dataset. There are multiple failures: people who present no key points, objects such as bags that have plenty of features, matches that connect completely different parts of the body, like shoulders with legs.

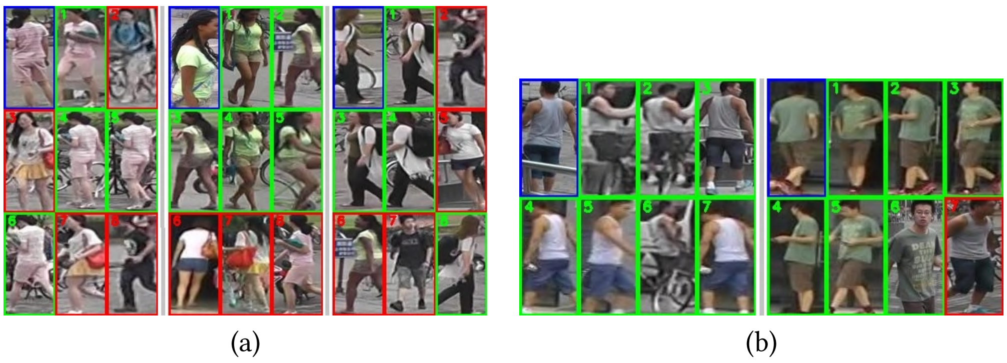


Figure 6.9: KNN applied to images elaborated with 11 real people and ResNet50 (a) and with 2 real people and GoogLeNet (b), with 9 images of each person. The query (images with the blue contour) is used to extract from the database the most similar pre-analysed images. The green contour depicts the correct extracted person, whereas the red contour highlights wrongly extracted persons.

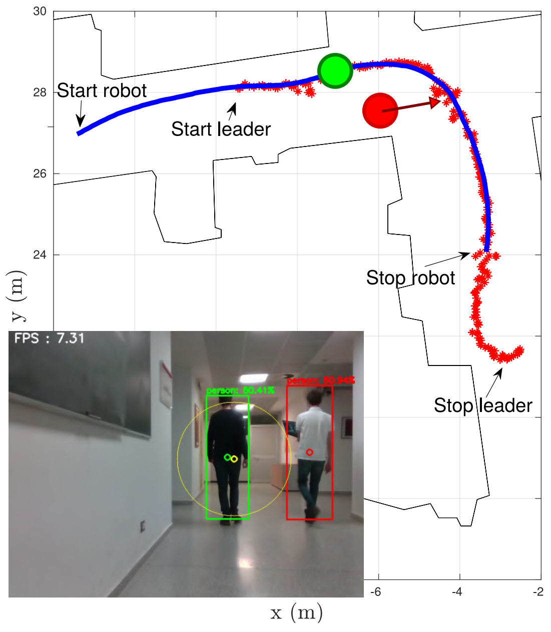
for each person (Figure 6.9-a). Instead, with 2 real human beings and 9 images per person, the KNN obtained only one false positive over 14 pictures. This result shows that the proposed KNN solution is appropriate for the application at hand (we are interested in only 2 classes, i.e., the path-finder and the other people). Similar results were obtained with ResNet50 and GoogLeNet, and we selected the second to work with the KNN since it achieved moderately faster computation times.

Tracking. For the image tracker, our aim is usually to process long real-time sequence with occasional total occlusions and changes of shape. Instead, since the complete camera pipeline is solved with a combination of detection, recognition, and tracking, the internal tracking task is simpler, and deals with changes of shape and partial occlusions. The requirement is to solve the task, and to deal with changes of shape and partial occlusions. The methods presented in Section 6.2.2 were evaluated in terms of execution time (for real-time implementation) and tracking performance:

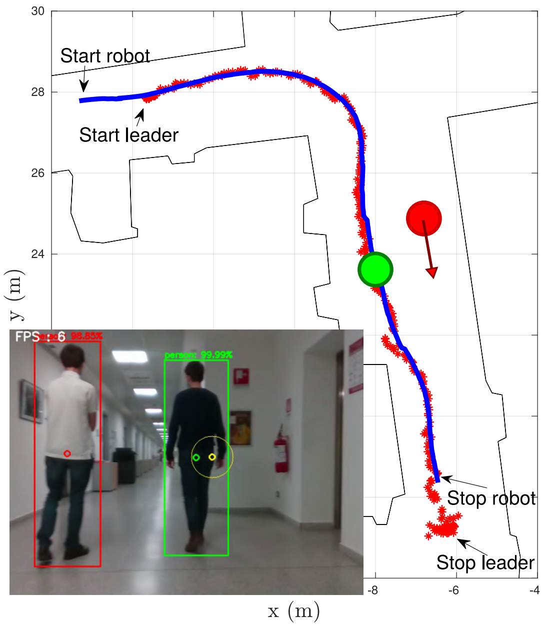
- MIL tracker is not a valid choice because it runs at few FPS and the newer version, namely the KCF, outperforms both its speed and accuracy.
- KCF tracker is the new version of MIL and it is one of the fastest and most reliable methods. However, it suffers the rapid change of appearance and, less relevant for our scenario, it does not manage total occlusions.

	MIL	KCF	MedFlow	CSRT	MOSSE	GOTURN	TLD
FPS	9	38	40	15	56	20	10

Table 6.2: Overview of the frame-per-second (FPS) rate of the image tracking algorithms. The performance was measured on an Nvidia Jetson TX2[8] GPU.



(a)



(b)

Figure 6.10: Experimental trajectories in a hallway. The path-finder and a pedestrian walk in the same corridor with (a) partially occluding trajectories (EXP-01) and (b) missing and recovering of the path-finder with the camera tracking (EXP-02). Solid blue lines depict the trajectories followed by the robot, while the red stars mark the measured path-finder positions. The green and red circles correspond to the positions of the path-finder and the other pedestrian, respectively, when the inlet camera snapshots are grabbed.

- MedFlow tracker works well only on rigid objects, hence it is not suitable human interaction applications.
- CSRT is one of the most reliable methods, and manage short-term total occlusions, despite the low FPS rate.
- MOSSE is focused on pure speed (i.e., very high FPS rate applications), but it is not very robust.
- GOTURN shows a good trade-off between reliability and speed.
- TLD is able to solve long total occlusions but easily fails on simpler scenarios by returning a lot of false positives, hence it does not fit our purpose.

Based on our results (shown in Table 6.2), the best trade-off choices from the application at hand were obtained with the KCF, CSRT, MOSSE, GOTURN trackers, and based on empirical evidence, we selected CSRT as the most suitable solution for our purposes.

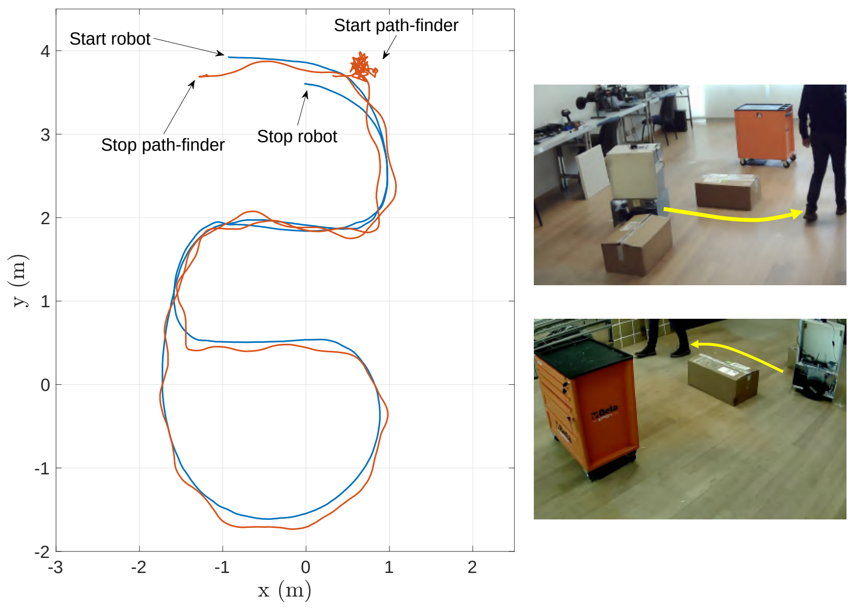


Figure 6.11: An example of the actual path-finder path (red line) and robot trajectory (blue line) for the maze-like environment of EXP-03.

6.5.2 Experiments with the mobile robot

The algorithms presented in Section 6.3 and Section 6.4 were executed on a Jetson TX2[8] (4-bit Nvidia Denver and Arm Cortex-A57 CPUs; 8GB LPPDR4; 256-core Nvidia® Pascal GPU) for the acquisition of the RGB-D data and the classification, while the LIDAR scans, the sensing data fusion, and the navigation control were executed on a Intel NUC[13] computing platform (Intel® Core™ i7-7567U with 2 cores, 16GB DDR4), both on board of the wheeled robot entirely assembled at the University of Trento (see Figure 6.4-a). All the reported experiments were carried out in our department at the University of Trento.

Reliability and robustness. In a first set of experiments, we aimed at testing the performance and robustness of the path-finder tracking algorithm. To this end, we recorded the real–time execution data in two different areas of an hallway with multiple exits and in different conditions. In Figure 6.10-a, the robot follows the path-finder while another pedestrian is walking nearby (EXP-01). The algorithm successfully rejected the disturbing effect of the second pedestrian and it correctly followed the path-finder. Similar results (EXP-02) were obtained for crossing trajectories or when the path-finder exits from the camera field of view for the right turn. Even in those cases, the other pedestrian is not wrongly classified as the path-finder, who is instead correctly tracked back after the turn (see Figure 6.10-b).

Accuracy. For a qualitative analysis of the tracking and navigation performance, we present in Figure 6.11 the comparison between the robot trajectory (blue line) with the actual position of the path-finder (red line). Both trajectories were captured with a network of eight OptiTrack cameras for ground truth reference in a maze-like environment (EXP-03). Notice the path-finder starting position standing in front of the robot during the bootstrap phase. The swinging path-finder trajectory is dictated by the OptiTrack tracked markers placed on the head of the human to avoid occlusions, hence oscillating with the footsteps. From this picture it is evident that, in sharp turns, the robot looses the image tracking

[13]https://ark.intel.com/content/www/us/en/ark/products/95065/intel-nuc-kit-nuc7i7bnh.html

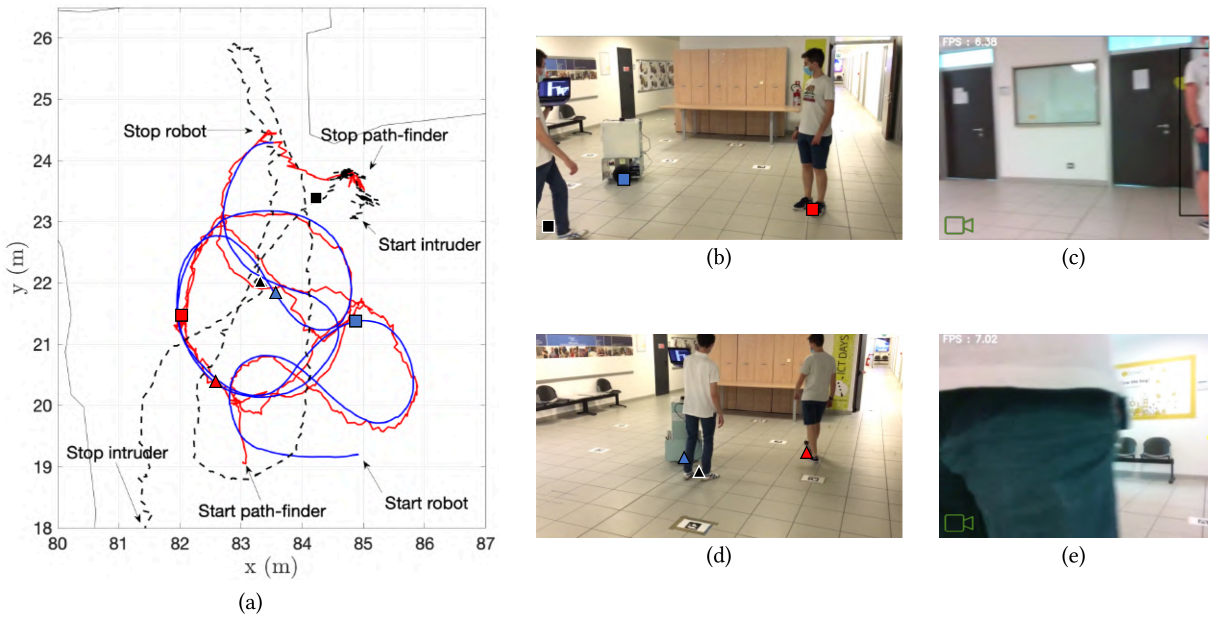
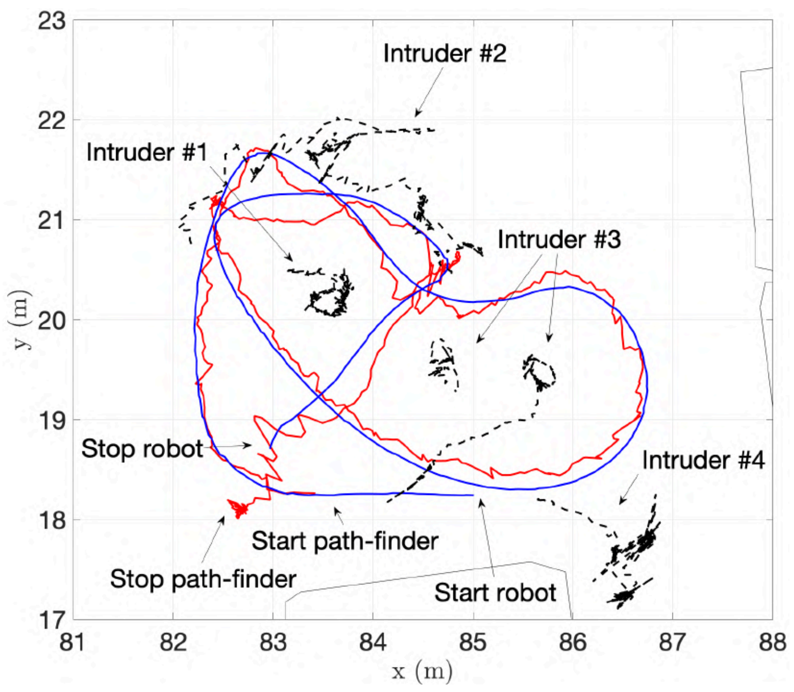


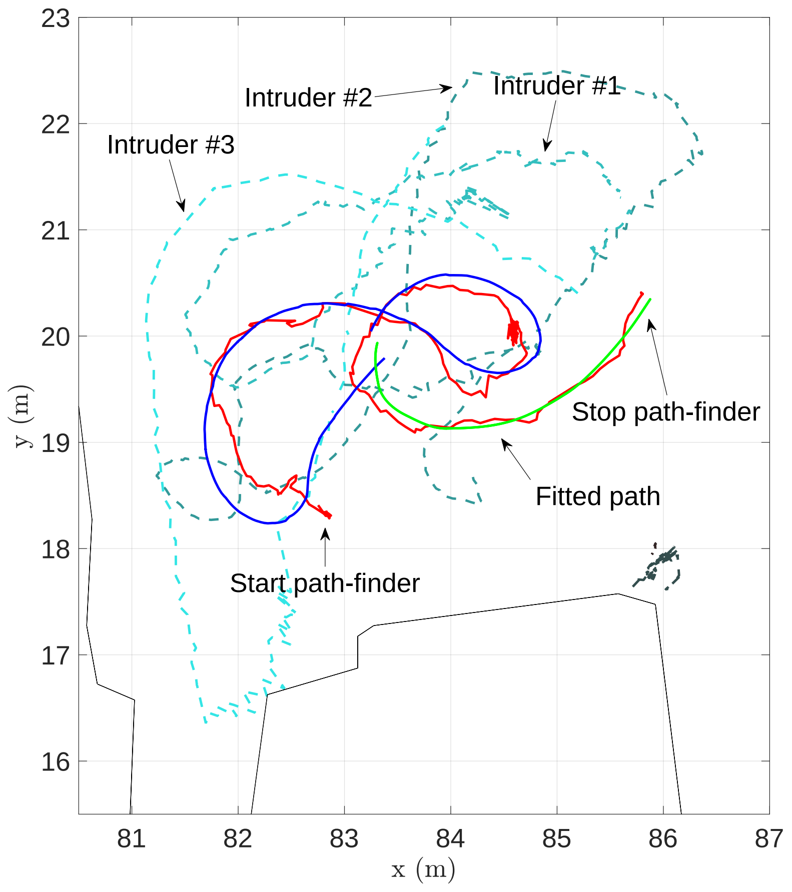
Figure 6.12: (a) Trajectory travelled by the robot in EXP-04 (blue line) and the measured path-finder path (red line), with an additional person acting as dynamic obstacle (black dashed line). The coloured squares represent the positions of robot, path-finder, and intruder when the snapshots (b,c) where grabbed, while the coloured triangles correspond to the snapshots (d,e). The pictures (c) and (e) show the robot's perspective.

of the path-finder, but it is nonetheless able to exactly follow the path by means of the fusion with the LIDAR data. Finally, we would like to point out that the error in the trajectory followed by the robot with respect to the human footsteps is in the range of ± 25 cm, i.e. the typical encumbrance of the human body.

Robustness and accuracy for extreme manoeuvres hindered by an intruder. In Figure 6.12-a we report the experimental results in an hallway test (EXP-04), where another human (i.e., a dynamic obstacle) moves in the scene. The path-finder was instructed to follow a winding path, which is a challenging scenario since the human necessarily and continuously exits from the field of view of the camera along the sharp turns. An additional element of complexity was given by the intruder (black dashed line) repeated interference with the robot operations. As visible in the plot, the path travelled by the robot (blue line) remained consistently aligned with the measured path-finder positions (red line). This experiments also shows the effectiveness of the LIDAR clustering and association algorithm presented in Section 6.3.3, which tracks the intruder and then rejects it for the evident inconsistencies with the visual data. Indeed, from Figure 6.12-c we notice that the path-finder is essentially outside the field of view of the camera, but the path to be followed is correctly tracked. The same proof of robustness can be seen in the snapshots of Figure 6.12-d,e where the camera is occluded by the intruder but the track is not lost.



(a)



(b)

Figure 6.13: (a) Tracking experiments with the path-finder (red line) and other persons in a crowded environment. In EXP-05 (a) multiple intruders (black dashed lines) are predominantly acting as bystanders, while in EXP-06 (b) they follow random paths (coloured dashed lines). The green line depicts the fitted path which will be followed by the robot.

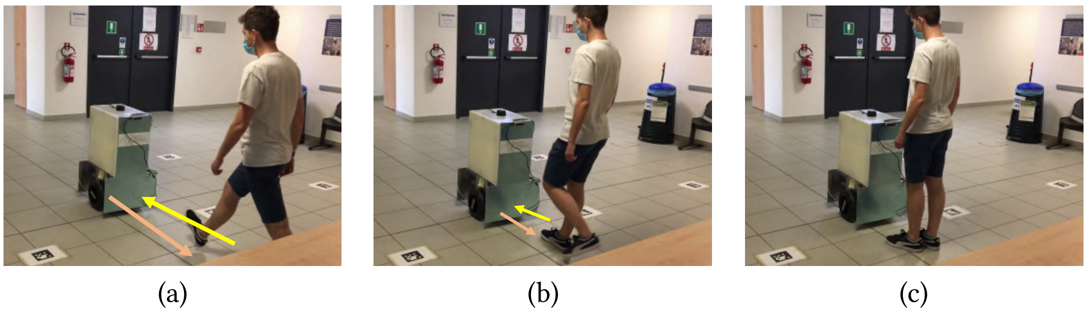


Figure 6.14: An example of the safety braking (EXP-07). The robot and a human walk towards each other (a), and reach a minimum distance (b) where the robot stops as not to collide with the person (c).

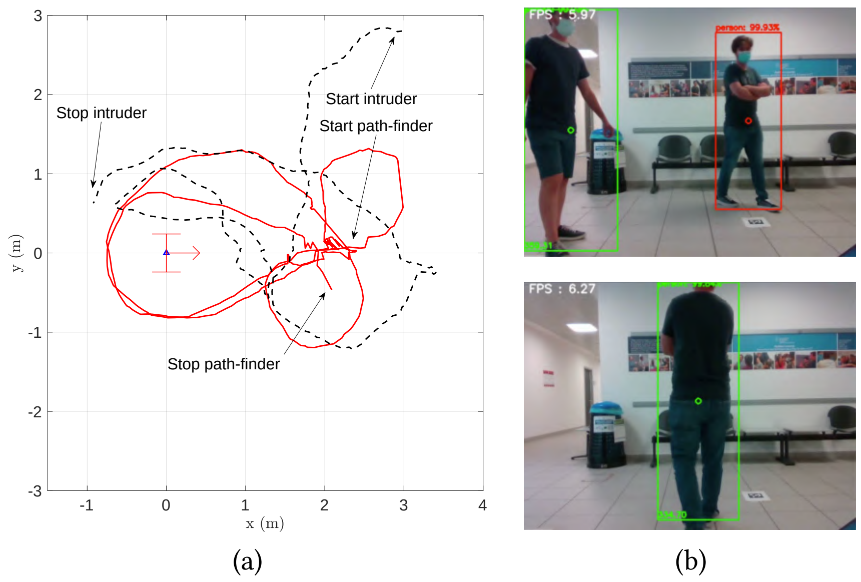


Figure 6.15: (a) Tracking experiment with the path-finder (red line) and an intruder (black dashed line) with highly similar visual features for EXP-08, expressed in $\langle L \rangle$. (b) Screenshot of a positive match of the path-finder (top) and a wrong classification of the intruder (bottom), which is recovered by the sensor fusion.

Crowded conditions. In a crowded scenario, the robot follows its path-finder along a winding path, meeting several other people in the same hallway. In the first case (EXP-05), the intruders made limited movements (quasi static condition), while the robot navigates between them. As reported in Figure 6.13-a the resulting robot trajectory is consistent with that of the path-finder.

In a more challenging case (EXP-06), also the intruders performed random trajectories, repeatedly obstructing the robot and occluding the path-finder but without compromising the correct robot path following (see Figure 6.13-b). Also in these two presented experiments, we note that our algorithm can track, together with the path-finder, all the nearby intruders. Our solution, thus, lends itself to a possible extension in which multiple path-finders are present.

Discerning the similarity between the path-finder and the intruder. Moreover, we tested the system safety (EXP-07, depicted in Figure 6.14) and the tracking performance of our system with two people sharing highly similar visual features, and reported the results expressed in the robot local reference frame $\langle L \rangle$ (EXP-08 of Figure 6.15-a). The screenshots of the camera tracking (Figure 6.15-b)

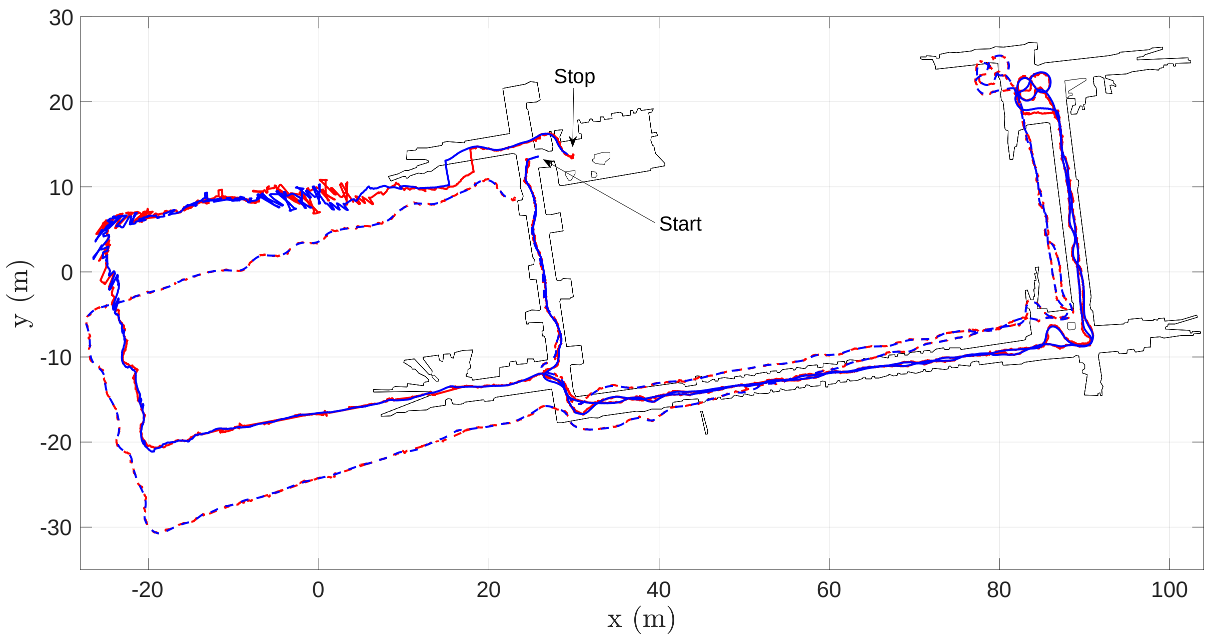


Figure 6.16: Trajectory travelled by the robot in (blue lines) and the measured path-finder path (red lines), in the global localisation frame $\langle W \rangle$ (solid lines) and in the relative localisation frame (dashed line). The known map is represented by the black line.

show moments in which the path-finder is correctly identified and the second person is classified as negative (top snapshot) and moments in which the intruder is misclassified as the path-finder (bottom). Nonetheless, even in presence of misclassification the sensor fusion with the LIDAR comes to rescue and the tracker correctly follows the path-finder (see the trajectories in Figure 6.15-a). The sequence in Figure 6.14 shows the path-finder in the worst condition of walking directly towards the robot. As we can see, the safety distance is correctly evaluated using the measurements obtained from the LIDAR, therefore the robot is able to stop before hitting any static or dynamic obstacle.

Navigation without the map. Finally, we tested the behaviour of the system in an a-priori unknown environment, thus renouncing to the availability of a global localisation system. Hence, in EXP-09, the path-finder and the robot positions are not expressed in the global frame $\langle W \rangle$ but in an arbitrary reference frame $\langle O \rangle$. Figure 6.16 reports the navigation task performed using only the odometry, along with the actual global localisation for comparison purposes and to overlay the odometry information on the environment map. The obtained trajectories (blue dashed line) show that odometry localisation trivially drifts; nonetheless, the actual path travelled by the robot (depicted with the blue solid line) accurately follows the path shown by the human (solid red line). This remarkable result shows how planning in an unknown environment can be effectively executed by the robot with the path-finder. Moreover, as can be observed in the left side of Figure 6.16, the navigation in the relative frame $\langle O \rangle$ turns to be more robust than in $\langle W \rangle$, which is useless when the robot is outside the a-priori known map (unless a SLAM solution is adopted).

Quantitative evaluation. In order to give quantitative evidence of the satisfaction of requirements **Q1** and **Q2** introduced in Section 6.1, we defined two metrics. Requirement **Q1** was measured by the percentage ratio $r_1\%$ between the camera frames where the path-finder is visible in the field of view of the camera over the total number of frames. We also report the ratio $r_2\%$ between the number of frames where the path-finder is correctly tracked over the number of frame where s/he is visible,

Experiment	In-frame ratio $r\%$ (**Q1**)		Curvilinear distance s (**Q2**)		
	visible/total $r_1\%$	correct/visible $r_2\%$	avg s	min s	max s
EXP-01	67.6%	83.7%	3.970	2.632	5.150
EXP-02	78.2%	92.6%	3.328	2.015	4.486
EXP-03	-	-	1.913	1.249	2.348
EXP-04	10.9%	68.6%	3.209	1.302	4.914
EXP-05	39.8%	79.9%	2.071	0.722	3.188
EXP-06	-	-	2.466	0.860	4.271
EXP-07	-	-	1.005	0.568	1.642
EXP-08	57.5%	86.0%, (62.8%)*	-	-	-

* $r_2\%$ = correct/tracking

Table 6.3: Summary of the two metrics for requirements **Q1** and **Q2** among the experiments reported in Section 6.5.2.

as an additional validating metric of the reliability of the vision module. The safety constraint **Q2** was evaluated by means of the curvilinear distance s between the path-finder and the robot along the clothoid curve. From the values of $r_1\%$ reported in Table 6.3 we can immediately notice that, as a major difference with respect to other visual servoing works presented in the literature, in our experiments the path-finder was not always detected by the camera. For instance, in EXP-01 and EXP-02, which took place in a long corridor, about 30% of the camera frames had no path-finder in sight, and yet the average recognition accuracy $r_2\%$ was fairly good. In EXP-04, the ratio $r_1\%$ drops to 10% due to the tortuous path taken by the path-finder. Despite this, the tracking was performed correctly (see Figure 6.12), and the recognition accuracy remained very good. In EXP-08, due to similarity between the two persons in the scene, a higher number of wrong detections occurred. For that reason, we also reported the ratio between the number of correct matches and the total number of detections. As could be expected, the ratio was lower than the one reported with the other experiments; however, we recall that with the feedback strategy of our pipeline the recognition module can overcome wrong matches of the path-finder (see Figure 6.15-a).

The results reported in Table 6.3 show that also the **Q2** requirement is satisfied in all the different scenario evaluated. In EXP-01 and EXP-02 the minimum distances are appropriately about 2 meters, and the path-finder is correctly followed even with distances above 5 m. The average curvilinear distance is maintained between 1 and 3 m, which ensures safety in addition to being widely recognised as the preferred spatial distance for social interactions (Rios-Martinez et al., 2015; Antonucci et al., 2019b), as also reviewed in Section 2.2. More dangerous values can be found for the EXP-07, since the experiment was specifically designed to trigger the robot emergency stop; nevertheless, the minimum distance reached is above 50 cm. The very high values reached by the maximum s distance in EXP-04, EXP-05 and EXP-06, i.e., experiments with tortuous trajectories, are due to the obstruction of the intruder persons on the robot path.

6.6 Comments

In this chapter, we have presented an approach for guiding a robot across a complex and congested environment. A human operator takes the role of a path-finder and the robot follows, moving in a close neighbourhood of the path physically marked by the human with her/his footsteps. This application requires a combination of state-of-the-art techniques for robust perception and path reconstruction. The experimental results show the high level of reliability and robustness reached by the proposed solution.

We envisage several interesting future developments. As regards the vision module, we can think of an improved version that can track multiple path-finders together or a single path-finder that suddenly changes its visual features during the follower task. Another interesting issue is the use of wearable haptic bracelets (Che et al., 2020) and the implementation of a protocol that the robot can use to notify to its path-finder the occurrence of exceptional conditions (e.g., when the path is too close to an obstacle and the robot cannot follow it within appropriate safety margins). Another important direction is a theoretical study of how the interaction between model-based approaches and neural networks can produce results with a guaranteed accuracy for people tracking, followed by a practical implementation. In this case, machine learning would not only be employed to classify the path-finder, but also to perform action recognition or skeleton motion prediction. Finally, it is worth investigating how the path information can be shared among multiple vehicles for transfer learning, even without any a-priori map knowledge.

Chapter 7

Conclusions

We are developing a new generation of robots that can move freely, take autonomous decisions, and interact directly with humans to execute tasks and deliver services. The expectations raised by this development are very high. People imagine a near future in which robots and humans will share public spaces, talk to each other, exchange objects and, to some extent, create emotional links. Is this a realistic perspective? In order to be up to the task, robots are required to be efficient for the tasks they are assigned (why else should we use a robot?) and they must preserve safety.

Indeed, although the problem at hand can be immensely complex, we dispose of software (in terms of new algorithms) and hardware (thus, computational power) that were technically unworkable until a couple of decades ago. Sensors for robotic platforms and autonomous cars have become cheaper and more reliable, allowing robots to perceive and measure the surrounding environment with great performance, obtaining detailed information. On the other hand, Artificial Intelligence has reached an incredible degree of understanding context, classifying the different entities, and interpreting human actions. Transferring this capability into the robotic domain is not as easy as it appears at first glance. Indeed, we have seen incredible progress in AI, but not such rapid progress for robotics. So, even if they should go hand in hand, many robotics applications are not designed to embed AI (and vice-versa). Some of the key reasons of this difference are the reliability issues of the level of success of the two fields. So, as the expected tasks that robots have to perform are expressed in the physical world, the bar of safety and reliability requirements has to be set very high (think for example to the vocal reaction to the relatively small number of accidents caused by autonomous cars in the past years).

However, in order to move between humans, safety is not enough: a robot has to be smooth in its reactions, it has to respect the personal space of the humans as much as possible, and it has to be predictable. Humans have sophisticated means to move in a socially acceptable way. Their social intelligence allows them to predict the intent of their peers by decoding a complex non-spoken language made of gestures, facial expressions, gaze, etc. Again, by putting together different novel methods a robot will be able to properly fuse all this huge amount of input data to produce the appropriate response, but this computing such an impressive size of information can be time-consuming, and, partially, unneeded for the considered use case, together with the fact that it is not unambiguously established what is the most suitable way to combine these data. Starting from these considerations we then decided on the focus of this thesis: the design of a modular framework for the autonomous navigation of mobile robots. Each level of the framework is well defined and confined to its area of competence, chosen based of the type and priority of the task to be performed, the characteristics of the information to be used, and the reaction time and space granted to the robot(s). The leading point of the framework is that the actions of the robot proactively consider the presence and intentions of people who are in the robot's surroundings, both with an active and passive interaction. Besides, the structure of the framework itself has a shape inspired by characteristics linked to relational conventions, such as proxemics. The second purpose of our proposed algorithm was to explore how to adequately combine model-based and learning-based approaches to obtain an efficient trade-off between the reliability of the system and the computational time of the modules, which are two crucial elements for an effective human-aware planner.

In Chapter 3, the main goal is to characterise the sensing and measurement system to be used on board the mobile platform and synthesise an estimation algorithm to f detect the human beings and their trajectories in the vicinity of the robot. Different types of sensors and fusion techniques were considered. In particular, we adopted a sensing system composed of a radar, a stereo camera, and a combination of camera and laser scanner. Together with the study of the characteristics and performance of the sensors, we have developed several people tracking algorithms and tested their effectiveness with an extensive experimental campaign. The operation and above all the type of information obtained from radar and LIDAR are easily understood. The types of sensors to which they belong allow having precise data on the position and distance of the objects and entities that are around the robot. The developed algorithms, therefore, served to filter and cluster appropriately the points measured by the sensors and apprehend their motion, using multi-model Bayesian estimation methods. Given the high abstraction of the inputs at disposal, we preferred a model-based approach to a learning method, using estimation filters combined with geometric and dynamic considerations on how to interpret the movement of a person. Indeed, despite the complex functioning of bipedal walking, a person's trajectory can be reduced to a set of the triad that make up his pose in 2D space, namely the (x, y) position and its θ orientation. From our experimental evaluation, it emerged that laser scanners are formidable for detection and ranging at high-frequency applications, but that in any case, they need to be used in combination with other sensors to complete the characteristics of the measured data (e.g., a group of reflex points actually belong to a person). Millimeter radar seemed also a promising tool for robotics applications. The low sensitivity to environmental conditions and obstructions is one of some interesting features of this type of sensor. However, the development and support of these devices at the moment are not that extensive. Future research directions may certainly include radars to analyze and correct their weak spots, which are mainly the high measurement noise and the fact that they are primarily intended for static operations. Cameras allow collecting much more information about the environment, the obstacles, and the human beings. In this case, automatic learning is an obligatory choice in order to process the number of insights contained in the image. In the literature several methods are present and fall under two broad categories: object detectors and pose estimators. The former identifies the human shape (or objects in general) by bounding a box around them, while the latter detects the skeleton joints and connects them resulting in pose estimates. From the experimental results, we have observed that pose estimators are computationally more expensive than object detectors and their output requires additional processing, even if the complete pose can tell more about the intentions of the persons, although the action or activity recognition is out of the studies reported in this thesis.

In conclusion, we have developed a fast and robust tracking system for the detection and positioning in the planar space of the persons that are in the robot's neighbourhood. The sensing system with the best performance is based on the 2D LIDAR and camera combination, where the first sensor is devoted mostly to the accurate positioning of the entities and the second is used for the recognition and classification of the persons. This configuration, which was then employed in the other works presented in this thesis, represents a suitable compromise between the accuracy of the measurements, the need for computing power, and the cost of sensors. The addition of a depth camera or the adoption of a 3D LIDAR allows gaining more contextual information such as the pose or the action the person is carrying out, or the obstacles and objects present, while greater safety can be ensured with the redundancy of sensors. Still, we can demonstrate that this simple sensing system is sufficient to carry out numerous applications for mobile robots, namely autonomous navigation, obstacles avoidance, and people following. As a side note, we can observe how the visual sensor coupled to the laser scanner gives the robot a perception space quite similar to that of humans. This space is large enough to encompass the action space where human-robot interactions take place. The tracking algorithm is robust to sensors' missdetections and targets temporary occlusions, and its filtering model, inspired to the nonholonomic nature of human walking, is correctly adapted to the behaviour of people's trajectories.

Some open issues are eligible for the next researches. The conducted experiments involved mostly one single moving agent in the sensing range of the sensors, thus the performance of the measurement and the tracking systems were evaluated and designed, respectively, to perceive one person at a time. This design choice was not a limitation for the approaches presented in this thesis, for example in the motion prediction of Chapter 4 we consider the motion of one pedestrian, while the framework presented in Chapter 5 can take into account multiple agents. Nevertheless, it will be interesting to expand the system capability to track multiple targets, as in this case the number of mutual occlusions will be higher. Notice that the tracking module described in Chapter 6 is already a preliminary working version of multi-agent tracking. Furthermore, the tracking system could be enhanced to work in outdoor scenarios. Outdoor conditions certainly represent a more dynamic and stimulating environment in which robots find themselves immersed in less structured scenarios with multiple movable entities (pedestrians, bicycles, cars, etc.). In future research, new filtering techniques will be studied to model the different dynamics of each agent, and further experiments will evaluate the characteristics of the sensors in the case of outdoor navigation.

Summarising, the main outcomes of this chapter are the following:

- A new tracking system based on the LIDAR-camera combination; faster and cheaper than a sensing system that relies on stereo camera or radar sensors;
- A lightened and robust tracking algorithm;
- A refined filtering model adapted to the behaviour of people's trajectories;
- The empirical demonstration that the Bayesian filtering can be used as a (behavioural) classifier.

The specific take of Chapter 4 is to use accurate predictions of human motions for robot plan synthesis in order to be inherently safe and compliant with unwritten social rules. Specifically, our proposed predictor relies on a structured neural network to infer the human future motion. The neurons of the network are organised according to the equations of the Social Force Model, which is a rather famous physics inspired dynamic model for representing agents motion in two-dimensional space. By embedding the prior knowledge of the dynamical model into the neural network, we expect a close approximation of the dynamics of human motion by the correct choice of parameters. Unlike traditional neural network approaches, we abstract the prediction of people's motion from the reference scenario, which requires specific datasets and may incur into overfitting, restricting the learning to the dynamic feature only. Indeed, the adoption of the Social Force Model has allowed us to include in the prediction also the effect of the surrounding static obstacles, which can be described in a general and abstract form, that modifies the resulting motion of the agents. In this way, our model can nicely model the navigation in indoor scenarios, and static obstacles can have different conformation between training data and testing environment, as they share the same representation.

The experimental results have shown that, despite the low number of learnable weights, the developed network correctly predicted the human trajectories, with performance comparable and sometimes better than other state of the art model. As an element of innovation compared to other works, we obtain far better results when we also consider the effects of the surrounding environment. Moreover, the complete framework, composed of a neural network and waypoint generation model, performs well both in simulation and in the implementation on the real mobile robot, since we had taken into account this final application since the choice of the algorithm's architecture. The lean structure of the network allows us to run the algorithm very quickly on the robot computing board.

As a further consideration, the model presented is an interesting use case of an explainable deep neural network, in which its internal parameters take on a specific meaning, being derived from a known mathematical model, and therefore it is easier to understand the behaviour of the trained network itself. In addition, a network with a structured architecture allows also to add custom constraints in internal points of the net or on its weights, in order to help convergence to the desired value and perform training in less time (or with less data).

The algorithm presented is the first example of this new type of neural network in the robotics field, and therefore it is open to several future developments. A first step will be the addition of multiple agents that interact with each other in the prediction, in order to align and compare our approach with the multi-agent predictors present in the literature. Secondly, other dynamic models, potentially better than SFM, could be incorporated into the network's structure. Finally, it would be interesting to increase the learning capabilities of the network by providing additional input data such as the orientation or pose of the person, or better contextual information about the goal or task pursued by humans.

Summarising, the advantages of the proposed approach are manifold:

- The number of neurons is significantly decreased (by one or two orders of magnitude) by wiring a model inside the neural network;
- The use of maximum likelihood evaluation of multiple target's endpoint deriving a less complex neural network than a monolithic solution;
- The possibility of creating synthetic datasets using known models, to create realistic data while reducing the waste of time;
- Because the neural network retains the model inside, its decisions can be explained in physical terms, which simplifies the interpretation of its inferences and the explanation of its possible errors.

Chapter 5 presents a hierarchical framework for multi-agent navigation in human-shared environments. The different levels of the framework include global path planning, local path planning, and reactive control. Each level relies on input information of increasing complexity and operates at different spaces of interest and updating frequencies. In this way, the tasks embedded in the framework are independent in their specific area of expertise, and their effects can be superimposed in the resulting robot action without stalling another task. We identified five requirements that must be satisfied when dealing with robot navigation with the compresence of humans, explained as follows. Safety guarantees are the first and most essential requirement. We have to ensure collision avoidance with obstacles, other robots, and human beings. This condition must be ensured even in the face of inaccurate robot localisation or estimations on the external entities. Beyond safety, we have to obtain socially aware motion planning. Indeed, the robot has to stay clear enough from the human private space, it has to follow smooth trajectories, and it should anticipate the human motion intentions. The presence of the robot will not intimidate the humans as long as the robot follows trajectories that are easily predictable by the pedestrian. When multiple robots are considered, a distributed coordination strategy between them is required. What is more, regardless of prior knowledge of the mission space, the navigation algorithm should quickly react to unmapped obstacles that are detected by on-board sensors. The last requirement concerns computation efficiency. This is important for two reasons: first, the robot must be able to compute control inputs at a high frequency in order to respond quickly to unexpected events, second, we can afford lean hardware with reduced costs and low energy consumption. In the state of the art literature, we cannot find a comprehensive solution that satisfies all the requirements mentioned above. Classic reactive methods (such as Force Field or Velocity Obstacle) are by construction computationally efficient and typically they can manage the multi-agent control. As the name suggests, these methods are designed to promptly react to the presence of obstacles, thus they can assure safety. However, if the estimation of the obstacles' position and velocity is not accurate, collisions cannot be ruled out. Moreover, being based on short-sighted decisions, a pure reactive approach can easily violate the socially aware requirement and led to deadlocking conditions in congested scenarios. Predictive methods synthesise a safe path taking into account local information and also a reliable human motion prediction. This means that, as long as the prediction fits the actual human trajectory and an appropriate cost function is selected, those methods can plan paths that are safe and socially aware. On the other hand, the computational power demand is higher and, again, safety cannot be ensured if the human motion prediction is inaccurate. Finally, learning-based methods can obtain a good level of safety and socially awareness. However, these techniques are unable of generalising social behaviour in different or dynamic environments. In addition, the computational requirements are usually acceptable for real-time navigation aims.

Thanks to the interaction between the three layers that compose our framework we actually can achieve all the requirements we have set. The global path is computed only upon request, typically at the beginning of the mission, by using the complete map of the environment. The global path is passed to the second layer, which generates the local path incorporating the human motion predictions. Thus, this layer produces a medium planning horizon, by using local information in a range of 5–10 m and it is called with a frequency of 1.25–2 Hz. Finally, the local path is passed to the reactive controller that computes the velocity control of the robot. This last layer has a reduced space of interest of 3 m, and it generates updated control inputs every 50 ms. Roughly speaking, the controller builds a safe region around the robot and it checks if the local path is safe: if not, it computes a safety-preserving deviation. These conditions can occur mainly if there is another robot on the planned path (as the local path does not consider the presence of other robots), or when the environment is highly dynamic, hence the updating frequency may be not fast enough. We tested the proposed framework on real robotic platforms fully developed at the University of Trento and we proved through extensive experiments the effectiveness of our approach. In particular, we have shown that our solution avoids collisions with humans but also deadlock between robots, and the coordination between the robots is efficient. The combination of local path planning with the human prediction allowed to perform avoidance manoeuvres with anticipatory behaviour, leaving the person undisturbed, and minimising the total deviation from the global (optimal) path. The experimental results validated the effectiveness of our approach and demonstrated that the specific modular structure of the framework and models of each individual layer are a promising solution to perform multi-robot navigation.

In the near future, we plan to integrate more sensors to obtain more accurate human motion predictions and hence improve the local path plan. The shown experiments were carried out by equipping the robots with the same sensing system already described in this thesis, therefore the agents relied on very simple input data. As mentioned also for the application of Chapter 4, different sensors will help to increase the overall performance of our approach. We underline that these possible changes would not be invasive for the rest of the framework, given its modular architecture. An assumption present that can be overcome is the actual non-cooperation between robots at the level of the local plan layer. Thus, further modifications can be done to integrate in the local path planner a distributed and computationally efficient (not sequential) multi-agent management system.

Summarising, with the proposed hierarchical architecture we have obtained:

- The fulfilment of five classes of requirements, including safe and socially-aware motion planning, multi-agent coordination, and computation efficiency;
- A modular framework in which each succeeding layer use a more detailed information on the environment but take decisions with a narrower spatio-temporal scope;
- Furthermore, the properties enforced at one layer are not invalidated by the layers below, so the navigation is not stalled by inconsistent decisions taken at different levels.

In Chapter 6 we describe an application for human autonomous following in which the robot can simultaneously follow the person and memorise the path travelled for future operations. Typically, visual servoing algorithms require that the human must always be inside the camera field of view. In our solution, instead, the person can perform sharp turns and complex paths at will, and the robot can correctly track her/him as long as at least one of its sensors keeps providing valid measures, even in the occurrence of temporary occlusions. Once the human operator has shown her/his visual features to the robot in the initialization phase, the latter can carry out the follower task even in the presence of other strangers in the way, who are avoided. In the event of an imminent collision with static (i.e., walls) or dynamic (the persons) obstacles, the robot stops to ensure safety and can later resume following the path if the space becomes clear again. Thus, in our approach, the human takes the role of path-finder, and the only required expertise is to simply walk and mark the path with her/his footsteps. The robot, for its part, must carry out a precise tracking of its path-finder, and follow him/her with a rapid response and socially-compliant trajectories, however, the generation of the optimal global path is enormously simplified because the responsibility is shifted to the person, who instead better knows how to navigate in complex or dynamic scenarios where many people are present. The robot's sensing system is an evolved version of the configuration presented in Chapter 3 and is composed of a 2D laser scanner and a depth camera. The application presents a combination of machine learning techniques and model-based approaches. Indeed, in the vision-based detection and recognition part of the framework, we use state-of-the-art neural networks to recognize and memorize the visual features of the human to be elected as leader. In the subsequent sensor fusion between the two sensors, on the other hand, we use a model-based solution to steadily track the positions of the path-finder. The tracked position are filtered assuming the same motion model developed in Chapter 3, and the robot's trajectory is generated using clothoid curves to obtain a smooth path by mimicking a human-like behaviour.

We tested the proposed framework on a real robotic platform and we proved through extensive experiments the effectiveness of our approach. We have shown that our algorithm successfully recognise and track the path-finder, and the travelled paths are consistent with those of the person. The disturbing effects of the intruders are rejected, and the path-finder can be found again in case the tracking is lost. In our experimental tests, we evaluated the approach also in the case of multiple people present in the scene, and we shown that the robot can maintain safety and avoid collisions. We can conclude that the experiments demonstrated the high level of reliability and robustness reached by the our solution.

Summarising, the proposed method:

- Represents a modern application of the teach-by-showing approach where the robot works intuitively with the human operator;
- Employs a robust combination of tracking filter and neural network to estimate and follow the path-finder's position, even when s/he falls outside of the camera's visual cone;
- Can comply with safety and reliability requirements thanks to the system's ability to distinguish between persons with similar features and resolve misclassifications due to illumination changes and partial occlusions.

Different points remain open and are reserved for future work. A first important direction is a theoretical study of how the interaction between model-based approaches and neural networks can produce results with a guaranteed accuracy for people tracking. That is, the idea is to embed the prediction model presented in Chapter 4 in the framework to increase the reactivity of the robot (since it will consider the social norms behind the human decisions), and to strengthen the tracking in case of temporary occlusions or missdetections. Another interesting issue is the use of wearable haptic bracelets and the implementation of a protocol that the robot can use to notify its path-finder the occurrence of exceptional conditions (e.g., when the path is too close to an obstacle and the robot cannot follow it within appropriate safety margins). Finally, it is worth investigating how the path information can be shared among multiple vehicles for transfer learning even without any a-priori map knowledge.

Bibliography

Abdar, Moloud, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. (2021). "A review of uncertainty quantification in deep learning: Techniques, applications and challenges". In: *Information Fusion* 76, pp. 243–297.

Aguirre, Eugenio and Miguel García-Silvente (2019). "Using a Deep Learning Model on Images to Obtain a 2D Laser People Detector for a Mobile Robot". In: *International Journal of Computational Intelligence Systems* 12.2, pp. 476–484.

Alahi, Alexandre, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese (2016). "Social lstm: Human trajectory prediction in crowded spaces". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 961–971.

Alcorn, Michael A, Qi Li, Zhitao Gong, Chengfei Wang, Long Mai, Wei-Shinn Ku, and Anh Nguyen (2019). "Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4845–4854.

Alonso-Mora, Javier, Andreas Breitenmoser, Martin Rufli, Paul Beardsley, and Roland Siegwart (2013). "Optimal reciprocal collision avoidance for multiple non-holonomic robots". In: *Distributed autonomous robotic systems.* Springer, pp. 203–216.

Amaoka, Toshitaka, Hamid Laga, and Masayuki Nakajima (2009). "Modeling the personal space of virtual agents for behavior simulation". In: *2009 International Conference on CyberWorlds.* IEEE, pp. 364–370.

Andre, D, A Nuhic, T Soczka-Guth, and Dirk Uwe Sauer (2013). "Comparative study of a structured neural network and an extended Kalman filter for state of health determination of lithium-ion batteries in hybrid electricvehicles". In: *Engineering Applications of Artificial Intelligence* 26.3, pp. 951–961.

Andreetto, Marco, Stefano Divan, Daniele Fontanelli, and Luigi Palopoli (2017). "Harnessing steering singularities in passive path following for robotic walkers". In: *2017 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 2426–2432.

Antonini, Gianluca, Santiago Venegas Martinez, Michel Bierlaire, and Jean Philippe Thiran (2006). "Behavioral priors for detection and tracking of pedestrians in video sequences". In: *International Journal of Computer Vision* 69.2, pp. 159–180.

Antonucci, A., V. Magnago, L. Palopoli, and D. Fontanelli (2019a). "Performance Assessment of a People Tracker for Social Robots". In: *2019 IEEE International Instrumentation and Measurement Technology Conference (I2MTC).* IEEE, pp. 1–6. DOI: 10.1109/I2MTC.2019.8826999.

Antonucci, Alessandro and Daniele Fontanelli (2018). "Towards a Predictive Behavioural Model for Service Robots in Shared Environments". In: *2018 IEEE Workshop on Advanced Robotics and its Social Impacts (ARSO).* IEEE, pp. 9–14. DOI: 10.1109/ARSO.2018.8625784.

Antonucci, Alessandro, Paolo Bevilacqua, Luigi Palopoli, Manuel Boldrer, and Daniele Fontanelli (2019b). "Motion Planning in Crowds: Proxemics as a Base for a Socially Acceptable Behaviour". In: *2019 1st Italian Conference on Robotics and Intelligent Machines (I-RIM).* I-RIM, pp. 15–16. DOI: 10.5281/zenodo.4782236.

Antonucci, Alessandro, Michele Corrà, Alessandro Ferrari, Daniele Fontanelli, Emiliano Fusari, David Macii, and Luigi Palopoli (2019c). "Performance analysis of a 60-GHz radar for indoor positioning and tracking". In: *2019 International Conference on Indoor Positioning and Indoor Navigation (IPIN).* IEEE, pp. 1–7. DOI: 10.1109/IPIN.2019.8911764.

Antonucci, Alessandro, Gastone Pietro Rosati Papini, Paolo Bevilacqua, Luigi Palopoli, and Daniele Fontanelli (2021a). "Efficient Prediction of Human Motion for Real-Time Robotics Applications With Physics-Inspired Neural Networks". In: *IEEE Access* 10, pp. 144–157. DOI: 10.1109/ACCESS.2021.3138614.

Antonucci, Alessandro, Paolo Bevilacqua, Stefano Leonardi, Luigi Palopoli, and Daniele Fontanelli (2021b). "Humans as Path-Finders for Safe Navigation". In: *arXiv preprint arXiv:2107.03079*.

Arechavaleta, Gustavo, Jean-Paul Laumond, Halim Hicheur, and Alain Berthoz (2008). "On the nonholonomic nature of human locomotion". In: *Autonomous Robots* 25.1-2, pp. 25–35.

Arras, Kai O, Oscar Martinez Mozos, and Wolfram Burgard (2007). "Using boosted features for the detection of people in 2d range data". In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, pp. 3402–3407.

Babenko, Boris, Ming-Hsuan Yang, and Serge Belongie (2010). "Robust object tracking with online multiple instance learning". In: *IEEE transactions on pattern analysis and machine intelligence* 33.8, pp. 1619–1632.

Bahari, Mohammadhossein, Saeed Saadatnejad, Ahmad Rahimi, Mohammad Shaverdikondori, Amir Hossein Shahidzadeh, Seyed-Mohsen Moosavi-Dezfooli, and Alexandre Alahi (2022). "Vehicle trajectory prediction works, but not everywhere". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17123–17133.

Baig, Mirza Waqar, Emilia Barakova, Carlo S Regazzoni, and Matthias Rauterberg (2014). "Realistic modeling of agents in crowd simulations". In: *2014 5th International Conference on Intelligent Systems, Modelling and Simulation*. IEEE, pp. 507–512.

Bajcsy, Andrea, Sylvia L Herbert, David Fridovich-Keil, Jaime F Fisac, Sampada Deglurkar, Anca D Dragan, and Claire J Tomlin (2019). "A scalable framework for real-time multi-robot, multi-human collision avoidance". In: *2019 international conference on robotics and automation (ICRA)*. IEEE, pp. 936–943.

Bar-Shalom, Yaakov, X Rong Li, and Thiagalingam Kirubarajan (2004). *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons.

Barredo Arrieta, Alejandro, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera (2020). "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI". In: *Information Fusion* 58, pp. 82–115. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2019.12.012. URL: https://www.sciencedirect.com/science/article/pii/S1566253519308103.

Basili, Patrizia, Murat Sağlam, Thibault Kruse, Markus Huber, Alexandra Kirsch, and Stefan Glasauer (2013). "Strategies of locomotor collision avoidance". In: *Gait & posture* 37.3, pp. 385–390.

Battaglia, Peter W, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. (2018). "Relational inductive biases, deep learning, and graph networks". In: *arXiv preprint arXiv:1806.01261*.

Belkada, Younes, Lorenzo Bertoni, Romain Caristan, Taylor Mordan, and Alexandre Alahi (2021). "Do Pedestrians Pay Attention? Eye Contact Detection in the Wild". In: *arXiv preprint arXiv:2112.04212*.

Bera, Aniket, Tanmay Randhavane, Rohan Prinja, and Dinesh Manocha (2017). "Sociosense: Robot navigation amongst pedestrians with social and psychological constraints". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 7018–7025.

Berg, Jur Van den, Ming Lin, and Dinesh Manocha (2008). "Reciprocal velocity obstacles for real-time multi-agent navigation". In: *2008 IEEE International Conference on Robotics and Automation*. IEEE, pp. 1928–1935.

Berg, Jur van den, Stephen J Guy, Ming Lin, and Dinesh Manocha (2011). "Reciprocal n-body collision avoidance". In: *Robotics research*. Springer, pp. 3–19.

Bertolazzi, Enrico and Marco Frego (2015). "G1 fitting with clothoids". In: *Mathematical Methods in the Applied Sciences* 38.5, pp. 881–897.

— (2018a). "Interpolating clothoid splines with curvature continuity". In: *Mathematical Methods in the Applied Sciences* 41.4, pp. 1723–1737.

— (2018b). “On the G2 Hermite interpolation problem with clothoids”. In: *Journal of Computational and Applied Mathematics* 341, pp. 99–116.

Bertolazzi, Enrico, Paolo Bevilacqua, and Marco Frego (2018). “Clothoids: a C++ library with Matlab interface for the handling of clothoid curves”. In: *Rend. Sem. Mat. Univ. Pol. Torino* 76.2, pp. 47–56.

Best, Andrew, Katelynn A Kapalo, Samantha F Warta, and Stephen M Fiore (2016). “Clustering social cues to determine social signals: developing learning algorithms using the" n-most likely states" approach”. In: *Unmanned Systems Technology XVIII*. Vol. 9837. International Society for Optics and Photonics, p. 98370L.

Bevilacqua, P., M. Frego, D. Fontanelli, and L. Palopoli (2018a). “Reactive Planning for Assistive Robots”. In: *IEEE Robotics and Automation Letters* 3.2, pp. 1276–1283. ISSN: 2377-3766. DOI: 10.1109/LRA.2018.2795642.

Bevilacqua, Paolo, Marco Frego, Daniele Fontanelli, and Luigi Palopoli (2018b). “Reactive planning for assistive robots”. In: *IEEE Robotics and Automation Letters* 3.2, pp. 1276–1283.

Boldrer, Manuel, Luigi Palopoli, and Daniele Fontanelli (2020a). “Lloyd-based Approach for Robots Navigation in Human-shared environments”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE/RSJ, pp. 6982–6989.

— (2020b). “Socially-Aware Multi-agent Velocity Obstacle Based Navigation for Nonholonomic Vehicles”. In: *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, pp. 18–25.

Boldrer, Manuel, Marco Andreetto, Stefano Divan, Luigi Palopoli, and Daniele Fontanelli (2020c). “Socially-aware Reactive Obstacle Avoidance Strategy based on Limit Cycle”. In: *IEEE Robotics and Automation Letters*.

Boldrer, Manuel, Alessandro Antonucci, Paolo Bevilacqua, Luigi Palopoli, and Daniele Fontanelli (2022). “Multi-agent navigation in human-shared environments: A safe and socially-aware approach”. In: *Robotics and Autonomous Systems* 149, p. 103979. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2021.103979. URL: https://www.sciencedirect.com/science/article/pii/S092188902100244X.

Bolme, David S, J Ross Beveridge, Bruce A Draper, and Yui Man Lui (2010). “Visual object tracking using adaptive correlation filters”. In: *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE, pp. 2544–2550.

Bonci, Andrea, Pangcheng David Cen Cheng, Marina Indri, Giacomo Nabissi, and Fiorella Sibona (2021). “Human-robot perception in industrial environments: A survey”. In: *Sensors* 21.5, p. 1571.

Broad, Alexander, Ian Abraham, Todd Murphey, and Brenna Argall (2018). “Structured neural network dynamics for model-based control”. In: *arXiv preprint arXiv:1808.01184*.

Burlet, Julien, Olivier Aycard, Anne Spalanzani, and Christian Laugier (2006). “Pedestrian tracking in car parks: an adaptive interacting multiple models based filtering method”. In: *2006 IEEE Intelligent Transportation Systems Conference*. IEEE, pp. 462–467.

Cao, Zhe, Tomas Simon, Shih-En Wei, and Yaser Sheikh (2017). “Realtime multi-person 2d pose estimation using part affinity fields”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7291–7299.

Cao, Zhe, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh (2019). “OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields”. In: *IEEE transactions on pattern analysis and machine intelligence* 43.1, pp. 172–186.

Cha, Elizabeth, Yunkyung Kim, Terrence Fong, Maja J Mataric, et al. (2018). “A survey of nonverbal signaling methods for non-humanoid robots”. In: *Foundations and Trends® in Robotics* 6.4, pp. 211–323.

Charalampous, Konstantinos, Ioannis Kostavelis, and Antonios Gasteratos (2017). “Recent trends in social aware robot navigation: A survey”. In: *Robotics and Autonomous Systems* 93, pp. 85–104.

Chavand, Florent, Etienne Colle, Y Chekhar, and EC N’zi (1997). “3-D measurements using a video camera and a range finder”. In: *IEEE Transactions on Instrumentation and Measurement* 46.6, pp. 1229–1235.

Che, Yuhang, Allison M Okamura, and Dorsa Sadigh (2020). “Efficient and Trustworthy Social Navigation via Explicit and Implicit Robot–Human Communication”. In: *IEEE Transactions on Robotics* 36.3, pp. 692–707.

Chen, Bao Xin, Raghavender Sahdev, and John K Tsotsos (2017a). “Person following robot using selected online ada-boosting with stereo camera”. In: *2017 14th conference on computer and robot vision (CRV)*. IEEE, pp. 48–55.

Chen, Changan, Yuejiang Liu, Sven Kreiss, and Alexandre Alahi (2019). “Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6015–6022.

Chen, Yu Fan, Michael Everett, Miao Liu, and Jonathan P How (2017b). “Socially aware motion planning with deep reinforcement learning”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 1343–1350.

Chen, Zhen-Chang and Chih-Wei Tang (2018). “Robust Pedestrian Tracking Using Interactive Multiple Model Particle Filter and Feature Matching”. In: *2018 3rd International Conference on Advanced Robotics and Mechatronics (ICARM)*. IEEE, pp. 480–485.

Cheng, J., H. Cheng, M. Q. . Meng, and H. Zhang (2018). “Autonomous Navigation by Mobile Robots in Human Environments: A Survey”. In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1981–1986. DOI: 10.1109/ROBIO.2018.8665075.

Cheng, Jiyu, Hu Cheng, Max Q-H Meng, and Hong Zhang (2018). “Autonomous navigation by mobile robots in human environments: a survey”. In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 1981–1986.

Chik, Sheng Fei, Che Fai Yeong, Lee Ming Su, Thol Yong Lim, Fend Duan, and Jun Hua Chin (2019). “Neural-Network Based Adaptive Proxemics-Costmap for Human-Aware Autonomous Robot Navigation”. In: *International Journal of Integrated Engineering* 11.4.

Chow, Tommy WS and Xiao-Dong Li (2000). “Modeling of continuous time dynamical systems with input by recurrent neural networks”. In: *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* 47.4, pp. 575–578.

Claes, Daniel and Karl Tuyls (2018). “Multi robot collision avoidance in a shared workspace”. In: *Autonomous Robots* 42.8, pp. 1749–1770.

Clark, Murray R (1994). “Application of Machine Learning Principles to Modeling of Nonlinear Dynamic Systems”. In: *Journal of the Arkansas Academy of Science* 48.1, pp. 36–40.

Cleveland, William S (1979). “Robust locally weighted regression and smoothing scatterplots”. In: *Journal of the American statistical association* 74.368, pp. 829–836.

Colombo, A., D. Fontanelli, D. Macii, and L. Palopoli (2014). “Flexible Indoor Localization and Tracking based on a Wearable Platform and Sensor Data Fusion”. In: *IEEE Trans. on Instrumentation and Measurement* 63.4, pp. 864–876. ISSN: 0018-9456. DOI: 10.1109/TIM.2013.2283546.

Colombo, Alessio, Daniele Fontanelli, Axel Legay, Luigi Palopoli, and Sean Sedwards (2013). “Motion Planning in Crowds using Statistical Model Checking to Enhance the Social Force Model”. In: *Decision and Control (cdc2013), 2013 Proc. of 53nd IEEE Conference on*. Firenze, Italy.

Cortes, Jorge, Sonia Martinez, Timur Karatas, and Francesco Bullo (2004). “Coverage control for mobile sensing networks”. In: *IEEE Transactions on robotics and Automation* 20.2, pp. 243–255.

Cosgun, Akansel, Emrah Akin Sisbot, and Henrik Iskov Christensen (2016). “Anticipatory robot path planning in human environments”. In: *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, pp. 562–569.

Croft, James L and Derek Panchuk (2018). “Watch where you’re going? Interferer velocity and visual behavior predicts avoidance strategy during pedestrian encounters”. In: *Journal of motor behavior* 50.4, pp. 353–363.

Cutting, James E, Peter M Vishton, and Paul A Braren (1995). “How we avoid collisions with stationary and moving objects.” In: *Psychological review* 102.4, p. 627.

Da Lio, Mauro, Daniele Bortoluzzi, and Gastone Pietro Rosati Papini (2020). “Modelling longitudinal vehicle dynamics with neural networks”. In: *Vehicle System Dynamics* 58.11, pp. 1675–1693.

De Angelis, G., A. Moschitta, and P. Carbone (2016). “Positioning Techniques in Indoor Environments Based on Stochastic Modeling of UWB Round-Trip-Time Measurements”. In: *IEEE Transactions on Intelligent Transportation Systems* 17.8, pp. 2272–2281. ISSN: 1524-9050.

De Vito, Luca, Octavian Postolache, and Sergio Rapuano (2014). “Measurements and sensors for motion tracking in motor rehabilitation”. In: *IEEE Instrumentation & Measurement Magazine* 17.3, pp. 30–38.

DeGroot, Morris H (1974). “Reaching a consensus”. In: *Journal of the American Statistical Association* 69.345, pp. 118–121.

Demeyere, Michal, Do Rurimunzu, and Christian Eugène (2007). “Diameter measurement of spherical objects by laser triangulation in an ambulatory context”. In: *IEEE Transactions on instrumentation and measurement* 56.3, pp. 867–872.

Deo, Nachiket and Mohan M Trivedi (2020). “Trajectory Forecasts in Unknown Environments Conditioned on Grid-Based Plans”. In: *arXiv preprint arXiv:2001.00735*.

Diallo, Alhassan Boner, Hiroyuki Nakagawa, and Tatsuhiro Tsuchiya (2021). “Adaptation Space Reduction Using an Explainable Framework”. In: *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, pp. 1653–1660.

Dondrup, Christian, Christina Lichtenthäler, and Marc Hanheide (2014). “Hesitation signals in human-robot head-on encounters: a pilot study”. In: *2014 9th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE, pp. 154–155.

Douglas, David H and Thomas K Peucker (1973). “Algorithms for the reduction of the number of points required to represent a digitized line or its caricature”. In: *Cartographica: the international journal for geographic information and geovisualization* 10.2, pp. 112–122.

Doyle, Daniel D, Alan L Jennings, and Jonathan T Black (2013). “Optical flow background subtraction for real-time PTZ camera object tracking”. In: *Instrumentation and Measurement Technology Conference (I2MTC), 2013 IEEE International*. IEEE, pp. 866–871.

Ebelt, R., A. Hamidian, D. Shmakov, T. Zhang, V. Subramanian, G. Boeck, and M. Vossiek (2014). “Cooperative Indoor Localization Using 24-GHz CMOS Radar Transceivers”. In: *IEEE Transactions on Microwave Theory and Techniques* 62.9, pp. 2193–2203.

Eiffert, Stuart and Salah Sukkarieh (2019). “Predicting Responses to a Robot’s Future Motion using Generative Recurrent Neural Networks”. In: *arXiv preprint arXiv:1909.13486*.

Evans, David J, Dan Cornford, and Ian T Nabney (2000). “Structured neural network modelling of multi-valued functions for wind vector retrieval from satellite scatterometer measurements”. In: *Neurocomputing* 30.1-4, pp. 23–30.

Everingham, Mark, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman (2007). “The PASCAL visual object classes challenge 2007 (VOC2007) results”. In:

Fan, T., P. Long, W. Liu, and J. Pan (2020). “Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios”. In: *International Journal of Robotics Research* 39.7, pp. 856–892. DOI: 10.1177/0278364920916531.

Farina, F., D. Fontanelli, A. Garulli, A. Giannitrapani, and D. Prattichizzo (Jan. 2017a). “Walking Ahead: The Headed Social Force Model”. In: *PLOS ONE* 12.1, pp. 1–23. DOI: 10.1371/journal.pone.0169734.

Farina, Francesco, Daniele Fontanelli, Andrea Garulli, Antonio Giannitrapani, and Domenico Prattichizzo (2017b). “Walking ahead: The headed social force model”. In: *PloS one* 12.1, e0169734.

Farmer, Michael E, Rein-Lien Hsu, and Anil K Jain (2002). “Interacting multiple model (IMM) Kalman filters for robust high speed human motion tracking”. In: *Object recognition supported by user interaction for service robots*. Vol. 2. IEEE, pp. 20–23.

Feldman, Jerome A, Mark A Fanty, Nigel H Goddard, and Kenton J Lynne (1988). “Computing with structured connectionist networks”. In: *Communications of the ACM* 31.2, pp. 170–187.

Ferrer, Gonzalo and Alberto Sanfeliu (2019). “Anticipative kinodynamic planning: multi-objective robot navigation in urban and dynamic environments”. In: *Autonomous Robots* 43.6, pp. 1473–1488.

Ferrer, Gonzalo, Anaís Garrell Zulueta, Fernando Herrero Cotarelo, and Alberto Sanfeliu (2017). “Robot social-aware navigation framework to accompany people walking side-by-side”. In: *Autonomous robots* 41.4, pp. 775–793.

Fiore, Stephen M, Travis J Wiltshire, Emilio JC Lobato, Florian G Jentsch, Wesley H Huang, and Benjamin Axelrod (2013). “Toward understanding social cues and signals in human–robot interaction: effects of robot gaze and proxemic behavior”. In: *Frontiers in psychology* 4, p. 859.

Fiorini, Paolo and Zvi Shiller (1998). “Motion planning in dynamic environments using velocity obstacles”. In: *The International Journal of Robotics Research* 17.7, pp. 760–772.

Flemisch, Frank O, Catherine A Adams, Sheila R Conway, Ken H Goodrich, Michael T Palmer, and Paul C Schutte (2003). *The H-Metaphor as a guideline for vehicle automation and interaction.* Tech. rep. RWTH Aachen University.

Foka, Amalia F and Panos E Trahanias (2010). “Probabilistic autonomous robot navigation in dynamic environments with human motion prediction”. In: *International Journal of Social Robotics* 2.1, pp. 79–94.

Fotiadis, Efstathios, Mario Garzón, and Antonio Barrientos (2013). “Human detection from a mobile robot using fusion of laser and vision information”. In: *Sensors* 13.9, pp. 11603–11635.

Fotios, Steve, Jim Uttley, and B Yang (2015). “Using eye-tracking to identify pedestrians’ critical visual tasks. Part 2. Fixation on pedestrians”. In: *Lighting Research & Technology* 47.2, pp. 149–160.

Fox, Dieter, Wolfram Burgard, and Sebastian Thrun (1997). “The dynamic window approach to collision avoidance”. In: *IEEE Robotics & Automation Magazine* 4.1, pp. 23–33.

Frego, Marco, Paolo Bevilacqua, Enrico Saccon, Luigi Palopoli, and Daniele Fontanelli (2020). “An iterative dynamic programming approach to the multipoint markov-dubins problem”. In: *IEEE Robotics and Automation Letters* 5.2, pp. 2483–2490.

Funahashi, Ken-ichi and Yuichi Nakamura (1993). “Approximation of dynamical systems by continuous time recurrent neural networks”. In: *Neural networks* 6.6, pp. 801–806.

Gao, Yuxiang and Chien-Ming Huang (2021). “Evaluation of Socially-Aware Robot Navigation”. In: *Frontiers in Robotics and AI*, p. 420.

Garcia, Pablo, Fernando Briz, Dejan Raca, and Robert D Lorenz (2007). “Saliency-tracking-based sensorless control of AC machines using structured neural networks”. In: *IEEE Transactions on Industry Applications* 43.1, pp. 77–86.

Gardill, Markus (2019). *Automotive Radar - An Overview on State of the Art Technology Slides.* Accessed: 2022-10-22. IEEE Microwave Theory and Techniques Society. URL: https://www.youtube.com/watch?v=P-C6_4ceY64.

Garrido, Santiago, Luis Moreno, Mohamed Abderrahim, and Dolores Blanco (2009). “FM2: a real-time sensor-based feedback controller for mobile robots”. In: *International Journal of Robotics & Automation* 24.1, p. 48.

Garzon Oviedo, Mario Andrei, Antonio Barrientos, Jaime Del Cerro, Andrés Alacid, Efstathios Fotiadis, Gonzalo R Rodríguez-Canosa, and Bang-Chen Wang (2015). “Tracking and following pedestrian trajectories, an approach for autonomous surveillance of critical infrastructures”. In: *Industrial Robot: An International Journal* 42.5, pp. 429–440.

Gate, Gwennael, Amaury Breheret, and Fawzi Nashashibi (2009). “Fast pedestrian detection in dense environment with a laser scanner and a camera”. In: *VTC Spring 2009-IEEE 69th Vehicular Technology Conference.* IEEE, pp. 1–6.

Gil, Óscar and Alberto Sanfeliu (2019). “Effects of a social force model reward in robot navigation based on deep reinforcement learning”. In: *Iberian Robotics conference.* Springer, pp. 213–224.

Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik (2014). “Rich feature hierarchies for accurate object detection and semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587.

Greydanus, Samuel, Misko Dzamba, and Jason Yosinski (2019). “Hamiltonian neural networks”. In: *Advances in Neural Information Processing Systems* 32.

Guerra, A., F. Guidi, D. Dardari, A. Clemente, and R. D'Errico (2017). "A Millimeter-Wave Indoor Backscattering Channel Model for Environment Mapping". In: *IEEE Transactions on Antennas and Propagation* 65.9.

Guidi, F., A. Guerra, D. Dardari, A. Clemente, and R. D'Errico (2016). "Environment Mapping with Millimeter-Wave Massive Arrays: System Design and Performance". In: *2016 IEEE Globecom Workshops (GC Wkshps).* Washington, DC, USA, pp. 1–6.

Güldenring, Ronja, Michael Görner, Norman Hendrich, Niels Jul Jacobsen, and Jianwei Zhang (2020). "Learning Local Planners for Human-aware Navigation in Indoor Environments". In: pp. 6053–6060.

Günther, Johannes, Elias Reichensdörfer, Patrick M Pilarski, and Klaus Diepold (2020). "Interpretable PID parameter tuning for control engineering using general dynamic neural networks: An extensive comparison". In: *Plos One* 15.12, e0243320.

Gupta, Agrim, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi (2018). "Social gan: Socially acceptable trajectories with generative adversarial networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2255–2264.

Haghani, Milad and Majid Sarvi (2018). "Crowd behaviour and motion: Empirical methods". In: *Transportation research part B: methodological* 107, pp. 253–294.

Hall, Edward Twitchell (1969). *The hidden dimension: man's use of space in public and private.* Bodley Head.

Hamandi, Mahmoud, Mike D'Arcy, and Pooyan Fazli (2019). "Deepmotion: Learning to navigate like humans". In: *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN).* IEEE, pp. 1–7.

Hamzah, Rostam Affendi and Haidi Ibrahim (2016). "Literature survey on stereo vision disparity map algorithms". In: *Journal of Sensors* 2016.

Han, Ruihua, Shengduo Chen, and Qi Hao (2020). "Cooperative Multi-Robot Navigation in Dynamic Environment with Deep Reinforcement Learning". In: *2020 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 448–454.

Harrington, D.O. (January, 1981). *The Visual Fields: A Textbook and Atlas of Clinical Perimetry, 5th ed.* St. Louis, Missouri: Mosby. ISBN: 9780801620591.

Hasan, Irtiza, Francesco Setti, Theodore Tsesmelis, Vasileios Belagiannis, Sikandar Amin, Alessio Del Bue, Marco Cristani, and Fabio Galasso (2019). "Forecasting people trajectories and head poses by jointly reasoning on tracklets and vislets". In: *IEEE transactions on pattern analysis and machine intelligence* 43.4, pp. 1267–1278.

Hashimoto, Masafumi, Tomoki Konda, Zhitao Bai, and Kazuhiko Takahashi (2010). "Laser-based tracking of randomly moving people in crowded environments". In: *2010 IEEE International Conference on Automation and Logistics.* IEEE, pp. 31–36.

Hayduk, Leslie A (1981). "The shape of personal space: An experimental investigation." In: *Canadian Journal of Behavioural Science/Revue canadienne des sciences du comportement* 13.1, p. 87.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Identity mappings in deep residual networks". In: *European conference on computer vision.* Springer, pp. 630–645.

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick (2017). "Mask r-cnn". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.

Hearst, Marti A., Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf (1998). "Support vector machines". In: *IEEE Intelligent Systems and their applications* 13.4, pp. 18–28.

Heiden, Eric, David Millard, Erwin Coumans, and Gaurav S Sukhatme (2020a). "Augmenting differentiable simulators with neural networks to close the sim2real gap". In: *arXiv preprint arXiv:2007.06045.*

Heiden, Tessa van der, Christian Weiss, Naveen Nagaraja Shankar, Efstratios Gavves, and Herke van Hoof (2020b). "Social navigation with human empowerment driven reinforcement learning". In: *arXiv preprint arXiv:2003.08158.*

Helbing, Dirk and Peter Molnar (1995). "Social force model for pedestrian dynamics". In: *Physical review E* 51.5, p. 4282.

Held, David, Sebastian Thrun, and Silvio Savarese (2016). "Learning to track at 100 fps with deep regression networks". In: *European conference on computer vision.* Springer, pp. 749–765.

Henriques, Joao F, Rui Caseiro, Pedro Martins, and Jorge Batista (2012). “Exploiting the circulant structure of tracking-by-detection with kernels”. In: *European conference on computer vision*. Springer, pp. 702–715.

Herrera, Daniel, Flavio Roberti, Marcos Toibero, and Ricardo Carelli (2016). “Human-Robot Interaction: Legible behavior rules in passing and crossing events”. In: *IEEE Latin America Transactions* 14.6, pp. 2644–2650.

Herrera, Daniel, Javier Gimenez, Matias Monllor, Flavio Roberti, and Ricardo Carelli (2019). “Cognitive social zones for improving the pedestrian collision avoidance with mobile robots”. In: *Revista Politécnica* 42.2, pp. 7–14.

Hochlehnert, Andreas, Alexander Terenin, Steindór Sæmundsson, and Marc Deisenroth (2021). “Learning contact dynamics using physically structured neural networks”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 2152–2160.

Hong, Jinpyo and Kyihwan Park (2011). “A new mobile robot navigation using a turning point searching algorithm with the consideration of obstacle avoidance”. In: *The International Journal of Advanced Manufacturing Technology* 52.5, pp. 763–775.

Honig, Shanee S, Tal Oron-Gilad, Hanan Zaichyk, Vardit Sarne-Fleischmann, Samuel Olatunji, and Yael Edan (2018). “Toward socially aware person-following robots”. In: *IEEE Transactions on Cognitive and Developmental Systems* 10.4, pp. 936–954.

Howard, Andrew G, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam (2017). “Mobilenets: Efficient convolutional neural networks for mobile vision applications”. In: *arXiv preprint arXiv:1704.04861*.

Hu, Nan, Michael Harold Lees, and Suiping Zhou (2013). “A pattern-based modeling framework for simulating human-like pedestrian steering behaviors”. In: *Proceedings of the 19th ACM Symposium on Virtual Reality Software and Technology*, pp. 179–188.

Huber, Markus, Yi-Huang Su, Melanie Krüger, Katrin Faschian, Stefan Glasauer, and Joachim Hermsdörfer (2014). “Adjustments of speed and path when avoiding collisions with another pedestrian”. In: *PloS one* 9.2, e89589.

Hur, H. and H. Ahn (2010). “A Circuit Design for Ranging Measurement Using Chirp Spread Spectrum Waveform”. In: *IEEE Sensors Journal* 10.11, pp. 1774–1778.

Ikeda, Tetsushi, Yoshihiro Chigodo, Daniel Rea, Francesco Zanlungo, Masahiro Shiomi, and Takayuki Kanda (2013). “Modeling and prediction of pedestrian behavior based on the sub-goal concept”. In: *Robotics* 10.

Islam, Md Jahidul, Jungseok Hong, and Junaed Sattar (2019). “Person-following by autonomous robots: A categorical overview”. In: *The International Journal of Robotics Research* 38.14, pp. 1581–1618.

ISO/IEC Guide 98-3:2008, (2008). *Uncertainty of measurement – Part 3: Guide to the expression of uncertainty in measurement (GUM:1995)*.

Jiang, Sicong, Jianing Zhang, Yunzhou Zhang, Feng Qiu, Dongdong Wang, and Xiaobo Liu (2018). “Long-term tracking algorithm with the combination of multi-feature fusion and YOLO”. In: *Chinese Conference on Image and Graphics Technologies*. Springer, pp. 390–402.

Jiang, Zhengqiang and Du Q Huynh (2017). “Multiple pedestrian tracking from monocular videos in an interacting multiple model framework”. In: *IEEE transactions on image processing* 27.3, pp. 1361–1375.

Jiang, Zhengqiang, Du Q Huynh, William Moran, and Subhash Challa (2011). “Appearance and motion based data association for pedestrian tracking”. In: *Image and Vision Computing New Zealand*, pp. 459–464.

Jin, Wanting, Paolo Salaris, and Philippe Martinet (2020). “Proactive-Cooperative Navigation in Human-Like Environment for Autonomous Robots”. In: *International Conference on Informatics in Control, Automation and Robotics*.

Johansson, Anders, Dirk Helbing, and Pradyumn K Shukla (2007). “Specification of the social force pedestrian model by evolutionary adjustment to video tracking data”. In: *Advances in complex systems* 10.supp02, pp. 271–288.

Johnson, Collin and Benjamin Kuipers (2018). “Socially-aware navigation using topological maps and social norm learning”. In: *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. ACM, pp. 151–157.

Kalal, Zdenek, Krystian Mikolajczyk, and Jiri Matas (2010). "Forward-backward error: Automatic detection of tracking failures". In: *2010 20th international conference on pattern recognition*. IEEE, pp. 2756–2759.

— (2011). "Tracking-learning-detection". In: *IEEE transactions on pattern analysis and machine intelligence* 34.7, pp. 1409–1422.

Kanda, Takayuki, Dylan F Glas, Masahiro Shiomi, and Norihiro Hagita (2009). "Abstracting people's trajectories for social robots to proactively approach customers". In: *IEEE Transactions on Robotics* 25.6, pp. 1382–1396.

Kanhere, O. and T. S. Rappaport (2018). "Position Locationing for Millimeter Wave Systems". In: *Proc. 2018 IEEE Global Communications Conference (GLOBECOM)*. Abu Dhabi, UAE, pp. 206–212.

Karamouzas, Ioannis, Peter Heil, Pascal Van Beek, and Mark H Overmars (2009). "A predictive collision avoidance model for pedestrian simulation". In: *International workshop on motion in games*. Springer, pp. 41–52.

Karamouzas, Ioannis, Brian Skinner, and Stephen J Guy (2014). "Universal power law governing pedestrian interactions". In: *Physical review letters* 113.23, p. 238701.

Karunarathne, Deneth, Yoichi Morales, Takayuki Kanda, and Hiroshi Ishiguro (2018). "Model of side-by-side walking without the robot knowing the goal". In: *International Journal of Social Robotics* 10.4, pp. 401–420.

Kassir, Abdallah and Thierry Peynot (2010). "Reliable automatic camera-laser calibration". In: *Proceedings of the 2010 Australasian Conference on Robotics & Automation*. ARAA.

Katyal, K, Gregory D Hager, and Chien-Ming Huang (2020). "Intent-aware pedestrian prediction for adaptive crowd navigation". In: *2020 International Conference on Robotics and Automation (ICRA)*.

Khambhaita, Harmish and Rachid Alami (2017). "Assessing the social criteria for human-robot collaborative navigation: A comparison of human-aware navigation planners". In: *2017 26th IEEE international symposium on robot and human interactive communication (RO-MAN)*. IEEE, pp. 1140–1145.

— (2020). "Viewing robot navigation in human environment as a cooperative activity". In: *Robotics Research*. Springer, pp. 285–300.

Kielar, Peter M and André Borrmann (2016). "Modeling pedestrians' interest in locations: A concept to improve simulations of pedestrian destination choice". In: *Simulation Modelling Practice and Theory* 61, pp. 47–62.

Kilic, Ergin, Melik Dolen, and A Bugra Koku (2011). "Long-term prediction of hydraulic system dynamics via structured recurrent neural networks". In: *2011 IEEE International Conference on Mechatronics*. IEEE, pp. 330–335.

Kim, Beomjoon and Joelle Pineau (2016). "Socially adaptive path planning in human environments using inverse reinforcement learning". In: *International Journal of Social Robotics* 8.1, pp. 51–66.

Kitazawa, Kay and Taku Fujiyama (2010). "Pedestrian vision and collision avoidance behavior: Investigation of the information process space of pedestrians using an eye tracker". In: *Pedestrian and evacuation dynamics 2008*. Springer, pp. 95–108.

Knapp, Mark L, Judith A Hall, and Terrence G Horgan (2013). *Nonverbal communication in human interaction*. Cengage Learning.

Knepper, Ross A and Daniela Rus (2012). "Pedestrian-inspired sampling-based multi-robot collision avoidance". In: *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, pp. 94–100.

Knorr, Alexander G, Lina Willacker, Joachim Hermsdörfer, Stefan Glasauer, and Melanie Krüger (2016). "Influence of person-and situation-specific characteristics on collision avoidance behavior in human locomotion." In: *Journal of experimental psychology: human perception and performance* 42.9, p. 1332.

Kok, Ven Jyn, Mei Kuan Lim, and Chee Seng Chan (2016). "Crowd behavior analysis: A review where physics meets biology". In: *Neurocomputing* 177, pp. 342–362.

Kothari, Parth, Sven Kreiss, and Alexandre Alahi (2021a). "Human trajectory forecasting in crowds: A deep learning perspective". In: *IEEE Transactions on Intelligent Transportation Systems*.

Kothari, Parth, Brian Sifringer, and Alexandre Alahi (2021b). "Interpretable social anchors for human trajectory forecasting in crowds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15556–15566.

Kretz, Tobias, Jochen Lohmiller, and Peter Sukennik (2018). “Some indications on how to calibrate the social force model of pedestrian dynamics”. In: *Transportation research record* 2672.20, pp. 228–238.

Kretzschmar, Henrik, Markus Spies, Christoph Sprunk, and Wolfram Burgard (2016). “Socially compliant mobile robot navigation via inverse reinforcement learning”. In: *The International Journal of Robotics Research* 35.11, pp. 1289–1307.

Kruse, Thibault, Amit Kumar Pandey, Rachid Alami, and Alexandra Kirsch (2013). “Human-aware robot navigation: A survey”. In: *Robotics and Autonomous Systems* 61.12, pp. 1726–1743.

Kuderer, Markus, Henrik Kretzschmar, Christoph Sprunk, and Wolfram Burgard (2012). “Feature-Based Prediction of Trajectories for Socially Compliant Navigation.” In: *Robotics: science and systems.*

Laganière, Robert, Sébastien Gilbert, and Gerhard Roth (2006). “Robust object pose estimation from feature-based stereo”. In: *IEEE Transactions on Instrumentation and Measurement* 55.4, pp. 1270–1280.

Lam, Chi-Pang, Chen-Tun Chou, Kuo-Hung Chiang, and Li-Chen Fu (2010). “Human-centered robot navigation—towards a harmoniously human–robot coexisting environment”. In: *IEEE Transactions on Robotics* 27.1, pp. 99–112.

Laoudias, C., A. Moreira, S. Kim, S. Lee, L. Wirola, and C. Fischione (2018). “A Survey of Enabling Technologies for Network Localization, Tracking, and Navigation”. In: *IEEE Communications Surveys Tutorials* 20.4, pp. 3607–3644.

Lasota, Przemyslaw A, Terrence Fong, Julie A Shah, et al. (2017). “A survey of methods for safe human-robot interaction”. In: *Foundations and Trends® in Robotics* 5.4, pp. 261–349.

Lee, Donghan, Chang Liu, Yi-Wen Liao, and J Karl Hedrick (2016). “Parallel interacting multiple model-based human motion prediction for motion planning of companion robots”. In: *IEEE Transactions on Automation Science and Engineering* 14.1, pp. 52–61.

Lerner, Alon, Yiorgos Chrysanthou, and Dani Lischinski (2007). “Crowds by example”. In: *Computer graphics forum.* Vol. 26. 3. Wiley Online Library, pp. 655–664.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick (2014). “Microsoft coco: Common objects in context”. In: *European conference on computer vision.* Springer, pp. 740–755.

Lin, Z., T. Lv, and P. T. Mathiopoulos (2018). “3-D Indoor Positioning for Millimeter-Wave Massive MIMO Systems”. In: *IEEE Transactions on Communications* 66.6, pp. 2472–2486.

Liu, Lucia, Daniel Dugas, Gianluca Cesari, Roland Siegwart, and Renaud Dubé (2020). “Robot Navigation in Crowded Environments Using Deep Reinforcement Learning”. In: pp. 5671–5677.

Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg (2016). “Ssd: Single shot multibox detector”. In: *European conference on computer vision.* Springer, pp. 21–37.

Loschonsky, M., C. Feige, O. Rogall, S. Fisun, and L. M. Reindl (2009). “Detection technology for trapped and buried people”. In: *Proc. 2009 IEEE MTT-S International Microwave Workshop on Wireless Sensing, Local Positioning, and RFID.* Cavtat, Croatia, pp. 1–6.

Loyez, C., N. Rolland, and M. Bocquet (2014). “UWB technology applied to millimeter-wave indoor location systems”. In: *Proc. 2014 International Radar Conference.* Lille, France, pp. 1–5.

Lu, David V and William D Smart (2013). “Towards more efficient navigation for robots and humans”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems.* IEEE, pp. 1707–1713.

Lu, Yiping, Aoxiao Zhong, Quanzheng Li, and Bin Dong (2018). “Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations”. In: *International Conference on Machine Learning.* PMLR, pp. 3276–3285.

Luber, Matthias, Johannes A Stork, Gian Diego Tipaldi, and Kai O Arras (2010). “People tracking with human motion predictions from social forces”. In: *2010 IEEE International Conference on Robotics and Automation.* IEEE, pp. 464–469.

Luber, Matthias, Luciano Spinello, Jens Silva, and Kai O Arras (2012). “Socially-aware robot navigation: A learning approach”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 902–907.

Lukezic, Alan, Tomas Vojir, Luka Čehovin Zajc, Jiri Matas, and Matej Kristan (2017). “Discriminative correlation filter with channel and spatial reliability”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6309–6318.

Luo, Linbo, Cheng Chai, Jianfeng Ma, Suiping Zhou, and Wentong Cai (2018). “ProactiveCrowd: Modelling Proactive Steering Behaviours for Agent-Based Crowd Simulation”. In: *Computer Graphics Forum*. Vol. 37. 1. Wiley Online Library, pp. 375–388.

Ma, Wei-Chiu, De-An Huang, Namhoon Lee, and Kris M Kitani (2017). “Forecasting interactive dynamics of pedestrians with fictitious play”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 774–782.

Madrigal, Francisco, Jean-Bernard Hayet, and Frédéric Lerasle (2014). “Intention-aware multiple pedestrian tracking”. In: *2014 22nd International Conference on Pattern Recognition*. IEEE, pp. 4122–4127.

Magnago, V., P. Bevilacqua, L. Palopoli, R. Passerone, D. Fontanelli, and D. Macii (2018). “Optimal Landmark Placement for Indoor Positioning using Context Information and Multi-sensor Data”. In: *Proc. IEEE Int. Instrumentation and Measurement Technology Conference (I2MTC)*. Houston, Texas (US): IEEE, pp. 1–6. ISBN: 978-1-5386-2222-3. DOI: 10.1109/I2MTC.2018.8409809.

Mangalam, Karttikeya, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon (2020). “It Is Not the Journey but the Destination: Endpoint Conditioned Trajectory Prediction”. In: *arXiv preprint arXiv:2004.02025*.

Mangalam, Karttikeya, Yang An, Harshayu Girase, and Jitendra Malik (2021). “From goals, waypoints & paths to long term human trajectory forecasting”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15233–15242.

Manso, Luis J, Ronit R Jorvekar, Diego R Faria, Pablo Bustos, and Pilar Bachiller (2019). “Graph Neural Networks for Human-aware Social Navigation”. In: *arXiv preprint arXiv:1909.09003*.

Masi, Filippo, Ioannis Stefanou, Paolo Vannucci, and Victor Maffi-Berthier (2021). “Thermodynamics-based Artificial Neural Networks for constitutive modeling”. In: *Journal of the Mechanics and Physics of Solids* 147, p. 104277.

Mavrogiannis, Christoforos, Alena M Hutchinson, John Macdonald, Patrícia Alves-Oliveira, and Ross A Knepper (2019). “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)*. IEEE, pp. 421–430.

Mavrogiannis, Christoforos I, Wil B Thomason, and Ross A Knepper (2018). “Social momentum: A framework for legible navigation in dynamic multi-agent environments”. In: *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 361–369.

Mazor, Efim, Amir Averbuch, Yakov Bar-Shalom, and Joshua Dayan (1998). “Interacting multiple model methods in target tracking: a survey”. In: *IEEE Transactions on aerospace and electronic systems* 34.1, pp. 103–123.

Mead, Ross and Maja J Matarić (2017). “Autonomous human–robot proxemics: socially aware navigation based on interaction potential”. In: *Autonomous Robots* 41.5, pp. 1189–1201.

Mendrzik, R., H. Wymeersch, and G. Bauch (2018). “Joint Localization and Mapping Through Millimeter Wave MIMO in 5G Systems”. In: *Proc. 2018 IEEE Global Communications Conference (GLOBECOM)*. Abu Dhabi, UAE, pp. 1–6.

Miller, Justin, Andres Hasfura, Shih-Yuan Liu, and Jonathan P How (2016). “Dynamic arrival rate estimation for campus Mobility On Demand network graphs”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 2285–2292.

Miller, Tim (2019). “Explanation in artificial intelligence: Insights from the social sciences”. In: *Artificial intelligence* 267, pp. 1–38.

Mokrzycki, Wojciech (1994). “Constraction of 3D deph map from binocular stereo”. In: *Proceedings of 5th International School on Computer Vision & Graphics Microcomputer*.

Möller, Ronja, Antonino Furnari, Sebastiano Battiato, Aki Härmä, and Giovanni Maria Farinella (2021). "A survey on human-aware robot navigation". In: *Robotics and Autonomous Systems* 145, p. 103837.

Monteiro, Gonçalo, Cristiano Premebida, Paulo Peixoto, and Urbano Nunes (2006). "Tracking and classification of dynamic obstacles using laser range finder and vision". In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–7.

Moussaïd, Mehdi, Dirk Helbing, and Guy Theraulaz (2011). "How simple rules determine pedestrian behavior and crowd disasters". In: *Proceedings of the National Academy of Sciences* 108.17, pp. 6884–6888.

Munkres, James (1957). "Algorithms for the assignment and transportation problems". In: *Journal of the society for industrial and applied mathematics* 5.1, pp. 32–38.

Mutlu, Bilge and Jodi Forlizzi (2008). "Robots in organizations: the role of workflow, social, and environmental factors in human-robot interaction". In: *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction.* ACM, pp. 287–294.

Nam, Cao Nguyen Khoa, Hee Jun Kang, and Young Soo Suh (2014). "Golf swing motion tracking using inertial sensors and a stereo camera". In: *IEEE Transactions on Instrumentation and measurement* 63.4, pp. 943–952.

Narayanan, Venkatraman, Bala Murali Manoghar, Vishnu Sashank Dorbala, Dinesh Manocha, and Aniket Bera (2020). "ProxEmo: Gait-based Emotion Learning and Multi-view Proxemic Fusion for Socially-Aware Robot Navigation". In: *arXiv preprint arXiv:2003.01062.*

Nazemzadeh, P., D. Fontanelli, D. Macii, and L. Palopoli (2017). "Indoor Localization of Mobile Robots through QR Code Detection and Dead Reckoning Data Fusion". In: *IEEE/ASME Transactions on Mechatronics* 22.6, pp. 2588–2599. ISSN: 1083-4435. DOI: 10.1109/TMECH.2017.2762598.

Neubeck, Alexander and Luc Van Gool (2006). "Efficient non-maximum suppression". In: *18th International Conference on Pattern Recognition (ICPR'06).* Vol. 3. IEEE, pp. 850–855.

Ng, H. J., W. Ahmad, M. Kucharski, J. Lu, and D. Kissinger (2017). "Highly-miniaturized 2-channel mm-wave radar sensor with on-chip folded dipole antennas". In: *Proc. IEEE Radio Frequency Integrated Circuits Symposium (RFIC).* Honolulu, HI, USA, pp. 368–371.

Nguyen, Thien-Minh, Muqing Cao, Shenghai Yuan, Yang Lyu, Thien Hoang Nguyen, and Lihua Xie (2021). "Viral-fusion: A visual-inertial-ranging-lidar sensor fusion approach". In: *IEEE Transactions on Robotics.*

Nikdel, Payam, Rakesh Shrestha, and Richard Vaughan (2018). "The hands-free push-cart: Autonomous following in front by predicting user trajectory around obstacles". In: *2018 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 4548–4554.

Nishimura, Mai and Ryo Yonetani (2020). "L2B: Learning to Balance the Safety-Efficiency Trade-off in Interactive Crowd-aware Robot Navigation". In: *arXiv preprint arXiv:2003.09207.*

Nishitani, Ippei, Tetsuya Matsumura, Mayumi Ozawa, Ayanori Yorozu, and Masaki Takahashi (2015). "Human-centered X–Y–T space path planning for mobile robot in dynamic environments". In: *Robotics and Autonomous Systems* 66, pp. 18–26.

Ogawa, Takashi, Hiroshi Sakai, Yasuhiro Suzuki, Kiyokazu Takagi, and Katsuhiro Morikawa (2011). "Pedestrian detection and tracking using in-vehicle lidar for automotive application". In: *2011 IEEE Intelligent Vehicles Symposium (IV).* IEEE, pp. 734–739.

Ohki, Takeshi, Keiji Nagatani, and Kazuya Yoshida (2010). "Collision avoidance method for mobile robot considering motion and personal spaces of evacuees". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems.* IEEE, pp. 1819–1824.

Olfati-Saber, Reza (2006). "Flocking for multi-agent dynamic systems: Algorithms and theory". In: *IEEE Transactions on automatic control* 51.3, pp. 401–420.

Oli, Siddharth, Bruno L'Esperance, and Kamal Gupta (2013). "Human Motion Behaviour Aware Planner (HMBAP) for path planning in dynamic human environments". In: *2013 16th International Conference on Advanced Robotics (ICAR).* IEEE, pp. 1–7.

Olivier, A., G. Bielsa, I. Tejado, M. Zorzi, J. Widmer, and P. Casari (2016). “Lightweight Indoor Localization for 60-GHz Millimeter Wave Systems”. In: *Proc. 13th Annual IEEE Int. Conf. on Sensing, Communication, and Networking (SECON).* London, UK, pp. 1–9.

Olivier, Anne-Hélène, Antoine Marin, Armel Crétual, Alain Berthoz, and Julien Pettré (2013). “Collision avoidance between two walkers: Role-dependent strategies”. In: *Gait & posture* 38.4, pp. 751–756.

Osher, Stanley and James A Sethian (1988). “Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations”. In: *Journal of computational physics* 79.1, pp. 12–49.

Pacchierotti, Elena, Henrik I Christensen, and Patric Jensfelt (2006). “Evaluation of passing distance for social robots”. In: *ROMAN 2006-The 15th IEEE International Symposium on Robot and Human Interactive Communication.* IEEE, pp. 315–320.

Papenmeier, Frank, Meike Uhrig, and Alexandra Kirsch (2019). “Human understanding of robot motion: the role of velocity and orientation”. In: *International Journal of Social Robotics* 11.1, pp. 75–88.

Parisi, Daniel R, Pablo A Negri, and Luciana Bruno (2016). “Experimental characterization of collision avoidance in pedestrian dynamics”. In: *Physical Review E* 94.2, p. 022318.

Park, Jin Hyoung, Francisco Arturo Rojas, and Hyun Seung Yang (2013). “A collision avoidance behavior model for crowd simulation based on psychological findings”. In: *Computer Animation and Virtual Worlds* 24.3-4, pp. 173–183.

Park, P., S. Kim, S. Woo, and C. Kim (2014). “A Centimeter Resolution, 10 m Range CMOS Impulse Radio Radar for Human Motion Monitoring”. In: *IEEE Journal of Solid-State Circuits* 49.5, pp. 1125–1134.

Pasquale, Frank (2017). “Toward a fourth law of robotics: Preserving attribution, responsibility, and explainability in an algorithmic society”. In: *Ohio St. LJ* 78, p. 1243.

Patompak, Pakpoom, Sungmoon Jeong, Itthisek Nilkhamhang, and Nak Young Chong (2019). “Learning Proxemics for Personalized Human–Robot Social Interaction”. In: *International Journal of Social Robotics*, pp. 1–14.

Pellegrini, Stefano, Andreas Ess, Konrad Schindler, and Luc Van Gool (2009). “You’ll never walk alone: Modeling social behavior for multi-target tracking”. In: *2009 IEEE 12th International Conference on Computer Vision.* IEEE, pp. 261–268.

Pettré, Julien, Jan Ondřej, Anne-Hélène Olivier, Armel Cretual, and Stéphane Donikian (2009). “Experiment-based modeling, simulation and validation of interactions between virtual walkers”. In: *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation.* ACM, pp. 189–198.

Pfeiffer, Mark, Giuseppe Paolo, Hannes Sommer, Juan Nieto, Rol Siegwart, and Cesar Cadena (2018). “A data-driven model for interaction-aware pedestrian motion prediction in object cluttered environments”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 1–8.

Pradeep, Yazhini C, Zhu Ming, Manuel Del Rosario, and Peter CY Chen (2016). “Human-inspired robot navigation in unknown dynamic environments”. In: *2016 IEEE International Conference on Mechatronics and Automation.* IEEE, pp. 971–976.

Prédhumeau, Manon, Julie Dugdale, and Anne Spalanzani (2019). “Adapting the Social Force Model for Low Density Crowds in Open Environments”. In:

Premebida, Cristiano, Oswaldo Ludwig, and Urbano Nunes (2009). “LIDAR and vision-based pedestrian detection system”. In: *Journal of Field Robotics* 26.9, pp. 696–711.

Presti, Liliana Lo and Marco La Cascia (2016). “3D skeleton-based human action classification: A survey”. In: *Pattern Recognition* 53, pp. 130–147.

Pun, Chi-Man, Xiao-Chen Yuan, and Xiu-Li Bi (2015). “Image forgery detection using adaptive oversegmentation and feature point matching”. In: *IEEE Transactions on Information Forensics and Security* 10.8, pp. 1705–1716.

Qin, Tong, Kailiang Wu, and Dongbin Xiu (2019). “Data driven governing equations approximation using deep neural networks”. In: *Journal of Computational Physics* 395, pp. 620–635. ISSN: 10902716. DOI: 10.1016/j.jcp.2019.06.042.

Raissi, Maziar, Paris Perdikaris, and George E Karniadakis (2019). “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational physics* 378, pp. 686–707.

Ratsamee, Photchara, Yasushi Mae, Kenichi Ohara, Tomohito Takubo, and Tatsuo Arai (2013). “Human–robot collision avoidance using a modified social force model with body pose and face orientation”. In: *International Journal of Humanoid Robotics* 10.01, p. 1350008.

Ratsamee, Photchara, Yasushi Mae, Kazuto Kamiyama, Mitsuhiro Horade, Masaru Kojima, and Tatsuo Arai (2015). “Social interactive robot navigation based on human intention analysis from face orientation and human path prediction”. In: *Robomech Journal* 2.1, p. 11.

Redmon, Joseph and Ali Farhadi (2017). “YOLO9000: better, faster, stronger”. In: *arXiv preprint.*

— (2018a). “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767.*

— (2018b). “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767.*

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi (2016a). “You Only Look Once: Unified, Real-Time Object Detection”. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

— (2016b). “You only look once: Unified, real-time object detection”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788.

Rehder, Eike, Florian Wirth, Martin Lauer, and Christoph Stiller (2018). “Pedestrian prediction by planning using deep neural networks”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA).* IEEE, pp. 1–5.

Rios-Martinez, Jorge, Anne Spalanzani, and Christian Laugier (2015). “From proxemics theory to socially-aware navigation: A survey”. In: *International Journal of Social Robotics* 7.2, pp. 137–153.

Ristani, Ergys and Carlo Tomasi (2018). “Features for multi-target multi-camera tracking and re-identification”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6036–6046.

Roehrl, Manuel A, Thomas A Runkler, Veronika Brandtstetter, Michel Tokic, and Stefan Obermayer (2020). “Modeling system dynamics with physics-informed neural networks based on Lagrangian mechanics”. In: *IFAC-PapersOnLine* 53.2, pp. 9195–9200.

Rohrig, C. and L. Telle (2011). “Real-Time Communication and Localization for a Swarm of Mobile Robots Using IEEE 802.15.4a CSS”. In: *2011 IEEE Vehicular Technology Conference (VTC Fall).* San Francisco, CA, USA, pp. 1–5.

Roth, Peter M., Martin Hirzer, Martin Köstinger, Csaba Beleznai, and Horst Bischof (2014). “Mahalanobis Distance Learning for Person Re-Identification”. In: *Person Re-Identification.* Ed. by Shaogang Gong, Marco Cristani, Shuicheng Yan, and Chen C. Loy. Advances in Computer Vision and Pattern Recognition. London, United Kingdom: Springer, pp. 247–267.

Rudenko, Andrey, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrila, and Kai O Arras (2019). “Human Motion Trajectory Prediction: A Survey”. In: *arXiv preprint arXiv:1905.06113.*

Rudenko, Andrey, Tomasz P Kucner, Chittaranjan S Swaminathan, Ravi T Chadalavada, Kai O Arras, and Achim J Lilienthal (2020). “THÖR: Human-Robot Navigation Data Collection and Accurate Motion Trajectories Dataset”. In: *IEEE Robotics and Automation Letters* 5.2, pp. 676–682.

Saarinen, Jari, Henrik Andreasson, Todor Stoyanov, and Achim J Lilienthal (2013). “Normal distributions transform Monte-Carlo localization (NDT-MCL)”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems.* IEEE, pp. 382–389.

Sadeghian, Amir, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese (2019). “Sophie: An attentive gan for predicting paths compliant to social and physical constraints”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1349–1358.

Sakour, Ibraheem and Huosheng Hu (2017). “Robot-assisted crowd evacuation under emergency situations: A survey”. In: *Robotics* 6.2, p. 8.

Sardar, Aziez, Michiel Joosse, Astrid Weiss, and Vanessa Evers (2012). “Don’t stand so close to me: users’ attitudinal and behavioral responses to personal space invasion by robots”. In: *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*. ACM, pp. 229–230.

Satake, Satoru, Takayuki Kanda, Dylan F Glas, Michita Imai, Hiroshi Ishiguro, and Norihiro Hagita (2009). “How to approach humans?: strategies for social robots to initiate interaction”. In: *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*. ACM, pp. 109–116.

— (2012). “A robot that approaches pedestrians”. In: *IEEE Transactions on Robotics* 29.2, pp. 508–524.

Sathyamoorthy, Adarsh Jagan, Jing Liang, Utsav Patel, Tianrui Guan, Rohan Chandra, and Dinesh Manocha (2020). “Densecavoid: Real-time navigation in dense crowds using anticipatory behaviors”. In: *arXiv preprint arXiv:2002.03038*.

Saunderson, Shane and Goldie Nejat (2019). “How robots influence humans: A survey of nonverbal communication in social human–robot interaction”. In: *International Journal of Social Robotics* 11.4, pp. 575–608.

Schneider, Nicolas and Dariu M Gavrila (2013). “Pedestrian path prediction with recursive bayesian filters: A comparative study”. In: *German Conference on Pattern Recognition*. Springer, pp. 174–183.

Schöller, Christoph, Vincent Aravantinos, Florian Lay, and Alois Knoll (2020). “What the Constant Velocity Model Can Teach Us About Pedestrian Motion Prediction”. In: *IEEE Robotics and Automation Letters*.

Schulz, Andreas T and Rainer Stiefelhagen (2015). “A controlled interactive multiple model filter for combined pedestrian intention recognition and path prediction”. In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE, pp. 173–178.

Sebastian, Meera, Santosh Balajee Banisetty, and David Feil-Seifer (2017). “Socially-aware navigation planner using models of human-human interaction”. In: *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE, pp. 405–410.

Seidl, David R and Robert D Lorenz (1991). “A structure by which a recurrent neural network can approximate a nonlinear dynamic system”. In: *IJCNN-91-Seattle International Joint Conference on Neural Networks*. Vol. 2. IEEE, pp. 709–714.

Senanayake, Ransalu and Fabio Ramos (2018). “Directional grid maps: modeling multimodal angular uncertainty in dynamic environments”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 3241–3248.

Shamsfakhr, Farhad, Alessandro Antonucci, Luigi Palopoli, David Macii, and Daniele Fontanelli (2022). “Indoor Localisation Uncertainty Control based on Wireless Ranging for Robots Path Planning”. In: *IEEE Transactions on Instrumentation and Measurement* 71, pp. 1–11. DOI: 10.1109/TIM.2022.3147316.

Sheen, D. M., D. L. McMakin, and T. E. Hall (2007). “Near Field Imaging at Microwave and Millimeter Wave Frequencies”. In: *Proc. 2007 IEEE/MTT-S International Microwave Symposium*. Honolulu, HI, USA, pp. 1693–1696.

Shinohara, Shigenobu, Hirofumi Yoshida, Hiroaki Ikeda, K-l Nishide, and Masao Sumi (1992). “Compact and high-precision range finder with wide dynamic range and its application”. In: *IEEE transactions on instrumentation and measurement* 41.1, pp. 40–44.

Shirmohammadi, Shervin and Alessandro Ferrero (2014). “Camera as the instrument: the rising trend of vision based measurement”. In: *IEEE Instrumentation & Measurement Magazine* 17.3, pp. 41–47.

Silva, Grimaldo and Thierry Fraichard (2017). “Human robot motion: A shared effort approach”. In: *2017 European Conference on Mobile Robots (ECMR)*. IEEE, pp. 1–6.

Simon, Tomas, Hanbyul Joo, Iain Matthews, and Yaser Sheikh (2017). “Hand Keypoint Detection in Single Images using Multiview Bootstrapping”. In: *CVPR*.

Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich (2015). “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.

Tadokoro, Satoshi, Masaki Hayashi, Yasuhiro Manabe, Yoshihiro Nakami, and Toshi Takamori (1995). “On motion planning of mobile robots which coexist and cooperate with human”. In: *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*. Vol. 2. IEEE, pp. 518–523.

Tamas, L, M Popa, Gh Lazea, I Szoke, and A Majdik (2010). “LiDAR and vision based people detection and tracking”. In: *Journal of Control Engineering and Applied Informatics* 12.2, pp. 30–35.

Tamura, Yusuke, Tomohiro Fukuzawa, and Hajime Asama (2010). “Smooth collision avoidance in human-robot coexisting environment”. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 3887–3892.

Toyoshima, Atsuki, Nozomi Nishino, Daisuke Chugo, Satoshi Muramatsu, Sho Yokota, and Hiroshi Hashimoto (2018). “Autonomous Mobile Robot Navigation: Consideration of the Pedestrian’s Dynamic Personal Space”. In: *2018 IEEE 27th International Symposium on Industrial Electronics (ISIE)*. IEEE, pp. 1094–1099.

Trautman, Peter and Andreas Krause (2010). “Unfreezing the robot: Navigation in dense, interacting crowds”. In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 797–803.

Truong, Xuan-Tung, Voo Nyuk Yoong, and Trung-Dung Ngo (2015). “RGB-D and laser data fusion-based human detection and tracking for socially aware robot navigation framework”. In: *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 608–613.

Vadakkepat, Prahlad and Liu Jing (2006). “Improved particle filter in sensor fusion for tracking randomly moving object”. In: *IEEE Transactions on Instrumentation and Measurement* 55.5, pp. 1823–1832.

Valero-Gomez, Alberto, Javier V Gomez, Santiago Garrido, and Luis Moreno (2013). “The path to efficiency: Fast marching method for safer, more efficient mobile robot trajectories”. In: *IEEE Robotics & Automation Magazine* 20.4, pp. 111–120.

Vassallo, Christian, Anne-Hélène Olivier, Philippe Souères, Armel Crétual, Olivier Stasse, and Julien Pettré (2017). “How do walkers avoid a mobile robot crossing their way?” In: *Gait & posture* 51, pp. 97–103.

— (2018). “How do walkers behave when crossing the way of a mobile robot that replicates human interaction rules?” In: *Gait & posture* 60, pp. 188–193.

Vicente, Alfredo Gardel, Ignacio Bravo Munoz, Pedro Jiménez Molina, and José Luis Lázaro Galilea (2009). “Embedded vision modules for tracking and counting people”. In: *IEEE Transactions on Instrumentation and Measurement* 58.9, pp. 3004–3011.

Vinciarelli, Alessandro, Maja Pantic, and Hervé Bourlard (2009). “Social signal processing: Survey of an emerging domain”. In: *Image and vision computing* 27.12, pp. 1743–1759.

Wan, Zhiqiang, Xuemin Hu, Haibo He, and Yi Guo (2017). “A learning based approach for social force model parameter estimation”. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 4058–4064.

Wang, D, DK Liu, NM Kwok, and KJ Waldron (2008). “A subgoal-guided force field method for robot navigation”. In: *2008 IEEE/ASME International Conference on Mechtronic and Embedded Systems and Applications*. IEEE, pp. 488–493.

Wang, Mengmeng, Yong Liu, Daobilige Su, Yufan Liao, Lei Shi, Jinhong Xu, and Jaime Valls Miro (2018). “Accurate and real-time 3-D tracking for the following robots by fusing vision and ultrasonar information”. In: *IEEE/ASME Transactions On Mechatronics* 23.3, pp. 997–1006.

Wang, Qiang, Yan Liu, and Juan Chen (2012). “Accurate indoor tracking using a mobile phone and non-overlapping camera sensor networks”. In: *Instrumentation and Measurement Technology Conference (I2MTC), 2012 IEEE International*. IEEE, pp. 2022–2027.

Watanabe, Atsushi, Tetsushi Ikeda, Yoichi Morales, Kazuhiko Shinozawa, Takahiro Miyashita, and Norihiro Hagita (2015). “Communicating robotic navigational intentions”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5763–5769.

Wei, Shih-En, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh (2016). “Convolutional pose machines”. In: *CVPR*.

Wiltshire, Travis J, Sierra L Snow, Emilio JC Lobato, and Stephen M Fiore (2014). "Leveraging social judgment theory to examine the relationship between social cues and signals in human-robot interactions". In: *Proceedings of the human factors and ergonomics society annual meeting*. Vol. 58. 1. SAGE Publications Sage CA: Los Angeles, CA, pp. 1336–1340.

Wiltshire, Travis J, Samantha F Warta, Daniel Barber, and Stephen M Fiore (2017). "Enabling robotic social intelligence by engineering human social-cognitive mechanisms". In: *Cognitive Systems Research* 43, pp. 190–207.

Wolcott, Ryan W and Ryan M Eustice (2014). "Visual localization within lidar maps for automated urban driving". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 176–183.

Wu, Jingyuan, Johannes Ruenz, and Matthias Althoff (2018). "Probabilistic map-based pedestrian motion prediction taking traffic participants into consideration". In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 1285–1292.

Wunderlich, Sarah, Johannes Schmölz, and Kolja Kühnlenz (2017). "Follow me: A simple approach for person identification and tracking". In: *2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*. IEEE, pp. 1609–1614.

Xu, Feiyu, Hans Uszkoreit, Yangzhou Du, Wei Fan, Dongyan Zhao, and Jun Zhu (2019). "Explainable AI: A brief survey on history, research areas, approaches and challenges". In: *CCF international conference on natural language processing and Chinese computing*. Springer, pp. 563–574.

Y. Bar-Shalom X. Rong Li, T. Kirubarajan (2001). *Estimation with Application to Tracking and Navigation – Theory, Algorithm and Software*. John Wiley and Sons.

Ye, Chen and Phil Webb (2009). "A sub goal seeking approach for reactive navigation in complex unknown environments". In: *Robotics and Autonomous Systems* 57.9, pp. 877–888.

Zanlungo, Francesco, Tetsushi Ikeda, and Takayuki Kanda (2011). "Social force model with explicit collision prediction". In: *EPL (Europhysics Letters)* 93.6, p. 68005.

Zhang, C., M. Kuhn, B. Merkl, M. Mahfouz, and A. E. Fathy (2006). "Development of an UWB Indoor 3D Positioning Radar with Millimeter Accuracy". In: *2006 IEEE MTT-S International Microwave Symposium Digest*. San Francisco, CA, USA, pp. 106–109.

Zhang, Daniel, Saurabh Mishra, Erik Brynjolfsson, John Etchemendy, Deep Ganguli, Barbara Grosz, Terah Lyons, James Manyika, Juan Carlos Niebles, Michael Sellitto, et al. (2021). "The ai index 2021 annual report". In: *arXiv preprint arXiv:2103.06312*.

Zhang, Qilong and Robert Pless (2004). "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. IEEE, pp. 2301–2306.

Zhang, Zhe and Limin Jia (2020). "Direction-decision learning based pedestrian flow behavior investigation". In: *IEEE Access*.

Zhen, Weikun, Yaoyu Hu, Jingfeng Liu, and Sebastian Scherer (2019). "A joint optimization approach of lidar-camera fusion for accurate dense 3-d reconstructions". In: *IEEE Robotics and Automation Letters* 4.4, pp. 3585–3592.

Zheng, Liang, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian (2015). "Scalable person re-identification: A benchmark". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1116–1124.

Ziebart, Brian D, Nathan Ratliff, Garratt Gallagher, Christoph Mertz, Kevin Peterson, J Andrew Bagnell, Martial Hebert, Anind K Dey, and Siddhartha Srinivasa (2009). "Planning-based prediction for pedestrians". In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 3931–3936.

Zitouni, M Sami, Andrzej Sluzek, and Harish Bhaskar (2019). "Visual analysis of socio-cognitive crowd behaviors for surveillance: A survey and categorization of trends and methods". In: *Engineering Applications of Artificial Intelligence* 82, pp. 294–312.