Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Department of Information Engineering
Master Thesis in Computer Engineering

People motion prediction for social navigation in crowded environments via context-based learning

Master Candidate

Andrea Savio

Student ID 2052420

Supervisor

Prof. Nicola Bellotto

Co-supervisor

Prof. Gloria Beraldo

Academic Year 2022/2023
12/10/23

Abstract

Human motion trajectory prediction plays a crucial role in enabling robots to navigate and interact safely and efficiently in crowded environments: proactive decision-making, obstacle avoidance, and natural human-robot interaction benefit greatly from the ability to adapt the robot's motion to the future. However, human motion trajectory prediction in social navigation poses significant challenges due to the complex nature of human behavior and the dynamic nature of social interactions. Many approaches focus on understanding how humans move in the world by learning how they act on the map from a bird-eye camera placed on a tall structure. Therefore, adapting these methods based on the onboard sensors of the robots is not straightforward: it is necessary that they possess a map of the area in which they are working and that they update it with the humans they detect using their sensors. Only after that, predictions can be made. Given the dynamics of crowded environments with both humans and obstacles moving often and at different speeds, the computational complexity due to the introduction and the continuous update of these maps could affect the robot's performance and reactivity. Moreover, people's behavior changes dramatically in relation to the context in which they are moving. To face these challenges, we propose a method for predicting human motion trajectories using only the robot's onboard sensors, namely a 2D lidar and an RGB-D camera, and based on context and on deep learning techniques trained on a state-of-the-art dataset, JackRabbot. The method employs a Long Short Term Memory model for learning trajectories, while the network learns in parallel from the data extracted from the context of the environment, using unsupervised learning. The method is then tested on popular social navigation datasets, ATC, ETH and UCY. Results show that this approach is slightly better when compared to a similar model based only on trajectory-learning. Finally, the model is tested in real life on the TIAGo++ robot situated at the Autonomous Robotics Laboratory of the University of Padova.

Abstract

La previsione delle traiettorie di movimento delle persone gioca un ruolo cruciale nel consentire ai robot di navigare e interagire in modo sicuro ed efficiente in ambienti affollati: il processo decisionale, l'elusione degli ostacoli e l'interazione naturale uomo-robot traggono grandi vantaggi dalla capacità di adattare il movimento del robot al futuro. Tuttavia, la previsione della traiettoria del movimento umano nella navigazione sociale pone sfide significative a causa della natura complessa del comportamento umano e della natura dinamica delle interazioni sociali. Molti approcci si concentrano sulla comprensione di come gli esseri umani si muovono nel mondo imparando come agiscono sulla mappa da una telecamera posizionata su una struttura alta. Adattare quindi questi metodi ai sensori di bordo dei robot non è semplice: è necessario che questi posseggano una mappa dell'area in cui lavorano e la aggiornino con gli esseri umani che rilevano tramite i loro sensori. Solo dopo si potranno fare delle previsioni. Considerata la dinamica degli ambienti affollati, con esseri umani e ostacoli che si muovono spesso e a velocità diverse, la complessità computazionale dovuta all'introduzione e al continuo aggiornamento di queste mappe potrebbe influenzare le prestazioni e la reattività del robot. Inoltre, il comportamento delle persone cambia radicalmente in relazione al contesto in cui si muovono. Per affrontare queste sfide, proponiamo un metodo per prevedere le traiettorie del movimento umano utilizzando solo i sensori di bordo del robot, vale a dire un lidar 2D e una fotocamera RGB-D, e basato sul contesto e su tecniche di deep learning addestrate su un dataset all'avanguardia, JackRabbot. Questo approccio sfrutta un modello Long Short Term Memory per imparare dalle traiettorie delle persone, mentre la rete impara in parallelo dalle informazioni estratte dal contesto attraverso metodi di unsupervised learning. Questo approccio è poi testato su importanti dataset usati nella social nvigation, ATC, ETH e UCY. I risultati dimostrano che questo approccio è leggermente migliore di un modello simile ma basato solo sulle traiettorie. Infine, il progetto è testato dal vivo sul robot TIAGo++ presente nel laboratorio di Autonomous Robotics dell'Università degli Studi di Padova.

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

SLAM Simultaneous Localization And Mapping

HRI Human-Robot Interaction

CSV Comma Separated Values

RL Reinforcement Learning

RRT Rapidly exploring Random Tree

DQN Deep-Q Network

DDQN Double Deep-Q Network

DDPG Deep Deterministic Policy Gradient

A3C Asynchronous Advantage Actor-Critic

RNN Recurrent Neural Networks

LSTM Long Short-Term Memory

GRU Gated Recurrent Unit

SFM Social Force Models

AT Anticipative Turn

ARP Anticipative Robot and Pedestrian's Propagation

APP Anticipative Pedestrian's PropagationAPP

HATEB Human Aware Timed Elastic Band

ASP Adaptive Social Planner

GUI Graphical User Interface

QTC Qualitative Trajectory Calculus

1 Introduction

1.1 Human-Aware Navigation

The problem of human presence in the environment may seem negligible: one could suppose that a human, after all, can be treated as a dynamic obstacle moving in the environment. However, in reality, that is not the case. Humans add a whole level of complexity to the robot navigation task: when dealing with humans it is necessary to take into account both physical safety and psychological safety. Physical safety consists in maintaining a minimum safe distance between robots and humans at all times, while to respect psychological safety the robot cannot afford to cause stress or annoyance to human beings, while also behaving as naturally as possible. To better characterize these rules, the following definitions are used in the literature [25]:

- **Comfort**: the absence of annoyance and stress for humans in interactions with robots;
- **Naturalness**: the similarity between robots and humans in low level behavior patterns;
- **Sociability**: the adherence to explicit high-level cultural conventions.

In short, in order to avoid discomfort to humans it is not sufficient to implement obstacle avoidance: the robot must also respect social norms and its intentions must be easily recognizable, as shown in Figure 1.1. Consequently, robotics has become an interdisciplinary field, involving mainly engineering but also, to a certain degree, psychology and sociology. In particular, the field of proxemics has proven useful for robotics. Proxemics [13] is the study of human use of space and the effects that population density has on behaviour, communication, and social interaction. According to proxemics, humans respect a virtual personal space around each other, and this model can be applied to robotics to comply with the concept of psychological safety. Another aspect of humans is their unpredictability: while generic dynamic obstacles behave similarly to one another, humans are highly variable in terms of personalities, cultures and behaviors. Different humans react differently to a stimulus, move differently and behave differently in the environment. This adds a layer of complexity in trying to generalize a robot behavior so that it is socially acceptable for everyone. Finally, other minor unrelated problems that may occur when dealing with humans regard the difficulty in detecting and tracking humans generally due to sensor types or position. For example, if the sensor is installed on the base of the robot, it may only be able to detect the legs of a person; if it is higher it may see only the head. Similarly, if the sensor is for example a camera, illumination changes may cause difficulties in human detection. The research field that embeds all the previously described aspects is known as Social Navigation.

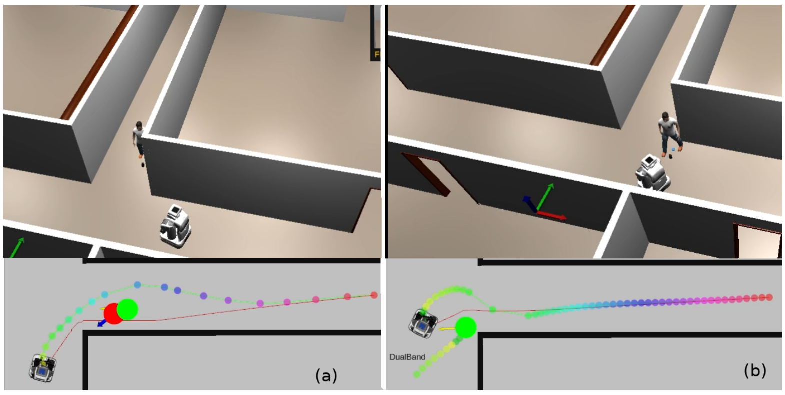


Figure 1.1: An example of good behavior in social navigation. The planner accounts for the possibility of encountering a person in the blind spot behind the wall (a), and modifies the trajectory as soon as the robot detects the human (b) [50].

1.1.1 Human-Robot Interaction

In some cases, robots also have to interact with people, for example by handing them objects. In this scenario, the concepts of maintaining a minimum distance from humans exposed previously need to be revised. The robot needs to grasp the context in which it is operating and act accordingly. In general, human-robot interactions (HRI) can be of three types:

- **Physical**: the robot and the person are in direct or indirect physical contact. The focus is on guaranteeing the safety of the humans.
- **Cognitive**: robot and person are involved in joint work.
- **Social/emotional**: the robot influences how the human responds both explicitly or implicitly.

In the cognitive type of interaction, the robot and the person are thinking and reacting to the same world and task, and this type of interaction to accomplish a goal was named joint cognitive system. These cognitive interactions can be divided in:

- **Taskable agent**: the robot is treated as an independent agent with a certain degree of autonomy and initiative.
- **Remote presence**: the robot is an extension of the human in an environment in which the human cannot be.
- **Assistive agent**: the robot is alongside the human to assist them.

HRI research includes a psychological aspect as well: for example, if the robot is a highly realistic humanoid, some people may feel a sense of uneasiness when interacting with it. This phenomenon is know as the "uncanny valley" effect [37].

1.2 People Motion Prediction

As explained, simply treating people as generic obstacles to avoid during navigation is not enough. Therefore, how do we deal with people in the environment in which we want to deploy our robot?

Firstly, it is important to address the problems related to human behaviour. Human behaviour is extremely unpredictable and highly dynamic: each person has individual characteristics, partly defined by one's culture, language, body and the situation in which they are moving. People also have social interactions with other people, which sometimes include moving in pairs, groups or standing still, waiting and so on. Every person movement derives from their past experiences and future intentions.

However, to make a robot operate in a crowded space, we surely need to address the unpredictability and dynamicity of human motion. Fortunately, there are ways to use a person's movement to predict their future positions in the world, so that we can ensure that our robot will behave as socially as possible.

In origin, this task would be accomplished using probabilistic models or social attention mechanisms: these approaches tried to model uncertainty, the former via probability and the latter via information retrieved from neighbouring agents in the environment.

Recently, with the explosion in popularity of machine learning, new methodologies quickly became popular. For example, one could employ reinforcement learning techniques to have the robot learn optimal decision-making policies to avoid people and satisfy social navigation constraints during movement. Another possibility involves using data-driven approaches, i.e. methods that leverage machine learning on large datasets to accurately predict the trajectories taken by people.

1.3 Applications

The social navigation with motion prediction field of study can be applied to many different fields:

- **Industry**: industrial robots are already very popular in manufacturing, but they typically are automated robots that quickly perform repetitive tasks and are sealed off from humans in their environment to avoid incidents and injuries. Autonomous robots could be employed alongside humans to aid them in moving heavy objects or operating complex tools, for example like in Figure 1.2;

- **Services**: service robots are designed to assist humans with various tasks, such as cleaning, cooking, or providing care to the elderly or disabled. These robots must navigate around humans and objects in a home or care environment safely;

- **Autonomous driving**: autonomous driving vehicles have gained attention in the last decade. Roads are usually populated by humans and therefore it is necessary to consider them while navigating to travel safely;

- **Healthcare**: healthcare robots are designed to assist medical professionals with tasks such as patient monitoring, medication delivery, and surgical procedures. Human-aware navigation enables these robots to navigate around patients and healthcare personnel safely. As an example, see Figure 1.3;

- **Agriculture**: autonomous robots can be used in agricultural tasks. For example, milking robots can assist humans when dealing with animals.

Another example of autonomous mobile robot used in agriculture is shown in Figure 1.4;

- **Space exploration**: as NASA plans to set foot on the Moon again with the Artemis program, and as Mars is the target of future crewed missions, rovers capable of working alongside humans(for example, in Figure 1.5) will be crucial for the success of these operations.
- **Education**: robots can be employed in classrooms or in museums. MINERVA [54] is an example of robot deployed in a museum.

These are only some of the possible uses of autonomous social robots. With future hardware and software advancements, the list of possible applications of this field can only grow.

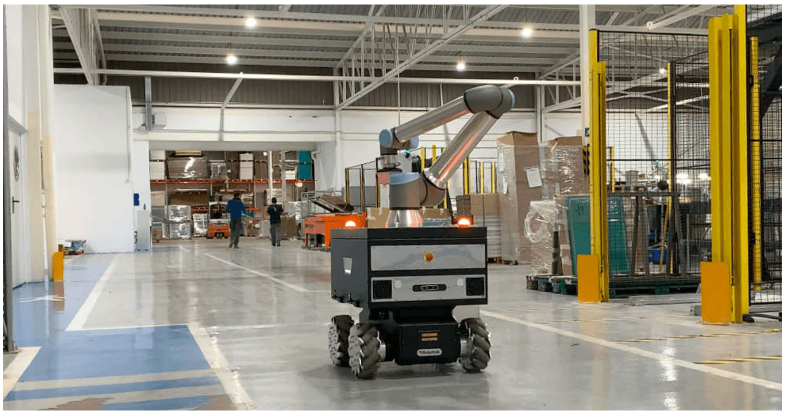


Figure 1.2: An example of autonomous robotics applied to industry.

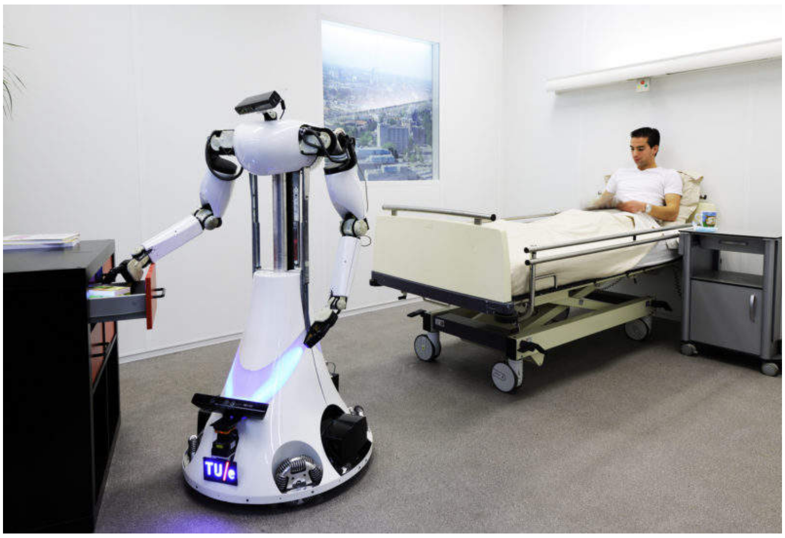


Figure 1.3: An example of autonomous robotics applied to healthcare.

Figure 1.4: An example of autonomous robotics applied to agriculture.

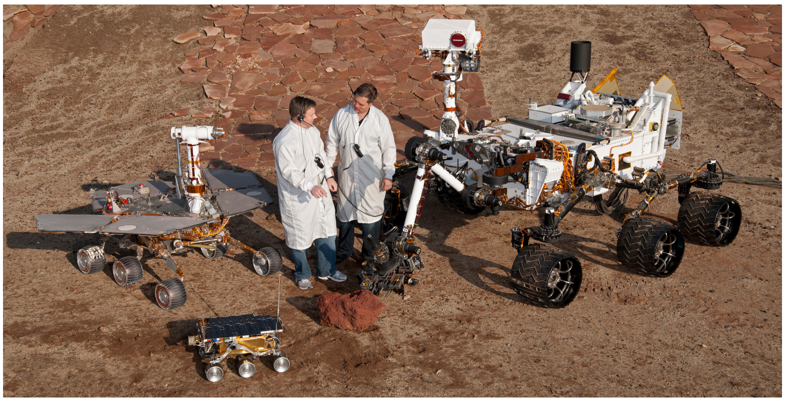


Figure 1.5: An example of autonomous robotics applied to space exploration.

1.4 Challenges

Despite substantial advancements in the field, social navigation retains many challenges.

Predicting the behavior of other agents and planning a path can be approached both as independent or linked processes [37]. In decoupled prediction and planning, the robot predicts the movement of the other agents but does not take into account how they react to its action: in the context of social navigation, this means treating humans as simple dynamic obstacles that do not respond to the agent actions. This causes some problems when dealing with humans: for example, the robot tends to enter a loop of unpredictability in which as the person moves unexpectedly the robot acts unexpectedly as well, which in turn produces another unexpected behavior from the human and so on. This oscillatory behavior is known as "reciprocal dance". Other examples include the robot blocking the person or disturbing their movement. In time uncertainty models for prediction became more and more complex, but treating humans as non-reactive dynamic obstacles was still problematic: at some point the uncertainty would explode and no viable path would be found ("freezing robot problem"). In order to avoid these issues a new approach involving Cooperative Collision Avoidance was proposed. This method included either explicit or implicit approaches. Explicit approaches use information about the structure of multi-agent collision avoidance to couple prediction and planning [37], for example with topological or geometric representations to model the coupling of multiple agent trajectories, or using game theory techniques. Implicit approaches use information about the principles underlying cooperative collision avoidance but do not set explicit constraints on its structure [37], for example by employing Reinforcement Learning models. In short, explicit approaches treat collision avoidance in a structured way, setting a-priori rules to follow, while implicit approaches rely on a learning model that learns the fundamentals of cooperative collision avoidance, without setting fixed rules to follow.

Human-robot coupling is more than a limit: it is a core abstraction for social navigation architectures. Unfortunately, coupled models become quickly computationally intractable, as each agent influences and is influenced by the other agents' movement. The trade-off between safety and efficiency in coupled models has many global optima with the same value: this means that additional constraints are required for optimization and it is not guaranteed that the coupled algorithm is doing what we want it to do.

Existing models often make the assumption that all agents in the environment take decisions with the same planning scheme. This is not true in general, humans behave in many different ways and the robot should be able to account for that.

The context in which the robot is situated has an effect on the planner. Different shapes of the space, timings or semantic maps require different behaviours from the robot and the planner, since crowded interactions are affected by those changes.

In addition to future location hints, intentions offer context-level signals that can aid pedestrians interpretation of robot motions [10]. Sometimes, pedestrians would even miss the robot if it did not signal its motion or its intention. Ideally, the intentions of the robot should be conveyed to pedestrians effectively to allow safe social navigation.

The concept of social spaces comes from proxemics [13]. Hall described four zones which denote the level of intimacy for interpersonal interactions:

- Intimate space: for embracing, touching or whispering, 0-0.5 meters;
- Personal space: for interactions among good friends or family, 0.5-1 meters;
- Social space: for interactions among acquaintances, 1-4 meters;
- Public space: used for public speaking, 4 or more meters.

Figure 1.6 represents social spaces. Robots must respect these spaces to satisfy the psychological safety constraints imposed by social navigation. Recently, research has been focusing on grouping of pedestrians: robots must also respect spaces generated within groups of people.

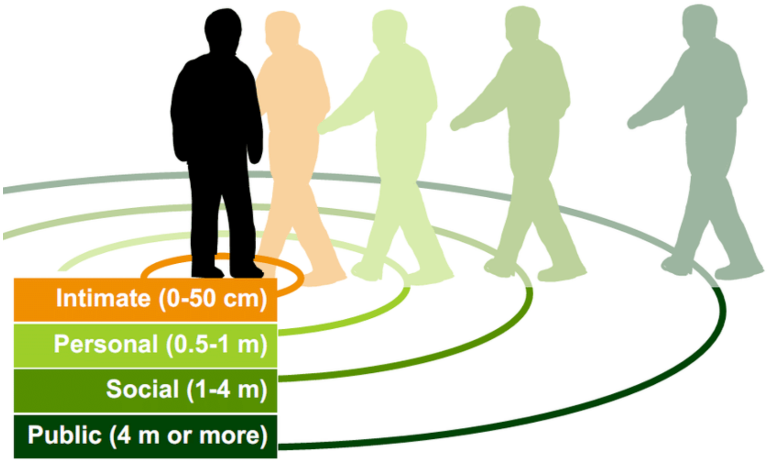


Figure 1.6: Hall social spaces.

1.5 Aim of the thesis

With the rapid advancements in robotics, mobile robots are becoming more and more integrated into daily life. As the environment in which these robots operate is highly dynamic and there are many constraints to be accounted for, regular planners are not enough anymore. Now planners need to generate effective trajectories, which consider the dynamics of the world in which the robot is moving and permit safe motions by taking into account human presence in their environment as well. Thus, the goal of this thesis is to propose an innovative method based on context for predicting the movement of people, so that planners can account for their trajectory and the robot can behave socially in a crowded environment.

The approach proposed in this thesis is part of the data-drive approaches: this work focuses on a data-driven approach that exploits a recurrent deep learning model to predict trajectories of people moving in the environment, and allows the robot to use these trajectories during motion planning.

Our approach is based on context: some of the key elements that influence people motion are the circumstances in which this motion is taking place. These circumstances include the environment where the person is moving, the relation with other people in the vicinity of the moving person, and the reason that drives the person's movement. We gather these important pieces of information via unsupervised learning and use this data alongside a classical trajectory-only recurrent learning model. This fusion of information from environment and past motion is then used to predict the future movements of people and the context of the environment. It is necessary that this prediction is performed as fast as possible so that it is possible to operate in real-time on the robot.

1.6 Structure of the thesis

This thesis is organized as follows: Chapter 1 contains a brief introduction on the field of human-aware navigation, on its challenges and applications, and on the the aim of this work.

In Chapter 2, a number of state-of-the-art approaches for social navigation and trajectory prediction is analyzed, with the objective of giving a general idea on the current state of research.

In Chapter 3 the methods and the tools used for accomplishing the goal of the thesis in terms of both hardware and software are illustrated, including the ROS framework, the TIAGo robot, standard simulators such as RViz and popular ones such as PedSim.

Chapter 4 contains an brief description of the datasets used for training and testing, including the pre-processing of JackRabbot.

Chapter 5 is the core of this work: a formulation of the proposed context-based approach in chapter is explained, along with an overview on the structure of the learning models chosen for training and testing.

Chapter 6 contains results collected from testing on the ATC and ETH/UCY datasets, as well as an explanation of the tests executed on the real TIAGo robot at the Autonomous Robotics Laboratory of the University of Padova.

Chapter 7 concludes this thesis with some final observations and propositions for a possible future evolution of this work.

Code is available at https://github.com/Andrea-Savio/trajectory_prediction.git.

2

State of the Art

The capability of robots to navigate and interact with people in social settings has become increasingly crucial in the current rapidly developing field of robotics. This chapter is reserved to the exploration of some of the state-of-the-art approaches, including classical approaches for path planning and advanced methods involving learning models for social navigation, including trajectory prediction for social navigation.

2.1 Classical path planning

As a starting point, classical path planning algorithms can be tested and compared to assess if they satisfy social navigation constraints and if they can be applied to move a robot in crowded scenarios. These algorithms are namely Dijkstra, A*, Rapidly exploring Random Tree (RRT) 1 and Artificial Potential Fields (APF) 2 [61]. Dijkstra's algorithm and A* are very famous graph search algorithms. Dijkstra will generate a path of minimum length at the cost of computation time, since it computes the shortest path to every possible node of the graph and therefore it has to search through many, potentially all, nodes while A* uses heuristics to estimate the current distance from the goal and to guide the expansion in the best direction, obtaining an almost-minimum path with lower computation costs, and even improved versions were developed over the years [66]. RRT is a sampling-based probabilistic search method, very computationally efficient but lacking in terms of optimality and predictability. Finally, the APT method is a very popular planner with great avoidance capabilities: each obstacle is assigned a high potential value while the goal has the lowest potential on the map, and the robot will try to move to the lowest potential following the gradient. Each of these methods has fundamentally the same issue: it does not take into account human necessities. A^*, Dijkstra and RRT are very straightforward global planning techniques: without a local planner, obstacle avoidance is treated as a reactive behavior based only on the sensor readings. As discussed previously, this is not enough for social navigation. On the other hand, APT is a local planner built to evade obstacles, but it does not consider human motion, it just follows the gradient down to the minimum potential. This could get the robot in the way of humans or fail/oscillate in certain scenarios. An example of trajectories planned by these classical algorithms can be seen in Figure 2.1.

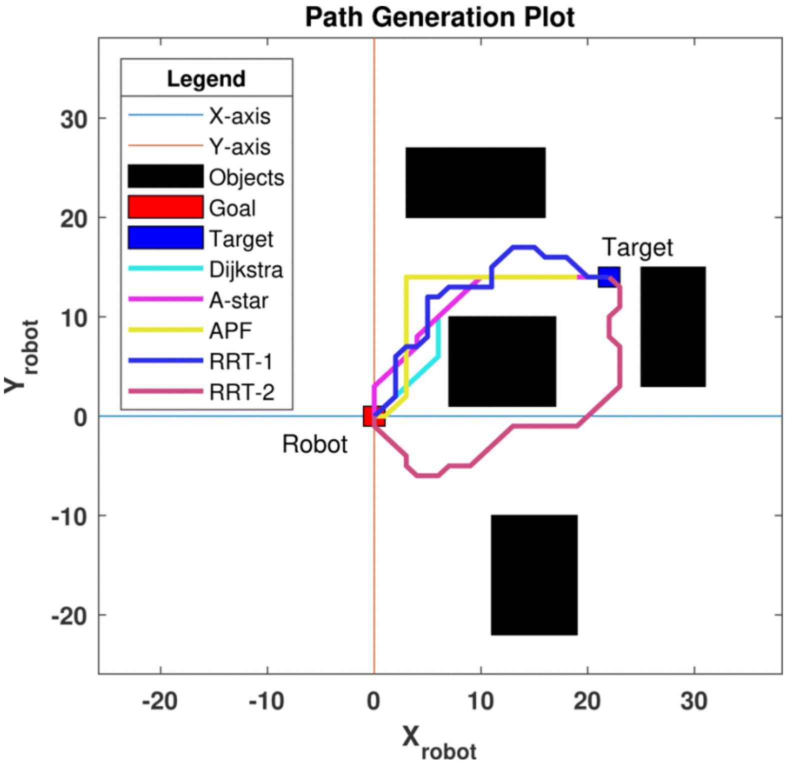


Figure 2.1: Example of trajectories planned by some classical path planning algorithms[61].

Algorithm 1 Rapidly-exploring Random Tree algorithm

```
Require: $x_{start}, x_{goal}, Map,$ Graph T
  $T.AddNode(x_{start})$
  while $x_{new} \neq x_{target}$ do
    $x_{rand} \leftarrow Random.State(Map);$
    $x_{near} \leftarrow Nearest.TreeNode(x_{rand}, T);$
    $x_{new} \leftarrow Nearest.Neighbour(x_{near}, x_{rand});$
    if $CheckNeighbours(x_{new}, Map)$ then
      $T.AddNode(x_{new});$
      $T.AddEdge(x_{new});$
    end if
  end while
  $path \leftarrow GetShortestPath(T, x_{start}, x_{target});$
  return path;
```

Algorithm 2 Artificial Potential Fields algorithm

```
Require: $x_{start}, x_{goal}, Map, ObstacleCoor, RobotVel, \delta_{obs}, \delta_{tar}$
  $Obs.Pot \leftarrow GetObstaclePot(Map, ObstacleCoor);$
  $Tar.Pot \leftarrow GetTargetPot(Map, x_{target});$
  $x_{pos} \leftarrow x_{start};$
  while $GetDistance(x_{pos}, x_{target}) \geq \delta_{tar}$ do
    if $DistanceToObstacle(x_{pos} \leq \delta_{obs}$ then
      $x_{grads} \leftarrow GetGradient(Tar.Pot - Obs.Pot);$
    else
      $x_{grads} \leftarrow GetGradient(Tar.Pot);$
    end if
    $x_{pos} \leftarrow x_{pos} + RobotVel \times x_{grads};$
    $path \leftarrow x_{pos};$
  end while
  return path;
```

2.2 Deep Reinforcement Learning

With the advancements in machine learning, new models have been tested to accomplish the social navigation task. Deep Reinforcement Learning became increasingly popular in social navigation [68]. This is due to the capability of working without a map and not depending much on sensor accuracy. In reinforcement learning, the agent tries to maximize the accumulated reward while obtaining a reward value based on its interaction with the environment. This agent could replace the localization, map building and local path planning modules of the navigation framework. DRL methods can be divided in value-based and policy-based approaches: value-based DRL obtains the agent's policy by updating the value function, while policy-based DRL optimizes the policy along the gradient to maximize the reward value. Value-based methods include Deep-Q Network (DQN) and Double DQN (DDQN), while policy-based methods include Deep Deterministic Policy Gradient (DDPG) and Asynchronous Advantage Actor-Critic (A3C). DQN and DDQN, as the name suggests, employ neural networks to learn to select the action of highest value (and also to evaluate it in DDQN). DDPC optimizes the policy via its gradient, while in A3C an actor selects an action using the policy gradient method and a critic evaluates the chosen action. In general, DRL techniques have proven their effectiveness, but are not perfect. Training deep learning models usually has very high costs in terms of computational effort, and as the models become more and more complex (for example by using RNNs, LSTMs or GRUs for memory ability) the costs increase excessively. Their effectiveness is also influenced by the gap between the real world and the virtual world in which the model was trained. Other approaches focus on estimating body, face, hands and feet to determine the activity that the person is engaging (standing, sitting, etc.) [41]. Figure 2.2 shows a possible framework used by deep reinforcement learning approaches.

2.3 Proactive Social Motion Model

Social Force Models (SFM) are a useful way to drive a mobile robot in high density conditions(see Figure 2.3) as they have reasonable computational costs[55]. The motion of pedestrians can be described as subject to social forces. These forces are a measure for the internal motivations of the individuals to per-

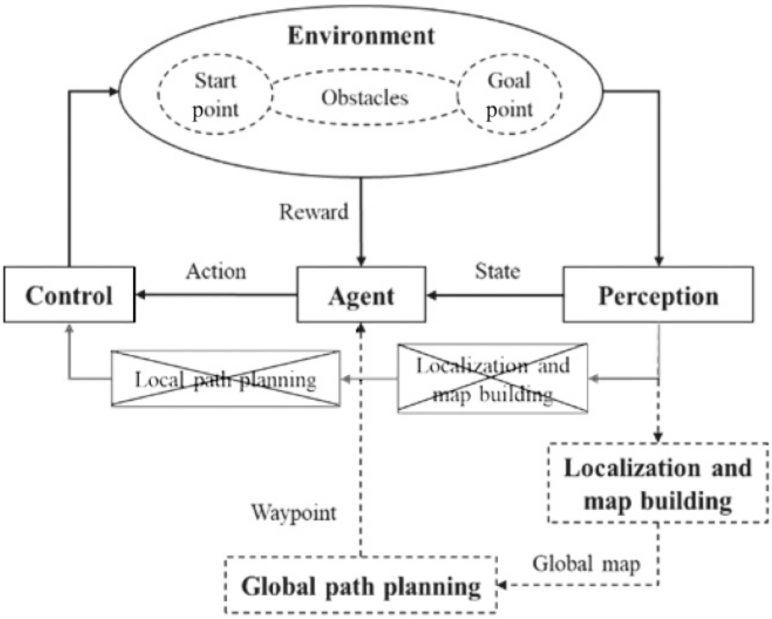


Figure 2.2: DRL-based navigation framework[68].

form certain actions. Social cues and social signals can be incorporated into the motion model, but SFM only deals with human features extracted from a single individual rather than the social characteristics of human-object and human-group interactions, causing a lack in robustness when working in situations with large groups of people. The solution proposed in [55] avoids this problem by extending the classical SFM with a human detection/tracking module and a social interaction module that identifies human interactions with objects and groups.

Figure 2.3: The motion of pedestrians is modeled using social forces.

2.4 Anticipative Models

Another approach is to implement a way of predicting the future movement of dynamic obstacles. These methods take the name of anticipative strategies [11].

> *Anticipation is the ability to react taking into account not only for actual situations, but also a prediction for a future time window.*

These methods are Anticipative Turn (AT) [11], Anticipative Robot and Pedestrian's Propagation (ARP) [11], Anticipative Pedestrian's Propagation (APP) [11], all represented in Figure 2.4. They propagate the robot and the human agents during a certain number of time steps in the local map (specifically AT propagates only the robot, ARP both the robot and the human agents and APP robot and agents with constant velocity). These strategies resulted quite successful both in simulated and in real environments, and in particular APP is able to avoid agents moving at up to 18 km/h in some situations, or even at higher velocities with a long-range sensor. The limit of 18 km/h is due to the fact that at that speed a robot equipped with a normal lidar sensor has less than a second to react, making it very difficult to avoid the incoming obstacle. Indoor experiments were performed where the robot encounters frontally and perpendicularly humans.

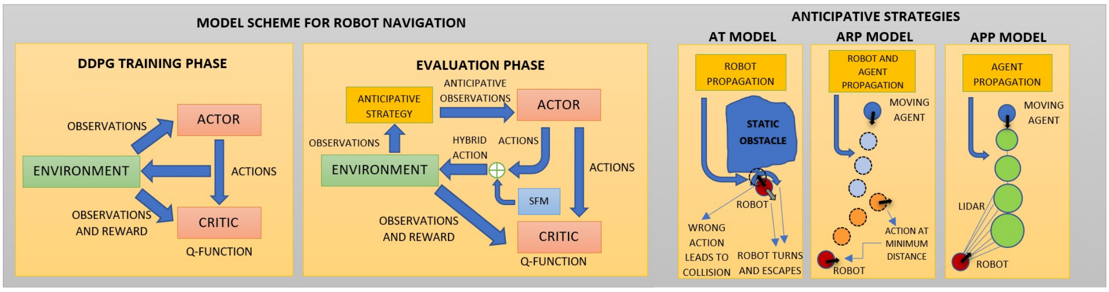


Figure 2.4: Model scheme of anticipative strategies implemented in the DRL robot navigation framework [11].

2.5 HATEB-2

HATEB-2 is a new framework for planning in social navigation [53]. It is based on the Timed-Elastic-Band approach[45][16], an efficient way of generat-

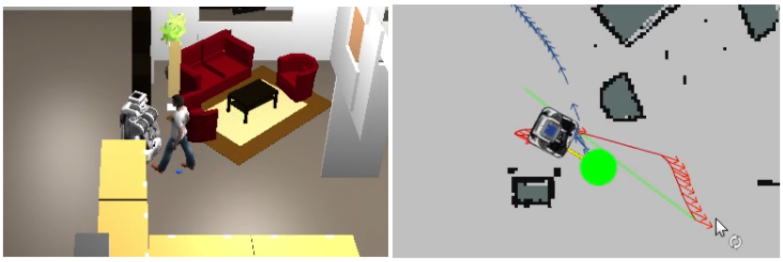


Figure 2.5: Narrow passage scenario in which the robot backs off to make way for the person [53].

ing a time-optimal trajectory by merging states, control inputs and time intervals. HATEB builds on TEB (see Algorithm 3) by also planning for both humans and robot, thus enabling human prediction, and HATEB-2 improves HATEB by adding decision making capabilities on top of planning. It works in 3 modalities named Single-Band, Dual-Band and VelObs. In Single-Band mode, the elastic band is applied only to the robot. This mode is the least computationally expensive of the 3 and is used when the robot is far from people. In Dual-Band mode, elastic bands are applied to all human agents and robot. Trajectories are generated simultaneously and the robot adapts itself to the predicted goals and motion of the humans. When running in this mode, the robot could encounter the entanglement problem: as HATEB assumes that all agents are continuously moving, if the humans stand still the robot stops and waits for their action, ignoring other solutions. To avoid this problem, the VelObs mode was introduced. VelObs works similarly to Dual-Band, except for the fact that the bands are added on the humans and their trajectories are predicted only if they are moving. The resulting prediction of the trajectories assumes that the moving people will keep moving at constant speed for the duration of the prediction window. Switching between these modes enables robots to find a solution for many complex social navigation problems. An an example, in Figure 2.5 HATEB-2 propagates both its and the person's future positions and the robot understands it needs to move backwards to let the human pass. In general, the results of the experiments have proven that HATEB-2 performs better than HATEB [53].

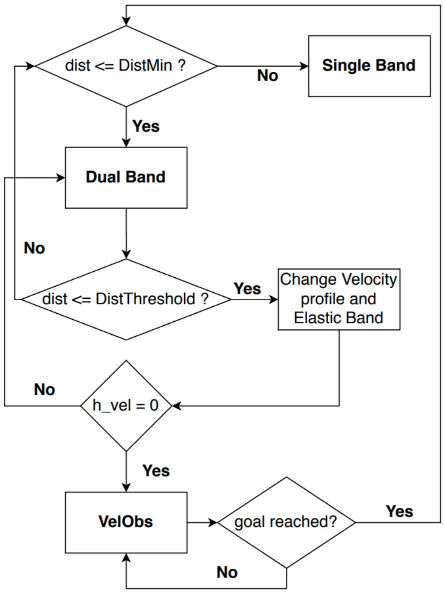


Figure 2.6: Transitions between modes in HATEB-2[53]. Dist is the current distance between the closest human and the robot, DistMin is the minimum value of dist to add a double band and DistThreshold is the minimum cutoff dist to initiate transition between Dual Band and VelObs. H vel is the velocity of human [53].

Algorithm 3 Timed-Elastic-Band procedure

```
Require: trajectory b, $x_{start}, x_{final}$
  Initialize or update trajectory;
  for all Iterations 1 to $I_{teb}$ do
    Adjust length n, resp. d of the trajectory;
    $b^* \leftarrow SolveNLP(b)$;
    Check feasibility;
    return (sub)optimal TEB $b^*$ and action $u_1^*$;
  end for
```

2.6 NeuroSyM

To further advance the field of human motion prediction, the authors of [39] propose a new neuro-symbolic approach that uses a-priori information on the interactions between observed agents in the environment. These interactions between agents in a neighbourhood are weighted differently by using a spatial representation technique known as Qualitative Trajectory Calculus (QTC)[15]. Spatial relations between pairs of interacting agents, like relative distance, velocity, and orientation, are represented by QTC symbols, which are then combined to describe models of pairs of moving agents.

The approach is evaluated on two state-of-the-art architectures: the first is the well-known Social Generative Adversarial Network (SGAN [12]), while the second is the Dual-stage Attention Recursive Neural Network (DA-RNN [43]). In the SGAN, NeuroSym influences the pooling mechanism of the model (Figure 2.7), where it represents human-human interactions by embedding first their relative poses in all the observed states of each agent through a dense layer, then weighing the embedded relative pose, and finally max-pooling the weighted embedding across neighbours in the global scene [39]. NeuroSym sensibly improves the performance of the original SGAN model. In the DA-RNN model, NeuroSym injects a Conceptual Neighbourhood Diagram [56] at the interface between the embedding and the softmax layers (Figure 2.8), which causes the update of the encoder attention weights with an a-priori knowledge of the reliability or stability of each input series. NeuroSym improves the performance of the NA-RNN model as well.

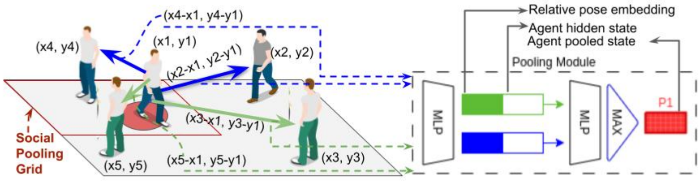


Figure 2.7: The neuro-symbolic generative adversarial network pooling mechanism proposed in [39].

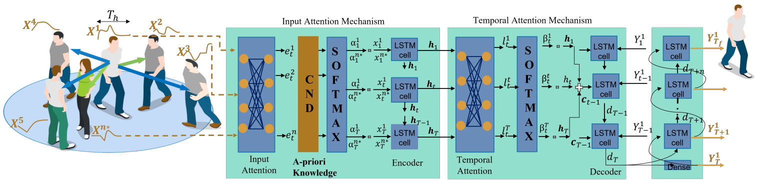


Figure 2.8: The neuro-symbolic approach for attention-based time-series prediction models (DA-RNN) proposed in [39].

2.7 Other Approaches

Many ideas were proposed over the years. In [50] the authors propose a model that, during planning, takes into account the possibility of humans emerging at any time from blind spots in the environment. The possible locations of these "invisible" humans are estimated by analyzing large separations between consecutive laser values scanned. In [3], a genetic algorithm was developed to address the planning phase of the social navigation problem. A genetic algorithm is a method inspired from biology which searches the optimal solution among chromosomes. It is composed of three stages: reproduction, selection and mutation or crossover. During reproduction new chromosomes will be generated; during selection the worst chromosomes will be pruned, and during mutation/crossover a Gaussian noise will be applied to the chromosomes to slightly modify them. This approach proved that the social navigation layers of ROS can be discarded in favour of a simple optimization algorithm. In [5] the authors implemented a social navigation system alongside classic shared operation. The social behavior allows the robot to follow a selected person while avoiding obstacles and keeping other people at appropriate distance. This approach proposes three modalities which can be applied to the robot: manual modality, shared modality and supervisory modality. In the end, the shared modality appears to be the best in term of reliability and required effort. In [44] the authors propose a model used for accompanying people in dynamic environments via an Adaptive Social Planner (ASP). This ASP anticipates the movement of people, including also other interaction forces between robot and environment, and includes a path evaluation to choose the best path among the generated possibilities. Other works focused on ways of challenging social navigation systems and planners with human movement simulators. In [10] the authors propose an intelligent human agent controller, an interface to control these agents, a GUI and metrics for evaluation. The goal is to avoid long, expensive and tiresome real-life testing and substitute it with fast and even parallel simulations. Finally, many comparisons between the performance of different planners can be found at [22].

2.8 Predicting motion trajectories of people

With the advent of recurrent learning models, capable of learning from time-series data and successfully predict future values, trajectory prediction became a reality. Similarly to the way humans can infer the path of other people moving around them, robots are now able to predict the position of humans in the near future. By using complex algorithms that fuse historical trajectory data and real-time sensor inputs, autonomous robots can forecast the path taken by any person in its surroundings. This enables robots to preemptively deal with human motion, avoid potential future collision and behave socially and predictably, therefore satisfying the constraints imposed by social navigation. Of course, human behaviour is often hardly predictable, as it depends on intention and cognitive processes that can be difficult to recognize from the outside of their mind. In the following table, a summary of a number of literature works regarding trajectory prediction are reported.

The following papers are divided into the following categories, that we highlight with different colours :

- Autonomous driving;
- Human-Robot Interaction;
- Very specific context of use, for example firefighting assistance;
- Based on top-view images for training/testing;
- Based on front-facing camera;
- Not related to social navigation;

The analyzed papers are classified as specified above based on their scope and their requirements. The reason for this classification is the necessity to find a "gap" (i.e. what is missing in the current state-of-the-art approaches) to work on. In the following tables, the gap found for each work is reported.

Table 2.1: Summary of papers read.

Paper	**Gap**
Group-based Motion Prediction for Navigation in Crowded Environments[59]	Motion prediction capabilities are short-term and do not scale with the number of agents, tested on classical human trajectory "top view" datasets
Long-term pedestrian trajectory prediction[19]	Single human trajectory to predict from previous observations.
Sparse to Dense Scale Prediction for Crowd Counting in High Density Crowd[23]	Uses heads to only count humans, images mainly from a top view.
Social and Scene-Aware Trajectory Prediction in Crowded Spaces[33]	Trained and tested with top view images, it won't work with a front facing camera.
Social LSTM: Human Trajectory Prediction in Crowded Spaces[2]	Trained and tested on top view images, it won't work with front facing camera. Does not take into account groups.
Group LSTM: Group Trajectory Prediction in Crowded Scenarios[6]	Trained and tested on top view images, it won't work with front facing camera.

Motion Planning Combines Psychological Safety and Motion Prediction for a Sense Motive Robot[32]	Uses human faces and facial behavior to predict trajectory, faces are hard to track in a crowded environments and from the distance due to possible occlusions and camera limitations.
Pedestrian Motion Trajectory Prediction in Intelligent Driving from Far Shot First-Person Perspective Video[7]	Autonomous driving scenario, higher velocities involved and ideally structured and less crowded spaces (authors assume the road is mostly inhabited by vehicles).
Context Attention: Human Motion Prediction Using Context Information and Deep Learning Attention Models[28]	Not based on crowds, different context. One-on-one, interaction activity.
A Continuous Learning Approach for Probabilistic Human Motion Prediction[64]	Focus on HRI, ambient changes and human kinematic structure (human joints position and movement prediction). Single human in front of the camera, no crowds.
A Deep Concept Graph Network for Interaction-Aware Trajectory Prediction[4]	Focus on autonomous driving, top view of trajectories used in testing, no reference to human crowds. Structured space (roads).
Autonomous Vehicle Parking in Dynamic Environments: An Integrated System with Prediction and Motion Planning[29]	Focus on autonomous driving, specifically on parking while predicting motion of other vehicles. Structured space (parking lot/road), few humans.
Conditioned Human Trajectory Prediction using Iterative Attention Blocks[42]	Top view of trajectories, focus on conditioning of the environment on humans (the relation between their movements and the structure of the world, e.g. humans walk on the sidewalk and not on grass/dirt).
Control-Aware Prediction Objectives for Autonomous Driving[38]	Focus on autonomous driving, predicting trajectories of cars on roads, top view of the world. Structured spaces.

Crossmodal Transformer Based Generative Framework for Pedestrian Trajectory Prediction[52]	Focus on autonomous driving, people only on crossroads, no crowds. Structured spaces.
Distributed Timed Elastic Band (DTEB) Planner: Trajectory Sharing and Collision Prediction for Multi-Robot Systems[9]	Focus on multi robot systems, communication and collaboration between agents that mutually share data.
HYPER: Learned Hybrid Trajectory Prediction via Factored Inference and Adaptive Sampling[18]	Focus on traffic and autonomous driving. Argoverse dataset, which contains motion prediction data based on top view in structured spaces.
KEMP: Keyframe-Based Hierarchical End-to-End Deep Model for Long-Term Trajectory Prediction[34]	Focus on autonomous driving, Argoverse and Waymu datasets used for training and testing, top view of the road and structured spaces.
Leveraging Smooth Attention Prior for Multi-Agent Trajectory Prediction[8]	Autonomous driving focused, environments with both vehicles and pedestrians but not necessarily crowds, structured spaces. Top view trajectories.
Meta-path Analysis on Spatio-Temporal Graphs for Pedestrian Trajectory Prediction[14]	Top view view of trajectories, training and testing done on classical ETH and UCY datasets.
Motion Primitives-based Navigation Planning using Deep Collision Prediction[40]	Uses on-board camera, but the task involves prediction of collision costs on objects, not trajectories. No crowds.
MultiPath++: Efficient Information Fusion and Trajectory Aggregation for Behavior Prediction[57]	Not based on camera images, Argoverse and Waymu datasets (top view of trajectories). Focus is more on the fusion of data from different sensor than motion prediction.

StopNet: Scalable Trajectory and Occupancy Prediction for Urban Autonomous Driving[24]	From on board images/scans (raw data) to top view, focus on autonomous driving and road agents in structured spaces.
Towards Efficient 3D Human Motion Prediction using Deformable Transformer-based Adversarial Network[17]	Focus on HRI and specifically human poses and kinematic model (human body joints position and possible motion). HUMAN3.6M dataset, AMASS benchmark.
Trajectory Prediction for Autonomous Driving with Topometric Map[63]	Focus on autonomous driving, topometric map is needed, KITTI dataset with GPS data. The map can be a limitation if the world is very dynamic.
Trajectory Prediction with Linguistic Representations[26]	Not based on camera images, it needs text to work: each situation (e.g. a car turning left) is explained with a sentence. Top view trajectories. Authors had to label part of the Argoverse dataset with text to train and test their model.
A Data-Efficient Approach for Long-Term Human Motion Prediction Using Maps of Dynamics[69]	Based on maps of dynamics, which are a way to encode spatial information about the dynamics at different locations. It uses the ATC dataset (images from high security cameras in a shopping centre). Maps can be a limitation.
A generic diffusion-based approach for 3D human pose prediction in the wild[46]	Focus on prediction of human poses, not trajectories. HUMAN3.6M dataset, single human in front of the camera.
Can We Use Diffusion Probabilistic Models for 3D Motion Prediction?[1]	Based on Human 3.6M and HumanEva-I datasets, single human poses and motion, no crowds. Authors mentioned that "Diffusion model cannot perfectly replace existing state-of-the-arts for both deterministic and stochastic motion prediction tasks".

Dynamic Control Barrier Function-based Model Predictive Control to Safety-Critical Obstacle-Avoidance of Mobile Robot[20]	Based only on LiDAR, no crowds. It uses the Kalman filter, which is effective but very basic.
Exploring Navigation Maps for Learning-Based Motion Prediction[47]	Focus on autonomous driving, use of navigation maps, Argoverse dataset. Navigation maps are a limitation.
Improving robot navigation in crowded environments using intrinsic rewards[36]	No trajectory prediction, use of intrinsic rewards only for improving navigation.
Improving the Generalizability of Trajectory Prediction Models with Frenét-Based Domain Normalization[65]	Focus on autonomous driving, Argoverse and Waymu datasets. The authors try to improve existing models (e.g. LSTM) by changing from the cartesian frame to the Frenét frame. Very small improvements on performance with respect to the same models without the Frenét-based domain normalization.
Moment-based Kalman Filter: Nonlinear Kalman Filtering with Exact Moment Propagation[48]	No reference to pedestrians/crowds, motion propagation of objects.
MVFusion: Multi-View 3D Object Detection with Semantic-aligned Radar and Camera Fusion[62]	Focus on detection and sensor fusion, no prediction of human trajectories, no crowds.
Pedestrian Crossing Action Recognition and Trajectory Prediction with 3D Human Keypoints[30]	Focus on autonomous driving and pedestrian trajectory prediction only when approaching crossings, i.e. focus on the intention of the human to cross the road. It uses head pose and movement to understand if the human wants to cross.

Situational Adaptive Motion Prediction for Firefighting Squads in Indoor Search and Rescue[35]	Very specific context, the robot has to predict the motion and follow the firefighter in a burning building.
Topological Trajectory Prediction with Homotopy Classes[58]	It divides the environment into classes and assigns each human to a class so that it can predict the trajectory that the human has to follow to reach the place of its class. Top view of the environment (ATC dataset).
TrafficBots: Towards World Models for Autonomous Driving Simulation and Motion Prediction[67]	Focus on autonomous driving, top view predictions (Waymu dataset), it simulates multi-agent behavior in structured spaces with few humans.
Trajectory and Sway Prediction Towards Fall Prevention[60]	Very specific context, prediction of imbalance (i.e. if a person risks to fall).
Context and Intention aware 3D Human Body Motion Prediction using an Attention Deep Learning model in Handover Tasks[27]	Handover task context, focus on HRI. Single human in front of the robot, no crowds.

2.9 Take-home message

Social navigation is not an easy task. There is no approach that is perfect or clearly superior to every other ones. Every method has its strengths and weaknesses, and thus developers have to choose the approach that works best for their requirements. Generally, classical approaches, despite representing the basis on which the field of social navigation was built, are outdated and outperformed. These approaches, such as rule-based systems and traditional path planning algorithms, formed the bedrock of social navigation research. While they provided initial insights and solutions, they had limitations in handling complex social dynamics and in adapting to dynamic environments. However, it is important to recognize their historical significance as they led researchers to the advancements we witness today. The field of social navigation has undergone a revolution thanks to the development of machine learning, deep learning, and reinforcement learning. These advanced methods have demonstrated capabilities in understanding and responding to social cues, human intentions, and complex social interactions. Advanced methods utilizing learning models provide better adaptability, robustness, and generalizability, and they can also learn from experience, continuously improving their performance. Nowadays robots can learn optimal navigation policies through reinforcement learning techniques such as Q-learning or Deep-Q networks. Many new methods employ some sort of learning paradigm in some way. Usually, authors then build new approaches on top of these learning models with new algorithms to improve performance. Looking ahead, social navigation is a dynamic and evolving field and future research should work on integrating social context, cultural variations, and ethical considerations into navigation algorithms. Other challenges include explainability, transparency, and human-centric design to improve acceptance of social robots.

3 Methods

In this chapter, the tools and the frameworks used for the development of this thesis will be briefly presented.

3.1 Robot Operating System (ROS)

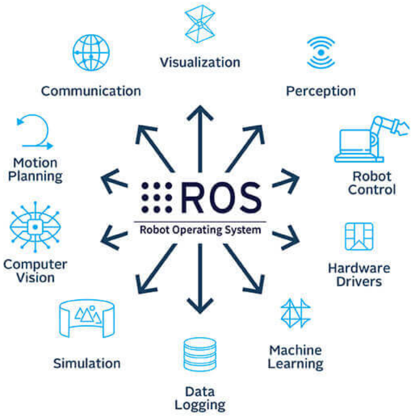


Figure 3.1: Robot Operating System [1]

Robots are complex systems that are difficult to build: motors, sensors, software and batteries must work together seamlessly to complete a task. To improve the portability of software among these systems, the Robot Operating System

[1] https://automationware.it/ros-eng/?lang=en

was proposed. The Robot Operating System (ROS), shown in Figure 3.1, is an open-source framework for software robot development. It provides services such as hardware abstraction, low-level device control, package management, and message-passing between processes. It also provides tools and libraries for obtaining, building, writing, and running code across multiple systems. ROS was developed with the goal of creating a flexible and modular framework that could support a wide range of robotic systems, from small hobbyist projects to large-scale industrial robots. The open-source nature of ROS has led to a large and active community of developers and users, who contribute to the development and improvement of the framework. It provides a standard set of communication protocols and data structures that allow different components of a robot system to communicate with each other seamlessly. ROS employs a node-based structure: ROS processes work as nodes in a graph structure, connected by edges called topics. Nodes communicate between each other via messages sent through topics and request and provide services to other nodes. The ROS Master is the main process: it registers and names each node and it tracks publishers and subscribers to topics and services. The ROS master allows the individual nodes to locate each other and sets up peer-to-peer communication between them.

3.1.1 Nodes

The use of nodes in the ROS framework provides several advantages:

- Crashes are isolated to individual nodes instead of the whole system, thus guaranteeing more stability;
- Code complexity is reduced as each node has its own code, instead of having a monolithic system;
- Implementation details are private to the node, which exposes itself to other nodes via a minimal API.

A complex system may have many nodes, where each one controls a single functionality of the system. It is better to have separated nodes for each functions instead of a single node implementing the complete system.

3.1.2 Topics

Topics are buses over which nodes exchange messages[2]. A node that is interested to some data subscribes to a topic, while a node that produces some data publishes to a topic. In general, a topic can have multiple publishers and subscribers and the nodes involved do not know with which other nodes they are communicating. Each topic is strongly typed and nodes can receive messages only if their type matches the message type used to publish to the topic.

3.1.3 Services

The many-to-many structure of the publish-subscribe mechanism is not well suited to situations where a request or an answer from a specific node is required (for example, in distributed systems). To avoid this problem, the request-reply mechanism is dealt with by using services, which are defined by a pair of messages, one for requesting and one for replying. Services give the possibility of interacting more strictly within nodes, enabling higher performance at the cost of robustness to provider changes.

Actions

In some circumstances, services may take a long execution time, and the user might want to cancel the request or receive periodic feedback about its progress. The actionlib package offers tools for building servers that carry out long-running, interruptible tasks.

3.2 Take It And Go (TIAGo) Robot

The TIAGo robot, shown in Figure 3.2 is a versatile humanoid robot developed by the Spanish robotics company PAL Robotics for indoor environments. It is designed to be a research platform for developing and testing advanced robotics applications, particularly in the fields of human-robot interaction and service robotics. It combines mobility, perception, navigation and human-robot interaction. It stands between 1.1 and 1.4 meters tall and weighs around 70

[2] http://wiki.ros.org/Topics

[3] http://wiki.ros.org/Robots/TIAGo

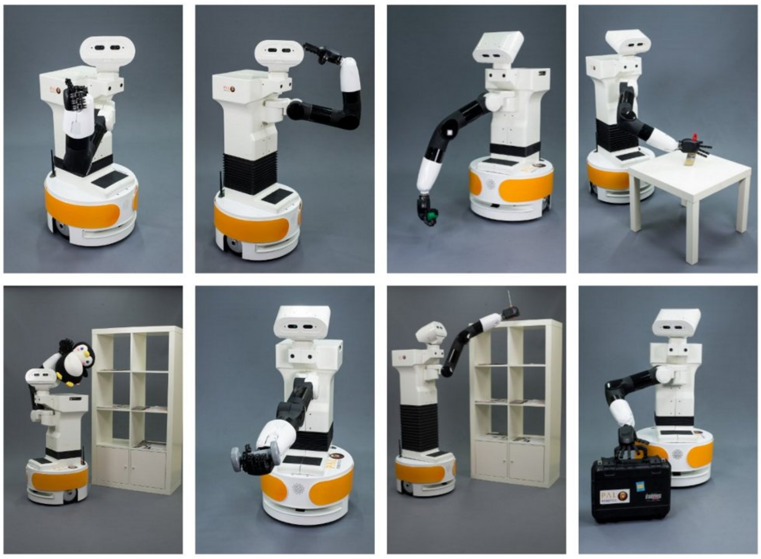


Figure 3.2: TIAGo [3].

kilograms. It has one arm (two in the TIAGo++ version) with seven degrees of freedom, and a two-fingered gripper for manipulating objects. It also has a 2D LiDAR sensor for navigation and obstacle avoidance, as well as a range of other sensors for perception and environmental monitoring. The TIAGo robot has been used in a range of research and development projects, including applications in healthcare, education, and industrial automation. It is particularly well-suited for tasks that require dexterous manipulation, such as object sorting and assembly.

The TIAGo base robot (Figure 3.3) is essentially the same robot as the TIAGo humanoid robot, but without the upper torso and arms. The TIAGo Base robot consists of a mobile base, which is equipped with a range of sensors and actuators, and is designed to be highly customizable and programmable. It is built on top of the same hardware and software platform as the TIAGo humanoid robot.

[4] http://wiki.ros.org/Robots/TIAGo-base

Figure 3.3: TIAGo base robot [4].

3.3 Simulators

Gazebo	Standard 3D rigid body simulator.
RViz	Standard 3D visualization tool for ROS.
PedSim	Tool for pedestrian simulation.
SocialGym	Simulator for social scenarios.
iGibson	Stanford robotics simulator based on Bullet.
CoppeliaSim	Versatile simulator built for many applications.
MORSE	Generic simulator for academic robots.

These simulators are popular choices among roboticists for visualization and testing in simulations of robots. Every simulator has its strengths and its weaknesses: some of them are light and reliable but limited, while others are heavy and can simulate also multiple agents, or realistic people motion. In the end, these simulators are indispensable tools for robotics research, development and testing.

3.3.1 Gazebo

Gazebo (logo in Figure 3.4) is a popular 3D robot simulation tool that is commonly used in the Robot Operating System (ROS) ecosystem. It allows users to create and simulate complex robotic systems in a realistic 3D environment[5]. It provides access to:

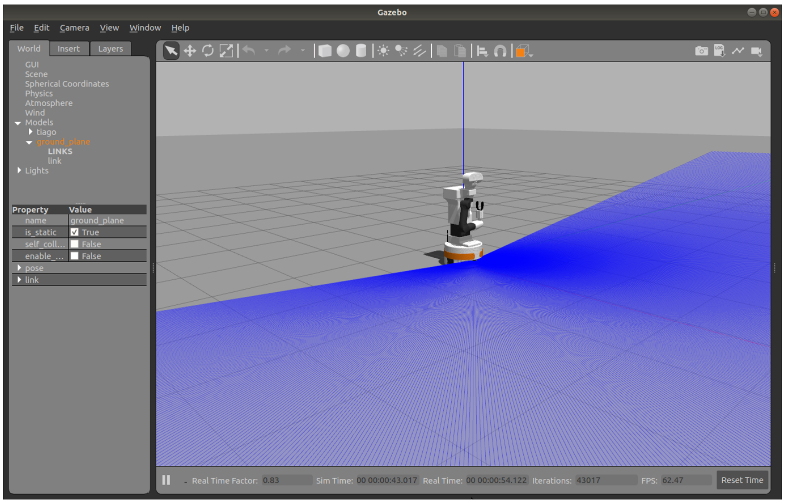
[5] https://gazebosim.org/libs/sim

Figure 3.4: Gazebo [6]

- **Dynamics simulation**: many high-performance physics engines are available through Gazebo Physics;
- **Advanced 3D graphics**: rendering engines such as OGREv2 are available for realistic rendering of environments,
- **Sensor and noise models**: generate sensor data from a wide range of sensors, including cameras, laser scanners, GPS and more;
- **Plugins**: custom plugins for robot, sensor and environment control are available;
- **Simulation models**: multiple robots are compatible with the simulation and new environments can be constructed using physically accurate models.

Gazebo also provides a range of powerful tools for visualization and analysis, including tools for 3D visualization, logging and visualization of simulation data, and support for real-time simulation and control. The key benefit of using Gazebo in ROS is that it allows users to test and develop their robotic systems in a safe and controlled environment, without the risk of damaging physical hardware. This can save time and resources, as well as making it easier to test and refine robotic systems before deploying them in the real world. In ROS, Gazebo is often used in conjunction with other tools, such as RViz, to create complete robot simulation systems. Gazebo allows users to simulate the physics of robots and their environments, including gravity, friction, collisions, and more. Figure 3.5 shows the TIAGo robot simulated in Gazebo.

[6] https://classic.gazebosim.org



Property	Value
name	ground_plane
is_static	☑ True
self_coll...	□ False
enable_...	□ False

Real Time Factor: 0.83 Sim Time: 00 00:00:43.017 Real Time: 00 00:00:54.122 Iterations: 43017 FPS: 62.47

Figure 3.5: TIAGo simulated in Gazebo.

3.3.2 RViz

RViz (logo at Figure 3.6) is a powerful 3D visualization tool that is commonly used in the field of robotics. It is an open-source software package that allows users to create interactive visualizations of robot models, sensor data, and other relevant information[7]. RViz provides a wide range of interactive tools for manipulating these visualizations, such as the ability to move and rotate the camera view, select and highlight specific components, and adjust the appearance of the models and data being displayed. In addition to its core visualization features, RViz can also be extended with plugins to add additional functionality. For example, there are plugins available for controlling robot motion, visualizing simulation data, and interfacing with external hardware. Users of RViz can design visual representations of robots that include the joints, connections, and other parts of the robots. Different 3D models, such as those made in CAD software or constructed using more basic geometric shapes, can be used for this. Data from sensors mounted on robots, such as cameras, LiDARs, or sonars, are also accessible to users. Both unprocessed sensor data and processed data, such as point clouds or occupancy grids, may be included in this. RViz is therefore an incredibly useful tool for anyone working in the field of robotics, as it allows users to easily create and interact with detailed 3D models and sensor data, making it easier to design, test, and debug robotic systems.

[7] https://github.com/ros-visualization/rviz

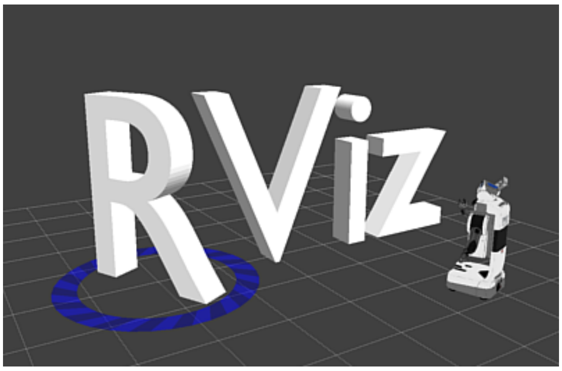


Figure 3.6: RViz.

3.3.3 PedSim

PedSim is an open-source software tool for simulating and analyzing pedestrian dynamics in indoor and outdoor environments based on SFM. It is built on top of the ROS framework, which makes it easy to integrate with other robotics and simulation tools. It is commonly used in the fields of robotics, computer vision, and transportation engineering to study and optimize pedestrian behavior and movement in crowded public spaces[8]. It features:

- Individual walking using social force model for very large crowds in real time;
- Group walking using the extended social force model;
- Social activities simulation;
- Sensors simulation (point clouds in robot frame for people and walls);
- XML based scene design;
- Extensive visualization using Rviz;
- Option to connect with gazebo for physics reasoning.

PedSim uses a combination of simulation and data analysis tools to model and analyze pedestrian behavior. It simulates the movement of large crowds of people in complex environments, taking into account factors such as walking speed, direction, and social behavior. It can also simulate the effects of obstacles and other environmental factors on pedestrian movement Figure 3.7 shows a simulated environment containing simulated humans and a robot.

[8] https://github.com/srl-freiburg/pedsim_ros

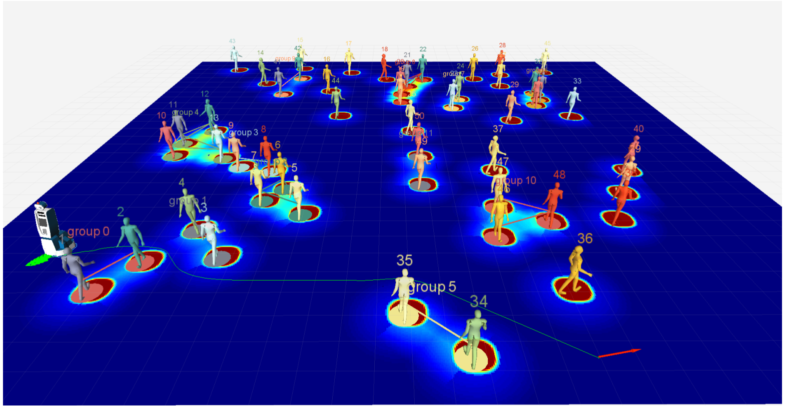


Figure 3.7: PedSim simulating a robot in a crowded environment.

3.3.4 SocialGym2

SocialGym2 is the updated version of the SocialGym simulation environment, which is designed for simulating and studying social behavior in human-robot collaboration scenarios. It is an open-source platform built on top of ROS, and it includes a range of pre-built environments and scenarios, as well as tools for creating custom ones [51]. It also contains tools for simulating cooperative and coordinated actions between humans and robots, and it provides support for complex multi-agent scenarios involving multiple robots and humans. Figure 3.8 shows the functional pipeline of SocialGym2.

Improvements on SocialGym1

- Multi-agent training;
- Control over the environment and simulator;
- Helper classes to implement rewards and observations;
- Tensorboard implementation to visualize the training process.

3.3.5 iGibson

iGibson (Interactive Gibson[9]) is an open-source simulation platform for building and testing intelligent robotic systems. It provides a realistic 3D environment for simulating the behavior and interactions of robots and their

[9] https://svl.stanford.edu/igibson/

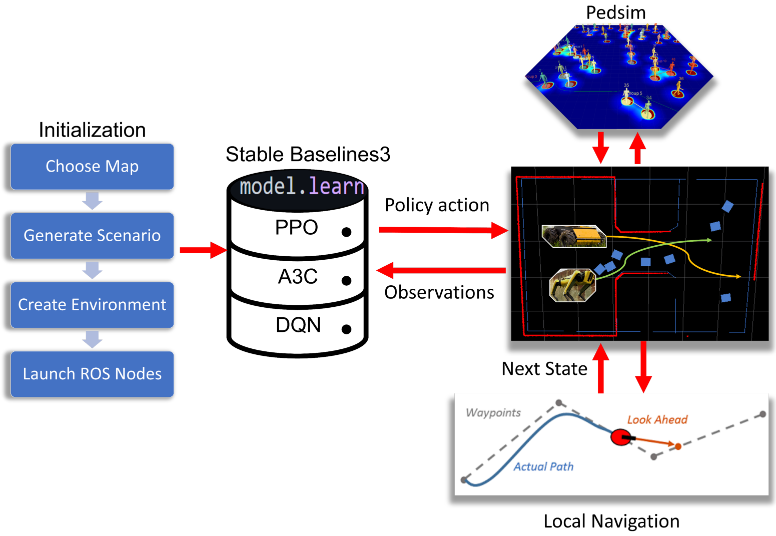


Figure 3.8: Structure of SocialGym2 [51].

surroundings. iGibson is designed to be highly customizable and modular, allowing researchers and developers to easily create and test complex robotic systems in a simulated environment. It allows robots to manipulate objects in the environment, open doors, and move through complex environments with obstacles and other challenges. It provides a range of tools and features for simulating physical interactions, including realistic physics simulation, dynamic lighting, and environmental effects like wind and rain. iGibson also provides support for a range of advanced robotics and computer vision features, such as semantic mapping, semantic segmentation, and object recognition. These features enable robots to perceive and understand their environment, and make intelligent decisions based on that information. Figure 3.9 shows an example of a home simulation.

3.3.6 CoppeliaSim

CoppeliaSim[10] (formerly known as V-REP, which stands for Virtual Robot Experimentation Platform) is a versatile and powerful robot simulation software. It is designed to simulate and visualize the behavior of robots and other robotic systems in various environments. CoppeliaSim provides a comprehensive framework for developing, testing, and evaluating robotic applications.

[10] https://www.coppeliarobotics.com/

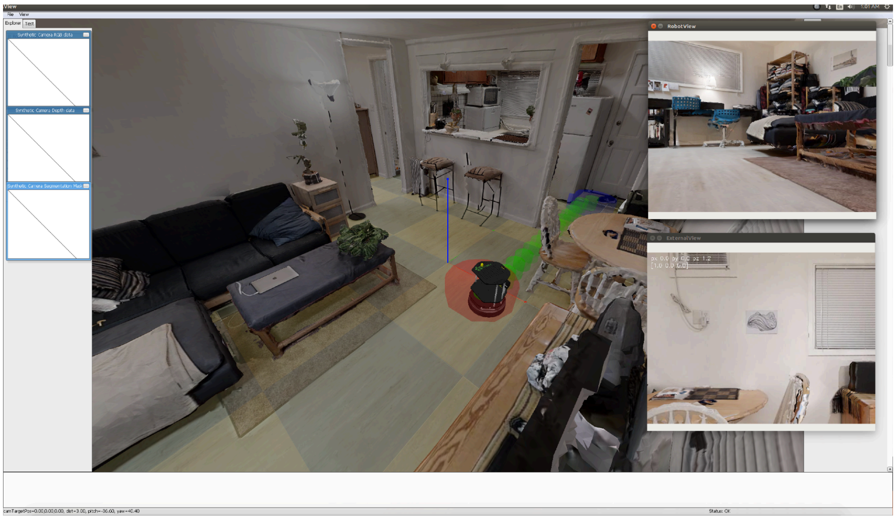


Figure 3.9: iGibson simulating a home-line environment.

CoppeliaSim offers a wide range of features, including:

- 3D Simulation Environment: It provides a 3D virtual world where you can create and simulate robots, environments, and objects. The simulation environment allows you to define and control the behavior of robots and test their performance in realistic scenarios;

- Physics Engine: CoppeliaSim includes a built-in physics engine that accurately simulates the dynamics and interactions of objects in the virtual environment. This enables realistic simulations of robot movements, collisions, and physical interactions;

- Robot Modeling and Control: It supports the creation and modeling of various types of robots, ranging from simple manipulators to complex humanoid robots. You can define robot kinematics, dynamics, and control strategies to simulate their behavior accurately;

- Sensor Simulation: CoppeliaSim allows you to simulate various sensors commonly used in robotics, such as cameras, lidars, proximity sensors, and force/torque sensors. This enables testing and evaluation of perception and sensing algorithms within the virtual environment;

- Programming Interfaces: It provides a wide range of programming interfaces (APIs) for different programming languages, including MATLAB, Python, C/C++, and Lua. These interfaces enable developers to interact with the simulation environment, control robots, and extract sensor data for algorithm development and testing;

- Remote Control and Communication: CoppeliaSim supports remote control and communication with external devices and software. It allows you to interface with real robots, control them from the simulation environment, or exchange data with other software systems.

3.3.7 MORSE

The MORSE simulator[11] is an open-source, flexible, and modular robotics simulator designed for research and development in the field of robotics and autonomous systems. MORSE stands for "Modular Open Robots Simulation Engine". Morse is:

- **Modular**: it is built with modularity in mind, allowing researchers and developers to easily extend its functionality by adding new robot models, sensors, actuators, and environments;
- **Scalable**: it can simulate single robots or large-scale multi-robot scenarios, making it suitable for testing algorithms related to swarm robotics, multi-agent systems, and collaborative tasks;
- **Community-driven**: it is an open-source project, and it benefits from contributions and support from a community of developers and researchers;
- **ROS integrated**: it allows researchers to test and develop ROS-based robotic applications in a simulated environment.

3.4 The SPENCER project

This thesis work focuses on context-based people motion trajectory prediction. Naturally, for this to work, a means of detecting and tracking people in the environment is needed. SPENCER [31] is an EU-funded research project in the area of robotics, covering in particular the fusion of perception, social understanding, and decision-making for autonomous robots. In particular, the spencer people tracking package contains a ROS-based multi-modal people detection and tracking framework. It employs both laser and camera sensors to detect people and it tracks them by fusing data from the different sensors via a fusion pipeline. As it was developed for its own robot, with its sensors and related ROS topics, it was necessary to adapt this package to the TIAGo robot. This was done by properly changing the input topics from which the framework takes the sensor data in the launch files. Moreover, since the TIAGo and SPENCER sensors publish on topics at a different rate, small changes that reflected this difference in the code were needed. Tracked people are published on

[11] `https://www.openrobots.org/morse/doc/stable/morse.html`

the /spencer/perception/tracked _persons topic with a custom message. An example of SPENCER detections can be seen in Figure 3.10, while Figure 3.11 shows SPENCER detecting the author of this thesis.

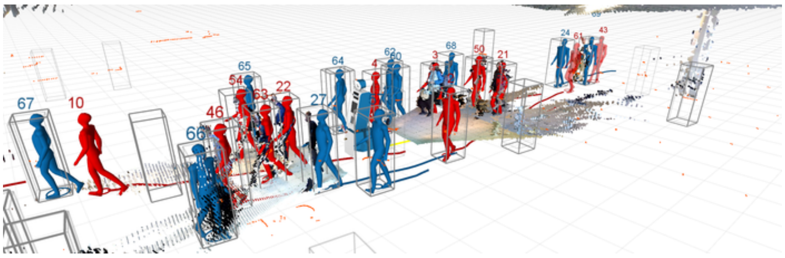
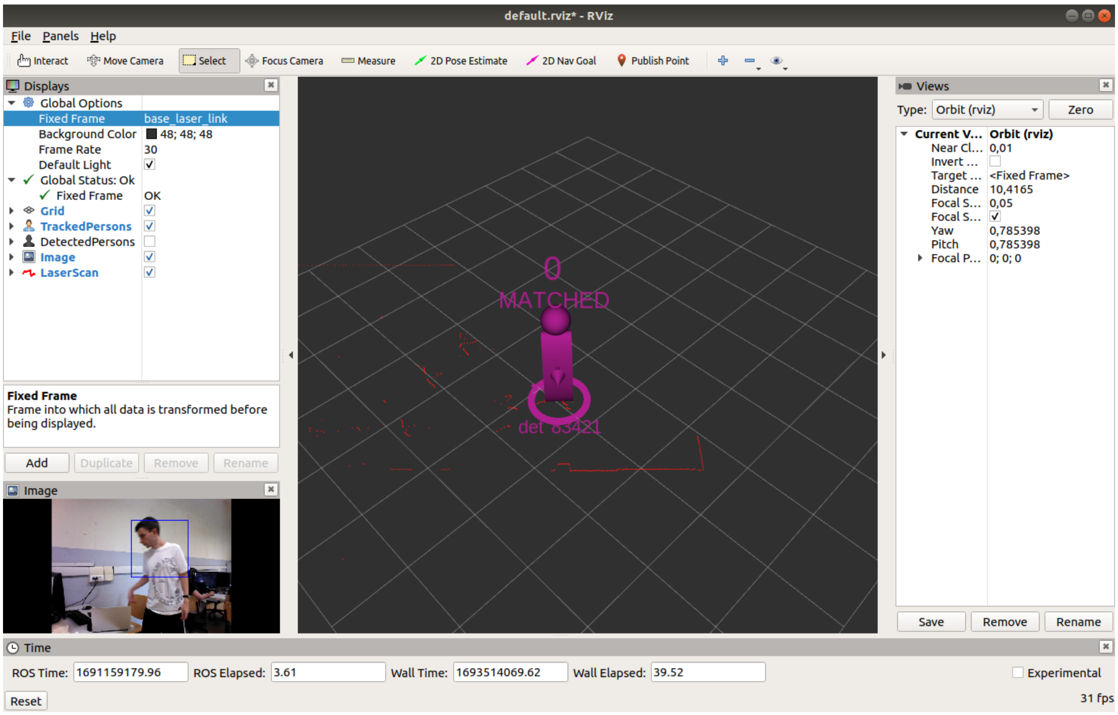


Figure 3.10: People detection and tracking using SPENCER [12]



Global Options	
Fixed Frame	base_laser_link
Background Color	■ 48; 48; 48
Frame Rate	30
Default Light	☑
✓ Global Status: Ok	
✓ Fixed Frame	OK
Grid	☑
TrackedPersons	☑
DetectedPersons	□
Image	☑
LaserScan	☑

Fixed Frame
Frame into which all data is transformed before being displayed.

Type: Orbit (rviz)

Current V...	Orbit (rviz)
Near Cl...	0,01
Invert ...	□
Target ...	<Fixed Frame>
Distance	10,4165
Focal S...	0,05
Focal S...	☑
Yaw	0,785398
Pitch	0,785398
Focal P...	0; 0; 0

ROS Time: 1691159179.96 ROS Elapsed: 3.61 Wall Time: 1693514069.62 Wall Elapsed: 39.52 □ Experimental

31 fps

Figure 3.11: Spencer tracking my movement, running on the TIAGo robot at the Autonomous Robotics Laboratory.

3.5 Hardware

The vast majority of this work was developed on an Intel i5 4460, 16 GB DDR3 RAM, GTX 1650 machine running Ubuntu 18.04.6 LTS and ROS Melodic.

[12] https://github.com/spencer-project/spencer_people_tracking

4 Datasets

In this chapter, the datasets used for training and testing will be examined.

4.1 JackRabbot

In order to employ a learning model for trajectory and context prediction, it is necessary to have data for training and testing. In particular, for this task, human motion data is needed, e.g. position over time, motion velocities and angles. Many datasets contain this kind of information from a top-view perspective, which is not acceptable for this work, as data from the point of view of the robot is needed. Thus, the choice falls on the JackRabbot dataset[1]. The JackRabbot dataset is a collection of sensor readings, images and videos, odometry and localization data, human bounding boxes annotations, action labels and body pose annotations collected from the homonymous robot at Standford University. Data is collected from the robot various sensors and is published at 7.5 Hz.

4.1.1 Data processing

The JackRabbot dataset is structured in many different folders, each containing different valuable data: RGB images, 2D and 3D pointclouds, labels for classification and, most importantly, 2D and 3D detections. 2D and 3D detections are collected in a series of large json files: each detection contains the

[1] `https://jrdb.erc.monash.edu/`

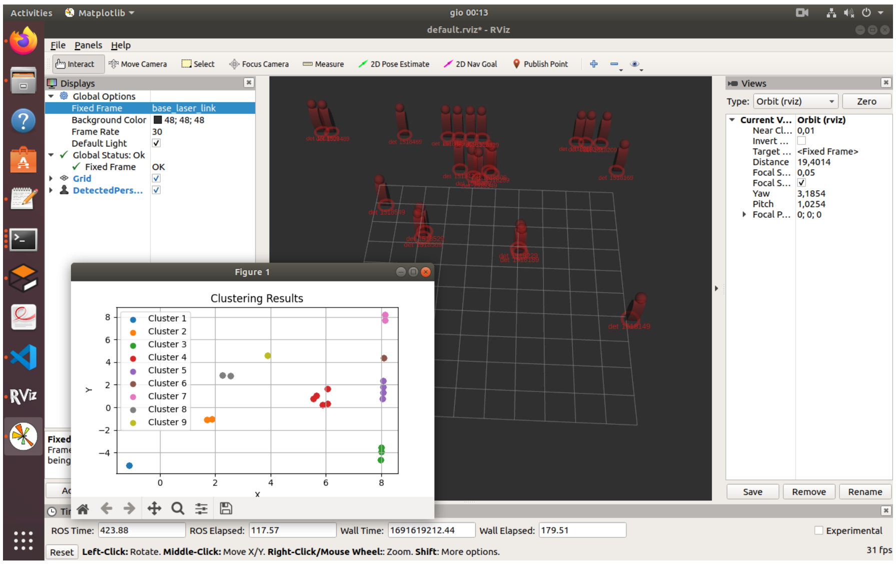
coordinates of its bounding box, its rotation angle, the id of the file associated with the detection, the label (currently pedestrian only) and the corresponding confidence score. The 3D detections are particularly useful because this work aims at predicting motion of humans from the perspective of the robot, which is exactly how these detections were recorded. The dataset_parser _multi _file.py contains the code employed to process the data so that it can be used with a learning model. Specifically, for each file a three-dimensional tensor is built: the first dimension is the batch size (i.e. the number of people detected), the second dimension is the sequence length (i.e. number of consecutive detections for each person) and the third dimension involves the features. Naturally, not every person is tracked (and detected) for the same amount of time, but the sequence length dimension must be uniform: to address this issue the maximum sequence length of the file is kept, and people with less detections have their sequences padded with their last known detection. The features involve both values for trajectory prediction and context prediction. For the former, the x and y coordinates and the rotation value is used, while the latter will be discussed in the next section.

4.1.2 Integrating JackRabbot with context

Thus, the JackRabbot dataset is a great source of data for the trajectory prediction task treated in this work. However, as explained before, the dataset (in particular the 3D labels directory) contains only the pose of the detection, its ID, its labels and a confidence score: there is only a reference to the social group with regards to the information needed for the context prediction task displayed in Subsection 5.2.1. However, this issue is easily solvable. First of all, since this approach is based on clustering, it is necessary to select an approach to compute the clusters. The choice falls on the DBSCAN (Density-Based Spatial Clustering of Applications with Noise, Algorithm 4) clustering algorithm, which is the most popular clustering algorithm in the literature. DBSCAN is used to cluster each detected person using its starting position, as seen in Figure 4.1. In this way, groups are created artificially and resemble closely the situation in which the original data was first recorded by the Stanford University researchers, despite some minor differences caused by the unknown time frame of each detection.

Algorithm 4 DBSCAN algorithm

```
Initialize an empty set $visited$ to keep track of visited points
Initialize an empty list of clusters $clusters$
for all unvisited points $p$ in $D$ do
  if point $p$ is not visited then
    Mark point $p$ as visited
    Find the neighbors of point $p$ within distance $\varepsilon$
    if number of neighbors $< MinPts$ then
      Mark point $p$ as Noise
    else
      Create a new cluster $C$ and add point $p$ to $C$
      Find the neighbors of point $p$ within distance $\varepsilon$
      for all neighbor $q$ of $p$ do
        if neighbor $q$ is not visited then
          Mark neighbor $q$ as visited
          Find the neighbors of point $q$ within distance $\varepsilon$
          if number of neighbors $>= MinPts$ then
            Add neighbors of $q$ to the neighbors of $p$
          end if
        end if
        if neighbor $q$ does not belong to any cluster then
          Add neighbor $q$ to cluster $C$
        end if
      end for
      Add cluster $C$ to the list of clusters
    end if
  end if
end for
return Clusters and Noise Points
```



default.rviz* - RViz

Displays

Global Options	
Fixed Frame	base_laser_link
Background Color	48; 48; 48
Frame Rate	30
Default Light	☑
Global Status: Ok	
Fixed Frame	OK
Grid	☑
DetectedPers...	☑

Views

Type: Orbit (rviz)

Current V...	Orbit (rviz)
Near Cl...	0,01
Invert ...	□
Target ...	<Fixed Frame>
Distance	19,4014
Focal S...	0,05
Focal S...	☑
Yaw	3,1854
Pitch	1,0254
Focal P...	0; 0; 0

ROS Time: 423.88 ROS Elapsed: 117.57 Wall Time: 1691619212.44 Wall Elapsed: 179.51 □ Experimental

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.

31 fps

Figure 4.1: An example of clusters generated in simulation.

4.2 ATC Shopping Center dataset

The ATC Shopping Centre dataset [2] is a collection of data regarding human motion, recorded, as the name suggests, in 2012 in the "ATC" shopping center in Osaka, Japan. The dataset contains both raw sensor readings and processed csv files of people detection data, captured at 30 Hz. It is a very large dataset, consisting of a total of 92 days of recordings: for this reason, testing on the complete dataset is too demanding both in terms of space and time. The literature deals with this issue by considering only a representative subset of the original dataset, consisting of 4 days [69]. The same is done for evaluating this work. Figure 4.2 shows some example frames with the map of the shopping center.

4.3 ETH and UCY

ETH [3] and UCY [4] are two publicly available datasets consisting of manually marked pedestrian identifiers and positions on recorded video, sampled at 2.5 Hz from a top view camera. Their name refers to the ETH Zurich University and to the University of Cyprus. Both ETH and UCY are further divided into

[2] https://dil.atr.jp/crest2010_HRI/ATC_dataset/

[3] https://icu.ee.ethz.ch/research/datsets.html

[4] https://www.ucy.ac.cy/?lang=en

subsets: ETH is composed of ETH-HOTEL and ETH-UNIV, while UCY contains UCY-ZARA01, UCY-ZARA02 and UCY-UNIV. For the purpose of this work, ETH-UNIV and UCY-ZARA01 will be used for testing. Figure 4.3 shows an example captured frame in the UCY dataset.

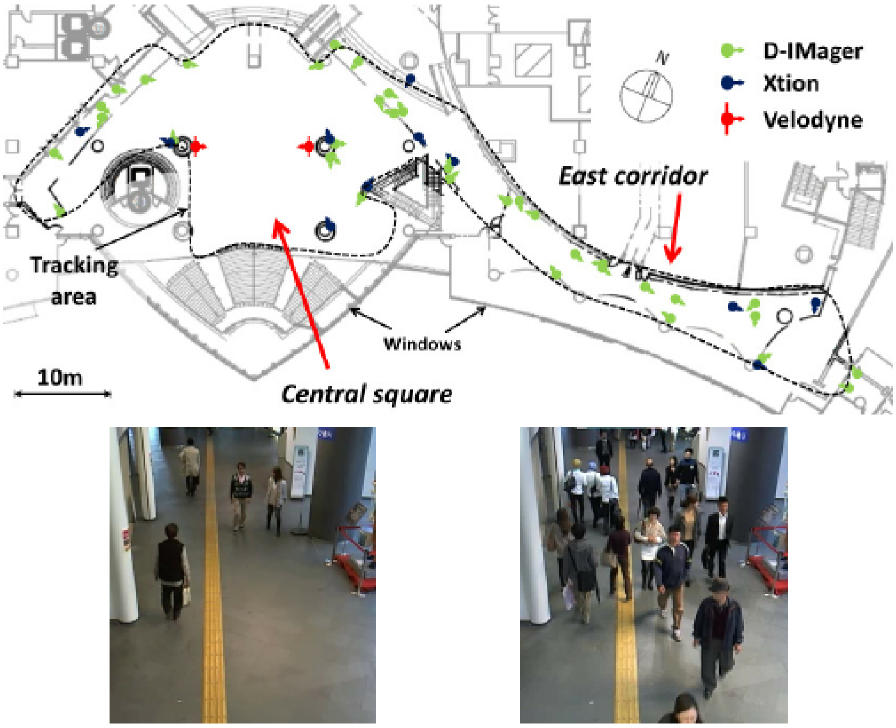


Figure 4.2: ATC shopping center.

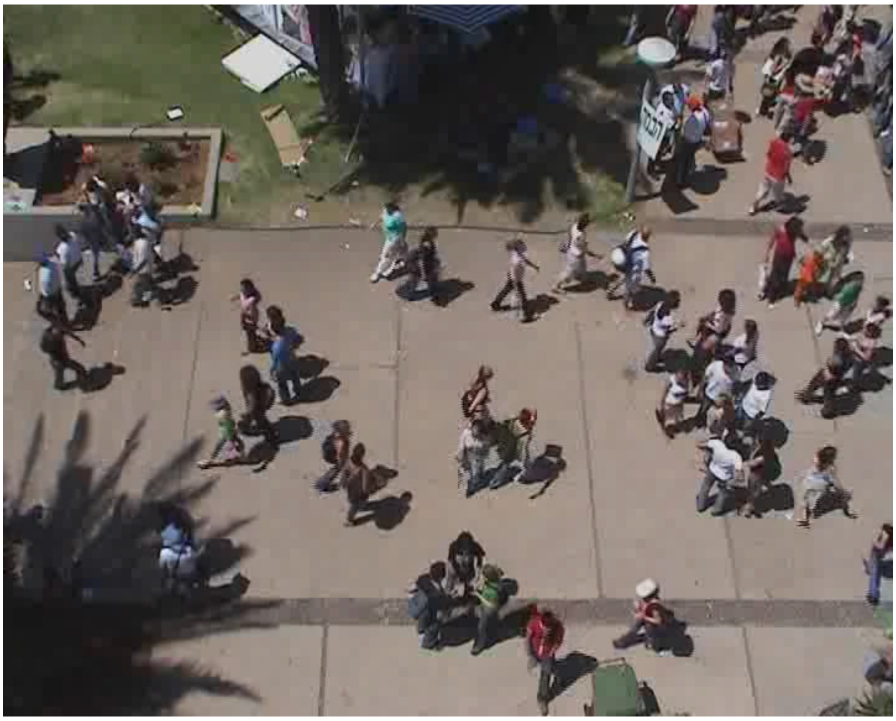


Figure 4.3: An example of an image from the UCY dataset.

5 Approach and model analysis

In this chapter, choices regarding the approach and the learning model proposed in this work will be presented.

5.1 Trajectory prediction

The first goal of this work is to predict the trajectory of a moving person in a specific time window, using a deep learning model, similarly to Figure 5.1. Considering the environment from the point of view of the robot (i.e., from its camera), trajectory is intended as a timed sequence of poses in 2D space. These poses should be predicted and published as a message on a specific topic, so that the motion planner can receive them and account for them during planning.

5.2 Reasoning about context

The idea of working with context understanding and prediction came while watching a video of an autonomous robot navigating through the crowd in an airport. The camera of the robot captured a great variety of people such as men, women, children, some of them in groups (children with their parents, people in queues or talking with other people in their surroundings) and some of them walking alone (chaotically). These observations could prove useful when dealing with social navigation: robots could optimize the path they are following by adapting to the situation they detect in front of them. Ideally,

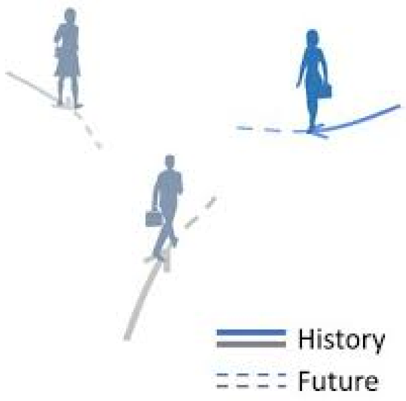


Figure 5.1: People trajectory prediction: the history trajectory is recorded and used to predict the future trajetory.

robots should distinguish between chaotic situations (e.g., people traversing the crowd alone) and more structured scenarios (e.g., queues in front of the security check/groups of people who know each other and are moving together or standing still).

5.2.1 Capturing context information

Firstly, where does context information lie in the environment? The first intuition comes from the fact that chaotic and ordered situations can be distinguished from the behaviour of the components of the situation. Ideally, in an ordered situation persons behave similarly to one another (e.g. standing still in a queue), while in chaotic situations behaviours vary greatly (e.g. different motion velocities, directions). Initially, this idea of "distinction" between chaos and order can be captured with clustering techniques: grouping people by similarity is a basis for understanding the context in which they are moving. The clustering algorithm chosen from this instance is the popular DBSCAN. As you can see in Figure 5.2, Figure 5.3, Figure 5.4 and Figure 5.5, this first approach was tested in simulation by deploying the robot in a world populated only with pedestrian models. After obtaining their positions via SPENCER, clustering is performed. Another source of context information can be computed from clusters: various indexes can be used to evaluate the quality of the clusters, taking inspiration from the literature [69], [1], [47], [65], [30], [67]:

- **Silhouette coefficient**: is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). It is commonly

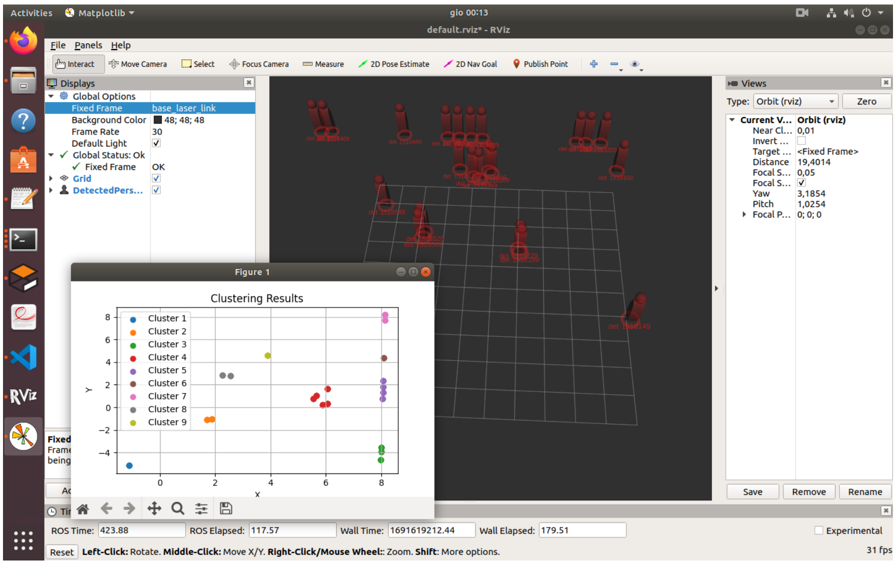
used to determine if the number of the computed clusters is acceptable;
Silhouette Score = $\frac{1}{N}\sum_{i=1}^{N}\frac{b(i)-a(i)}{\max\{a(i),b(i)\}}$

- **Davies-Bouldin index**: the average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score;
  Davies-Bouldin Index = $\frac{1}{N}\sum_{i=1}^{N}\max_{j\neq i}\left(\frac{S_i+S_j}{d(C_i,C_j)}\right)$

- **Calinski-Harabasz Index**: is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). Here cohesion is estimated based on the distances from the data points in a cluster to its cluster centroid and separation is based on the distance of the cluster centroids from the global centroid;
  Calinski-Harabasz Index = $\frac{\mathrm{Tr}(B)}{\mathrm{Tr}(W)}\times\frac{N-k}{k-1}$

- **Dunn Index**: It is calculated as the lowest intercluster distance (i.e. the smallest distance between any two cluster centroids) divided by the highest intracluster distance (i.e., the largest distance between any two points in any cluster). The higher the index, the better the clustering.
  Dunn Index = $\frac{\min_{i\neq j}\min_{x\in C_i,y\in C_j}d(x,y)}{\max_i\max_{x,y\in C_i}d(x,y)}$

These indexes reflect some qualities of the context: where these values are higher, the clustering is better, i.e., the environment is more structured and the situation is more ordered. This is particularly clear in Figure 5.4 and Figure 5.5.



Global Options
Fixed Frame base_laser_link
Background Color 48; 48; 48
Frame Rate 30
Default Light ☑
✓ Global Status: Ok
✓ Fixed Frame OK
Grid ☑
DetectedPers... ☑

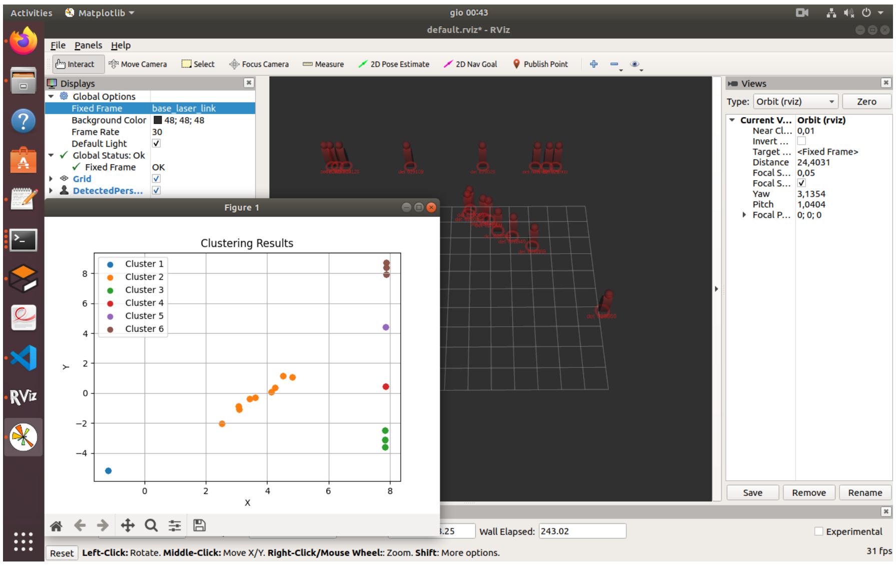
Type: Orbit (rviz)
Current V... Orbit (rviz)
Near Cl... 0,01
Invert ... ☐
Target ... <Fixed Frame>
Distance 19,4014
Focal S... 0,05
Focal S... ☑
Yaw 3,1854
Pitch 1,0254
Focal P... 0; 0; 0

ROS Time: 423.88 ROS Elapsed: 117.57 Wall Time: 1691619212.44 Wall Elapsed: 179.51 ☐ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options.

31 fps

Figure 5.2: An example of clusters generated in simulation.

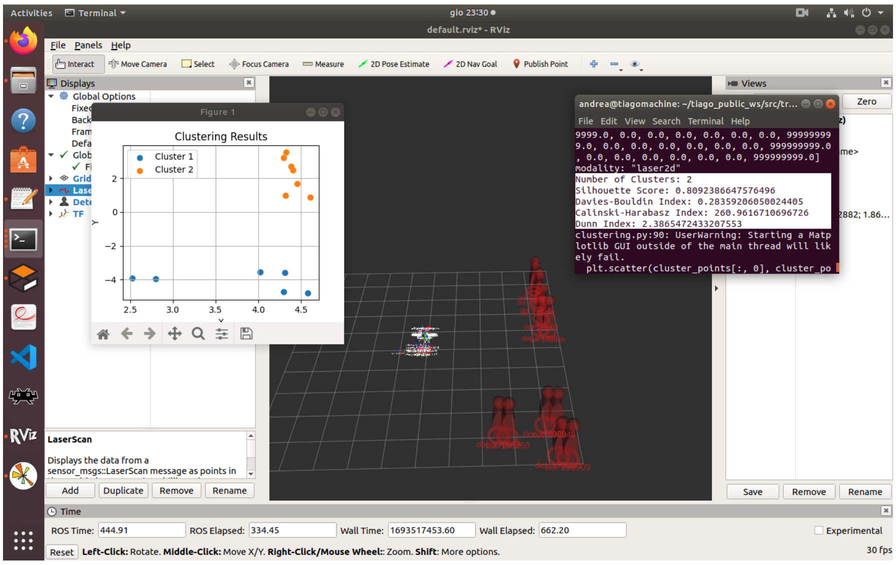


Global Options
Fixed Frame base_laser_link
Background Color 48; 48; 48
Frame Rate 30
Default Light ✓
Global Status: Ok
Fixed Frame OK
Grid ✓
DetectedPers... ✓

Type: Orbit (rviz) Zero
Current V... Orbit (rviz)
Near Cl... 0,01
Invert ... □
Target ... <Fixed Frame>
Distance 24,4031
Focal S... 0,05
Focal S... ✓
Yaw 3,1354
Pitch 1,0404
Focal P... 0; 0; 0

.25 Wall Elapsed: 243.02

Figure 5.3: Another similar example of people clustering, this time with people forming two queues: one in the middle of the space and the other on the back.



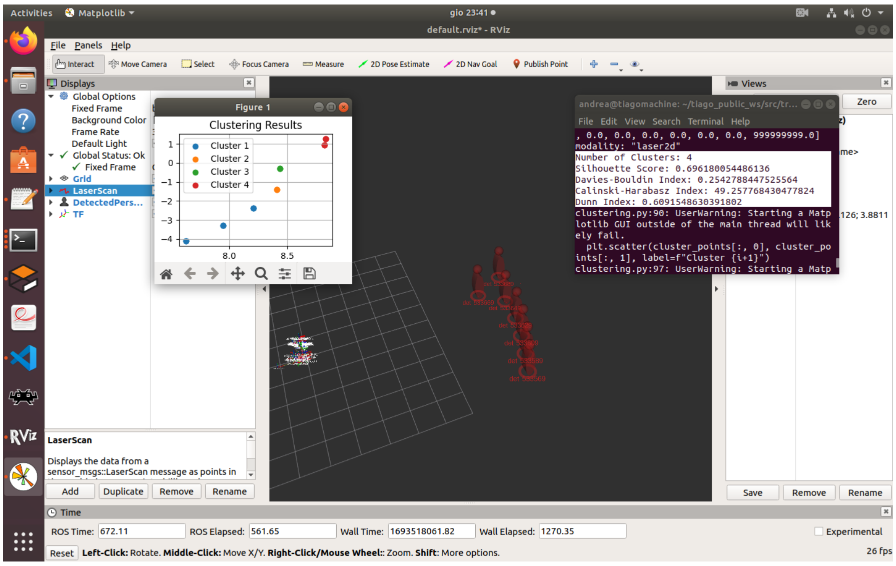
```
9999.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 99999999
9.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 999999999.0
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 999999999.0]
modality: "laser2d"
Number of Clusters: 2
Silhouette Score: 0.8092386647576496
Davies-Bouldin Index: 0.28359206050024405
Calinski-Harabasz Index: 260.9616710696726
Dunn Index: 2.3865472433207553
clustering.py:90: UserWarning: Starting a Matp
lotlib GUI outside of the main thread will lik
ely fail.
  plt.scatter(cluster_points[:, 0], cluster_po
```

LaserScan

Displays the data from a sensor_msgs::LaserScan message as points in

ROS Time: 444.91 ROS Elapsed: 334.45 Wall Time: 1693517453.60 Wall Elapsed: 662.20

Figure 5.4: Another similar example of people clustering. Here the groups of people are far apart and easily recognizable, and the indexes computed are high.



```
, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 999999999.0]
modality: "laser2d"
Number of Clusters: 4
Silhouette Score: 0.696180054486136
Davies-Bouldin Index: 0.2542788447525564
Calinski-Harabasz Index: 49.257768430477824
Dunn Index: 0.6091548630391802
clustering.py:90: UserWarning: Starting a Matp
lotlib GUI outside of the main thread will lik
ely fail.
  plt.scatter(cluster_points[:, 0], cluster_po
ints[:, 1], label=f"Cluster {i+1}")
clustering.py:97: UserWarning: Starting a Matp
```

Figure 5.5: Another similar example of people clustering. This time the algorithm is unable to find a good clusters due to the low distance threshold set. The computed indexes are lower than the previous case, as you can see.

How can such data help in predicting the context in which people are moving? The idea is to input the number of clusters, indexes, angle and velocity of each person to the learning model. The network will then output a value (regression), which can be treated as a measure of how ordered/chaotic is the environment: this is due to the fact that the indexes in Subsection 5.2.1 increase with the quality of the clusters. In turn, if the quality of the clusters is high (scores are high), the situation is well structured, and thus, more organized. Thresholding will then be used to classify the context of the environment into "chaotic", "mixed" and "structured" are the names of the labels chosen, each one corresponding to a state of the environment. The thresholds chosen reflect the behaviour of the clustering indexes: under -0.5 the context is chaotic, between -0.5 and 0.5 it is mixed and over 0.5 it is structured. These thresholds are inspired from the literature [21], and adjusted to fit our task by working similarly to the clustering index.

After the clusters have been generated, it is possible to compute the indexes discussed in Subsection 5.2.1. On a side note, the velocities of each person are based on the definition of velocity ($v = \delta x/t$) and are straightforward to compute, even without performing clustering. For simplicity, the assumption of constant speed throughout the motion is made. The silhouette coefficient, Davis-Bouldin index and Calinski-Harabasz index are computed using the standard scikit-learn function, while concerning the Dunn index, the implementation of the Algorithm 5 was used.

Algorithm 5 Dunn Index algorithm

```
Require: $Set of datapoints X, Cluster assignments C$
Ensure: Dunn Index value
  $D_{\text{min}} \leftarrow +\infty$
  $D_{\text{max}} \leftarrow -\infty$
  for all clusters $i, j$ where $i \neq j$ do
    for all points $x$ in cluster $C_i$ do
      for all points $y$ in cluster $C_j$ do
        $d \leftarrow$ EuclideanDistance($x, y$)
        if $d < D_{\text{min}}$ then
          $D_{\text{min}} \leftarrow d$
        end if
      end for
    end for
    for all points $x, y$ in cluster $C_i$ do
      $d \leftarrow$ EuclideanDistance($x, y$)
      if $d > D_{\text{max}}$ then
        $D_{\text{max}} \leftarrow d$
      end if
    end for
  end for
  return $\frac{D_{\text{min}}}{D_{\text{max}}}$
```

5.3 Modeling the memory based on past observations

The trajectory prediction task consists of computation on data indexed in time order: trajectories are in fact a sequence of points captured at consecutive intervals in time (discrete-time data). This particular structure of the data requires an adequate model capable of capturing this temporal relation. The class of recurrent neural networks addresses this necessity. Recurrent neural networks are characterized by recurrent connections: at each stage the network receives both the input from the current time-step and the hidden state from the previous time-step. This hidden state contains values that represent information related to the inputs of the previous time-steps. In this way, the hidden state acts as a "memory" object that is capable of keeping track of past inputs. Similarly to classical neural networks, during training recurrent neural networks also back-propagate the gradient to adjust the weights. However, the back-propagation through recurrent connections can cause the gradient to "vanish" (i.e., it tends to 0, if the original gradient is small) or "explode" (i.e., it tends to infinity, if the original gradient is large). The vanishing gradient problem is well-known and explored in the field of deep learning. This issue is (partly) solved in state-of-the-art recurrent neural networks by deploying Long Short Term Memory models (see Figure 5.6 and Subsection 5.3.1).

5.3.1 Long Short Term Memory

The main characteristic of the Long Short Term Memory model is its structure: a LSTM unit is composed of a memory cell, an input gate, an output gate and a forget gate. The memory cell stores information over time, capturing even long temporal relations, and it can be read or written on during training. The gates are used to control the flow of data through the network:

- The input gate activates when information from the current time-step needs to be saved in the memory cell;
- The output gate activates when the current data stored in the memory cell should be used as output;
- The forget gate activates when the data stored in the memory cell is no longer useful and is thus discarded.

Instead of classical back-propagation, LSTMs employ back-propagation through time (BPTT), which allows the model to compute the gradient across multiple time steps. LSTMs are very powerful and have been adopted in various fields, such as:

- Natural language processing;
- Speech recognition;
- Time-series forecasting (as in this work);
- Robotics;
- Music composition.

Figure 5.6 shows a representation of a LSTM unit.

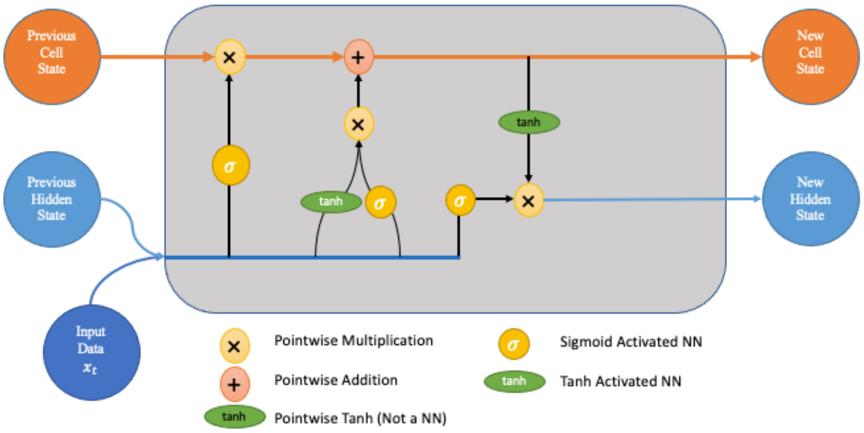


Figure 5.6: LSTM unit. You can see the input and the previous hidden state forming the new hidden state, and the previous cell state contributing to the new cell state. Each sigmoid represents a gate: the first is the input gate, the second is the forget gate and the third and final is the output gate.

5.4 Our solution

This section contains the description of the new model we propose to tackle the task of trajectory prediction in crowded environments. First, in Subsection 5.4.1 we describe our proposed context-based model. The proposed context-based model works with LSTM layers for the trajectory part, but incorporates also context information captured from the environment to improve the quality of the predicted trajectories.

5.4.1 The proposed context-based model

The context-based model is our new proposed solution to the problem of trajectory prediction. It uses both the recorded motion trajectory of humans and the information related to the context, captured as explained in Subsection 5.2.1, to make a prediction. These collections of data are concatenated and fed to the rest of the network: in this way, the model learns to couple the situation of the environment with the motion of people, improving its understanding of the world and of people motion. It is also necessary to define a time-window, i.e., how far in time we want to predict the motion of a person. Considering the 7.5 Hz sampling rate of the JackRabbot dataset, which will be used for training, ideally each second of trajectory corresponds to 7.5 input or output poses. So, for instance, if we want a prediction window of 5 seconds, the model needs to take in input 35 poses and outputs 35 poses as well.

The architecture of the trajectory part of the model relies on a 3-layered LSTM. The context data is fed to a fully connected layer; the outputs of the trajectory side and the context side of the network are then concatenated and fed to other fully connected layers. In short, the architecture is as follows (see Figure 5.7):

- 3 LSTM layers with dropout taking the trajectories of tracked humans;
- A fully connected layer taking in input the variation in position, velocity, angle, number of clusters, Silhouette, Davies-Bouldin, Calinski-Harabasz and Dunn scores;
- A concatenation layer, which concatenates the outputs of the final LSTM layer of the trajectory-related side of the network with the output of the fully connected layer of the context side of the network;
- 3 fully connected layers process the concatenated information;
- The network is finally divided: a fully connected layer outputs the predicted trajectory, while another parallel fully connected layer outputs the predicted context.

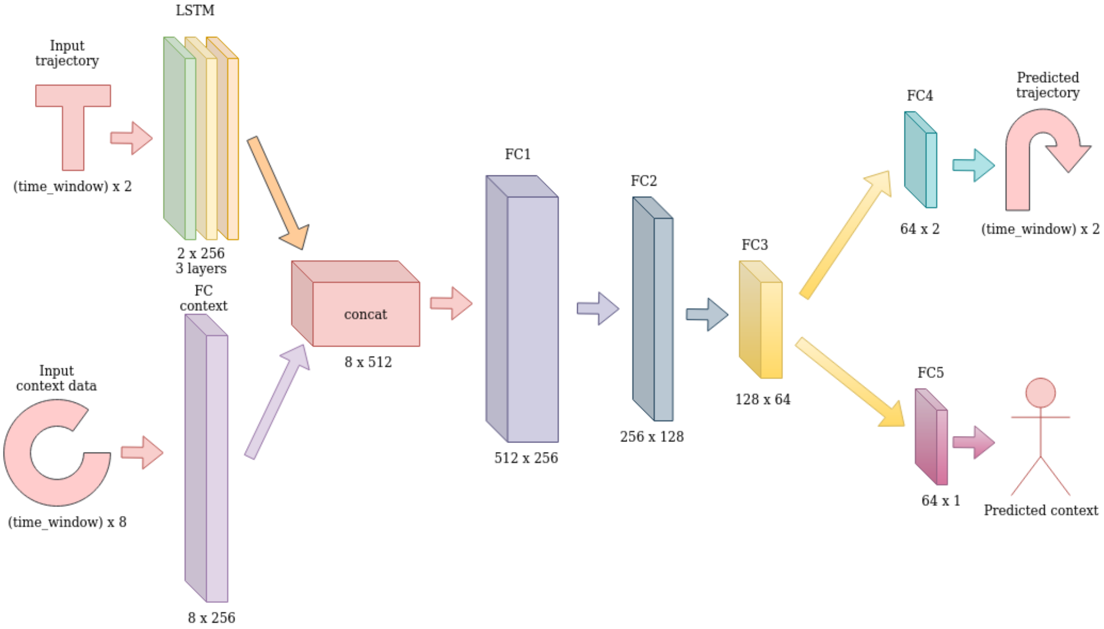


Figure 5.7: Structure of the proposed context-based model.

As already stated, this work proposes trajectory prediction based on context-learning. However, in order to evaluate this approach, it is necessary to compare it to standard state-of-the-art trajectory prediction models, in particular to those based only on trajectories. For this reason, both a context-free model and a context-based model were built. The next subsection describes the context-free model. This model's purpose is to serve as an initial evaluation of our solution: the new context-based model we propose, which is described in Subsection 5.4.1.

5.4.2 The proposed context-free model

The context-free model needs only the recent motion trajectory of humans to work, i.e., a temporal sequence of their poses up to the current time. As explained before, this is the perfect use-case of LSTM models. Finally, the model architecture is based on state-of-the-art trajectory prediction architectures and resembles the trajectory part of our proposed context-based model (see Figure 5.8):

- The 3 LSTM layers are the core of the model, as expected. The number of layers was initially larger to match the literature works, but it was tuned down in order to further diminish the vanishing gradient problem;
- A dropout layer is added after the LSTM layers in order to reduce overfitting. The dropout probability is set to 0.2;
- A series of fully connected layers closes the model, with dropout set to 0.2

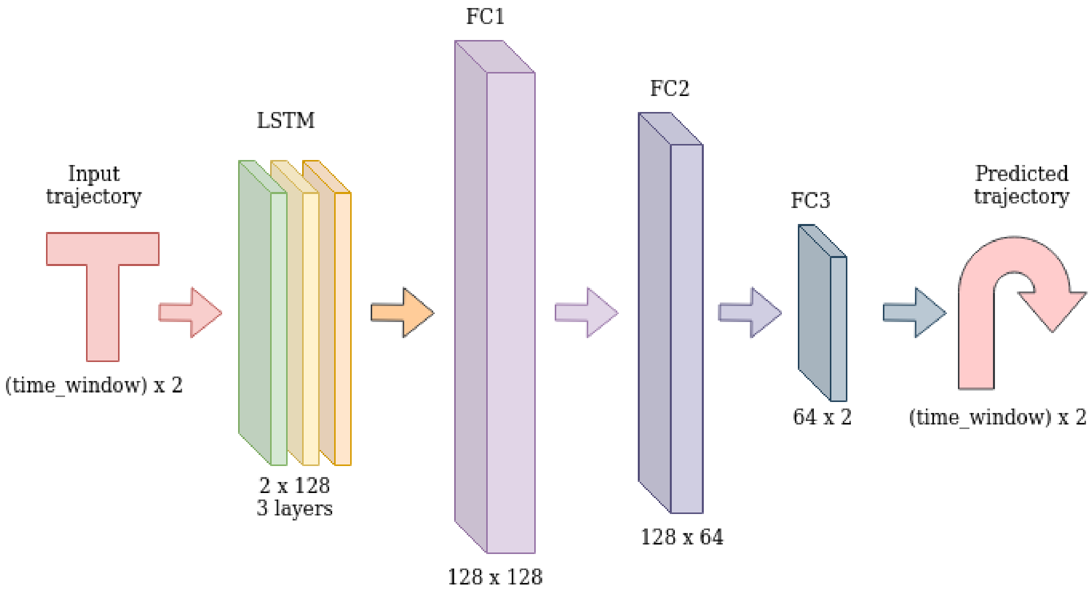


Figure 5.8: Structure of the proposed context-free model.

5.5 Training

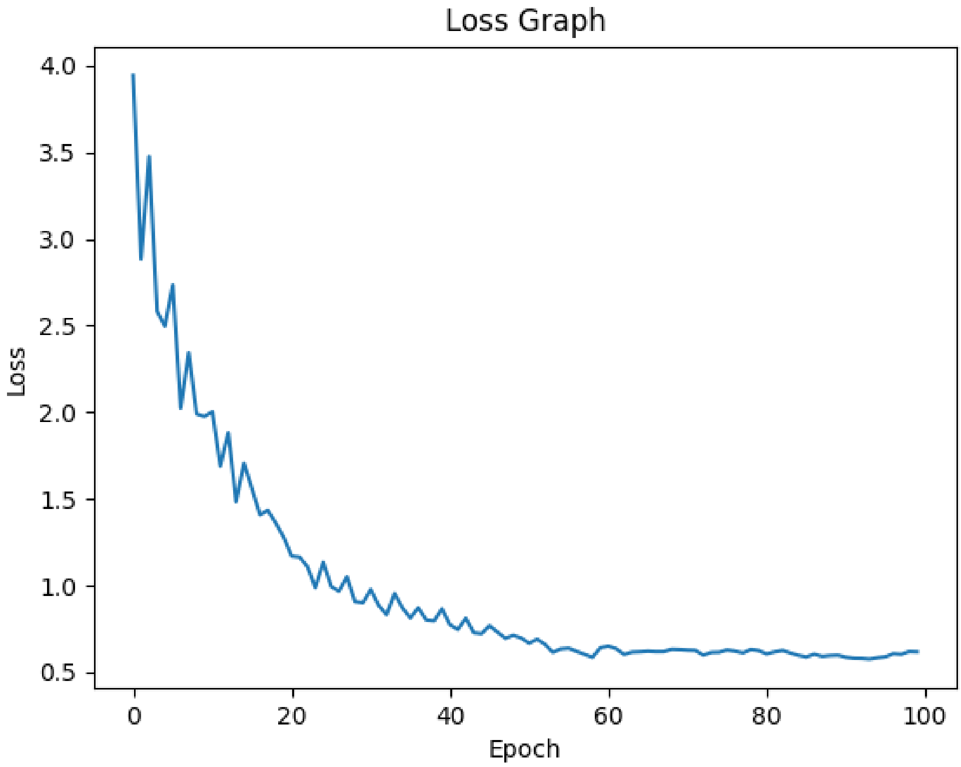


Figure 5.9: Training loss of the context-based model, trained on JackRabbot for 100 epochs.

The two models were trained as similarly as possible in terms of hyperparameters, in order to obtain a fair comparison. In particular:

- The chosen loss function is the Root-Mean-Square Error (RMSE). The reason of this choice lies in the effect of the root on the loss value: RMSE particularly penalizes large error values, which is preferred for this task. In fact, large errors in the predicted trajectory have a serious effect (e.g., planning the motion of the robot without accounting for a person which is moving close to it, but was predicted to be far from the robot). On the contrary, minor errors should not affect the planning of the robot too much. The loss of the proposed context-based model over 100 epochs is plotted in Figure 5.9;

- The chosen optimizer is Adam, with its learning rate set to 0.001. Adam is a popular optimization algorithm, preferred to standard stochastic gradient descent due to its efficiency and its adaptability to problems that deal with large data or parameters;

- Scaling the data has proven useful to prevent the LSTM layers to get stuck in local minima and perform poorly. The popular StandardScaler from the scikit-learn library was used here: it standardizes the features by removing the mean and scaling to unit variance;

- Both models were trained incrementally to 100 epochs, with a 5 seconds time-window for prediction. Checkpoints and other models are present in the models directory of the git repository;

- Concerning the network specific hyperparameters (number of layers, layer size, etc...), the values were chosen empirically by looking for acceptable complexity without sacrificing accuracy too much. The reason for this choice is our goal of possibly running the model directly on-board the robot, which is not very powerful, and as close to real-time as possible.

6 Evaluation

The results of the evaluation phase of this work are examined in this chapter. First, we present the evaluation metrics chosen to evaluate the performance of our proposed context-based model. Then, we present the results of the tests on the ATC dataset of the context-based and of the context-free models, and the results of the tests on the ATC, ETH and UCY datasets of the context-based model, comparing it to state-of-the-art methods.

6.1 Evaluation Metrics

To evaluate the performance of the proposed context-based model and of the simple context-free model, the following metrics have been considered:

- **Average Displacement Error (ADE)**: it refers to the mean square error over all predicted points of every trajectory and the ground truth points.

$$\text{ADE} = \frac{1}{N}\sum_{t=1}^{N} \|\hat{p}_t - p_t\| \tag{6.1}$$

- **Final Displacement Error (FDE)**: is the distance between the final predicted position and the final ground truth position.

$$\text{FDE} = \|\hat{p}_N - p_N\| \tag{6.2}$$

- **Miss Rate (MR)**: Number of predicted trajectories within a certain distance from the ground truth trajectories over the total number of trajectories.

$$\text{Miss Rate (MR)} = \frac{\text{Number of Predictions Within Range}}{\text{Total Number of Predictions}} \tag{6.3}$$

- **MaxDist**: Maximum distance between a predicted trajectory and the ground truth trajectory.
- **Intersection over Union (IoU)**: area of the intersection of the trajectories over the area of the union of the trajectories. Since this measure deals with areas, 2D bounding boxes are built on top of each point of each trajectory in order perform the computation.

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}} \tag{6.4}$$

6.2 Testing and Analysis

The testing phase is structured as follows:

- The selected files of the ATC or ETH/UCY datasets are split into smaller pieces, processed (trajectories extracted and context data added) and the resulting tensors are saved in .pt format for later use;
- The trajectories extracted from the tensors are fed to the learning model and the resulting predicted trajectories are saved in a large csv file;
- The csv file containing the predicted trajectories for the whole dataset day file is read and the evaluation metrics are computed.

The trajectory-based model and the context-based model are compared by testing and computing the metrics on the ATC dataset, over 3 different prediction time windows: 5, 7.5 and 10 seconds. Both models were trained on a 5-seconds time window.

Table 6.1: Comparison of the two models using a 5 seconds prediction window.

Model	**Context-free**	**Context-based**
ADE(m)	4.456	4.074
FDE(m)	6.323	6.002
MR	0.651	0.683
MaxDist(m)	13.04	13.14
IoU	0.443	0.489

The context-based model is compared to state-of-the-art approaches on the ATC and ETH/UCY datasets.

Table 6.2: Comparison of the two models using a 7.5 seconds prediction window.

Model	**Context-free**	**Context-based**
ADE(m)	10.45	11.03
FDE(m)	13.01	12.92
MR	0.443	0.422
MaxDist(m)	31.07	26.03
IoU	0.321	0.309

Table 6.3: Comparison of the two models using a 10 seconds prediction window.

Model	**Context-free**	**Context-based**
ADE(m)	16.89	16.77
FDE(m)	22.34	23.04
MR	0.278	0.269
MaxDist(m)	47.32	45.89
IoU	0.175	0.182

Table 6.5: Comparison on the ETH-UNIV for 5 seconds prediction window, with models VLSTN, SRNN and MESRNN from [14], and SATNN and Trajectron++ from [42]. If a metric was not computed in the original work, a "-" is placed on the corresponding entry of the table.

Model	**Context-based**	**VLSTN**	**SRNN**	**MESRNN**	**SATTN**	**Trajectron++**
ADE(m)	0.521	0.031	0.015	0.013	0.33	0.41
FDE(m)	0.528	0.072	0.033	0.026	3.92	1.07
MR	0.977	-	-	-	-	-
MaxDist(m)	5.111	-	-	-	-	-
IoU	0.787	-	-	-	-	-

Table 6.6: Comparison on the UCY-ZARA01 for 5 seconds prediction window, with models VLSTN, SRNN and MESRNN from [14], and SATTN and Trajectron++ from [42]. If a metric was not computed in the original work, a "-" is placed on the corresponding entry of the table.

Model	**Context-based**	**VLSTN**	**SRNN**	**MESRNN**	**SATTN**	**Trajectron++**
ADE(m)	0.507	0.059	0.022	0.007	0.20	0.30
FDE(m)	0.518	0.157	0.056	0.016	0.52	0.77
MR	0.978	-	-	-	-	-
MaxDist(m)	5.110	-	-	-	-	-
IoU	0.788	-	-	-	-	-

Table 6.4: Comparison on the ATC dataset for a 10 seconds prediction window. If a metric was not computed in the original work, a "-" is placed on the corresponding entry of the table.

Model	**Context-based**	**CLiFF-LHMP[69]**
ADE(m)	16.77	8.826
FDE(m)	23.04	17.441
MR	0.269	-
MaxDist(m)	47.32	-
IoU	0.175	-

Table 6.7: Comparison on the UCY-ZARA01 for 5 seconds prediction window, with other models LSTM and S-LSTM from [42]. If a metric was not computed in the original work, a "-" is placed on the corresponding entry of the table.

Model	**Context-based**	**LSTM**	**S-LSTM**
ADE(m)	0.507	0.410	0.470
FDE(m)	0.518	0.880	1.00
MR	0.788	-	-
MaxDist(m)	5.110	-	-
IoU	0.978	-	-

Regarding the comparison between the context-based approach proposed in this work and the simpler context-free model (see Tables 6.1, 6.2 and 6.3), when predicting over the same time window used during training, the context-based model performs a bit better on almost every metric. In particular:

- ADE: the context-based model obtains a 9.6% increase in performance;
- FDE: the context-based model obtains a 5.1% increase in performance;
- MR: the context-based model obtains a 4.6% increase in performance;
- MaxDist: the context-based model obtains a 0.7% decrease in performance. This is the only metric that decreases;
- IoU: the context-based model obtains a 11.3% increase in performance;

In short, the context-based model performs up to 11% better (based on results of Table 6.1), which proves it is capable of generalizing better in average and predict trajectories closer to the ground truth. The only metric which is (very slightly) worse is the MaxDist: since the difference is almost nonexistent, it is not a serious issue, but interesting observations could be made. This is probably due to the effect of the context data on a particular instance of the predicted trajectory.

When increasing the time window for the predictions, in general, the metrics get worse quite a lot: this result is in part expected, as the time window used during training is 5 seconds and the models have learnt to generalize on that time frame, but in the end the MaxDist metric increases too dramatically.

In particular, comparing the two models on 7.5 and 10 seconds time windows:

- The variation in performance in almost every metric is extremely low, sometimes in favour of the context-based model and sometimes not;
- The MaxDist metric of the context-based model is lower in every case;

Again, the performance gets worse when increasing the time window. On a positive note, despite being more complex, the performance of the proposed context-based model are similar to the context-free model: the context data does influence the model in a negative way when predicting on different time windows. On the negative side, the context data does not necessarily help the model to deal with longer time windows, with the exception of the improvements on the MaxDist metric: on average, the context model performs similarly to the context-free one, but at least the maximum distance achieved from the former is lower then the one achieved from the latter.

Regarding the comparison between the context-based model and the state of the art models (see Tables 6.4, 6.5, 6.6, 6.7):

- In the comparison on the ATC dataset, the context-based model performs a bit worse than the state-of-the-art model (CLiFF-LHMP) with respect to ADE and FDE;
- It is important to note that in the original work, the CLiFF-LHMP model is intended for long term trajectory prediction (from 10 seconds up to 60 seconds);
- The advantage of the CLiFF-LHMP approach lies in the use of maps of dynamics: exploiting learned motion patterns, CliFF-LHMP is able to generate a map with explicit knowledge about obstacle, allowing it to predict realistic trajectories that follow the layout of the environment;
- In the comparison on ETH-UNIV and UCY-ZARA01 datasets, the context-based model performs similarly to SATTN and Trajectron++, even with a small improvement on the FDE metric;
- The performance of VLSTN, SRNN and MESRNN are outstanding, with MESRNN almost reaching a centimeter accuracy on average on trajectories predicted;
- The advantage of the MESRNN method lies in the use of meta-paths obtained from moving neighbours to improve trajectory prediction.
- The state-of-the-art models used here are in general very complex and run deeper than our context-based model. As our model has to eventually run on the hardware of a TIAGo robot with very limited computing capabilities, it was necessary to trade off the accuracy for minor complexity. It is acceptable to predict the motion with a small error as long as it is done fast enough for the robot to use this prediction during planning;
- In general, performance is comparable with works presented in [42]. These methods are also the most similar to our model, considering they are based on LSTM as well;
- As already stated, the models of [14] are very promising. The meta-path approach could be fused with our context model for a possible future work.

Regarding the comparison between the context-based approach and the context-free approach, some important observations can be made:

- Firstly, despite having the trajectory-learning part of the network in common, the context-based model is naturally more complex, as it takes in more inputs, and therefore the first layers of the network are larger, even though after the concatenation the dimensions of the following fully connected layers are the same in both models. This may have played a part in the increase in performance of the context-based model: in addition to the important context information retrieved and used by the network, the capability of the model to capture more information with more weights has a role on its effectiveness;

- Secondly, as explained in Section 4.1.1 of this work, not every person registered in the dataset has the same number of detections: this means that it was necessary to perform 0-padding to standardize the dimension of the tensors to be given in input to the network. The people with few detections were removed from the training dataset since the 0-padding would influence the performance of the models due to the high number of zeros. However, some 0-padded sequences still remain, and they may have partly affected the networks. However, it is expected that this is a very common issue, which means that state-of-the-art models have had to make similar assumptions when dealing with padding;

- Finally, the standardization of the scaler on input data may have an effect on performance: as explained in Chapter 5, scaling was performed on the data to improve training and reduce overfitting. The scaler was obviously fitted on the training data, in particular on trajectory coordinates. However, the context-based model takes in input the context information (namely, the indexes reported in Subsection 5.2.1) as well. Most of these indexes are in a range (commonly from -1 to 1), and express the situation of the whole environment. Therefore, they don't need scaling and are fed to the network as they are.

Figures 6.1, 6.2, 6.3 show some examples of predicted trajectories (in red) compared with the ground truth (in blue), over a 5 second prediction window.

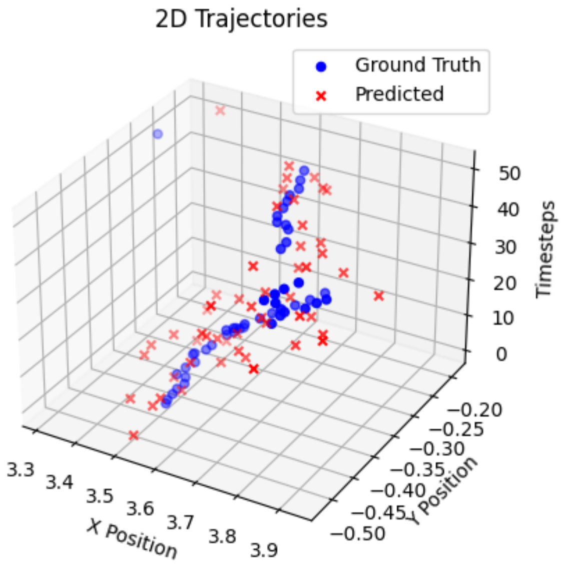


Figure 6.1: An example of an acceptable predicted trajectory compared to the ground truth.

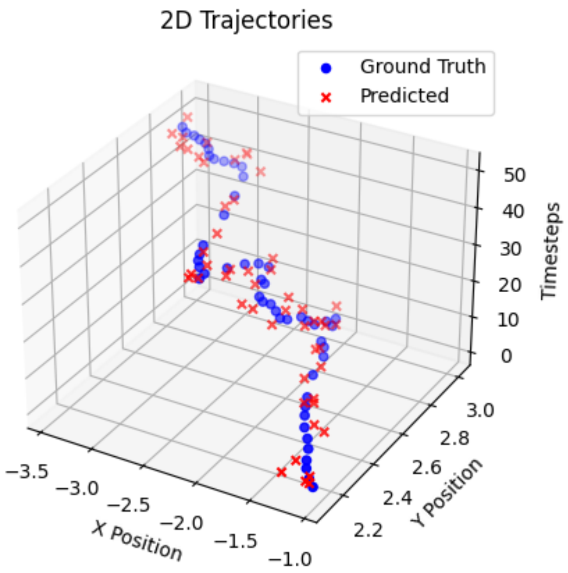


Figure 6.2: An example of a good predicted trajectory compared to the ground truth.

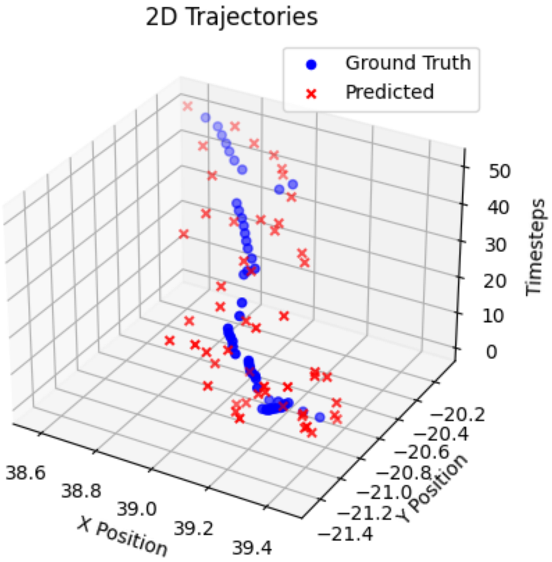


Figure 6.3: An example of a bad predicted trajectory compared to the ground truth.

6.3 Testing on the TIAGo robot

The final part of the evaluation phase consists in testing this system on a real robot moving in the real world. As discussed in chapter 3, the chosen robot is a TIAGo (in particular, a TIAGo++) situated in the Autonomous Robotics Laboratory of the University of Padova. For our purpose, we used some existing packages as well: the SPENCER people tracking package [31] for detecting, tracking and obtaining positions of people in the environment, and the CoHAN planner [49], which is a state-of-the-art planner based on the Human-Aware Timed Elastic Band [53].

The experiments were performed as follows (see Figure 6.4):

- A number of white cardboards are placed on the floor, so that the chair and table legs are hidden and the laser does not detect possible human legs in those positions;
- The robot is placed in the middle of the robotics laboratory entrance hallway, at an even distance from the side walls. The SPENCER people tracker package is launched;
- The CoHAN planner is started, and a goal placed around the first right corner and towards the end of the room is given to the robot;
- The predictor node is started: it starts collecting positions of people and feeding them to the learning model for prediction;
- Another node publishes both the tracked persons and the predictions on the tracked_humans topic used by CoHAN for planning.
- TIAGo then starts moving to reach its goal, while a person moves towards it at the same time;

The predictor node is the main node of this work: it uses the context-based model we propose and collects the positions of the people detected in the environment. Once it has enough data for the desired time window for prediction, it feeds this data to the model. The output predicted trajectories are then processed into a message of type TrackedHumans that is passed to the CoHAN planner. The planner will then use this message to plan a path.

The goal of these experiments is to prove that, during navigation, TIAGo adapts its path to the people it tracks and their possible future positions and behaves respecting social navigation constraints. Despite eventually encountering some issues (e.g. the robot delocalizing in some cases and turning on itself, responding slowly to the person in front of it or behaving oscillatorily), the robot

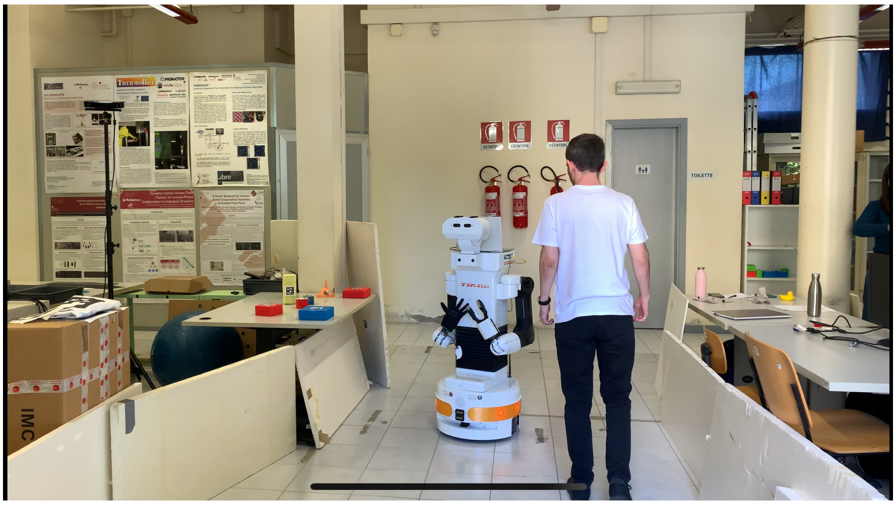
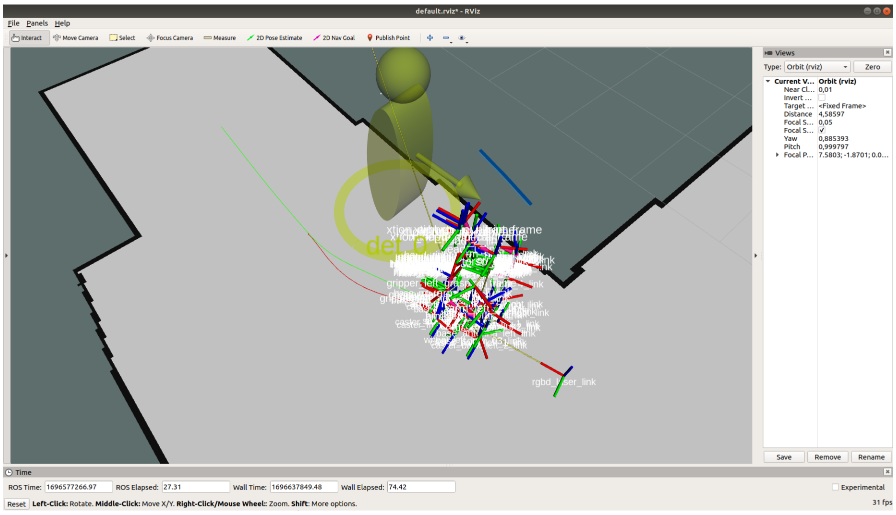


Figure 6.4: This figure shows the testing environment we used: the robot is navigating towards the author and avoiding him as soon it detects him. Notice the white cardboards hiding the chairs and the table legs.

properly handled the social space during navigation with human presence in the environment. Figure 6.5 shows how the CoHAN local planned trajectory (red line) changed with respect to the global plan (green line) when the robot detected a person moving in its way: the planner modifies its planned path to give space to the person and avoid them during motion.



Type: Orbit (rviz)

Current V...	Orbit (rviz)
Near Cl...	0,01
Invert ...	□
Target ...	<Fixed Frame>
Distance	4,58597
Focal S...	0,05
Focal S...	☑
Yaw	0,885393
Pitch	0,999797
Focal P...	7.5803; -1.8701; 0.0...

ROS Time: 1696577266.97 ROS Elapsed: 27.31 Wall Time: 1696637849.48 Wall Elapsed: 74.42

□ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options.

31 fps

Figure 6.5: This figure shows the robot (depicted as the structure forming from its frames) moving towards the detected person (in yellow), and changing the planned local path (in red) to avoid him and respect social norms.

7 Conclusion

The goal of this thesis work was to design and test a new trajectory prediction method that exploits the information embedded into the context of the environment, in particular into the positions, orientations and motion of people with respect to each other. Trajectory prediction is a very complex task. Human behaviour is very diverse among different people: it is highly dynamic, it can change drastically and rapidly, and it is influenced not only by the current state but also by past and future motion and intentions. Moreover, humans rarely move alone, but are in constant interactions with each other, when avoiding, communicating or grouping with other people. Beyond such challenges, to make the robot socially navigate and interact with people, the estimation of the human motion has to be done in real-time. To achieve this, a clustering approach was applied alongside classical trajectory learning. The results suggested that this approach proved to be up to 11% better than a context-free model (see Chapter 6, in particular Table 6.1), which is a good starting point for a possible future evolution of this concept to improve its efficacy. However, some complex state-of-the-art approaches obtain outstanding performance, almost precise to the centimeter. The deployment on the TIAGo++ robot was done at the Autonomous Robotics Laboratory of the University of Padova: the robot was given a goal and had to socially navigate while avoiding a person walking in its direction. This preliminary real-world test proved that the robot is able to conform to social rules in real-time and it is able to correctly adapt the planned path to avoid people, despite some challenges related to localization difficulties and oscillatory behaviours.

7.1 Future works

Future work will be focused on improving the performance by adding data from other datasets and assessing other networks. One possible change to make, is to train the model on more or different datasets, and see how it performs. For example, in this work the ATC dataset has been used for testing purposes only, but its large quantity of data could be exploited better in training.
The next step could be integrating this context-based approach with the very powerful meta-path enhanced methods [14] that perform so well on ETH and UCY. Another possible idea consists of integrating information from other people in the environment (e.g. their distances with respect to a chosen human) to improve the prediction accuracy of the person we are analyzing. Finally, a possibility is to use deeper and more complex models. However, in order to be deployable on the robot, it is necessary to carefully evaluate how much more it can handle.

In conclusion, the future of trajectory prediction for social navigation holds great promise. Addressing the complexities and ethical considerations while embracing emerging technologies will enable us to create autonomous systems that coexist seamlessly with humans in an increasingly dynamic and diverse world.

References

[1] Hyemin Ahn, Esteve Valls Mascaro, and Dongheui Lee. "Can We Use Diffusion Probabilistic Models for 3D Motion Prediction?" In: *arXiv preprint arXiv:2302.14503* (2023).

[2] Alexandre Alahi et al. "Social lstm: Human trajectory prediction in crowded spaces". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 961–971.

[3] Alberto Bacchin, Gloria Beraldo, and Emanuele Menegatti. "Learning to plan people-aware trajectories for robot navigation: A genetic algorithm". In: *2021 European Conference on Mobile Robots (ECMR)*. 2021, pp. 1–6. DOI: 10.1109/ECMR50962.2021.9568804.

[4] Yutong Ban et al. "A deep concept graph network for interaction-aware trajectory prediction". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 8992–8998.

[5] Gloria Beraldo et al. "Shared Autonomy for Telepresence Robots Based on People-Aware Navigation". In: *Intelligent Autonomous Systems 16: Proceedings of the 16th International Conference IAS-16*. Springer. 2022, pp. 109–122.

[6] Niccoló Bisagno, Bo Zhang, and Nicola Conci. "Group lstm: Group trajectory prediction in crowded scenarios". In: *Proceedings of the European conference on computer vision (ECCV) workshops*. 2018, pp. 0–0.

[7] Yingfeng Cai et al. "Pedestrian Motion Trajectory Prediction in Intelligent Driving from Far Shot First-Person Perspective Video". In: *IEEE Transactions on Intelligent Transportation Systems* PP (Jan. 2021), pp. 1–16. DOI: 10.1109/TITS.2021.3052908.

[8] Zhangjie Cao et al. "Leveraging Smooth Attention Prior for Multi-Agent Trajectory Prediction". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 10723–10730.

[9] Yiu Ming Chung, Hazem Youssef, and Moritz Roidl. "Distributed Timed Elastic Band (DTEB) Planner: Trajectory Sharing and Collision Prediction for Multi-Robot Systems". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 10702–10708.

[10] Anthony Favier, Phani-Teja Singamaneni, and Rachid Alami. "Challenging Human-Aware Robot Navigation with an Intelligent Human Simulation System". working paper or preprint. June 2022. URL: `https://hal.laas.fr/hal-03684245`.

[11] Óscar Gil and Alberto Sanfeliu. *Robot Navigation Anticipative Strategies in Deep Reinforcement Motion Planning*. 2022. arXiv: `2210.08280 [cs.RO]`.

[12] Agrim Gupta et al. *Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks*. 2018. arXiv: `1803.10892 [cs.CV]`.

[13] Edward T. Hall. "A System for the Notation of Proxemic Behavior". In: *American Anthropologist* 65.5 (1963), pp. 1003–1026. ISSN: 00027294, 15481433. URL: `http://www.jstor.org/stable/668580` (visited on 04/17/2023).

[14] Aamir Hasan, Pranav Sriram, and Katherine Driggs-Campbell. "Metapath Analysis on Spatio-Temporal Graphs for Pedestrian Trajectory Prediction". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 617–624.

[15] Shyamanta Hazarika et al. "A Qualitative Trajectory Calculus to Reason about Moving Point Objects". In: Jan. 2012, pp. 147–167. ISBN: 9781616928704. DOI: `10.4018/978-1-61692-868-1.ch004`.

[16] Van Bay Hoang et al. "A Time-Dependent Motion Planning System for Mobile Service Robots in Dynamic Social Environments". In: *2021 International Conference on System Science and Engineering (ICSSE)*. 2021, pp. 464–469. DOI: `10.1109/ICSSE52999.2021.9538489`.

[17] Yu Hua et al. "Towards Efficient 3D Human Motion Prediction using Deformable Transformer-based Adversarial Network". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 861–867.

[18] Xin Huang et al. "HYPER: Learned hybrid trajectory prediction via factored inference and adaptive sampling". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2906–2912.

[19] Zhe Huang et al. "Long-Term Pedestrian Trajectory Prediction Using Mutable Intention Filter and Warp LSTM". In: *IEEE Robotics and Automation Letters* 6.2 (Apr. 2021), pp. 542–549. DOI: 10.1109/lra.2020.3047731. URL: https://doi.org/10.11092Flra.2020.3047731.

[20] Zhuozhu Jian et al. "Dynamic Control Barrier Function-based Model Predictive Control to Safety-Critical Obstacle-Avoidance of Mobile Robot". In: *arXiv preprint arXiv:2209.08539* (2022).

[21] Bhagyashri Abhay Kelkar, Sunil F. Rodd, and Umakant P. Kulkarni. "Estimating distance threshold for greedy subspace clustering". In: *Expert Systems with Applications* 135 (2019), pp. 219–236. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2019.06.011. URL: https://www.sciencedirect.com/science/article/pii/S0957417419304117.

[22] Harmish Khambhaita and Rachid Alami. "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)*. 2017, pp. 1140–1145. DOI: 10.1109/ROMAN.2017.8172447.

[23] Sultan Daud Khan and Saleh Basalamah. "Sparse to dense scale prediction for crowd couting in high density crowds". In: *Arabian Journal for Science and Engineering* 46.4 (2021), pp. 3051–3065.

[24] Jinkyu Kim et al. "Stopnet: Scalable trajectory and occupancy prediction for urban autonomous driving". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 8957–8963.

[25] Thibault Kruse et al. "Human-aware robot navigation: A survey". In: *Robotics and Autonomous Systems* 61.12 (2013), pp. 1726–1743. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2013.05.007. URL: https://www.sciencedirect.com/science/article/pii/S0921889013001048.

[26] Yen-Ling Kuo et al. "Trajectory prediction with linguistic representations". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2868–2875.

[27] Javier Laplaza, Francesc Moreno-Noguer, and Alberto Sanfeliu. "Context and Intention aware 3D Human Body Motion Prediction using an Attention Deep Learning model in Handover Tasks". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 4743–4748.

[28] Javier Laplaza, Francesc Moreno-Noguer, and Alberto Sanfeliu. "Context Attention: Human Motion Prediction Using Context Information and Deep Learning Attention Models". In: *ROBOT2022: Fifth Iberian Robotics Conference: Advances in Robotics, Volume 1*. Springer. 2022, pp. 102–112.

[29] Jessica Leu et al. "Autonomous Vehicle Parking in Dynamic Environments: An Integrated System with Prediction and Motion Planning". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 10890–10897.

[30] Jiachen Li et al. "Pedestrian Crossing Action Recognition and Trajectory Prediction with 3D Human Keypoints". In: *arXiv preprint arXiv:2306.01075* (2023).

[31] Timm Linder et al. "On multi-modal people tracking from mobile platforms in very crowded and dynamic environments". In: May 2016, pp. 5512–5519. DOI: 10.1109/ICRA.2016.7487766.

[32] Hejing Ling, Guoliang Liu, and Guohui Tian. "Motion planning combines psychological safety and motion prediction for a sense motive robot". In: *arXiv preprint arXiv:2010.11671* (2020).

[33] Matteo Lisotto, Pasquale Coscia, and Lamberto Ballan. "Social and scene-aware trajectory prediction in crowded spaces". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*. 2019, pp. 0–0.

[34] Qiujing Lu et al. "KEMP: Keyframe-Based Hierarchical End-to-End Deep Model for Long-Term Trajectory Prediction". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 646–652.

[35] Nils Mandischer, Frederik Schicks, and Burkhard Corves. "Situational Adaptive Motion Prediction for Firefighting Squads in Indoor Search and Rescue". In: *arXiv preprint arXiv:2306.02705* (2023).

[36] Diego Martinez-Baselga, Luis Riazuelo, and Luis Montano. "Improving robot navigation in crowded environments using intrinsic rewards". In: *arXiv preprint arXiv:2302.06554* (2023).

[37] Christoforos Mavrogiannis et al. *Core Challenges of Social Robot Navigation: A Survey*. 2021. arXiv: 2103.05668 [cs.RO].

[38] Rowan McAllister et al. "Control-aware prediction objectives for autonomous driving". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 01–08.

[39] Sariah Mghames et al. "A Neuro-Symbolic Approach for Enhanced Human Motion Prediction". In: *2023 IEEE International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2023.

[40] Huan Nguyen et al. "Motion primitives-based navigation planning using deep collision prediction". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 9660–9667.

[41] Ngoc Anh Pham, Lan Anh Nguyen, and Xuan Tung Truong. "Socially aware robot navigation framework: Social activities recognition using deep learning techniques". In: *2021 8th NAFOSTED Conference on Information and Computer Science (NICS)*. 2021, pp. 381–385. DOI: 10.1109/NICS54270.2021.9701551.

[42] Aleksey Postnikov, Aleksander Gamayunov, and Gonzalo Ferrer. "Conditioned Human Trajectory Prediction using Iterative Attention Blocks". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 4599–4604.

[43] Yao Qin et al. *A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction*. 2017. arXiv: 1704.02971 [cs.LG].

[44] Ely Repiso, Anais Garrell, and A. Sanfeliu. "Adaptive Social Planner to Accompany People in Real-Life Dynamic Environments". In: *International Journal of Social Robotics* (Dec. 2022). DOI: 10.1007/s12369-022-00937-3.

[45] Christoph Rösmann, Frank Hoffmann, and Torsten Bertram. "Timed-Elastic-Bands for time-optimal point-to-point nonlinear model predictive control". In: *2015 European Control Conference (ECC)*. 2015, pp. 3352–3357. DOI: 10.1109/ECC.2015.7331052.

[46] Saeed Saadatnejad et al. "A generic diffusion-based approach for 3D human pose prediction in the wild". In: *arXiv preprint arXiv:2210.05669* (2022).

[47] Julian Schmidt et al. "Exploring Navigation Maps for Learning-Based Motion Prediction". In: *arXiv preprint arXiv:2302.06195* (2023).

[48] Yutaka Shimizu et al. "Moment-based Kalman Filter: Nonlinear Kalman Filtering with Exact Moment Propagation". In: *arXiv preprint arXiv:2301.09130* (2023).

[49] Phani Teja Singamaneni, Anthony Favier, and Rachid Alami. "Human-Aware Navigation Planner for Diverse Human-Robot Ineraction Contexts". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021.

[50] Phani Teja Singamaneni, Anthony Favier, and Rachid Alami. *Watch out! There may be a Human. Addressing Invisible Humans in Social Navigation*. 2022. arXiv: `2211.12216 [cs.RO]`.

[51] Zayne Sprague et al. "SOCIALGYM 2.0: Simulator for Multi-Agent Social Robot Navigation in Shared Human Spaces". In: *arXiv preprint arXiv:2303.05584* (2023).

[52] Zhaoxin Su et al. "Crossmodal transformer based generative framework for pedestrian trajectory prediction". In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 2337–2343.

[53] Phani Teja S. and Rachid Alami. "HATEB-2: Reactive Planning and Decision making in Human-Robot Co-navigation". In: *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. 2020, pp. 179–186. DOI: `10.1109/RO-MAN47096.2020.9223463`.

[54] Sebastian Thrun et al. "MINERVA: A second-generation museum tour-guide robot". In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*. Vol. 3. IEEE. 1999.

[55] Xuan-Tung Truong and Trung Dung Ngo. "Toward Socially Aware Robot Navigation in Dynamic and Crowded Environments: A Proactive Social Motion Model". In: *IEEE Transactions on Automation Science and Engineering* 14.4 (2017), pp. 1743–1760. DOI: `10.1109/TASE.2017.2731371`.

[56] Nico Van de Weghe and Philippe De Maeyer. "Conceptual Neighbourhood Diagrams for Representing Moving Objects". In: Oct. 2005, pp. 228–238. ISBN: 978-3-540-29395-8. DOI: `10.1007/11568346_25`.

[57] Balakrishnan Varadarajan et al. “Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 7814–7821.

[58] Jennifer Wakulicz et al. “Topological Trajectory Prediction with Homotopy Classes”. In: *arXiv preprint arXiv:2301.09821* (2023).

[59] Allan Wang, Christoforos Mavrogiannis, and Aaron Steinfeld. *Group-based Motion Prediction for Navigation in Crowded Environments*. 2022. arXiv: 2107.11637 [cs.RO].

[60] Weizhuo Wang et al. “Trajectory and Sway Prediction Towards Fall Prevention”. In: *arXiv preprint arXiv:2209.11886* (2022).

[61] Arjun Rajeev Warrier et al. “Implementation of Classical Path Planning Algorithms for Mobile Robot Navigation: A Comprehensive Comparison”. In: *2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*. 2022, pp. 1–6. DOI: 10.1109/ICECCME55909.2022.9988092.

[62] Zizhang Wu et al. “Mvfusion: Multi-view 3d object detection with semantic-aligned radar and camera fusion”. In: *arXiv preprint arXiv:2302.10511* (2023).

[63] Jiaolong Xu et al. “Trajectory Prediction for Autonomous Driving with Topometric Map”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 8403–8408.

[64] Jie Xu et al. “A Continuous Learning Approach for Probabilistic Human Motion Prediction”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 11222–11228.

[65] Luyao Ye, Zikang Zhou, and Jianping Wang. “Improving the Generalizability of Trajectory Prediction Models with Frenet-Based Domain Normalization”. In: *arXiv preprint arXiv:2305.17965* (2023).

[66] Jing Zhang et al. “Research on Effective Path Planning Algorithm Based on Improved A* Algorithm”. In: *Journal of Physics: Conference Series* 2188.1 (Feb. 2022), p. 012014. DOI: 10.1088/1742-6596/2188/1/012014. URL: https://dx.doi.org/10.1088/1742-6596/2188/1/012014.

[67] Zhejun Zhang et al. “TrafficBots: Towards World Models for Autonomous Driving Simulation and Motion Prediction”. In: *arXiv preprint arXiv:2303.04116* (2023).

[68] Kai Zhu and Tao Zhang. “Deep reinforcement learning based mobile robot navigation: A review”. In: *Tsinghua Science and Technology* 26.5 (2021), pp. 674–691. DOI: 10.26599/TST.2021.9010012.

[69] Yufei Zhu et al. “A Data-Efficient Approach for Long-Term Human Motion Prediction Using Maps of Dynamics”. In: *arXiv preprint arXiv:2306.03617* (2023).

Acknowledgments

I would like to express my gratitude to everyone that has supported me throughout this journey.

To my co-supervisor Prof. Gloria Beraldo and my supervisor Prof. Nicola Bellotto for their support and time spent on following and guiding my work during these months.

To my family, for the continuous support, for allowing me to do whatever I desired and for giving me the time and space necessary to achieve my goals.

Finally, to my colleagues and friends, for the years spent together, in both hard and fun times.