UNIVERSITY OF OULU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Aayush Kafle

HUMAN AWARE ROBOT NAVIGATION

Master's Thesis
Degree Programme in Computer Science and Engineering
June 2022

Kafle A. (2022) Human Aware Robot Navigation. University of Oulu, Degree Programme in Computer Science and Engineering, 58 p.

ABSTRACT

Human aware robot navigation refers to the navigation of a robot in an environment shared with humans in such a way that the humans should feel comfortable, and natural with the presence of the robot. On top of that, the robot navigation should comply with the social norms of the environment. The robot can interact with humans in the environment, such as avoiding them, approaching them, or following them. In this thesis, we specifically focus on the approach behavior of the robot, keeping the other use cases still in mind. Studying and analyzing how humans move around other humans gives us the idea about the kind of navigation behaviors that we expect the robots to exhibit. Most of the previous research does not focus much on understanding such behavioral aspects while approaching people. On top of that, a straightforward mathematical modeling of complex human behaviors is very difficult. So, in this thesis, we proposed an Inverse Reinforcement Learning (IRL) framework based on Guided Cost Learning (GCL) to learn these behaviors from demonstration. After analyzing the CongreG8 dataset, we found that the incoming human tends to make an O-space (circle) with the rest of the group. Also, the approaching velocity slows down when the approaching human gets closer to the group. We utilized these findings in our framework that can learn the optimal reward and policy from the example demonstrations and imitate similar human motion.

Keywords: Approaching Humans, Guided Cost Learning, Inverse Reinforcement Learning, O-Space, Optimization

TABLE OF CONTENTS

FOREWORD

This thesis is written as the requirement for the Master's in Computer Science and Engineering degree program at the University of Oulu, Oulu, Finland. This thesis is part of the work that I did during my time as a research assistant at the Perception Engineering (PE) group of the Ubiquitous Computing (UBICOMP) department at the University of Oulu. This thesis provides a foundation for navigating telepresence robots while approaching humans.

First, I want to express my warmest gratitude to my thesis supervisor Dr. Markku Suomalainen who guided and supported me throughout this thesis. I would also like to thank my technical supervisors Basak Sakcak, and Kshitij Tiwari for continuous comments and feedback on the thesis. A special mention to my wife Richa Adhikari and my family members for keeping me motivated during this process. I would also like to thank my friends and colleagues who were there to listen to my complaints. I am grateful to the UBICOMP department and PE group, who gave me the opportunity, funding, and resources during my thesis.

And lastly, I am grateful to the ITEE, and the University of Oulu family for making my journey memorable.

Oulu, June 23rd, 2022

Aayush Kafle

LIST OF ABBREVIATIONS AND SYMBOLS

2D	2 Dimensional
3D	3 Dimensional
AS-EKF	Augmented State-Extended Kalman Filter
CNN	Convolution Neural Network
CSV	Comma Separated Value
D3QN	Deep Q Learning Network
DS-RNN	Decentralized Structural -Recurrent Neural Network
DNN	Deep Neural Network
DWA	Dynamic Window Approach
EEG	ElectroEncephaloGram
EM	Expectation Maximization
F-Formation	Facing-Formation
GCL	Guided Cost Learning
HAN	Human Aware Navigation
HMD	Head Mounted Display
HRI	Human Robot Interaction
ICS	Inevitable Collision States
IRL	Inverse Reinforcement Learning
MDP	Markov Decision Process
MLE	Maximum Log-likelihood Estimation
PDA	Preliminary Data Analysis
QTC	Qualitative Trajectory Calculus
RGB-D	Red Green Blue -Depth
RL	Reinforcement Learning
ROS	Robot Operating System
RRT	Random Recurrent Tree
PaCcET	Pareto Concavity Elimination Transformation
PDA	Preliminary Data Analysis
PGM	Probabilistic Graphical Method
VR	Virtual Reality
SFM	Social Force Model
SVM	Support Vector Machines
m	meter
fps	frames per second
s	second
$^\circ$	degree
Hz	Hertz
A	set of actions
a	action
a_0	initial action
a_t	action at time t
c	intermediate complex number

$D(\pi, \pi^*)$	divergence between π and π^*
$f(s, a)$	feature functions for s and a
$H()$	entropy
m	maximum margin
O_t	optimization variable at time t
p_1	player 1
p_2	player 2
p_3	player 3
p_4	player 4
$P_a(s, s')$	probability of going to s' from s by executing a
$Q_\pi(s, a)$	action-value function for s and a
$Q^*(s, a)$	optimal action-value function for s and a
R_ψ	reward for parameter ψ
r_c	radius of circle
S	set of states
s	state
s'	next state
s_0	initial state
s_t	state at time t
t	time
$V_\pi(s)$	value function for s
$V^*(s)$	optimal value function for s
w	intermediate complex number
(x, y)	horizontal and vertical positions in world coordinate
(x_1, y_1)	position of player 1
(x_2, y_2)	position of player 2
(x_3, y_3)	position of player 3
(x_4, y_4)	position of player 4
(x_c, y_c)	center of circle
(x_{grid}, y_{grid})	horizontal and vertical positions in grid coordinate
(x_{max}, y_{max})	maximum x and y value in world coordinate
(x_{min}, y_{min})	minimum x and y value in world coordinate
(x_{res}, y_{res})	x and y resolution in grid coordinate
Z	parity function
z_1	complex number $x_2 + iy_2$
z_2	complex number $x_3 + iy_3$
z_3	complex number $x_4 + iy_4$
β	policy parameter
θ	orientation in world coordinate
θ_{grid}	orientation in grid coordinate
θ_{max}	maximum orientation in world coordinate
θ_{min}	minimum orientation in world coordinate
θ_{res}	orientation resolution in grid coordinate
∞	infinity
γ	discount factor
ψ	reward parameter

Π	set of policies
π	policy
π^*	optimal policy
ΔL_ψ	MLE gradient of ψ
$\delta\pi_\psi$	loss for reward network
$\delta\pi_\beta$	loss for policy network
τ	trajectory
τ_{traj}	set of generated trajectories
τ_{samp}	set of both generated and demonstrated trajectories
τ_{demo}	set of demonstrated trajectories
$\tau_{\hat{traj}}$	sampled set of generated trajectories
$\tau_{\hat{demo}}$	sampled set of demonstrated trajectories
$E[]$	expected value
$exp()$	exponential function
$log()$	logarithmic function
$max()$	maximize
$min()$	minimize
$p()$	probability
$p(\|)$	conditional probability
^{T}	transpose
$tan()$	tangent
$tan^{-1}()$	arc$tan()$
$\|\|z\|\|$	L2 norm of z
$\in$	belongs to
$\sim$	sample
$\sum()$	sum
$\prod()$	product

1. INTRODUCTION

Remote working has been normal since the COVID-19 pandemic, and the practice tends to grow in the future [1]. Most professional, academic, or personal interactions, such as meetings, seminars, lectures, or even musical shows, were conducted through video calling technologies. The popularity boom of remote meeting software such as Zoom, Teams, Slack, etc. [2] shows that remote interactions and communications were the key factors that helped maintain similar productivity during such a difficult time. Nevertheless, the interaction over present video calling/conferencing technologies does not feel real and natural and limits the mobility as in the actual scenario.

Telepresence provides a sensation to the user of a different physical location from their actual location with the help of audio and visual devices such as cameras and screens. In other words, the user experiences teleportation with an enhanced sense of connectedness. Telepresence, in general, is a vague concept, and one can argue that a video calling app is a telepresence technology. It is valid to some extent, but the important thing we are missing is that telepresence technology should be able to give the experience of virtually being in another environment. Virtual Reality (VR) technology, using Head-Mounted Displays (HMD), can achieve the objective to a large extent. It uses two lenses for eyes, headphones/speakers for ears, and body tracking sensors to track the movement of the head or other body parts. The system tries to deceive human perception by showing and playing the information about a different environment A telepresence robot with integrated telepresence technology can be the intersection between the virtual and real environment of such interactions. Their application spans domains of industries and more unstructured environments such as offices, public spaces, universities, hospitals, and houses.

Humans are considered dynamic obstacles for robots in most research. However, human behavior is often not considered for making predictions based on movement. Those that model the behavior of humans in the environment, do so only to avoid collision with them. There is little research on how to approach to have an interaction rather than avoid humans in an environment. Also, there is almost no such research that focuses on telepresence robots. Although there are state-of-the-art robots with a high degree of autonomy for navigation in deployed domains [3], due to the unpredictable nature of the environment and humans, efficient interaction during navigation is challenging to accomplish. Sequential methods [4], and mathematical modeling [5] have been used to approach people by understanding human behaviors. The problem is that these methods fail to capture the dynamics of the environment and cannot be scaled.

Since humans intrinsically follow these basic characteristics, we can study human behaviors for different navigation scenarios including how to approach people, and implement these behaviors into the robot. Yet, things are not that easy because it's hard to understand and generalize human behaviors. Previous methods have tried using sequential methods [4], and mathematical modeling [5] to approach people. The problem here is that these methods fail to capture the dynamics of the environment, and cannot be scaled. Even though human behaviors are difficult to model as a mathematical function, we can assume that humans subconsciously optimize some quantity while making rational decisions. So if we model such quantity as a function of learnable parameters, we can learn the parameters using learning algorithms like

Inverse Reinforcement Learning (IRL) [6] by looking at the examples of humans behaving in such scenarios. Then, we can use these learned parameters to replicate these behaviors in the robot.

In IRL, the quantity that we want to learn the parameters of and optimize is called reward. Similarly, the actions that lead to those rewards are behaviors or, in technical terms, policy. We have to model our environment into states in such a way that there are actions for different states and there are rewards for those actions. Also, the states change according to the actions. IRL is different from Reinforcement Learning (RL) because, in RL, we have to learn an optimal policy when the reward is given. Whereas, in IRL, we have to estimate the reward from the expert demonstration examples. But one issue of IRL is that, at each optimization step, we have to solve a full RL problem to find the optimal policy for the new reward. The guided Cost Learning (GCL) algorithm handles the problem pretty well by using Deep Neural Networks (DNN) to approximate rewards and policy [7].

We take the trajectory examples from the CongreG8 dataset [8] because they have the trajectories of the same scenario that we are trying to study, i.e., approaching and joining a group. First, we preprocess the dataset and convert it into the format that can be used in the IRL framework. During this process, we analyze the trajectories and try to understand the properties that can be helpful while using them in the IRL framework.

We want robots to understand the environment better so that they can be seamlessly integrated into our daily lives. There is also a big gap in research on behavioral expectations and execution when it comes to robot navigation around humans. Although the research in HAN is ramping up some speed, there is still so much work to be done to better understand the scenario, and its properties and integrate it with telepresence technologies. In this thesis, I propose a learning framework that can learn the behavior of a specific scenario, i.e., while joining a group, using already present examples. Also, I analyze a dataset to find the properties that can be applied to make some assumptions for the framework. The framework helps us to learn the rewards associated with the trajectories of this specific scenario, and generate a path that has similar properties as that of examples.

The main objectives of this thesis are to analyze a dataset for the approaching scenario of HAN, establish the mathematical foundation for an IRL framework, and structure the dataset into the input format required by the framework to learn the reward functions and policy associated with the example trajectories.

1.1. Structure of Thesis

The Chapter 1 of this thesis introduces the thesis, motivation for the thesis, and provides the objectives of this research. The Chapter 2 consists of the relevant background concept and mentions the past works in HAN, telepresence, learning from demonstration, and available datasets. The Chapter 3 has a detailed description of the proposed framework. This section describes the dataset, data preprocessing methods, and algorithms. Similarly, the mathematical formulation of the problem and the approach to solving the problem are also explained here. The results and illustration obtained in the various stages of our study are presented in Chapter 4. The discussion and analysis of the obtained results and future research ideas are also done in this chapter. Finally, the Chapter 5 consists of the conclusion of the thesis.

2. BACKGROUND AND RELATED WORKS

This section describes the background concepts and technical terms of HAN and telepresence. It also mentions the relevant past works for HAN, learning by demonstration, and datasets available.

2.1. Human Aware Navigation

Human Aware Navigation is the intersection between HRI and robot motion planning [9]. HRI involves the study of interaction dynamics between humans and robots [10]. Robot motion planning is to plan and execute the navigation of the robot from a start position to a goal position.

As the research in robotics navigation is increasing, HAN is also becoming a very broad topic. Different aspects of HAN include criteria for robot navigation, additional criteria for coexisting with humans, and different navigation scenarios. The minimum criterion to satisfy in robot navigation is to avoid collisions by using robust navigation strategies. Additional criteria are to enable the smooth coexistence of robots and humans in the same environment is to make humans feel comfortable, natural, and social. In a normal case, the humans around the robots should be able to acknowledge and accept the existence of a robot. In the telepresence case, the humans around the robot and the humans virtually inside the robot should feel comfortable, natural, and social. In a broad sense, the scope of HAN is to provide a general navigation framework that fits the essential criteria [9].

Scientists and engineers have successfully deployed autonomous robots in the same environment with humans [11] such as: Rhino [12], Robox [13–15], Minerva [16], Rackham [17], Mobot [18], and Cice [19]. These robots had robust navigation modules that would avoid humans or other obstacles and navigate through an environment. The strategies used were stop-and-wait, i.e., wait until the path was cleared or a minimal required obstacle distance, which is a minimum distance the robot should maintain when it sees an obstacle on its path. These methods mostly focused on avoiding critical situations and failed to address the social aspect of navigation.

When navigating a robot in an environment with humans, the robot might cause discomfort, harm, or surprise to the humans there. For example, a guider-robot might move too slow or fast and might fail to avoid distraction to the person and the person cannot follow it. When moving around a person, failing to maintain appropriate distance might annoy or frighten the humans, depending on the context. Unnecessary long routes, getting in the way of humans, overreacting near obstacles (suddenly stopping or making noise), and approaching from behind are some other examples that could discomfort humans due to robot navigation.

While comfort is a subjective concept, and it’s difficult to fulfill individual criteria, some minimum things can be done to make humans feel comfortable. Humans, in general, feel uncomfortable when the agents they are interacting with are either too close or too far in terms of physical distance. So, the majority of literature for solving the comfort issue in HAN considers the distance between the robot and humans [9]. A virtual space known as “Proxemics” [20], and its derivations, which define interaction distances mutually respected by humans based on context and relationship, are quite popular in HAN. More details about proxemics can be seen in Table 1.

Table 1. Proxemics distance depending on relationship and context [20].

Type	Specification	Details
Intimate	0 – 45 cm	Embracing, Touching, Whispering
Personal	45 – 120 cm	Friends
Social	1.2 – 3.6 m	Acquaintances and Strangers
Public	> 3.6 m	Public Speaking

Comfortable navigation can be ensured by eliminating obvious causes of discomfort. Defining the area around humans as a cost function and potential field are effective solutions, as they balance the goal of navigation and caused discomfort. For e.g., a robot has to pass through a human in a confined space, using cost function or potential field, robots can achieve the goal even by moving close to a human. But if we just define the area around the human as forbidden, then the robot is stuck since there is no room for compromise [21–23]. Another idea could be signaling at an appropriate distance before performing anything that might cause discomfort [24].

Several studies have tried to make robots move similar to humans to make robots acceptable to humans because it makes robots predictable, intuitive, or "natural" [9]. This behavior may also sometimes cause humans some discomfort, the "uncanny valley" [25]. This issue has not caused any impact on robot navigation so far because the issue only about looks [9]. So we can safely say that the benefit of making robots more human-like outweighs the risks.

One of the methods used by researchers to make the robot motion natural is to make the motion smooth. The velocity and geometry of the path can be made smooth by minimizing the jerk of the motion and making the robots move more human-like [26]. For relative motion, sharing the direction [27], and following a formation [28] are some properties exhibited by humans. Approaching from the front [29] rather than from the back can be another approach to make the motion feel more natural.

Additionally, robots are expected to comply with norms of the society. Examples of those norms can be: taking a side to move in narrow corridors, not causing unnecessary discomfort or disturbance, staying in the queue, letting people leave the room before entering through the doors, and so on. Most of the robots are hard-coded to behave in a certain way while following the social norms like not violating the distance norm, taking a side, or circumventing a group of people rather than passing through it [28, 30, 31]. Robots are sometimes made to divert from the shortest path while crossing the roads or standing in a queue [32] to value the social aspect of navigation.

All three criteria of HAN; comfort, naturalness, and sociability are interlinked with each other. Compromising any one of them seems to affect another to some extent. So, balancing these criteria can be a better idea than trying to break them and achieve them separately. In summary, some ideas that can be implemented are: respecting personal zones and affordance space, avoiding erratic or disturbing motion along with culturally scorned behaviors, modulating speed, direction while approaching, and gaze direction [9].

To realize these requirements, a general navigation framework can be established. Kruse et.al. [9] suggested one such framework, which can be divided into deliberate planning and reactive planning group. There is also a third part, sensing and prediction that provides the information about the environment through sensors and prediction algorithms during both deliberating and acting stage. The block diagram is presented in Figure 1

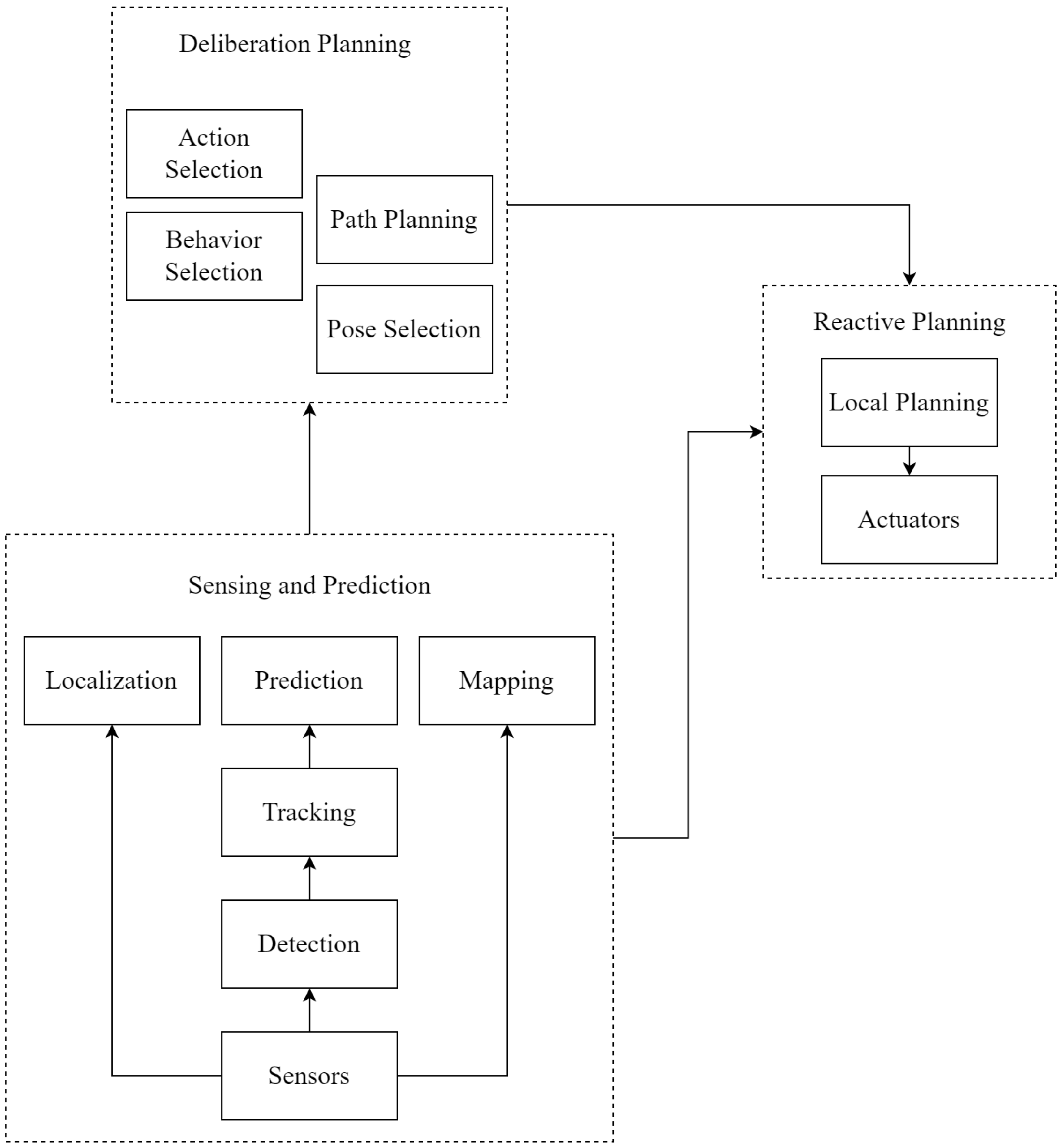


Figure 1. A framework to execute HAN, suggested by [9]

The first part of this framework is the sensing and prediction part. This part helps to produce a geometric representation of the environment by using data from different sensors. Sensing and prediction of the environment play a vital role because of the influence of static and moving obstacles in the navigation process. Prediction of human behavior is difficult because it involves different factors depending on the context and intent of both humans and robots, the limitation of sensors, and also it does not scale very well. Prediction helps in planning steps in such a way that the robot can avoid the obstacles by avoiding being in the position where the obstacle might be in the future. Prediction can be done using geometric reasoning as well as learning using data or by a combination of both. A trajectory is normally straight with constant velocity but slows down while changing direction [33], stochastic grid map of likelihood position [34], a potential field for attraction/repulsion [35], continuous probabilistic models (uncertainty) [36], stochastically maximizing the expectation of random paths to avoid collision (this method uses both prediction and local planning together) [37], using social forces while walking in a formation or group [38], etc. are the popular ideas for prediction using geometric reasoning.

While the generalization of prediction may fail in some special cases, the calculation is also expensive with an increase in information, so it does not scale. There also lies the challenge of modeling uncertainty, which gives rise to the freezing robot problem and requires human cooperation to solve it [3]. Learning, on the other hand, improves over time as it gathers data to adapt in special cases with comparatively low online computation costs. Data library can be used to predict short-term and long-term trajectories using Expectation-Maximization (EM) and converting it into a classification problem [39]. IRL and MDP-based methods are popular nowadays because they have performed quite well for unstructured environments [40]. Most of the learning-based methods at present are coupled with the planning steps [3].

Deliberation focuses on the action, pose selection, path planning, and finally behavior selection. A planner has to make all these decisions to find the optimal solution. Action planning and behavior planning are similar, where the former decides on “what” and “whether”, and the latter on the “how” of the motion planning. Pose selection is the position in which the robot will perform some specific task like stopping or manipulation [9]. Path planning is to find a valid solution from start to goal. Pose selection can be done to comply with our requirements of comfort, naturalness, and social norms. For this part, a dynamic potential field can be created around the goal or obstacle based on context [21].

Path planning yields a set of waypoints that have to be followed by the robot for optimally achieving the navigation goal from the start position to the goal position. Graph-based search in a 2D map (which has the information about the environment, and obstacles) is the most common approach. Graphs that have square grids, arbitrary lattices, or expanding random trees are used in this application [9]. Kinematic constraints are also to be considered as well, as the design of a good cost function is also challenging. A cost function depends on the combination of action and context. If we represent a cost function in a 2D map, it becomes the costmap. This costmap can be utilized by the global planner to construct the plan. The costmap can be composed of different cost functions that capture different scenarios that might occur during the navigation like object padding, occlusion, hidden zones, zones of high or low noise, comfort distance, visibility, interaction regions, pass through left, inertia, crowd density, velocity similarity, etc [9]. Similarly, a time dimension can be added in planning to avoid blocking the path for each robot however, the computation cost is increased exponentially, so the temporal planning is more commonly done in the local planning step [36].

Selection of behaviors can be done to perform interaction tasks such as avoiding and interacting with humans by defining appropriate behaviors for different scenarios. State machines can be used to switch between different behaviors, like approaching, following, patrolling, searching, etc [4, 41]. Potential fields are also a commonly used approach to define such behaviors and modulate the velocity either during approaching or going away or keeping the distance [21, 28]. Behavior can also be modified in the local planner by constraining the velocity in different contexts [42].

The reactive planning part is the main execution of the motion. Safety, obstacle avoidance, and smoothness of the model are handled here. It is also called local planning because it only plans for some distance or time in the future [43]. It defines the motor responses to perceived static or moving obstacles. Hard constraints to avoid the Inevitable Collision States (ICS) are imposed in this stage in addition to some other constraints to make the motion more acceptable for humans [44]. Sampling-based algorithms like Dynamic Window Approach (DWA) [45] and velocity obstacle [46] approaches are the most common local planning approaches.

There can be multiple scenarios under HAN. The scenarios are dependent on the application and categorized accordingly. Our idea is to group them according to the type of interaction that they are having with the humans in terms of navigation. Some interactions between robots with humans are avoiding them, approaching them, following them, staying in a certain formation, and so on.

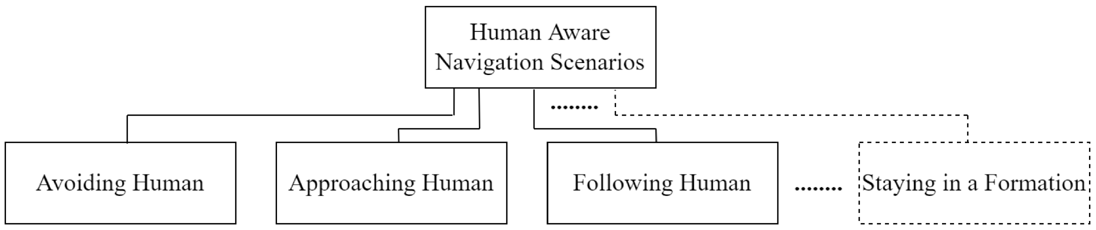


Figure 2. Different scenarios under HAN.

2.1.1. Related Works

This section consists of a brief description of works done in general HAN, followed by the works done in avoiding people and approaching people.

Navigation for human-robot interaction tasks by Althaus et al. presented a state machine-based method to control a robot to behave naturally and human-like while interacting with humans in the environment [28]. The interactions were, entering the room, approaching a group, being part of the group, and leaving the group. This method provided a simplified approach to tackling the problem which was specific to one case so, the generalization and scaling into a different environment is difficult, however, the concept of formation and proxemics can be used in designing other methods. Huttenrauch et al. studied the spatial relationships between robots and humans when they are interacting. The study was more focused on understanding the appropriate movement behaviors in interaction scenarios, which are crucial in non-explicit communication such as signaling of intent [47]. Gomez et al. proposed a mathematical formulation for the general formulation of the social path planning problem. The formulation considers scenarios such as single or several humans, O-space formulation, treating a group as obstacles, and so on [48].

In the thesis by Rios et al., they proposed a risk-based navigation framework for mobile robots [32]. The proposed methods integrated traditional approaches of prediction of moving obstacles with social considerations such as proxemics, formation, activity space, etc. The models attempted to address the estimated discomfort and risk of disturbance on the path caused by robot navigation. However, dynamic adjustment to different scenarios and the consideration of their shape and appearance of them can be addressed in the framework. Bevilacqua et al. maximized human comfort in a Human Aware Navigation scenario by solving an optimal control problem to generate smooth trajectories [49]. The solution consisted of a two-tier approach where the first generated waypoints and the second module would optimize the path to enhance the user comfort. The cost function consisted of several parameters that would represent different dimensions of user comfort.

Social Force Model (SFM) [50] is a popular approach for modeling pedestrian dynamics. In simple terms, this model measures the aggregate virtual force (maybe attraction or repulsion depending on context) exerted on the pedestrian by goals, obstacles, other pedestrians, etc. This concept can be applied to applications such as the formation of groups, group dynamics, and also human-robot navigation behaviors. The system described by Hansen et al. used motion-pattern analysis to estimate the intent of the human and move the robot accordingly. The system also used potential fields around the human obstacle based on the intent estimate, and the robot would move towards the region of the lowest value. This research aimed to make robots more acceptable and natural by respecting the personal distance in different scenarios [21]. Qualitative Trajectory Calculus (QTC) [51] utilizes the relative spatial positioning of the robot and human and encodes it into a state-space representation. QTC combined with distance measure and probabilistic behavior model was applied into a sequential model to study the joint spatial behavior of humans and robots. This approach can be applied to study behaviors such as guiding, approaching, departing, or coordinating in narrow spaces between humans and robots. A computational framework of proxemics-based data-driven probabilistic models of social signaling in human-robot interaction [52] used a concept that features such as physical, psychological, and psycho-physical determine the proxemics behavior during interaction. On top of that, an interaction potential-based trajectory planner and reactive proxemic controller were developed to navigate the robot to conduct social interactions with the human.

In the research by Martinez et al., a multi-robot system architecture was developed to guide a group of humans. The robot followed strategies to localize multi-humans, and create a trajectory to control the group [38]. An architecture for multi-robot communication was also developed for this purpose. The Social Force Model (SFM) and center of gravity concept were the fundamentals for formulating the trajectory to control the group. Lam et al. proposed a navigation algorithm for the harmonious coexistence of humans and robots in the same environment. The navigation algorithm used different rules regarding sensitive zones and yielded a virtual sensitive field that included different scenarios. Then a motion planning algorithm would plan and navigate the robot in that environment. However, inaccuracies in the movement tracking of humans adversely affected the performance of this algorithm [53].

Banisetty et al. introduced a Pareto Concavity Elimination Transformation (PaCcET) based planner which was integrated into Robot Operating System's (ROS) local planner [54]. The approach successfully worked on scenarios such as a hallway, art gallery, queue, and group interaction. The robot performed well in terms of efficiency and cost in scenarios where the robot required prior knowledge on what interaction task to perform.

The review paper by He et al. reviewed several robot motion planning methods, including classical as well as more modern reinforcement learning-based methods. Classical methods have different stages starting from map acquisition, discrete path searching, and trajectory generation to trajectory tracking using local planners. Reinforcement learning-based methods can have the map of the environment and optimize using learning algorithms or can be mapless and have end-to-end architecture. The latter method can also be extended to work on multi-robot planning problems. However, there are still many performance and scalability issues to tackle before deploying these algorithms in real, unstructured environments [55].

HAN: Avoiding Humans

A concept called reciprocal velocity obstacles [56] can be used for autonomous collision and oscillation-free navigation among static and dynamic obstacles. Reciprocal Velocity Obstacles is a modification of velocity obstacles [57] designed to overcome the oscillation problem observed in the original method. This concept works in two-dimension space, and multi-agent scenarios (1000 agents) and also can be extended to work in high-speed obstacles and three-dimensional space. A proactive social motion model [58] used ideas from the extended social force model and hybrid reciprocal velocity obstacle technique to enable a robot to navigate safely and socially in a crowded dynamic environment.

For stabilizing the formation of multiple moving agents, Tanner et al. developed a decentralized controller that worked in a cooperative approach using local navigation functions [59]. The agents used relative orientation and position to form a stable formation and avoid a collision. This was achieved by fine-tuning the parameters of the navigation functions, which are based on the geometry of the space and the degree of interconnection between the agents. The approach was tested in the simulated environment of three and four moving agents. By tackling socially aware robot navigation as a learning problem rather than a traditional model-based problem, better human relative motion behavior can be reproduced [60]. The research took the annotated surveillance data, learned the human relative motion behavior using unsupervised learning to compute a dynamic costmap, and finally plan socially acceptable avoidance maneuvers.

By explicitly modeling the negative examples (collision or discomfort) for socially-aware robot navigation, Liu et al. were able to reduce the collision rates compared to different state-of-the-art methods [61]. The researchers proposed a social contrastive learning method and formulate a loss that represented sufficient information for distinguishing the positive and negative events during navigation. The research by Finean et al. predicted the human motion in a dynamic environment and used trajectory optimization-based motion planning [62]. The human motion prediction was carried out in multiple stages using image processing algorithms, mapping, and intent recognition. The information was integrated into the occupancy gridmap and finally, the trajectory optimization algorithm was used to plan the path. Finally, the verification of the results was done both in simulation and in physical robots.

HAN: Approaching Humans

Researchers developed a multi-stage method to initiate the conversation between robot and human [4]. The proposed model included predicting the walking behavior, choosing the target, planning the approach path, and finally, non-verbally indicating the intention to start the conversation.

Kato et al. designed and analyzed polite approaching behaviors for a customer service robot by replicating the behavior exhibited by a customer and service staff before initiating conversations [63]. First, the data was collected and then categorized into different intention categories using Support Vector Machine (SVM) for estimation. A state transition model was designed for the robot to exhibit one of three behaviors depending on the observed intention of the customer. The researchers found out that the model was effective for initiating a polite conversation however, the modeling is not generalizable to other deployment scenarios or places with different social norms. The robot designed by Repiso et al. was capable of approaching and engaging people [5]. The robot was able to predict the best encounter point to engage itself in the interaction by modifying its pose and orientation. The calculation for the encounter point was done by using the gradient descent method that considered the predictions of human motions. It also used the extended social force model to include the dynamic goal. The verification and validation of the concept in terms of social acceptance were done in simulation as well as in real-life robots for various scenarios.

A proactive approach method for human service robots was proposed in [64]. This method tackled the problem of estimating the pose and orientation of a human, and the risk of unacceptability and discomfort caused by robot motion, by proactively approaching the human in a socially acceptable manner. For doing so, the researchers integrated the motion and interaction modules using behavior trees. A strategy for the robot on how to approach people, not in a friendly way but for admonishment by Mizumaru et al. [65]. First, the researchers observed the human motion for both admonishment and friendly scenarios. Then, they implemented the findings in the robot and verified the effectiveness of a real robot.

2.2. Virtual Reality (VR) and Telepresence

VR technology can be used to enable an immersive telepresence experience during HAN scenarios. Because of the reason that this area is relatively new, the research in this area is quite behind compared to other areas. The comfort and naturalness aspect of the navigation holds in the case of telepresence, but in a slightly different viewpoint, the viewpoint of the operator. The contemporary research also focuses on creating comfortable and intuitive immersive telepresence technologies. On top of that, there are researches creating remotely controllable telepresence robots.

Desai et al. described the essential features of a modern telepresence robot as guidelines. The features include video, audio, user interface, physical features, and autonomous behaviors [66]. Tsui et al. mentioned the iterative design process of a semi-autonomous telepresence robot, which is a follow-up on a previous paper [67,68]. The papers have a detailed explanation of each stage of development, along with

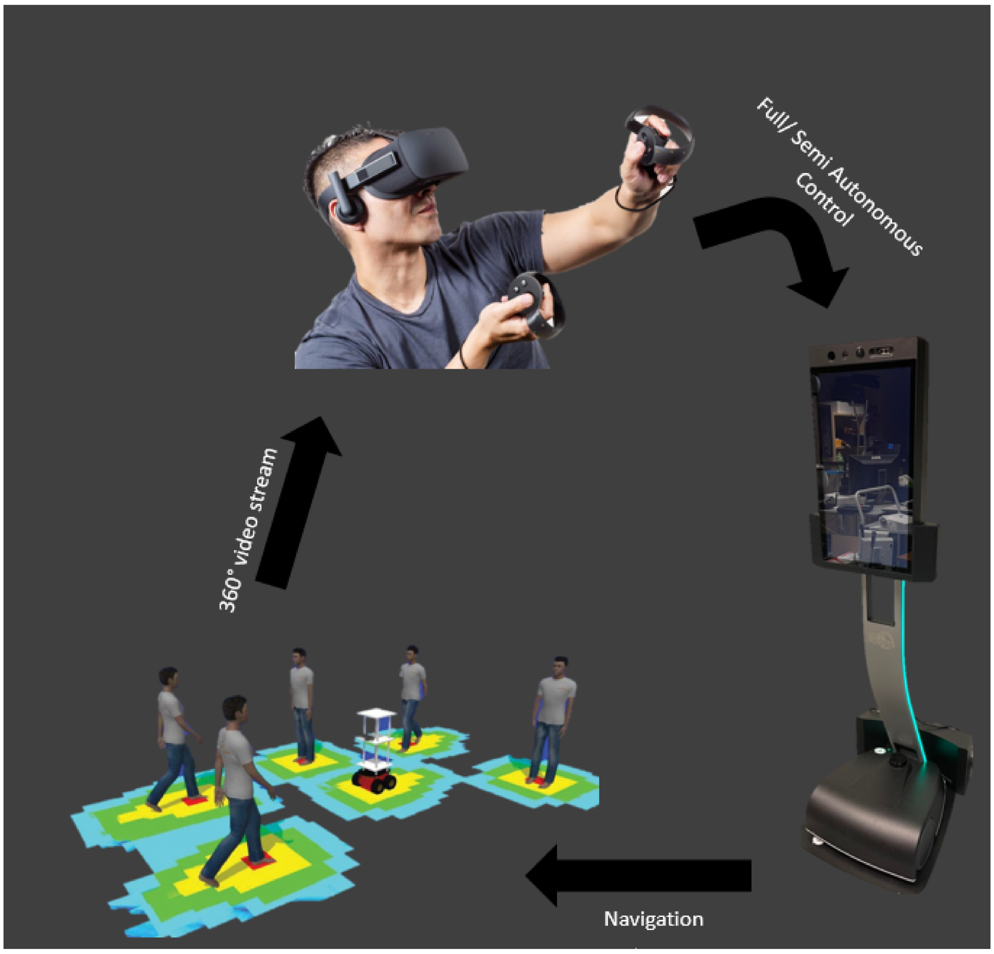


Figure 3. VR and telepresence integration with HAN. Here, the person using a HHMD and controller feels as if he is virtually immersed in the robot body, where he can experience different HAN scenarios.

the requirements and design constraints they encountered during the whole platform development process.

Researchers have also designed telepresence robots that can be controlled by brain signals [69]. The system used ElectroEncephaloGram (EEG) signals from the brain of the operator and fed them to a pattern recognition algorithm. Then, a shared control algorithm was used to drive the telepresence robot (fitted with a camera) in the environment and complete the desired task.

Suguitan et al. demonstrated multiple perspectives of telepresence robot application in real life [70]. The remote-controlled robot had cameras that enabled it to experience the robot in either a first-person perspective or a third-person perspective. To tackle the problem of communication, Augmented State Extended Kalman Filter (AS-EKF) was used to estimate the true position of the telepresence robot, eliminating the uncertainty in localization caused by delay in communication and data processing [71].

The motion sickness in a highly automated transport system was in conflicting nature with the journey time for an automated vehicle [72]. Also, the driving style had a more severe effect on journey time and motion sickness compared to vehicle speed or road width, while the higher vehicle speed resulted in short journey time and relatively lower motion sickness compared to low vehicle speed. The researchers used multi-objective optimization in a predefined road for this research.

In the research by Mimnaugh et al., researchers performed preference, comfort, and naturalness test in smooth path motion and piecewise linear path motion for the immersive telepresence robot. They found out that the participants preferred the path that had one meter per second forward speed and rated the path with the least amount of turns comfortable [73]. In another paper, they concluded that both comfort and naturalness had a strong effect on path preference, although the naturalness measure was subjective to the participants. [74]

Another study focused on autonomously moving telepresence robot motion behavior's effect on the sickness and comfort of the participants [75]. There was a moderate correlation between the user's comfort and the sickness they experienced, but there was no association between sickness with the choice of the comfortable path. Also, the effect of turning speed was not correlated with either comfort or sickness. However, lack of predictability could have been the possible reason for the discomfort. The users also felt more comfortable and less sick when there was an increase in distance between the walls/ obstacles with the camera.

To improve the comfort and reduce the sickness in telepresence experience, Suomalainen et al. proposed an unwinding rotation technique while in motion [76]. Unwinding was done by reversing the default rotation of the camera so that the user has control over the rotation movement while taking turns during motion. It was found that the users preferred the control while moving rather than the default rotation of the camera.

2.3. Learning from Demonstrations

In simple terms, learning from demonstrations is to learn the dynamics of decision-making for the observations and model it mathematically with the help of a set of examples of action for observations carried out by an expert. It helps to understand the rationality of the decision and unfold unknown complex dynamics involved during the process. Furthermore, it enables the learned system to take action for similar observations in the future. A system can learn from demonstration by using algorithms such as imitation learning [77] and IRL [6].

IRL is a subfield of machine learning that uses expert demonstration examples and estimates the rewards associated with those demonstrations. Now, these rewards can be used to derive a policy that explains the behavior of the expert. The Chapter 3 contains more details about IRL.

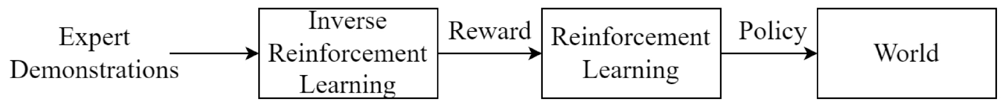


Figure 4. Top level view of learning from demonstration using IRL, here the IRL learns reward which is utilized by RL to derive good policies to act on the world.

Researchers have been using the idea of learning from demonstrations for robotic applications. Since deriving the policies by hand is hard, intractable, and requires a lot of theoretical expertise, there is room for performance issues due to imperfect modeling. Learning methods are getting popularity because it is simple to implement and efficient [6].

Researchers learned different driving styles using demonstrations as examples to understand the dynamics of driving styles that make the vehicles safe and reliable [78]. A linear parameterized sum of features associated with driving styles was used as the cost function for the inverse reinforcement learning model to understand the driving style of different drivers. A modular approach proposed by Kim et al. for path planning of robots in a human environment using inverse reinforcement learning [79]. The first module extracted the features to characterize the state information (velocity and density of obstacles) using a depth camera sensor. The second module used inverse reinforcement learning to learn expert behaviors from the demonstration in different states. Finally, the planning module integrated the prior information and used a path planning algorithm to determine the shortest path. Also, the low-level planning and execution including the immediate obstacle avoidance were handled by a typical approach. Ramirez et al. studied inverse reinforcement learning in two scenarios that took into account social norms for navigation [80]. First, the planner learned the appropriate way to approach a person in an open area without static obstacles. Later, the information was translated into learning the costmap using linear combinations of continuous functions to generate a path.

Deep reinforcement learning to develop a time-efficient navigation policy that respects the common social norms [81]. This method modeled the behavior of two humans navigating at walking speed. Further, the approach was generalized to work in an environment with multiple agents. A method for constructing an HRI policy for multimodal probabilistic future scenarios was developed by Schmerling et al. [82]. This method learned the policy from a multimodal probability distribution for future human actions by sampling the human responses to the actions predicted by the robot. This concept was tested in a highway-like scenario using simulation.

Attention-based deep reinforcement learning network can be used to model crowd-robot interaction [83]. By jointly modeling human-robot and human-human interaction, the network was able to anticipate the crowd dynamics and navigate into the time-efficient paths. Similarly, a dual social attention deep reinforcement learning method found a feasible, collision-free path for a robot in a crowded environment [84]. Here, the dual social attention network would model the complex interaction of the neighboring agents that helped the collision-free and efficient navigation of the robot in that environment.

A reinforcement learning-based end-to-end learning method used Deep Q Learning Network (D3QN) with convolution neural network (CNN) as its core network was successful in learning to navigate a robot by avoiding obstacles using images from Red Green Blue-Depth (RGB-D) camera [85]. A robust behavior cloning pipeline was employed to train and deploy the human driving behaviors into autonomous vehicles by Samak et al. [86]. The steering angle was predicted from a neural network model that used camera images as input. The velocity commands were derived from the steering output using a control function proposed in the paper. Extensive testing in various simulated scenarios was carried out to analyze the computational efficiency and robustness of the proposed pipeline.

By balancing the social preference with the desire to reach the goal, a model was able to learn the human-aware robot navigation using inverse reinforcement learning [87]. The proposed system used a force sensor to record the user input during navigation and optimized a cost function to learn the behavior while navigating. The approach was verified by deploying it into a real robot and using a user-based survey. The approach proposed by Smith et al. predicted the future human intention to generate a socially robust human avoidance behavior in the robot while remaining robust to the uncertainty of the human intention and motion variance [88]. This method employed a novel framework of cooperative MDP that considers the previous studies done on human motion, intention, and preference. Here, the behavior of humans is modeled into a discrete behavior model and the Co-MDP algorithm, which is a similar but extended form of multi-agent MDP, predicts the action integrating both human and robot behaviors and their joint transition function.

Decentralized Structural Recurrent Neural Network (DS-RNN) [89] learned the spatial ad temporal relationship between robot and crowd to make decisions when navigating through it. A model-free reinforcement learning method was applied to learn in an unsupervised setting. Later, the policy was deployed and verified into a real-world robot. Cui et al. used model-based reinforcement learning to learn the policies to navigate through a crowded environment [90]. The reward functions were designed to capture social conventions such as distance from other humans. This method successfully avoided the obstacles and navigated through the crowd.

2.4. Available Dataset

We can find several datasets of HAN scenarios developed by researchers for research. Most of the datasets are not for the general research of HAN but their specific study. In particular, we searched for the dataset that is most relevant to our research with the possibility of using it in our research. A dataset is typically composed of trajectory data in a human environment that can be used to model the dynamics associated with the environment.

The Edinburgh Informatics Forum Pedestrian Dataset [91] consists of over 90000 trajectories of pedestrians. The main objective of the research was to understand pedestrian behavior in open spaces and entries. The data was taken from above. ETH Pedestrian Dataset [92] used overhead cameras to track multi-person trajectories. It contained over 650 trajectories and was more than 25 minutes in length. Crowds UCY Dataset wanted to do the crowd simulation [93] by collecting 200 trajectories data. Similarly, Train Station Dataset [94] used 33 minutes of footage by annotating it as key-point trajectories to conduct crowd behavior analysis.

The Stanford Dome Dataset [95] took it to the next level by using 60 videos and, 20000 detected participants. It used bounding boxes in the videos to track the trajectories. The PETS Dataset [96] multipurpose dataset was released in 2009 as a challenge that annotated crowd density, and person count in a 5-gigabyte video. It was raw data and the detection and estimation were to be done by the researchers using it for their research.

The CongreG8 [8] is a dataset that has the trajectories along with body marker data from the participants for an approaching scenario. It has 385 trajectories data for human approaching and 38 robot approaching trials.

3. PROPOSED FRAMEWORK

This chapter describes the proposed IRL framework. First, we describe our study scenario, followed by the description of the dataset that we used and the data preparation. Then, we formulate our problem mathematically, explain the environment of the framework and finally, describe the algorithms used. In each section, we justify our design choices, too.

3.1. Navigation Scenario and Dataset

Among the various interaction scenario in human-aware robot navigation, we are interested in the scenario where the robot approaches a group of humans and tries to join their group, as in figure 5. In this section, we describe the scenario of our study and the detailed specification for the study environment.

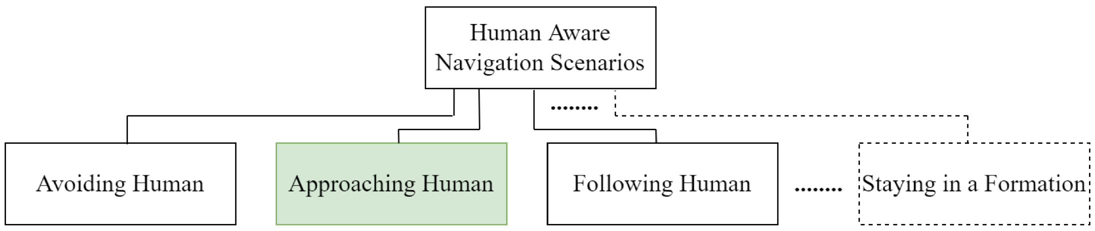
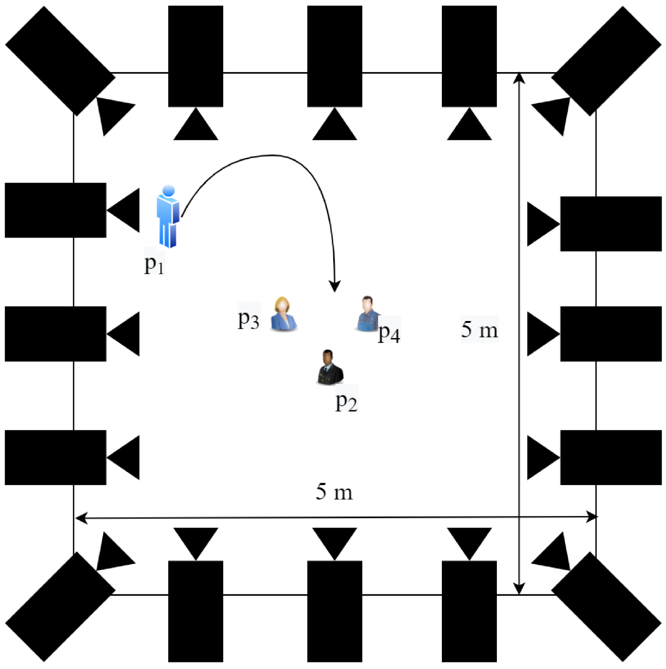


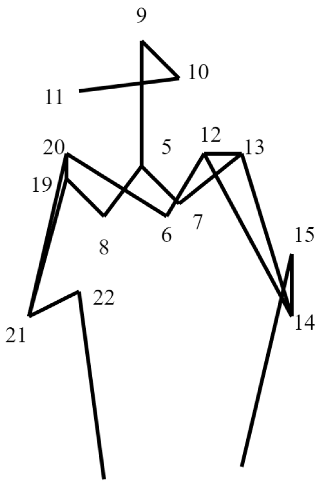
Figure 5. Among different navigation scenarios, we are interested in approaching humans (shaded in green)

The dataset [8] has full-body motion capture of the movement of the subjects during a scenario where a human tries to approach and join a group of humans. The participants play the game “who is the spy” [97], and the adjudicator approaches the group. The dataset consists of 380 human approach trajectories. The duration of each trajectory is between 2-30 seconds and the frequency is 120 frames per second (fps). Each frame of data contains the 3D positions of all corrected markers and the 3D position and rotation (quaternion) of skeletal bones of all four participants. In total, there are 37 markers and 21 skeletal bones for each human. The dataset is available in Comma Separated Value (CSV) file format. The scenario of data collection and marker positions can be seen in figure 6. An example of a trajectory plot can be seen in figure 7

There are three main reasons for selecting this dataset for our research. First, we had limited time, so it would have been challenging for us to design a whole new setup for data collection. Second, the dataset has high-quality data that was perfect for us to study the approaching behavior of the human and replicate it in the robot. Finally, the dataset contains the marker position and orientation which is already processed and corrected by the creators and used by other researchers in their experiments. So, we don’t need to run tracking algorithms and clean the data before using it.

We have designed our environment to accommodate both the scenario and the dataset. The environment space is a 6×6 m^2 square space which we divided into a grid cell of 0.1x0.1 m^2. So, we have 60 discrete rows and columns each, and a total





(a) (b)

Figure 6. (a) shows the scenario of data collection using 16 motion capture cameras, (b) shows the position of bone markers (above the chest, 13/37 markers) in the body.

of 3600 grid cells. We chose the grid size in such a way that a human can move a maximum of 1 step in either of eight directions per time step. In total, we have nine movement possibilities (8 directions and one, not moving) at each time step, as you can see in table 2. Similarly, the resolution for orientation is 10°. Furthermore, for simplicity, we consider the humans in the group to be stationary, i.e., we only consider the final position of the participants that are in the group.

Table 2. Movement and their effect in position per time step

S.No.	Movement	Change in Grid (x, y)
1	No Movement	(0, 0)
2	Up	(0, 1)
3	Up-Right	(1, 1)
4	Right	(1, 0)
5	Down-Right	(1, -1)
6	Down	(0, -1)
7	Down-Left	(-1, -1)
8	Left	(-1, 0)
9	Up-Left	(-1, 1)

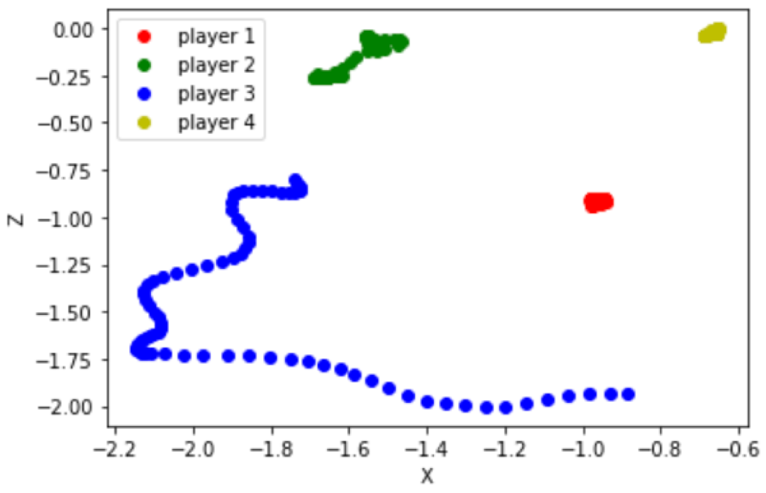


Figure 7. Example trajectory plot, here player 3 (blue) is joining the group that already has three humans (green, red, and yellow)

3.2. Data Preparation

This section describes the process that we used for preparing the data before using it in our project. We will sequentially mention the process together with the reason for using the specific method. Figure 8 illustrates our data preparation pipeline.

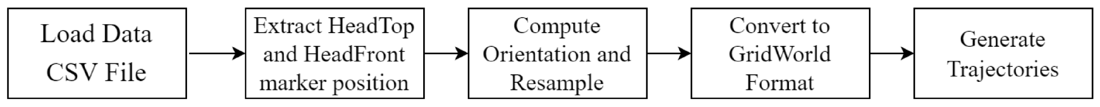

Load Data CSV File
Extract HeadTop and HeadFront marker position
Compute Orientation and Resample
Convert to GridWorld Format
Generate Trajectories

Figure 8. Data preparation pipeline that generates trajectory from given raw data

As mentioned above, the raw data is in CSV file format. So, the first step in data preparation was to load the data from CSV files. Each trajectory had its file so, we had to run a loop to load every file. We used pandas DataFrame [98] for loading the files and other processing steps.

Then, we extracted the marker positions HeadTop and HeadFront of each player at each time step, which are markers 9 and 10 respectively from figure 6(b). We only need HeadTop and HeadFront positions from now on because they are sufficient to give the position and orientation of the participants.

Next, we computed the orientation of the players, which is the same as the orientation of the vector that originates from the HeadTop marker and ends in the HeadFront marker. For calculating orientation, we used the following equation:

$$\theta = \tan^{-1}(\frac{y_2 - y_1}{x_2 - x_1})$$

where θ is the orientation, (x_1, y_1) and (x_2, y_2) are the marker positions of HeadTop and HeadFront respectively, for each participant. After calculating orientation, we resampled the frame rate from 120 Hz to 12 Hz (10 times reduction). We did this because the original frame rate was too high and there would not have been any actions for multiple frames. Decreasing the frame rate also reduces the memory and time requirement during data processing.

The next thing we did was to convert the real-world position and orientation into our discrete grid world format. As mentioned above, the resolution for the position is 0.1 m whereas for orientation is 10°. Also, the lower bound and higher bounds for position and orientation in the real world are (-3, 3), and $(-\pi, \pi)$, respectively. So, we used the following equations to calculate the grid coordinates from world coordinates.

$$(x_{grid}, y_{grid}) = (\frac{x - x_{min}}{x_{res}}, \frac{y - y_{min}}{y_{res}})$$

$$\theta_{grid} = \frac{\theta - \theta_{min}}{\theta_{res}}$$

Here, (x_{grid}, y_{grid}), and θ_{grid} are the position and orientation of a player in the grid world which is derived from the position(x, y) and orientation(θ) from world coordinates. Also, x_{min}, y_{min}, θ_{min}, x_{res}, y_{res}, and θ_{res} are the low bound for x, y, θ, and resolution for x, y, and θ. The table 3 has value of these parameters.

Table 3. Grid world parameters

Parameters	Value	Unit
x_{min}	-3	meter (m)
y_{min}	-3	meter (m)
θ_{min}	$-\pi$ or 180	radian or degree (°)
x_{res}	0.1	meter (m)
y_{res}	0.1	meter (m)
θ_{res}	10	degree (°)

Finally, we label the data (which coordinate belongs to the approaching participant and which coordinates belong to the group's participants) and generate trajectories in terms of state and action pairs, which we will describe in the later section.

3.3. Problem Formulation

To solve a problem by using mathematical tools such as algorithms, logic, or calculation, we need to properly define the problem in mathematical terms. This process is also known as problem formulation. This section describes how we use mathematics to structure the problem in that terms and related mathematical concepts along with their evolution. First, we introduce the concept of the Markov Decision Process (MDP), followed by Reinforcement Learning (RL) and Inverse Reinforcement Learning (IRL). Finally, we describe the application of these concepts in our research.

3.3.1. Markov Decision Process

An MDP [99] is a discrete stochastic process that provides a framework for modeling the situations where we know the probability of occurrence of some event depends on the action. MDP is typically used to make decisions in scenarios when there is some uncertainty associated with the outcome provided an action by an agent.

An MDP is a tuple with four elements:

- S is the set of states.
- A is the set of actions.
- $P_a(s, s')$ is the state transition probability which is equal to $p(s_{t+1} = s'|s_t = s, a_t = a)$.
- $R_a(s, s')$ is the immediate reward.

At any time t, the process is in any one state s from the set of states S in MDP. The agent in charge can take any action a from the set of actions available, A and the process moves to a new state s'. Here, the choice of a and resultant s' yields a reward determined by the reward function $R_a(s, s')$. Also, the probability that the state moves from s to s' by taking action a is given by the state transition function $P_a(s, s')$. We can see that in MDP, the s' depends on the s and a but is conditionally independent of all previous states and actions. This property is also known as Markov Property. Figure 9 is a simple example of MDP. It has three states and two probable actions for each state. Also, there are transition functions for each state-action pair, whereas the rewards associated with state and actions are sparse. We can imply that rewards for all the other cases are zero in this example.

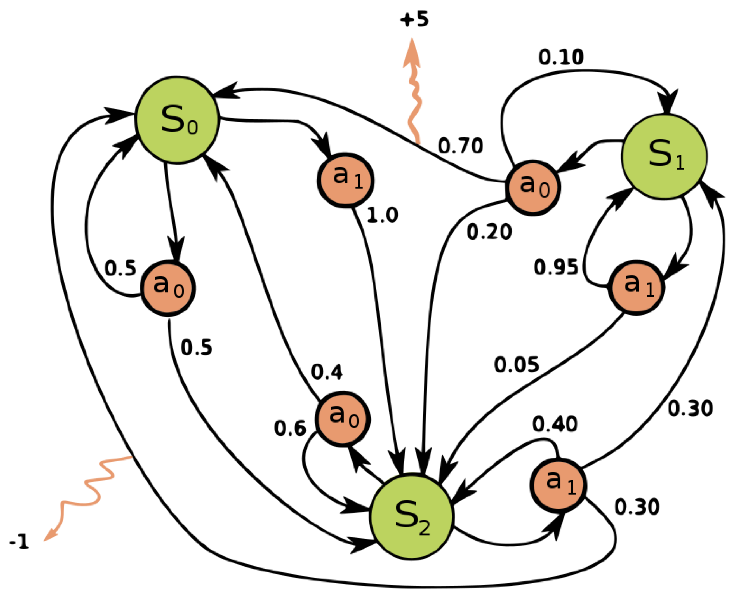


Figure 9. An example of MDP where the green circles represent the states, orange circles represent actions, numbers on the arrow represent state transition probabilities, and orange arrows represent associated rewards. Image from [100]

The main goal of MDP is to find a good policy that helps the decision-maker to make the decision. A policy ($\pi : S \rightarrow A$) is a function that maps S to A. It provides an action a for a state s. When an MDP is combined with $\pi(s)$, each state has an associated action, resulting in a Markov chain-like behavior.

For a policy to be good, it should maximize the long term reward. In that case, the policy is called optimal policy and denoted by π^*. The expected reward over an infinite horizon over π is given by:

$$R(\tau) = E\left[\sum_{t=0}^{\infty} \gamma^t R_{a_t}(s_t, s'_t)\right]$$

where γ is the discount factor ($0 \leq \gamma \leq 1$), which describes the immediate importance of reward compared to a time step in the future. For example, γ = 0.75 means that the same reward that we get in the next step is worth three forth the reward that we get now. Similarly, the path or trajectory τ is the ordered list of state-action pairs ($\tau = [(s_0, a_0), (s_1, a_1), ,,,]$), which denotes the action taken at each state by using $\pi(s)$.

3.3.2. Reinforcement Learning

The purpose of Reinforcement Learning (RL) is to solve the MDP problem, or in other words, to determine the optimal policy through a series of trial and error. For determining the optimal policy, the RL algorithm should maximize the cumulative reward. An RL problem can have the knowledge about the state transition function (model-based) or not (model-free).

As we can see in the figure 10, an RL problem consists of three main entities which have their respective roles. An agent acts in an environment, whereas an external viewer observes the environment and interprets it into state representation, and assigns an associated reward.

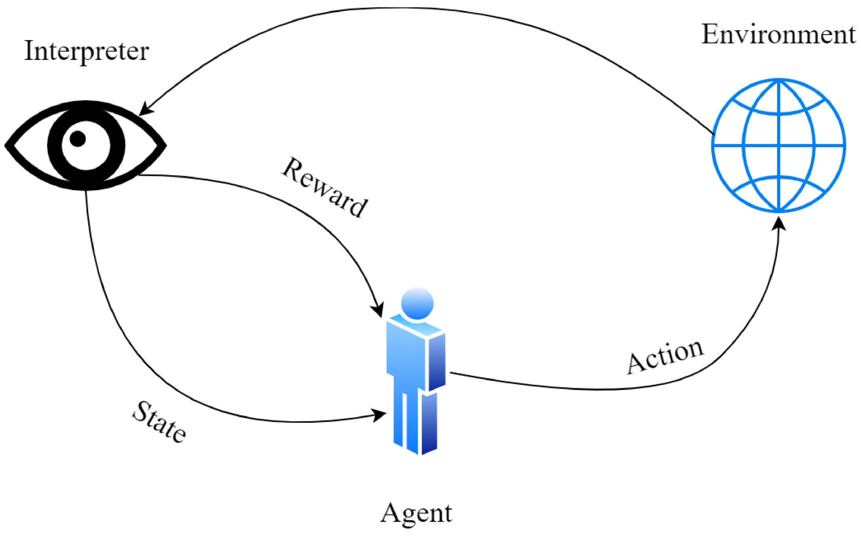


Figure 10. A typical RL framework.

There is more than one algorithm to solve the RL problem. Before knowing more about the algorithms to solve RL problems, we should first structure an RL problem.

For a RL problem, we are provided with an MDP $(S, A, P_a(s, s'), R_a(s, s'))$, discount factor γ, an environment (simulated or real) where we can do multiple trials. Now, the objective for RL is to learn a policy π, such that it maximizes the expected cumulative reward. A policy function $\pi(s)$, as described above, gives an action for a state. For a stochastic process, we can define a policy function $\pi(a, s) = p(a_t = a|s_t = s)$. In other words, $\pi(a|s)$ gives the probability of taking action a when the state is s at any time t.

If the process starts from any state s, then the total expected reward by following a policy π is known as the value function, $V_\pi(s)$ which is given by the following equation:

$$V_\pi(s) = E[R|s_0 = s] = E\left[\sum_{t=0}^{\infty} \gamma^t R_{a_t}(s_t, s_{t+1})|s_0 = s\right]$$

Here, s_{t+1} is the next state after s_t under π. The value function can be broken down into a recursive relationship as follows:

$$V_\pi(s) = \sum_{a \in A} \pi(a|s) \left[R_a(s, s') + \gamma \sum_{s' \in S} P_a(s, s') V_\pi(s')\right]$$

Similarly, the action-value function $Q_\pi(s, a)$ is the expected reward when starting from s, taking action a, and then following the policy π. The equation for Q-value function is as follows:

$$Q_\pi(s, a) = E[R|s_0 = s, a_0 = a] = E\left[\sum_{t=0}^{\infty} \gamma^t R_{a_t}(s_t, s_{t+1})|s_0 = s, a_0 = a\right]$$

Here, the $V_\pi(s)$ has the following relation with $Q_\pi(s, a)$:

$$V_\pi(s) = \sum_{a \in A} \pi(a|s) Q_\pi(s, a)$$

Therefore, we can write:

$$Q_\pi(s, a) = R_a(s, s') + \gamma \sum_{s' \in S} P_a(s, s') V_\pi(s')$$

Finally, the optimal policy π^* that we are so interested in, is the policy that maximizes the $V_\pi(s)$, and $Q_\pi(s, a)$ among all the policies. Also, the optimal value is the maximum value function over all the policies:

$$V^*(s) = \max_\pi V_\pi(s)$$

Similarly, the optimal action-value is the maximum action-value function over all the policies:

$$Q^*(s, a) = \max_\pi Q_\pi(s, a)$$

For simplicity, we define a policy π to be better than or equal to ($\geq$) another policy π', if the result of the value function of π is greater than or equal to that of π', for all the states s in S.

$$\pi \geq \pi', if V_\pi(s) \geq V_{\pi'}(s), \forall s$$

Therefore, we can say that there exists an optimal policy π^* that is better than or equal to all the other policies, $\pi^* \geq \pi$ for all the policies π. All the optimal policies achieve optimal value and action-value function:

$$V_{\pi^*}(s) = V^*(s)$$

$$Q_{\pi^*}(s, a) = Q^*(s, a)$$

3.3.3. Inverse Reinforcement Learning

While the motivation behind RL is to learn the optimal policy for an MDP when rewards are provided, through trial and error; the motivation behind the Inverse Reinforcement Learning problem is to learn the reward for an MDP when expert demonstrations are provided, as we can see in figure 11.

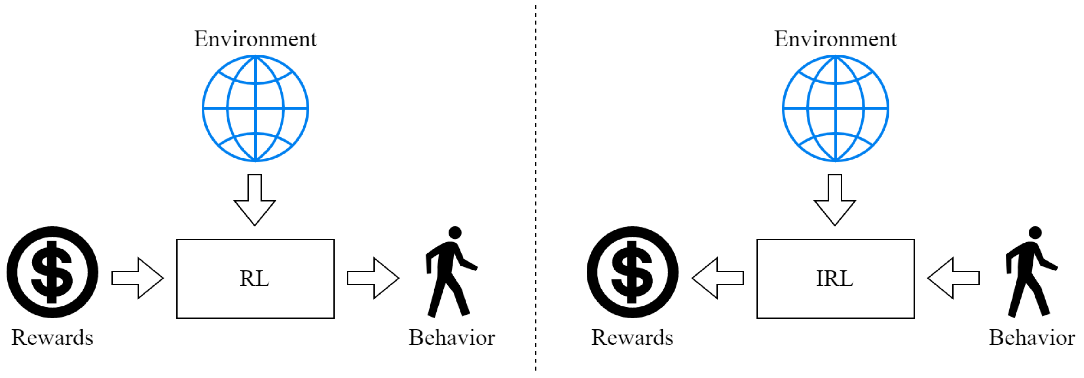


Figure 11. Difference between RL and IRL concept.

One of the ideas to solve the IRL problem is to use feature matching. Here, we assume that the reward function is the linear, parameterized sum of features. The features depend on the state and action. Now, the goal of IRL is to learn the value of those parameters. The linear reward function is:

$$R_\psi(s, a) = \sum_i \psi_i f_i(s, a) = \psi^T f(s, a)$$

, where ψ is the parameter vector that we want to learn, i ∈{0, 1, . . . , N-1} N is the number of features, and $f_i(s, a)$ are the feature functions.

For feature matching, we learn the reward function for which the optimal policy has the same expected value for these features. Let π^{R_ψ} be the optimal policy for R_ψ, then we can pick ψ such that the expected value of features for π^{R_ψ} is equal to the expected value of features for the optimal policy π^*:

$$E_{\pi^{R_\psi}}[f(s, a)] = E_{\pi^*}[f(s, a)]$$

. It means that, if we select the right features, we can assume the expected features from the expert demonstrations, to be equal to the optimal policy. However, the state-actions for π^{R_ψ} are marginal under π^{R_ψ} because we need to estimate the reward by going through each state-action pair, and the optimal policy π^* is still unknown. We can approximate the expected value of features from the optimal policy by averaging the features from expert samples, but multiple different ψ vectors may result in equal feature expectations. So, the problem is still ambiguous.

We can maximize the difference in between the expected reward we obtained from π^* and all other policies Π to disambiguate the problem,

$$\max_{\psi,m} m, \quad \text{such that} \quad \psi^T E_{\pi^*}[f(s,a)] \geq \max_{\pi\in\Pi} \psi^T E_{\pi}[f(s,a)] + m$$

. Here, m is the margin. It basically means that we need to find a vector ψ that maximizes the expected reward for the policy from expert demonstration from all other policy by the maximum margin. This is similar to the maximum margin principle concept from Support Vector Machine (SVM) [101]. Burrowing the idea from SVM, we can simplify the problem as:

$$\min_{\psi} \frac{1}{2}||\psi||^2, \quad \text{such that} \quad \psi^T E_{\pi^*}[f(s,a)] \geq \max_{\pi\in\Pi} \psi^T E_{\pi}[f(s,a)] + D(\pi,\pi^*)$$

where $D(\pi,\pi^*)$ is some calculated divergence between π and π^*.

Still, the idea of maximizing the margin is adopted from somewhere else and is arbitrary. Also, it does not address the issue of suboptimal behaviors from experts. So, we add optimality variables to our problem, and its probability at any time t is represented by:

$$p(O_t|s_t,a_t) = exp(R(s_t,a_t))$$

We can represent the optimality variable in Probabilistic Graphical Model (PGM) as in figure 12.

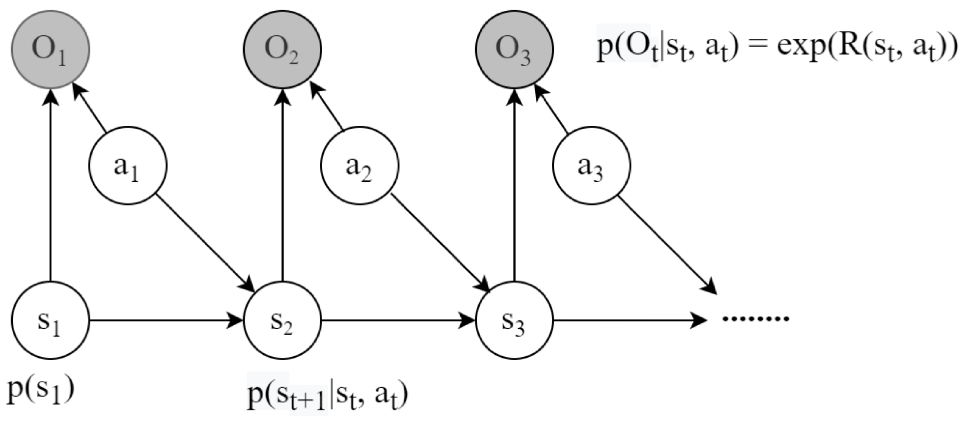


Figure 12. Representation of our problem as PGM and introduction of optimality variable O.

Now, the probability of getting a trajectory, given the expert was working optimally, is:

$$p(\tau|O_{1:T}) = \frac{p(\tau,O_{1:T})}{p(O_{1:T})}$$

which is equivalent to:

$$p(\tau|O_{1:T}) \propto p(\tau)\prod_t exp(R(s_t,a_t)) = p(\tau)exp(\sum_t R(s_t,a_t))$$

In this PGM, our aim is to learn the reward parameters ψ. So, by adding ψ to the optimality variable, we get:

$$p(O_t|s_t, a_t, \psi) = exp(R_\psi(s_t, a_t))$$

and

$$p(\tau|O_{1:T}, \psi) \propto p(\tau)exp(\sum_t R_\psi(s_t, a_t))$$

Since we already have samples $\{\tau_i\}$ from $\pi^*(\tau)$, we use Maximum Likelihood Estimation (MLE) [102] to maximize the probability of the trajectories that we have observed. The $p(\tau)$ is independent of the parameter vector ψ, so we can ignore it for the MLE:

$$\max_\psi \frac{1}{N}\sum_{i=1}^{N} logp(\tau|O_{1:T}, \psi) = \max_\psi \frac{1}{N}\sum_{i=1}^{N} R_\psi(\tau_i) - logZ$$

Here, N is the total number of sample trajectories and $log(Z)$ is the log normalizer. The normalizer Z, also known as the partition function, is equal to:

$$Z = \int p(\tau)exp(R_\psi(\tau))\, d\tau$$

By taking the gradient of our MLE problem, we get:

$$\Delta_\psi L = \frac{1}{N}\sum_{i=1}^{N} \Delta_\psi R_\psi(\tau_i) - \frac{1}{Z}\int p(\tau)exp(R_\psi(\tau))\Delta_\psi R_\psi(\tau)\, d\tau$$

which we can write as:

$$\Delta_\psi L = E_{\tau\sim\pi^*(tau)}[\Delta_\psi R_\psi(\tau_i)] - E_{\tau\sim p(\tau|O_{1:T},\psi)}[\Delta_\psi R_\psi(\tau)]$$

In the above equation, we can see that, the gradient of MLE can be obtained by subtracting the expected value of the gradient of reward under $p(\tau)$ given current ψ from the expected value of the gradient of reward under optimal policy π^*. So, we can estimate the first term from the expert samples whereas, we can run the inference on soft-optimal policy for current reward parameters to estimate the second term. To determine the soft-optimal policy, we should solve the whole problem of reinforcement learning for the current reward.

3.4. Proposed Approach

This section consists of the description of the MDP that we used, along with the IRL algorithm.

3.4.1. Environment

As already explained in earlier sections, to solve an IRL problem, we need to properly define an MDP $(S, A, P_a(s, s'), R_a(s, s'))$. In our research, the state vector is

$$s = (x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$$

where $(x_{1,...,4}, y_{1,...,4}) \in \{0, 1, 2, \ldots, 60\}$. Similarly, (x_1, y_1) is the coordinate of the approaching participant p_1, whereas (x_2, y_2), (x_3, y_3), and (x_4, y_4) are the coordinates of the participants in the group p_2, p_3, and p_4, respectively. Since we are considering the group stationary, the (x_2, y_2), (x_3, y_3), and (x_4, y_4) are the final positions for respective participants.

The action space consists of one-step movement in each eighth direction of the 2D plane, and one, no movement as we can see in Table 2. So, the action only affects the movement of p_1, as we are learning the behavior of the person approaching the group. We can see the example of state transition in Figure 13.

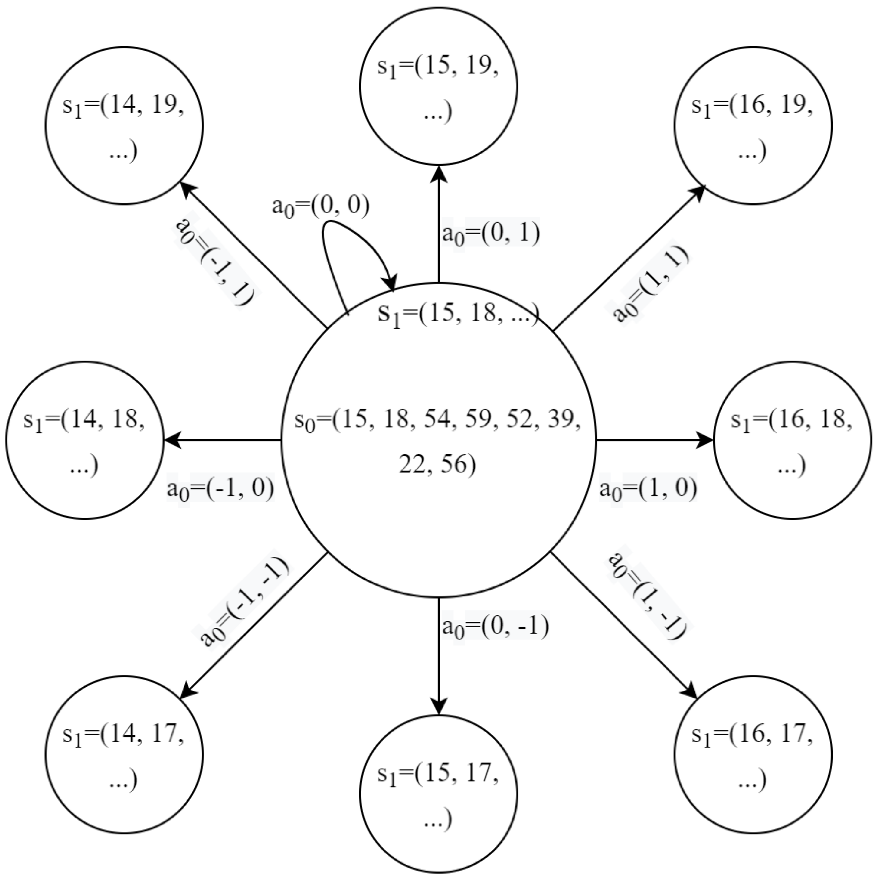


Figure 13. An example of state transition under all nine actions. s_0 is the current state, a_0 is the action when the state is s_0, and s_1 is the future state.

For generating the path, we implemented our path generator based on the MDP and a policy generator. Alternatively, we can also call it the simulator. We design the simulator such that it generates a path that has similar properties to the samples from the dataset. The simulation environment follows the flow chart in Figure 14.

For initialization, we generate a random state vector that is also the start state of the path, $s_0 = (x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4)$ that follows the following conditions. We formulated these conditions for the start state based on our finding of Preliminary Data Analysis (PDA) of the dataset, which we have explained in Chapter 4.

1. The radius of the virtual circle (radius r_c, and center (x_c, y_c)) formed by the position of p_2, p_3, and p_4 is between the 3 and 11 units. i.e.:$3 \leq r_c \leq 11$. We can

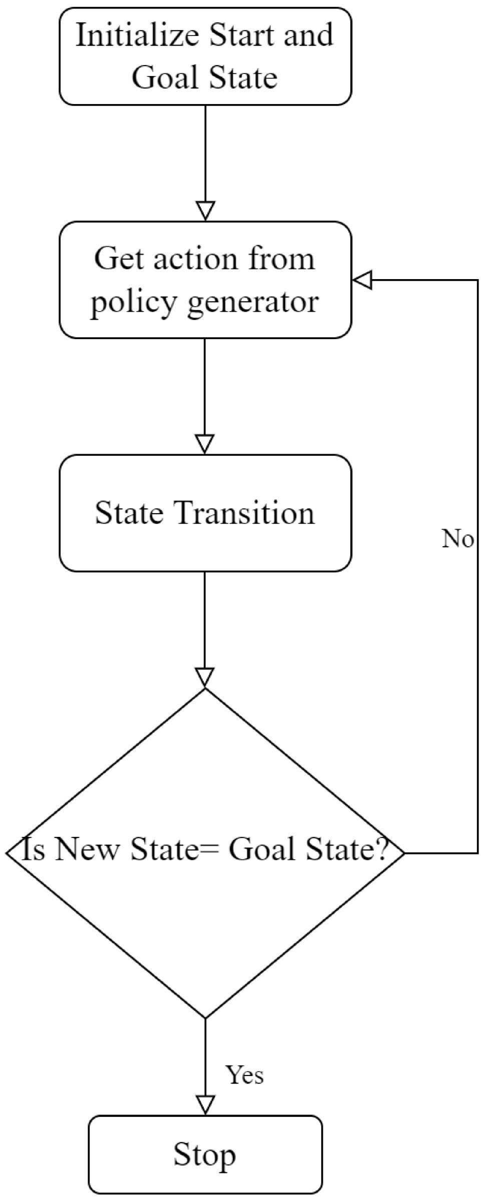

Initialize Start and Goal State
Get action from policy generator
State Transition
No
Is New State= Goal State?
Yes
Stop

Figure 14. The flow chart for generating path.

calculate the radius r_c, and center (x_c, y_c) by using the following mathematical process:

Let us assume the complex numbers $z_1 = x_2 + iy_2$, $z_2 = x_3 + iy_3$, and $z_4 = x_4 + iy_4$; a new complex number $w = \frac{z_3 - z_1}{z_2 - z_1}$. Using the following equation, we can derive a complex number c that has the required information.

$$c = \frac{(z_1 - z_2)(w - ||w||^2)}{2i\Im(w)} - z_1$$

Then the center $(x_c, y_c) = (-\Re(c)), -\Im(c))$, and $r_c = ||c + z_1||$

2. The distance between the center of the virtual circle formed by the group and the starting position of the approaching participant be greater than or equal to $\sqrt{7{,}5}$ times the radius of the virtual circle. i.e.:
$$(x_1 - x_c)^2 + (y_1 - y_c)^2 \geq 7{,}5 r_c^2$$

Referring to the PDA, the goal state condition for our case is: the position of p_1 touches the circumference of the virtual circle made by the group. Mathematically:
$$(x_1 - x_c)^2 + (y_1 - y_c)^2 \leq r_c^2$$

After initializing the start state and defining the goal condition, the simulator starts generating actions for the states. A policy generator is the policy function that generates an action for a state. The next section explains the details of the policy generator. State transition occurs when the simulator applies the action, and we get the new state. Figure 13 represents an example of how the state transition occurs. Now, the next step is to check the goal condition, if the new state satisfies the goal condition, the simulator stops the simulation and returns the path list, else it uses the policy generator to get an action for the new state; the process repeats until the new state satisfies the goal condition.

3.4.2. IRL Algorithm

Since, our state space is large (for one participant it is 3600), enumerating all state-action tuples is impractical. Also, we need to solve the RL problem at each parameter gradient step to get a soft optimal policy, which is not feasible because of the unknown dynamics of our problem. One thing that we can do at this stage is to use a policy that is not optimal, but a suboptimal policy. It means that we can use a policy that is not fully optimized but is better than the previous step, i.e, in the direction of optimality. This idea is the basis of the algorithm that we are using in this research. Guided Cost Learning (GCL) [7] learns the optimal reward values from the human demonstrations by using Deep Neural Networks (DNN).

The GCD is a DNN-based IRL algorithm that simultaneously learns the optimal cost and policy from the expert demonstrations. It optimizes the same IRL equation that we mentioned in previous sections. Here, we represent the reward function and the policy function as DNN and use Algorithm 1 to optimize the respective networks.

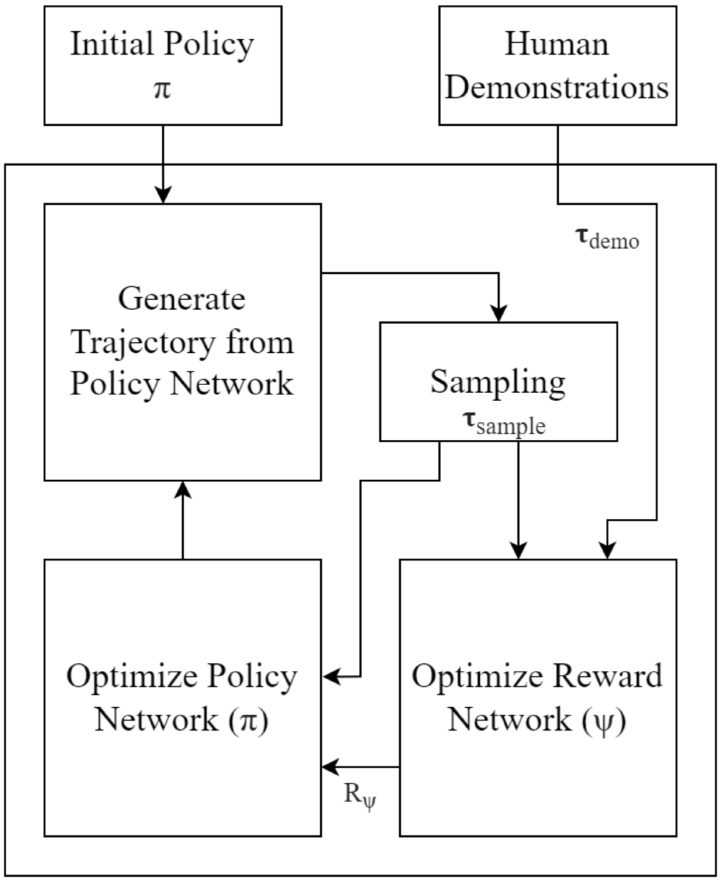

Initial Policy π
Human Demonstrations
τdemo
Generate Trajectory from Policy Network
Sampling
τsample
Optimize Policy Network (π)
Optimize Reward Network (ψ)
Rψ

Figure 15. Guided Cost Learning Method.

Algorithm 1. Guided Cost Learning

```
Input  : Examples of expert demonstration $\tau_{demo}$
Output: Optimized reward parameters $\psi$, and policy $\pi^*$
1  Initialize random policy network $(\pi)$
2  for $i \leftarrow 0$ to $I$ do
3      Generate samples $\tau_{traj}$
4      Append samples $\tau_{samp} \leftarrow \tau_{traj} \cup \tau_{samp}$
5      for $k \leftarrow 0$ to $K$ do
6          Sample demonstration batch $\hat{\tau}_{demo} \subset \tau_{demo}$
7          Sample generated batch $\hat{\tau}_{samp} \subset \tau_{samp}$
8          Append demonstration and generated batch to generated batch
             $\hat{\tau}_{samp} \leftarrow \hat{\tau}_{samp} \cup \hat{\tau}_{demo}$
9          Estimate $\delta L_{\psi}$ from $\hat{\tau}_{samp}$ and $\hat{\tau}_{demo}$
10         Update parameters $(\psi)$ of reward network $(R_{\psi})$
11     end
12     Estimate $\delta L_{\beta}$ for the policy network $\pi$ using $\tau_{traj}$ and updated reward
         network$(R_{\psi})$
13 end
```

Here, δL_ψ, and δL_β are the loss functions for the reward network and policy network, respectively. N and M are the total number of sampled trajectories from $\hat{\tau}_{demo}$ and $\hat{\tau}_{samp}$, respectively. The equation to estimate δL_ψ is:

$$\delta L_\psi = \frac{1}{N} \sum_{\tau_i \in \tau_{demo}} R_\psi(\tau_i) + log \frac{1}{M} \sum_{\tau_j \in \tau_{samp}} z_j exp(-R_\psi(\tau_j))$$

where, z_j is the importance weight of the sampled trajectory, which is equal to $z_j = (p(\tau_j))^{-1}$. The policy generator also estimates the value of z_j. We can estimate δL_β for the policy network by using:

$$L_\beta = -\frac{1}{M} \sum_{\tau_j \in \tau_{traj}} \left[\sum_{a_k \in A} [log(p(\tau_j))\pi_\psi(s, a_k))] R_\psi(\tau_j) - H(\tau_j) \right]$$

The term $H(\tau_j)$ is the entropy of the trajectory, whose value is equal to $H(\tau_j) = -\sum_{\tau_j \in \tau_{traj}} p(\tau_j) log(p(\tau_j))$. M in the above equation denotes the total number of generated trajectory τ_{traj}. For optimization, we used Adam optimizer [103] and pytorch environment [104].

4. RESULTS AND DISCUSSION

This section consists of the analysis of results that we obtained in various steps of our research. This section also discusses the major issues that we encountered during those steps and the significance of the results. First, we explain the results, followed by a discussion of the thesis. Finally, we suggest some possible future research directions.

4.1. Results

The dataset that we used has 385 trajectories of a human approaching a group of three humans to initiate a conversation. Figure 16 illustrates the frequency distribution of the euclidean distance between the start position and end goal position of the approaching human p_1. The mean distance is 1.44 m, and the graph shows the balanced distribution of distance, with the minimum and maximum distance being 0.22 m and 2.77 m. The size of the environment and the distance between the start position and goal are in the range that can be used to conduct experiments in our case. We can also create similarly sized simulation environments and test the approach trajectories.

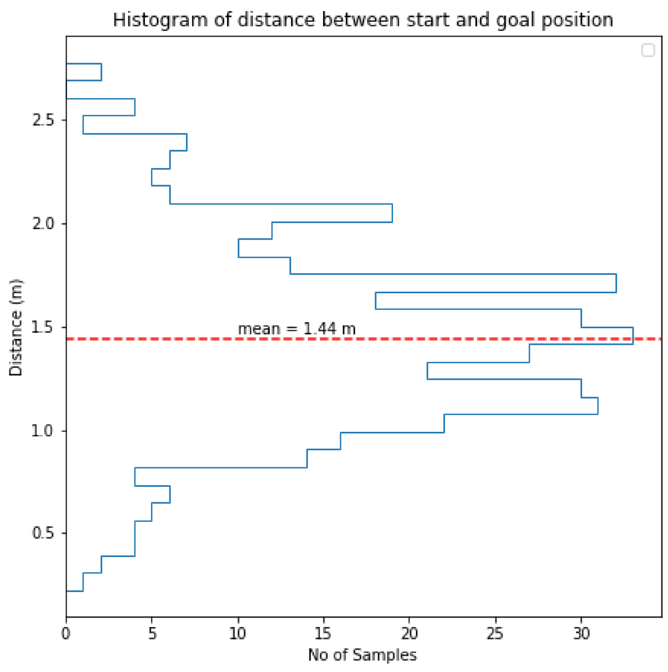


Figure 16. Histogram of distance between start and goal position of the trajectories from the dataset.

By down-sampling the original high frame rate of 120 fps, we converted the dataset into a frame rate is 12 fps. Each trajectory in the dataset lasts between 2.42 sec to 32.83 sec, which translates to 29 frames and 394 frames, respectively. The mean duration of each trajectory is 9.44 sec or roughly 113 frames. The frequency distribution of time taken in for each trajectory is in Figure 17. The distribution plot shows that the time distribution is not quite balanced and has some positive skewness. The duration of each trajectory is decent in terms of usage in our application. Had it been longer than this, it would have been difficult to use in our study. Similarly, shorter trajectories would have been pretty much useless for us, considering the information they would have carried.

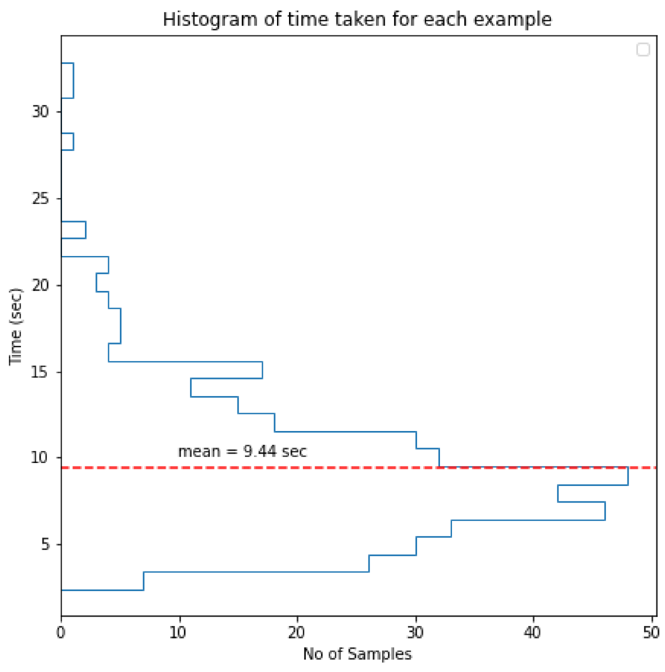


Figure 17. Histogram of time taken for each trajectory of the dataset.

The distribution of the average speed for each trajectory looks almost symmetric in Figure 18. The mean average speed is 0.45 sec, whereas the fastest and the slowest trajectory speeds are 0.08 m/s and 0.96 m/s respectively. The second-fastest speed is quite low compared to the fastest speed and measures 0.84 m/s. If we compare the results with the standard human walking speed (mean speed = 0.869 [105]), our mean speed is quite low. It may be because while humans are approaching another human or group for making an interaction, they tend to slow down as they get closer to the target. We can verify this concept through our result in the next paragraph.

Figure 19 compares the average speed of the whole trajectory with the speed of the final stages of the trajectory (last 2 seconds of the trajectory). We can see here that the approaching human slows down when it is approaching the proximity of the group by a factor of 4.5 times on average. It suggests that designing a robot that approaches humans intending to have an interaction should significantly limit its approaching velocity.

Another interesting observation is that the approaching human p_1 has the goal position very close ($\approx$ 0.17 m on average) to the circumference of the virtual circle made by the target group. Figure 20 shows the examples where the trajectory (blue color), ends almost exactly on the boundary of the circle. We know the fact that we

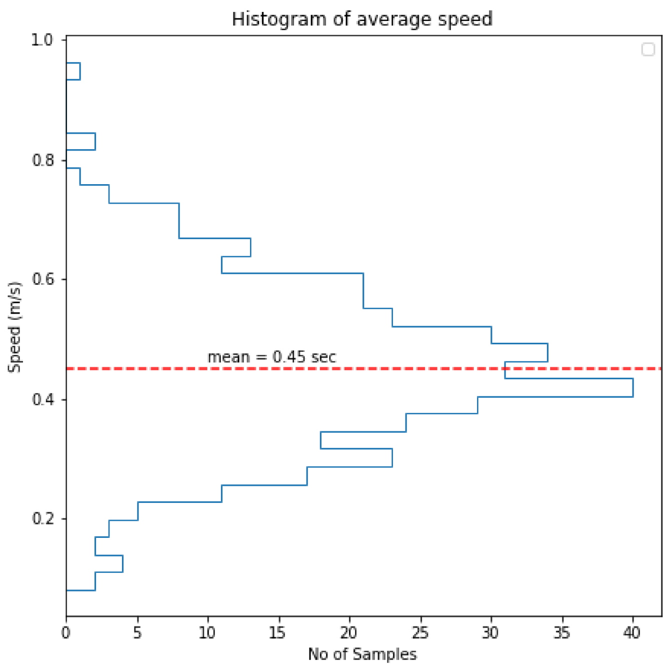


Figure 18. Histogram of average speed of p_1 from the dataset.

can create a circle in a 2D plane by using any three noncollinear points in the plane. So, fitting three people in a circle's circumference is trivial. However, our observations show that the fourth point fits the circle with a small margin, which is interesting and gives us the foundation for applying it further in our research. The comparison between the radius of the virtual circle and, the distance between the goal position and the center of the circle in Figure 21 shows that both the mean and median differences are 0.17 m. If we analyze the difference, we can cover that distance in almost a third of a second, referencing the average speed of 0.45 m/s in Figure 18. We have used this implication for formulating the goal condition in Chapter 3.

Figure 22 illustrates the difference in the distance between the final position of p_1 and the center of the virtual circle with the radius of the virtual circle made by the group. At this point, we can assume that they are close enough to consider that touching the circumference of the virtual circle can be the goal of our navigation.

In Figure 23, we can see that the minimum, maximum, and mean radius of the virtual circle are roughly 0.3 m, 0.6 m, and 1.1 m respectively. So, while designing the condition of initializing the start state, we have used these statistics to define the minimum and maximum radius of the virtual circle made by the group.

For initialization of the start state, we have another condition that the distance between the start position of p_1 should be greater or equal than $\sqrt{7{,}5}$ times the radius of the circle. The formulation of that condition was according to our finding in, Figure 24 which shows that on average, the aforementioned distance is almost 2.75 times the radius. For ease of implementation, we used an approximate scale quantity of $\sqrt{7{,}5}$ while checking the condition.

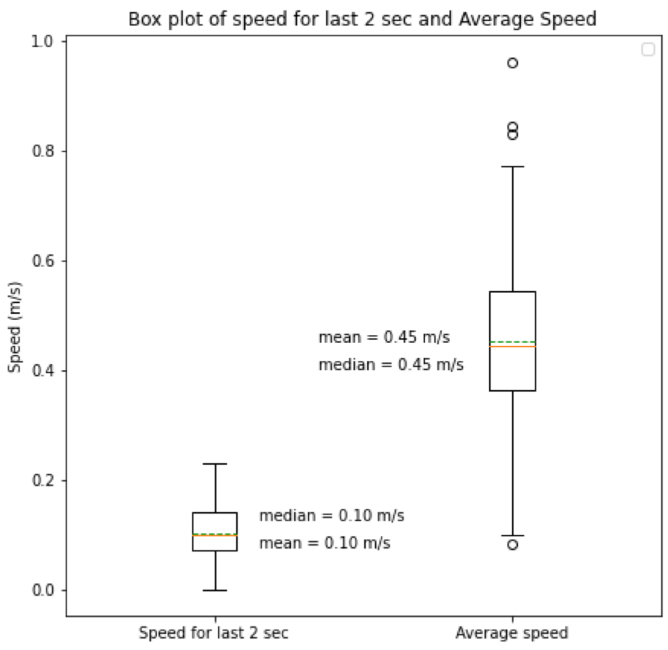


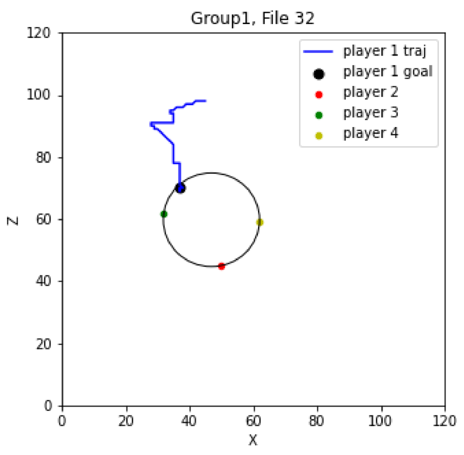
Figure 19. Comparison between the speed while approaching the group with the average speed for the trajectories from dataset.

4.2. Discussion

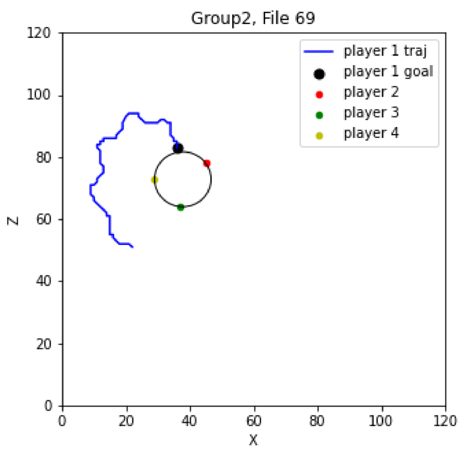
This thesis focused on learning the navigation behaviors for approaching a group of people. We are trying to learn the behavior from humans because we want to generalize it and apply it to a telepresence robot. The main objectives of this thesis were to analyze the dataset, establish a mathematical formulation for an IRL framework, and structure the dataset to a format that can be used by the framework to learn the associated reward and policy.

From PDA, we found some interesting results that align with the literature in HAN. First, we found out that the approaching human tends to form a circle with the rest of the group while interacting. This finding aligns with the O-hypothesis [28]. We have used this finding to establish conditions for initializing states while generating the samples. We also derived a condition to check if the approaching human reached the goal state during simulation, based on the same findings.

Another interesting finding that agrees with the past research is the approaching speed. The mean average speed of the trajectories is slow compared to the average walking speed of humans. The reason for this might be that the human slows down when it reaches close to the group. The humans moved significantly slower in the last 2 seconds of their trajectory when compared to the average speed of the trajectory. It suggests that the humans modulate their speed according to the distance from other humans while approaching them, as suggested by past research [21, 28]. However, this finding can be used to verify the acceptability of the trajectories generated by the



(a)



(b)

Figure 20. (a), and (b) both are the examples that verify the O-space hypothesis from [28]

learned policy network. For example, one of the feature functions could utilize the relation between the velocity and distance to the goal.

Furthermore, we can see abnormalities in the dataset considering typical approach behavior. Figure 25 shows that there is a sharp change in direction of the trajectory at some point during the navigation. In normal circumstances, we would expect the approaching human to take a direct or shortest path to the group proximity. However, the participant seems to move in one direction and suddenly changes the direction to join the group, which looks strange in the first place. We can explain this behavior by looking at the data collection process. Since the recording of the dataset took place when the participants were playing a game, there was a trigger point after which, p_1 started moving towards the group with an intention to join the group. Before that trigger, p_1 was just moving along a random path without any specific intention, waiting for the trigger signal. Since these trajectories are not natural behaviors and are caused by game rules, we should filter them out for better results.

For the learning framework, we developed an environment that generated the initial state of trajectories. Those initial states had similarities with the initial states of expert trajectories because we used the properties from the dataset about the size of the circle and the distance of the human from the center of the circle. We can see an example of the initial state generated by our method in Figure 26. Here, the position of all four humans satisfies the conditions that we talked about earlier. By doing a random walk from this initial condition, we generated the path similar to Figure 27.

We faced some challenges while designing the framework. The first challenge was the choice of the reward function. Generally, reward functions are supposed to represent the features of navigation. Human behavior is so complex that we chose not to only use pre-defined cost functions since those often fail to capture the complete behavior. Thus, we propose using a DNN as the reward function; the network would learn to adjust its weight to give maximum rewards. Also, the weights of the reward network can learn the features that we would have missed if we used a certain set of feature functions.

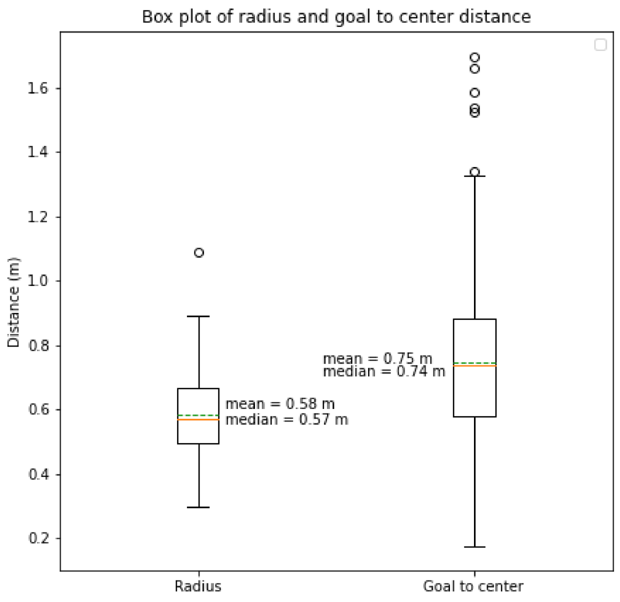


Figure 21. Comparison of the distance between the final position of p_1 and the center of the virtual circle with the radius of the virtual circle made by the group.

The IRL framework that we proposed optimizes the parameters of the reward network by doing two operations simultaneously. First, it maximizes the expected reward for the expert trajectories, and second, it penalizes the expected reward for trajectories obtained from the soft optimal policy divided by a normalization factor. The normalization factor is the function of the probability of a certain trajectory, provided we know the optimality variable. But we know that initially, the trajectories picked by the policy generator network are not truly optimal, and they get closer to an optimal trajectory as the training progresses. So the normalization factor, which is the probability of a certain trajectory given optimality, increases with training. So, the penalizing term in the loss function of the reward network decreases with training. This means that the policy generator is optimizing and at one point gives the optimal policy.

The proposed framework can be used with other datasets. For example, we can collect data for another scenario or collect a set of new data for the same scenario, we can use the proposed framework to learn the associated reward and policy with a few modifications. If we use a dataset with a similar environment, we need to change the conditions for initialization and goal by doing a statistical analysis of the dataset to find the properties such as the radius of the virtual circle, the distance between the start and goal position. However, if we want to change the scenario or the environment, a new environment has to be developed for simulation along with some modifications in policy and reward networks too. However, chunks of present implementation can be reused in the new environment.

In our framework, we propose a generic policy gradient method for optimizing the policy network. One of the possible future directions of research can be experimenting

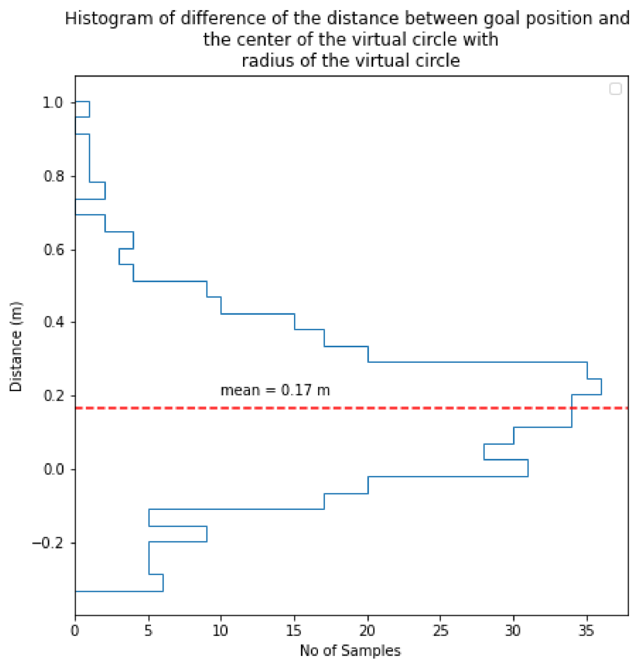


Figure 22. Histogram of difference of the distance between the final position of p_1 and the center of the virtual circle with the radius of the virtual circle made by the group.

with other policy optimization networks, such as REINFORCE and actor-critic algorithms [106]. Without proper experiments, it is not clear which algorithm is better in different scenarios, since learning algorithms perform differently in different scenarios. So, next after this is to conduct experiments with different networks and compare the results. Finally, if the trigger issue described in the results section appears to cause any issue during training, we can also collect our own data.

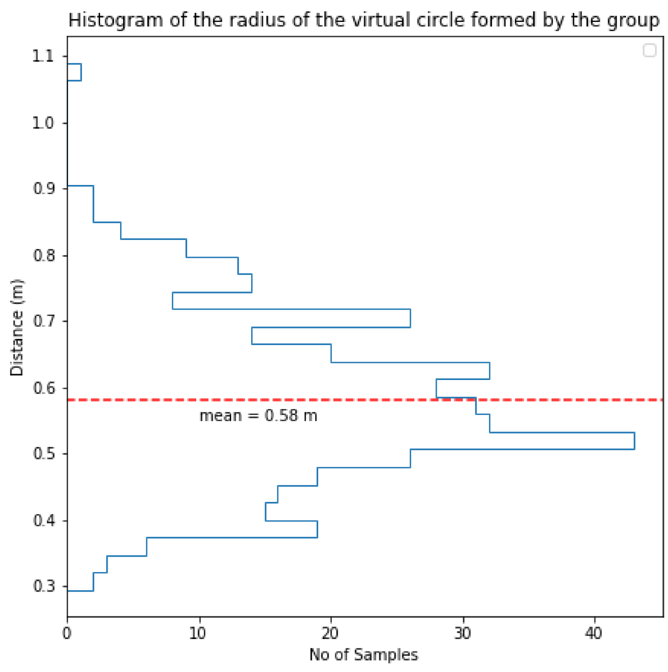


Figure 23. Histogram of the radius of the virtual circle made by the group.

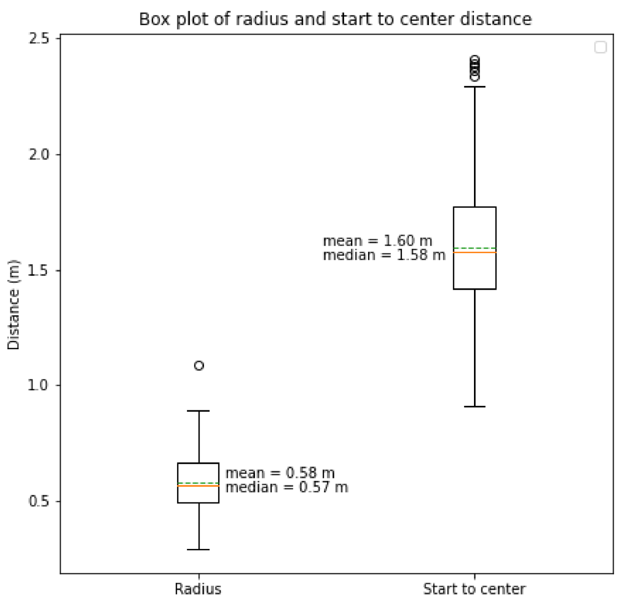


Figure 24. Comparison of the distance between the start position of p_1 and the center of the virtual circle with the radius of the virtual circle made by the group.

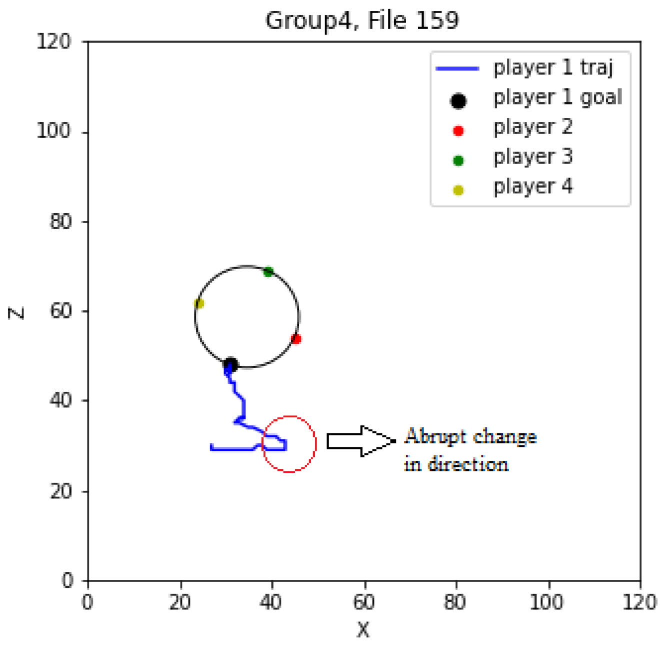


Figure 25. An example of abrupt turn taken by p_1 due to trigger.

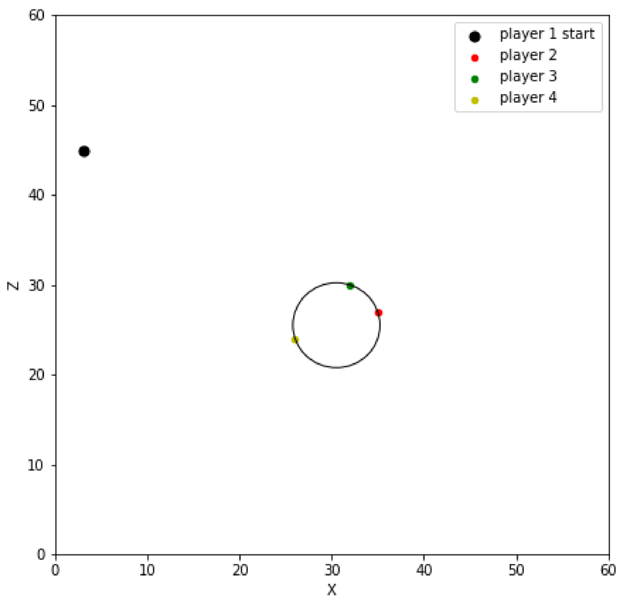


Figure 26. An example of generated start state of the trajectory.

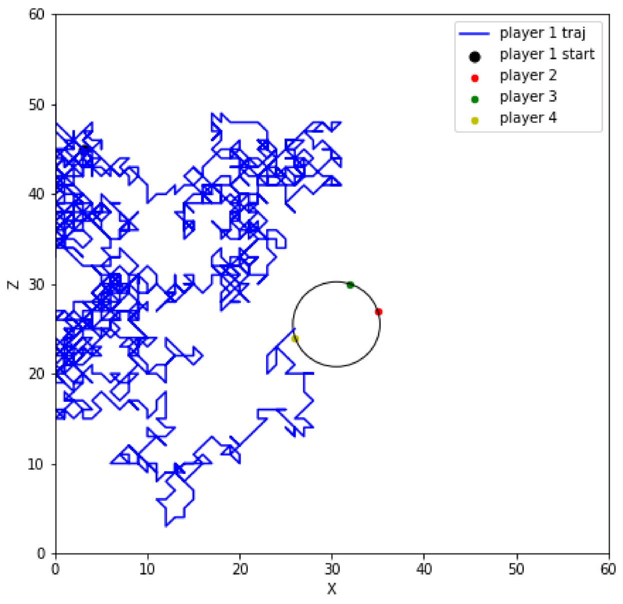


Figure 27. Path generated by random walk for initial state in Figure 26.

5. CONCLUSION

This thesis focused on understanding the human behavior while approaching a group to find a policy that generates human-like trajectories for an autonomous mobile telepresence robot in similar scenarios. To this end, we analyzed a dataset of human trajectories collected in a similar case. We found that people, while interacting as a group, tend to make a circle, and a person who is approaching the group slows down as he or she gets closer to it. Based on the conditions derived from analyzing the dataset of trajectories, we proposed an IRL framework that can be used to simultaneously learn the optimal reward and policy from human demonstrations.

Future works include a user study to test whether the policy learned using the proposed framework is acceptable to people, and a comparison against different methods to find a policy. Proper experiments are needed to claim that the framework performs better than other methods. Especially, there lies a lot of choices in policy network, so it can be studied and experimented in more detail.

6. REFERENCES

[1] Bryan Robinson P. (2022), Remote work is here to stay and will increase into 2023, experts say. URL: `https://www.forbes.com/sites/bryanrobinson/2022/02/01/remote-work-is-here-to-stay-and-will-increase-into-2023-experts-say/?sh=33ab1a7820a6`.

[2] (2021), Zoom, microsoft teams, and slack have exploded due to the covid-19 pandemic. can they hold onto this growth? URL: `https://glginsights.com/articles/zoom-microsoft-teams-and-slack-have-exploded-due-to-the-covid-19-pandemic-can-they-hold-onto-this-growth/`.

[3] Mavrogiannis C., Baldini F., Wang A., Zhao D., Trautman P., Steinfeld A. & Oh J. (2021) Core challenges of social robot navigation: A survey. arXiv preprint arXiv:2103.05668 .

[4] Satake S., Kanda T., Glas D.F., Imai M., Ishiguro H. & Hagita N. (2009) How to approach humans? strategies for social robots to initiate interaction. In: Proceedings of the 4th ACM/IEEE international conference on Human robot interaction, pp. 109–116.

[5] Repiso E., Garrell A. & Sanfeliu A. (2018) Robot approaching and engaging people in a human-robot companion framework. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp. 8200–8205.

[6] Argall B.D., Chernova S., Veloso M. & Browning B. (2009) A survey of robot learning from demonstration. Robotics and autonomous systems 57, pp. 469–483.

[7] Finn C., Levine S. & Abbeel P. (2016) Guided cost learning: Deep inverse optimal control via policy optimization. In: International conference on machine learning, PMLR, pp. 49–58.

[8] Yang F., Gao Y., Ma R., Zojaji S., Castellano G. & Peters C. (2021) A dataset of human and robot approach behaviors into small free-standing conversational groups. PloS one 16, p. e0247364.

[9] Kruse T., Pandey A.K., Alami R. & Kirsch A. (2013) Human-aware robot navigation: A survey. Robotics and Autonomous Systems 61, pp. 1726–1743.

[10] Meyers R.A. et al. (2009) Encyclopedia of complexity and systems science, vol. 9. Citeseer.

[11] Jensen B., Tomatis N., Mayor L., Drygajlo A. & Siegwart R. (2005) Robots meet humans-interaction in public spaces. IEEE Transactions on Industrial Electronics 52, pp. 1530–1546.

[12] Burgard W., Cremers A.B., Fox D., Hähnel D., Lakemeyer G., Schulz D., Steiner W. & Thrun S. (1999) Experiences with an interactive museum tour-guide robot. Artificial intelligence 114, pp. 3–55.

[13] Arras K.O., Tomatis N. & Siegwart R. (2003) Robox, a remarkable mobile robot for the real world. In: Experimental robotics VIII, Springer, pp. 178–187.

[14] Philippsen R. & Siegwart R. (2003) Smooth and efficient obstacle avoidance for a tour guide robot. In: Proceedings. 2003 IEEE International Conference on Robotics and Automation, September 14-19, 2003, The Grand Hotel, Taipei, Taiwan, vol. 1, IEEE Operations Center, vol. 1, pp. 446–451.

[15] Siegwart R., Arras K., Jensen B., Philippsen R. & Tomatis N. (2003) Design, implementation and exploitation of a new fully autonomous tour guide robot. In: Proceedings. ASER0́3, 1st International Workshop on Advances in Service Robotics: March 13-15, 2003-Bardolino, Italy, Fraunhofer IRB-Verl., pp. 146–151.

[16] Thrun S., Beetz M., Bennewitz M., Burgard W., Cremers A.B., Dellaert F., Fox D., Haehnel D., Rosenberg C., Roy N. et al. (2000) Probabilistic algorithms and the interactive museum tour-guide robot minerva. The International Journal of Robotics Research 19, pp. 972–999.

[17] Clodic A., Fleury S., Alami R., Chatila R., Bailly G., Brethes L., Cottret M., Danes P., Dollat X., Elisei F. et al. (2006) Rackham: An interactive robot-guide. In: ROMAN 2006-the 15th IEEE International Symposium on Robot and Human Interactive Communication, IEEE, pp. 502–509.

[18] Nourbakhsh I.R., Kunz C. & Willeke T. (2003) The mobot museum robot installations: A five year experiment. In: Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453), vol. 4, IEEE, vol. 4, pp. 3636–3641.

[19] Macaluso I., Ardizzone E., Chella A., Cossentino M., Gentile A., Gradino R., Infantino I., Liotta M., Rizzo R. & Scardino G. (2005) Experiences with cicerobot, a museum guide cognitive robot. In: Congress of the Italian Association for Artificial Intelligence, Springer, pp. 474–482.

[20] Hall E.T., Birdwhistell R.L., Bock B., Bohannan P., Diebold Jr A.R., Durbin M., Edmonson M.S., Fischer J., Hymes D., Kimball S.T. et al. (1968) Proxemics [and comments and replies]. Current anthropology 9, pp. 83–108.

[21] Hansen S.T., Svenstrup M., Andersen H.J. & Bak T. (2009) Adaptive human aware navigation based on motion pattern analysis. In: RO-MAN 2009-The 18th IEEE International Symposium on Robot and Human Interactive Communication, IEEE, pp. 927–932.

[22] Kirby R., Simmons R. & Forlizzi J. (2009) Companion: A constraint-optimizing method for person-acceptable navigation. In: RO-MAN 2009-The 18th IEEE International Symposium on Robot and Human Interactive Communication, IEEE, pp. 607–612.

[23] Svenstrup M., Bak T. & Andersen H.J. (2010) Trajectory planning for robots in dynamic human environments. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 4293–4298.

[24] Pacchierotti E., Christensen H.I. & Jensfelt P. (2005) Human-robot embodied interaction in hallway settings: a pilot user study. In: ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005., IEEE, pp. 164–171.

[25] Mori M. (1970) Bukimi no tani [the uncanny valley]. Energy 7, pp. 33–35.

[26] Arechavaleta G., Laumond J.P., Hicheur H. & Berthoz A. (2008) On the nonholonomic nature of human locomotion. Autonomous Robots 25, pp. 25–35.

[27] Gockley R., Forlizzi J. & Simmons R. (2007) Natural person-following behavior for social robots. In: Proceedings of the ACM/IEEE international conference on Human-robot interaction, pp. 17–24.

[28] Althaus P., Ishiguro H., Kanda T., Miyashita T. & Christensen H.I. (2004) Navigation for human-robot interaction tasks. In: IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004, vol. 2, IEEE, vol. 2, pp. 1894–1900.

[29] Hayashi K., Shiomi M., Kanda T., Hagita N. & Robotics A. (2012) Friendly patrolling: A model of natural encounters. In: Proc. RSS, p. 121.

[30] Pandey A.K. & Alami R. (2010) A framework towards a socially aware mobile robot motion in human-centered dynamic environment. In: 2010 IEEE/RSJ international Conference on Intelligent Robots and systems, IEEE, pp. 5855–5860.

[31] Helbing D. (1991) A mathematical model for the behavior of pedestrians. Behavioral science 36, pp. 298–310.

[32] Rios-Martinez J. (2013) Socially-aware robot navigation: combining risk assessment and social conventions. Hal. Inria Frace .

[33] Granata C. & Bidaud P. (2012) A framework for the design of person following behaviors for social mobile robots. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 4652–4659.

[34] Tadokoro S., Hayashi M., Manabe Y., Nakami Y. & Takamori T. (1995) On motion planning of mobile robots which coexist and cooperate with human. In: Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots, vol. 2, IEEE, vol. 2, pp. 518–523.

[35] Hoeller F., Schulz D., Moors M. & Schneider F.E. (2007) Accompanying persons with a mobile robot using motion prediction and probabilistic roadmaps. In: 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 1260–1265.

[36] Kushleyev A. & Likhachev M. (2009) Time-bounded lattice for efficient planning in dynamic environments. In: 2009 IEEE International Conference on Robotics and Automation, IEEE, pp. 1662–1668.

[37] Althoff D., Kuffner J.J., Wollherr D. & Buss M. (2012) Safety assessment of robot trajectories for navigation in uncertain and dynamic environments. Autonomous Robots 32, pp. 285–302.

[38] Martinez-Garcia E.A., Akihisa O. et al. (2005) Crowding and guiding groups of humans by teams of mobile robots. In: IEEE Workshop on Advanced Robotics and its Social Impacts, 2005., IEEE, pp. 91–96.

[39] Bennewitz M. (2004) Mobile robot navigation in dynamic environments. Ph.D. thesis, Freiburg (Breisgau), Univ., Diss., 2004.

[40] Ziebart B.D., Ratliff N., Gallagher G., Mertz C., Peterson K., Bagnell J.A., Hebert M., Dey A.K. & Srinivasa S. (2009) Planning-based prediction for pedestrians. In: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 3931–3936.

[41] Pandey A.K. & Alami R. (2009) A step towards a sociable robot guide which monitors and adapts to the person's activities. In: 2009 International Conference on Advanced Robotics, IEEE, pp. 1–8.

[42] Feil-Seifer D. & Matarić M. (2011) People-aware navigation for goal-oriented behavior involving a human partner. In: 2011 IEEE International Conference on Development and Learning (ICDL), vol. 2, IEEE, vol. 2, pp. 1–6.

[43] Bautin A., Martinez-Gomez L. & Fraichard T. (2010) Inevitable collision states: a probabilistic perspective. In: 2010 IEEE international conference on robotics and automation, IEEE, pp. 4022–4027.

[44] Fraichard T. (2007) A short paper about motion safety. In: Proceedings 2007 IEEE International Conference on Robotics and Automation, IEEE, pp. 1140–1145.

[45] Fox D., Burgard W. & Thrun S. (1997) The dynamic window approach to collision avoidance. IEEE Robotics & Automation Magazine 4, pp. 23–33.

[46] Chan N., Kuffner J. & Zucker M. (2008) Improved motion planning speed and safety using regions of inevitable collision. In: 17th CISM-IFToMM symposium on robot design, dynamics, and control, pp. 103–114.

[47] Hüttenrauch H., Eklundh K.S., Green A. & Topp E.A. (2006) Investigating spatial relationships in human-robot interaction. In: 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 5052–5059.

[48] Gómez J.V., Mavridis N. & Garrido S. (2013) Social path planning: Generic human-robot interaction framework for robotic navigation tasks. In: 2nd Intl. workshop on cognitive robotics systems: replicating human actions and activities.

[49] Bevilacqua P., Frego M., Bertolazzi E., Fontanelli D., Palopoli L. & Biral F. (2016) Path planning maximising human comfort for assistive robots. In: 2016 IEEE Conference on Control Applications (CCA), IEEE, pp. 1421–1427.

[50] Helbing D. & Molnar P. (1995) Social force model for pedestrian dynamics. Physical review E 51, p. 4282.

[51] Hanheide M., Peters A. & Bellotto N. (2012) Analysis of human-robot spatial behaviour applying a qualitative trajectory calculus. In: 2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication, IEEE, pp. 689–694.

[52] Mead R. & Matarić M.J. (2017) Autonomous human–robot proxemics: socially aware navigation based on interaction potential. Autonomous Robots 41, pp. 1189–1201.

[53] Lam C.P., Chou C.T., Chiang K.H. & Fu L.C. (2010) Human-centered robot navigation—towards a harmoniously human–robot coexisting environment. IEEE Transactions on Robotics 27, pp. 99–112.

[54] Banisetty S.B., Forer S., Yliniemi L., Nicolescu M. & Feil-Seifer D. (2021) Socially aware navigation: a non-linear multi-objective optimization approach. ACM Transactions on Interactive Intelligent Systems (TiiS) 11, pp. 1–26.

[55] He Z., Wang J. & Song C. (2021) A review of mobile robot motion planning methods: from classical motion planning workflows to reinforcement learning-based architectures. arXiv preprint arXiv:2108.13619 .

[56] Van den Berg J., Lin M. & Manocha D. (2008) Reciprocal velocity obstacles for real-time multi-agent navigation. In: 2008 IEEE international conference on robotics and automation, Ieee, pp. 1928–1935.

[57] Fiorini P. & Shiller Z. (1998) Motion planning in dynamic environments using velocity obstacles. The international journal of robotics research 17, pp. 760–772.

[58] Truong X.T. & Ngo T.D. (2017) Toward socially aware robot navigation in dynamic and crowded environments: A proactive social motion model. IEEE Transactions on Automation Science and Engineering 14, pp. 1743–1760.

[59] Tanner H.G. & Kumar A. (2005) Formation stabilization of multiple agents using decentralized navigation functions. In: Robotics: Science and systems, vol. 1, Boston, vol. 1, pp. 49–56.

[60] Luber M., Spinello L., Silva J. & Arras K.O. (2012) Socially-aware robot navigation: A learning approach. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, pp. 902–907.

[61] Liu Y., Yan Q. & Alahi A. (2021) Social nce: Contrastive learning of socially-aware motion representations. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 15118–15129.

[62] Finean M.N., Petrović L., Merkt W., Marković I. & Havoutis I. (2022) Motion planning in dynamic environments using context-aware human trajectory prediction. arXiv preprint arXiv:2201.05058 .

[63] Kato Y., Kanda T. & Ishiguro H. (2015) May i help you?-design of human-like polite approaching behavior. In: 2015 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI), IEEE, pp. 35–42.

[64] Naik L., Palinko O., Bodenhagen L. & Krüger N. (2021) Multi-modal proactive approaching of humans for human-robot cooperative tasks. In: 2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN), IEEE, pp. 323–329.

[65] Mizumaru K., Satake S., Kanda T. & Ono T. (2019) Stop doing it! approaching strategy for a robot to admonish pedestrians. In: 2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI), IEEE, pp. 449–457.

[66] Desai M., Tsui K.M., Yanco H.A. & Uhlik C. (2011) Essential features of telepresence robots. In: 2011 IEEE Conference on Technologies for Practical Robot Applications, IEEE, pp. 15–20.

[67] Tsui K.M., Norton A., Brooks D.J., McCann E., Medvedev M.S. & Yanco H.A. (2013) Design and development of two generations of semi-autonomous social telepresence robots. In: 2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA), IEEE, pp. 1–6.

[68] Tsui K.M., Norton A., Brooks D.J., McCann E., Medvedev M.S., Allspaw J., Suksawat S., Dalphond J.M., Lunderville M. & Yanco H.A. (2014) Iterative design of a semi-autonomous social telepresence robot research platform: a chronology. Intelligent Service Robotics 7, pp. 103–119.

[69] Escolano C., Antelis J.M. & Minguez J. (2011) A telepresence mobile robot controlled with a noninvasive brain–computer interface. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42, pp. 793–804.

[70] Suguitan M. & Hoffman G. (2021) You are (not) the robot: Variable perspective motion control of a social telepresence robot. In: Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems, pp. 1–4.

[71] Das B. & Dobie G. (2021) Delay compensated state estimation for telepresence robot navigation. Robotics and Autonomous Systems 146, p. 103890.

[72] Htike Z., Papaioannou G., Siampis E., Velenis E. & Longo S. (2021) Fundamentals of motion planning for mitigating motion sickness in automated vehicles. IEEE Transactions on Vehicular Technology .

[73] Mimnaugh K.J., Suomalainen M., Becerra I., Lozano E., Murrieta-Cid R. & LaValle S.M. (2021) Analysis of user preferences for robot motions in immersive telepresence. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp. 4252–4259.

[74] Mimnaugh K.J., Suomalainen M., Becerra I., Lozano E., Murrieta-Cid R. & LaValle S.M. (2021) Defining preferred and natural robot motions in immersive telepresence from a first-person perspective. arXiv preprint arXiv:2102.12719 .

[75] Suomalainen M., Mimnaugh K.J., Becerra I., Lozano E., Murrieta-Cid R. & LaValle S.M. (2021) Comfort and sickness while virtually aboard an autonomous telepresence robot. In: International Conference on Virtual Reality and Mixed Reality, Springer, pp. 3–24.

[76] Suomalainen M., Sakcak B., Widagdo A., Kalliokoski J., Mimnaugh K.J., Chambers A.P., Ojala T. & LaValle S.M. (2022) Unwinding rotations improves user comfort with immersive telepresence robots. arXiv preprint arXiv:2201.02392 .

[77] Hussein A., Gaber M.M., Elyan E. & Jayne C. (2017) Imitation learning: A survey of learning methods. ACM Computing Surveys (CSUR) 50, pp. 1–35.

[78] Kuderer M., Gulati S. & Burgard W. (2015) Learning driving styles for autonomous vehicles from demonstration. In: 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 2641–2646.

[79] Kim B. & Pineau J. (2016) Socially adaptive path planning in human environments using inverse reinforcement learning. International Journal of Social Robotics 8, pp. 51–66.

[80] Ramírez O.A.I., Khambhaita H., Chatila R., Chetouani M. & Alami R. (2016) Robots learning how and where to approach people. In: 2016 25th IEEE international symposium on robot and human interactive communication (RO-MAN), IEEE, pp. 347–353.

[81] Chen Y.F., Everett M., Liu M. & How J.P. (2017) Socially aware motion planning with deep reinforcement learning. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp. 1343–1350.

[82] Schmerling E., Leung K., Vollprecht W. & Pavone M. (2018) Multimodal probabilistic model-based planning for human-robot interaction. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 3399–3406.

[83] Chen C., Liu Y., Kreiss S. & Alahi A. (2019) Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning. In: 2019 International Conference on Robotics and Automation (ICRA), IEEE, pp. 6015–6022.

[84] Zeng H., Hu R., Huang X. & Peng Z. (2021) Robot navigation in crowd based on dual social attention deep reinforcement learning. Mathematical Problems in Engineering 2021.

[85] Ruan X., Ren D., Zhu X. & Huang J. (2019) Mobile robot navigation based on deep reinforcement learning. In: 2019 Chinese control and decision conference (CCDC), IEEE, pp. 6174–6178.

[86] Samak T.V., Samak C.V. & Kandhasamy S. (2020) Robust behavioral cloning for autonomous vehicles using end-to-end imitation learning. arXiv preprint arXiv:2010.04767 .

[87] Kollmitz M., Koller T., Boedecker J. & Burgard W. (2020) Learning human-aware robot navigation from physical interaction via inverse reinforcement learning. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp. 11025–11031.

[88] Smith T., Chen Y., Hewitt N., Hu B. & Gu Y. (2021) Socially aware robot obstacle avoidance considering human intention and preferences. International Journal of Social Robotics , pp. 1–18.

[89] Liu S., Chang P., Liang W., Chakraborty N. & Driggs-Campbell K. (2021) Decentralized structural-rnn for robot crowd navigation with deep reinforcement learning. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 3517–3524.

[90] Cui Y., Zhang H., Wang Y. & Xiong R. (2021) Learning world transition model for socially aware robot navigation. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 9262–9268.

[91] Majecka B. (2009) Statistical models of pedestrian behaviour in the forum. Master's thesis, School of Informatics, University of Edinburgh .

[92] Pellegrini S., Ess A., Schindler K. & Van Gool L. (2009) You'll never walk alone: Modeling social behavior for multi-target tracking. In: 2009 IEEE 12th international conference on computer vision, IEEE, pp. 261–268.

[93] Lerner A., Chrysanthou Y. & Lischinski D. (2007) Crowds by example. In: Computer graphics forum, vol. 26, Wiley Online Library, vol. 26, pp. 655–664.

[94] Zhou B., Wang X. & Tang X. (2012) Understanding collective crowd behaviors: Learning a mixture model of dynamic pedestrian-agents. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, pp. 2871–2878.

[95] Robicquet A., Sadeghian A., Alahi A. & Savarese S. (2016) Learning social etiquette: Human trajectory understanding in crowded scenes. In: European conference on computer vision, Springer, pp. 549–565.

[96] Ferryman J. & Shahrokni A. (2009) Pets2009: Dataset and challenge. In: 2009 Twelfth IEEE international workshop on performance evaluation of tracking and surveillance, IEEE, pp. 1–6.

[97] Contributor B. (2016), Who is the spy (talking game). URL: `https://busyteacher.org/24039-who-is-the-spy-talking-game.html`.

[98] pandas development team T. (2020), pandas-dev/pandas: Pandas. URL: `https://doi.org/10.5281/zenodo.3509134`.

[99] Bellman R. (1957) A markovian decision process. Indiana Univ. Math. J. 6, pp. 679–684.

[100] Wikipedia (2022), Markov decision process — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Markov%20decision%20process&oldid=1091621976. [Online; accessed 13-June-2022].

[101] Cortes C. & Vapnik V. (1995) Support-vector networks. Machine learning 20, pp. 273–297.

[102] Myung I.J. (2003) Tutorial on maximum likelihood estimation. Journal of mathematical Psychology 47, pp. 90–100.

[103] Kingma D.P. & Ba J. (2014) Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 .

[104] Paszke A., Gross S., Massa F., Lerer A., Bradbury J., Chanan G., Killeen T., Lin Z., Gimelshein N., Antiga L. et al. (2019) Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems 32.

[105] Koilias A., Nelson M.G., Anagnostopoulos C.N. & Mousas C. (2020) Immersive walking in a virtual crowd: The effects of the density, speed, and direction of a virtual crowd on human movement behavior. Computer Animation and Virtual Worlds 31, p. e1928.

[106] Bahdanau D., Brakel P., Xu K., Goyal A., Lowe R., Pineau J., Courville A. & Bengio Y. (2016) An actor-critic algorithm for sequence prediction. arXiv preprint arXiv:1607.07086 .