Learning to Navigate in Mixed Human-Robot Crowds via an Attention-Driven Deep Reinforcement Learning Framework

Ibrahim Khalil Kabir , Muhammad Faizan Mysorewala * , Yahya Osais , Ali Nasir

Posted Date: 26 September 2025

doi: 10.20944/preprints202509.2260.v1

Keywords: crowd navigation; deep reinforcement learning; human-robot interaction; robot navigation; socially aware navigation; knowledge extraction; attention mechanism

Article

Learning to Navigate in Mixed Human-Robot Crowds via an Attention-Driven Deep Reinforcement Learning Framework

Ibrahim K. Kabir [1,2], Muhammad F. Mysorewala [1,2,3,*], Yahya I. Osais [4,5,6] and Ali Nasir [1,3]

[1] Control and Instrumentation Department, King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, 31261 Saudi Arabia
[2] Interdisciplinary Research Center for Smart Mobility and Logistics, King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, 31261 Saudi Arabia
[3] Interdisciplinary Research Center for Intelligent Manufacturing and Robotics, King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, 31261 Saudi Arabia
[4] Information & Computer Science Department, King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, 31261 Saudi Arabia
[5] Interdisciplinary Research Center for Intelligent Secure Systems, King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, 31261 Saudi Arabia
[6] Interdisciplinary Research Center for Communication Systems and Sensing, King Fahd University of Petroleum & Minerals (KFUPM), Dhahran, 31261 Saudi Arabia
* Correspondence: mysorewala@kfupm.edu.sa

Abstract

The rapid growth of technology has introduced robots into daily life, necessitating navigation frameworks that enable safe, human-friendly movement while accounting for social aspects. Such methods must also scale to situations with multiple humans and robots moving simultaneously. Recent advances in Deep Reinforcement Learning (DRL) have enabled policies that incorporate these norms into navigation. This work presents a socially aware navigation framework for mobile robots operating in environments shared with humans and other robots. The approach, based on single-agent DRL, models all interaction types between the ego robot, humans, and other robots. Training uses a reward function balancing task completion, collision avoidance, and maintaining comfortable distances from humans. An attention mechanism enables the framework to extract knowledge about the relative importance of surrounding agents, guiding safer and more efficient navigation. Our approach is tested in both dynamic and static obstacle environments. To improve training efficiency and promote socially appropriate behaviors, Imitation Learning is employed. Comparative evaluations with state-of-the-art methods highlight the advantages of our approach, especially in enhancing safety by reducing collisions and preserving comfort distances. Results confirm the effectiveness of our learned policy and its ability to extract socially relevant knowledge in human-robot environments where social compliance is essential for deployment.

Keywords: crowd navigation; deep reinforcement learning; human-robot interaction; robot navigation; socially aware navigation; knowledge extraction; attention mechanism

1. Introduction

The field of robotics has advanced rapidly in recent decades with robots being increasingly deployed in human environments such as factory floors, restaurants, airports, and hospitals [1]. In shared spaces with humans, robots must avoid collisions and move in ways that do not cause discomfort or harm to nearby people. Human navigation in shared spaces is governed by social conventions or norms, which assist people in understanding intentions and moving in ways that ensure mutual comfort [2]. Robot navigation in the presence of humans, termed socially aware navigation or crowd navigation, applies these rules as well as proxemics [3], which defines how space

 doi:10.20944/preprints202509.2260.v1

is regulated when interacting and navigating. As a requirement, for navigation to be considered socially aware, it must take into account comfort, naturalness, and sociability [4].

Existing works in Socially Aware Navigation (SAN) have generally been categorized into three areas: reaction-based methods, proactive methods, and learning-based methods [5]. Reaction-based methods involve robots reacting with other mobile agents using one-step interaction approaches [6]. They can be considered as purely rule-based, without any prediction or learning happening. The Velocity Obstacle (VO) [7], Optimal Reciprocal Collision Avoidance (ORCA)[8], and Social Force Model (SFM) [9] are such methods. However, they are heavily dependent on mobility models and capture only simple interactions. Moreover, planning is short-sighted and often unnatural. Proactive methods, often called trajectory-based methods, predict the behavior of humans through their trajectories and plan a path to follow. These methods have smoother interactions within human social norms [10–12]. Unfortunately, the computational cost of predicting all possible paths in these methods becomes very high, especially when the state space expands with increasing crowd size.

Learning-based methods have shown promise in recent years, specifically Reinforcement Learning (RL) and Deep Reinforcement Learning (DRL) approaches. The Collision Avoidance with DRL (CADRL) [14] was one of the earliest methods to use DRL for collision avoidance in robot navigation by replacing hand-crafted techniques with DRL. The authors then extended it in [15] to consider social norms and learn socially compliant behavior. However, the over-reliance on reciprocal assumption led to overly conservative behaviour with the effectiveness being severely reduced in complex scenarios. In [16], LSTM-RL was developed by using the Long-Short Term Memory (LSTM) to encode other agents and model the collective impact of the crowd. It sequentially processes the current states of each neighbor in reverse order of proximity to the robot. In large-scale crowds, it is necessary to identify the most critical individuals for navigation and learn accurate representations of the crowd.

Various studies have proposed different approaches such as, Graph Convolutional Networks (GCN) [33], Graph Attention Network (GAT) [34], Self-Attention Mechanisms [35], and Transformers [36]. Some methods employ Imitation Learning (IL) to improve sample and training efficiency by initializing training with a set of expert navigation demonstrations in a manner similar to supervised learning [17,18,35]. This is then used to obtain a navigation model based on the training data.

Multiple mobile robots may navigate in an environment cooperatively with the same goal, or with different goals. Some of these robots may be designed to follow social norms and be socially aware. It is therefore important to design navigation frameworks that help mobile robots move through crowded environments with both people and robots. Frameworks must be developed to accurately model different interaction types in environments with multiple humans and robots. For the case of multiple robots in the same environment, the robots must be able to differentiate between themselves and humans. Most methods that explore Multirobot SAN treat each robot as a separate RL agent and model the framework as a Multi-Agent Reinforcement Learning (MARL) framework with focus on cooperative navigation [20–22]. This approach, however, has serious challenges which include: (1) Non-stationarity resulting from continually changing policies of the agents during their learning processes, (2) Dimensional explosion with increasing number of agents when scaling, (3) Learning optimal policies in the MARL system [19]. Additionally, methods like these do not model the diversity in the interaction types, and thus model all robots in the same way.

Given all these drawbacks and challenges, we therefore propose a Socially Aware Navigation framework based on single-agent Deep Reinforcement Learning with Imitation-Learning based pre-training and initialization for navigation of multiple robots within the presence of humans. The main contributions of this work are as follows:

- A navigation framework based on single-agent deep reinforcement learning, which allows the ego robot to move according to social norms among humans and other robots that follow a predefined collision avoidance behaviour.
- The use of imitation learning for socially aware single agent navigation in environments shared other robots and humans.

 doi:10.20944/preprints202509.2260.v1

- A model of the full social awareness for the ego robot and partial social awareness for other robots to reflect realistic interactions, enabling the extraction of socially relevant knowledge from human-robot interactions.

The rest of the paper is organized as follows. Section 2 discusses the related work, Section 3 presents and discusses the methodology. Section 4 presents the obtained results along with discussions. Finally, Section 5 gives a conclusion of this research.

2. Related Work

The earliest works in this field all consider a single robot navigating in presence of humans in a socially aware manner. Robots primarily rely on their perception and reasoning capabilities to interpret human behavior and predict future human actions yet lack feedback from the crowd. These methods often treat robot navigation in social environments as a one-way human-robot interaction problem, which could pose challenges as the size of the crowd increases. To accurately model robot navigation in human environments as a two-way human-robot interaction problem, [35] developed the Social Attention Reinforcement Learning (SARL) approach which explicitly models interactions between the robot and the crowd, specifically human-human and human-robot interactions. It uses a self-attention mechanism to find the aggregate impact of the crowd and prioritizes most relevant interactions. It also incorporates imitation learning for initialization of training using ORCA [8]. Building upon this work, [37] proposed Social Obstacle Avoidance using Deep Reinforcement Learning (SOADRL) which processes static and dynamic obstacle information independently, effectively solving the difficulty of navigating with sensors only given a restricted field of view within crowds.

In [23], the most important individuals with respect to navigation are highlighted within the crowd. The work in [24] suggested simulating danger zones for the robot in order to react in real time to human actions. The definition of these zones is based on tracking human behavior in real-time and then storing all the potential actions that individuals may do at any given moment. The robot learns to navigate away from these danger zones so as to achieve dependable and safe navigation.

Multirobot approaches in SAN have primarily focused on multi-agent cooperative navigation within Multi-Agent RL frameworks. These works have essentially developed frameworks with Centralized Training and Decentralized Execution (CTDE) of the agents and trained policies respectively [22,26–28]. These methods treat each robot as an RL agent with its separate action, observations, and rewards. In [22], a multirobot Multi-agent RL approach was developed. It employed separate hybrid spatio-temporal transformers to model both human-robot and robot-robot interactions then used a multi-modal transformer to align and fuse these features, creating a comprehensive state representation that includes both human-robot and robot-robot interactions. It also used an attention mechanism to determine the importance of each agent relative to others, enhancing the robots ability to make socially-aware decisions.

The approach in [26], uses an Actor-Critic RL approach, with cooperative planning, using a Temporal-Spatial graph-based encoder to obtain the significance of social interactions. It also incorporates a K-step look ahead reward mechanism to allow robots to consider future consequences hence avoiding aggressive short-sighted decisions. Robot-robot interactions are captured using a centralized value-network that shares necessary information about each robot needed to facilitate accurate decision-making. While [21] similarly uses the CTDE paradigm, it models interactions differently. It uses a Graph Neural Network (GNN) to model interactions between robots and humans combining their positions in the field of view to get an accurate representation of the environment. Then an edge selector mechanism is used to identify the most important interactions between agents by creating a sparse graph and applying attention on the nodes. The authors in [29] proposed Heterogeneous Relational Deep Reinforcement Learning (HeR-DRL), which is a single agent Multirobot crowd navigation framework. It uses a two-layer heterogeneous GNN to transfer and aggregate information among the robots and humans navigating in the environment and used separate subgraphs to model the relationships between the different types of robots and humans. It however modelled robot-robot

 doi:10.20944/preprints202509.2260.v1

interactions same as human-robot interactions and modelled only the controlled robot which is the RL agent with capability of social awareness and all other robots in the environment with no social awareness.

3. Methodology

3.1. Assumptions

This work is based upon some fundamental assumptions which are presented below:

- This work is based on a Single agent deep reinforcement learning framework where an agent learns in an environment that includes other robots and humans as part of the environment.
- The robot that learns the optimal policy is called the ego robot while the other robots in the environment are referred to as the other robots.
- The ego robot has a full view of the environment while the other robots have a partial view of the environment.
- All the robots are modeled as holonomic, i.e. they can move in any direction instantly without rotation constraints.
- Humans and other robots in the environment follow the ORCA policy.
- There is no explicit communication of navigation intent between ego robot, other robots and humans, they can only observe the states of each other.
- The navigation is modeled as point-to-point navigation scenarios in Two-Dimensional (2D) plane.

3.2. Problem Formulation

Many real-world applications only require robot navigation in only two dimensions. Their navigation objective is to determine the most efficient and safe route from the initial position to the target location. The problem of a robot navigating in a 2D Euclidean space environment with multiple humans and multiple robots with social considerations can be formulated as a Markov Decision Process (MDP) within an RL framework. The MDP is defined by the tuple $\langle S, A, P, R, \gamma \rangle$, where S is the state space, A the action space, P is the transition probability function, R is the reward function, and γ is the discount factor. Together, these elements form the foundation for the agent's ability to make decisions and learn within the environment. By selecting an optimal action in a given state, the agent maximizes the cumulative reward and the algorithm steadily improves its performance.

3.2.1. State Space

The state space comprises all the environment information that the agent can observe. This includes the states of the humans, other robots, and ego robot. The state space S includes the states of the ego robot, humans, and other robots. It comprises of the position $p = [p_x, p_y]$, velocity $v = [v_x, v_y]$, and radius of each agent along with the heading angle, preferred velocity, and goal position p_g of the ego robot. The state of the ego robot at time is given by

$$s_t^{er} = [d_g, v_{pref}, \theta, r, v_x, v_y] \tag{1}$$

where $d_g = ||p - p_g||_2$ is the distance of the ego robot to the goal, v_{pref} is the preferred velocity of the ego robot, θ is the heading angle of the ego robot, r is the radius of the robot, v_x, and v_y are the velocity components of the ego robot. The state of the i-th human at time t is denoted as

$$s_t^{h_i} = [p_x^{h_i}, p_y^{h_i}, v_x^{h_i}, v_y^{h_i}, r^{h_i}, d_a^{h_i}, r^{h_i} + r, c^{h_i}] \tag{2}$$

where $p_x^{h_i}$,and $p_y^{h_i}$ are components of the position of the human, $v_x^{h_i}$ and $v_y^{h_i}$ are the velocity components, r^{h_i} is the radius of the human, and $d_a^{h_i}$ is the distance between the ego robot and the human. Similarly, the state of the j-th other robot at time t is given by

$$S_t^{or_j} = [p_x^{or_j}, p_y^{or_j}, v_x^{or_j}, v_y^{or_j}, r^{or_j}, d_a^{or_j}, r^{or_j} + r, c^{or_j}] \tag{3}$$

 doi:10.20944/preprints202509.2260.v1

where $p_x^{or_j}, p_y^{or_j}$ are the components of the position of the other robot, $v_x^{or_j}, v_y^{or_j}$ are velocity components of the other robot, r^{or_j} is the radius of the other robot, and $d_a^{or_j}$ is the distance between the other robot and the ego robot. As in [29] we also define a category flag c^* which is 1 if human and 0 if it is an Other robot to help identify the type of agent in the joint state and hence the type of interaction.

We utilize the above state information to construct the joint state, s_t^{jn}, which is a combination of the states of the ego robot, humans, and other robots. To simplify computation and reduce redundancy, the ego robot is set at the origin of a polar coordinate system, and the x-axis becomes directed toward the goal position, which is always relative to the position of the ego robot.

3.2.2. Action Space

The action space encompasses the full range of actions the agent can perform. In this case, the action space is discrete, and allows the ego robot to select any combination of speed and direction for smooth and unrestricted movement. We use holonomic kinematics for the robots, and assume that ego robot's velocity can be immediately reached after receiving the action command. We use a large action space to allow more flexible action selection while appropriately limiting it to prevent too much computational burden. We use a two-dimensional action space $[v_t, \phi_t]$, where v_t represents the velocity at time t, taking 5 discrete values uniformly distributed between 0 and v_{pref}, and ϕ_t denotes the heading angle at time t taking 16 discrete values uniformly distributed between 0 and 2π. The final combination then yields a total of 80 discrete action possibilities: A speed command $v_t \in [0, v_{pref}]$, and a heading angle command $\phi_t \in [0, 2\pi]$.

3.2.3. Reward Function

The reward function assigns a numerical score (reward or penalty) to the agent's actions based on how desirable those actions are in a given state. We selected a reward function that will help in achieving social awareness in the navigation framework. Our reward framework includes an arrival reward, a collision penalty, an uncomfortable distance penalty, and a time-based penalty (set as zero).

The reward function used is given below and was adapted from [35]. Let H be the set of humans and OR be the set of other robots. The expression for the cumulative or total reward R_T is expressed as

$$R_T = R_g + R_c + R_d \tag{4}$$

where R_g is the reward for navigating to the goal, R_c is the penalty for collisions, and R_d is the discomfort penalty which only applies to interactions with humans. These are all individually defined below.

$$R_g = \begin{cases} 1, & \text{if } p_t = p_g \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

$$R_c = \begin{cases} -0.25, & \text{if } d_a^i < r + r^i \\ 0, & \text{otherwise} \end{cases} \quad \text{for all } i \in H \cup OR. \tag{6}$$

$$R_d = \sum_{h \in H} f(d_{min}^h, d_{th}^H) \tag{7}$$

$$f(d_{min}, d_{th}) = \begin{cases} d_{min} - d_{th}, & \text{if } d_{min} < d_{th} \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

where d_g is the ego robot distance to the goal, and d_a^i is the ego robot distance from another agent i at timestep t. Because the comfort term applies only to humans, $d_{th}^H = 0.2m$ is the threshold distance, below which humans will feel discomfort, and d_{min} is the assigned value of minimum distance of separation between the ego robot and the j-th human at timestep t. For Other robots, $d_{th}^O R = 0$ by design, so they contribute nothing to R_d.

 doi:10.20944/preprints202509.2260.v1

3.2.4. Optimal Policy and Value Function

The value function V_s represents the expected total reward an agent can accumulate by being in a particular state and following a certain policy afterwards. In this work, the goal is to find the optimal policy and an optimal value function $V^*(s_t^{jn})$, which is given by

$$V^*(s_t^{jn}) = \max_a \mathbb{E}_{s'}\left[r(s_t^{jn}, a_t) + \gamma V^*(s_t^{jn'})\right] \tag{9}$$

where γ is a value between 0 and 1 (but closer to 1) termed the discount factor, and $r(s_t^{jn}, a_t)$ is the reward for taking action a_t in the joint state. The optimal policy π^*, which maps the joint state s_t^{jn} to an action a_t is given by

$$\pi^*(c) = \arg\max_{a_t} Q^*(s_t^{jn}, a_t) \tag{10}$$

where s_t^{jn} is the current joint state and a_t is the chosen action.

3.3. Interaction Modeling

Five distinct interaction types exist in the environment are: Human-Human, Human-Ego Robot, Ego Robot-other robot, Human-other robot, and other robot-other robot interactions, and are shown in Figure 1, and their distinct relationships are shown in an agent interaction matrix in Table 1.

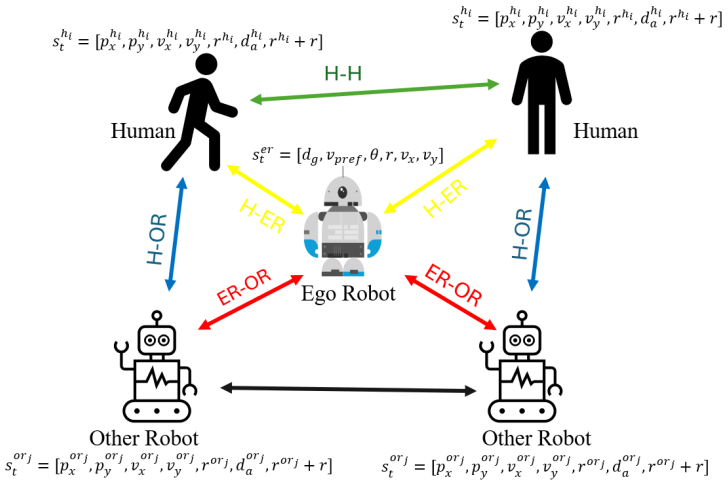

Figure 1. Categories of agents and their interaction relationship.

3.3.1. Human-Human Interaction

Human agents are modeled using the ORCA policy, which enables reciprocal collision avoidance based on local sensing and velocity adjustments. Each human responds reactively to the motion of nearby humans, maintaining safe distances and navigating toward individual goals while avoiding collisions. This interaction captures the decentralized and socially compliant nature of human navigation in shared spaces.

3.3.2. Human-Ego Robot Interaction

The ego robot employs a reinforcement learning policy trained to exhibit socially aware behavior. It perceives and interprets the state of surrounding humans, including position, velocity, and relative orientation, to make navigation decisions that align with human social norms. The policy accounts for proxemic constraints and collision risk, allowing the robot to adjust its trajectory appropriately in the presence of humans.

3.3.3. Ego Robot-Other robot Interaction

While the ego robot is the only agent trained via reinforcement learning, other robots in the environment are modeled using the ORCA policy. These other robots are treated as dynamic obstacles

 doi:10.20944/preprints202509.2260.v1

by the ego robot, so it does not enforce avoiding entering the personal space or discomfort zone of the other robots, rather it only does collision avoidance, because robots are not social beings so do not experience discomfort.

3.3.4. Human-Other Robot Interaction

Interactions between humans and other robots are governed by the same reciprocal collision avoidance principles as human–human interactions. There is no social awareness in this interaction, only purely collision avoidance. However, other robots are designed to maintain slightly larger safety margins when navigating around humans. This ensures that their presence in the environment reflects behaviour of robots.

3.3.5. Other Robot-Other Robot Interaction

Interactions among the other robots follows standard ORCA dynamics. Each robot independently computes its velocity based on the positions and velocities of its neighbors, including other robots. These interactions are symmetric and reciprocal, enabling decentralized collision avoidance without explicit communication. Although these robots are not trained, their behavior is sufficient to support socially aware motion of other agents in the environment.

Table 1. Agent interaction behaviour matrix.

Agent	In presence of Human	In presence of Ego Robot	In presence of other robot
Human	Reciprocal collision avoidance	Reciprocal collision avoidance	Reciprocal collision avoidance
Ego Robot	Implements learned DRL policy with social awareness	-	Reciprocal collision avoidance (no social awareness)
Other Robot	Reciprocal collision avoidance with larger safety margin	Reciprocal collision avoidance (no social awareness)	Reciprocal collision avoidance

3.4. System Architecture

Our framework is designed with three modules and designed to address socially aware navigation in multi-robot scenarios. It comprises an Interaction Module, a Pooling Module, and a Planning Module as shown in the overall architecture in Figure 2. While the modular principle is conceptually related to [35], our formulation is specifically adapted to capture both human–robot and robot–robot interactions through unique input representations, modified attention computations, and a training strategy that supports navigation of multiple robots within a single-agent DRL formulation. These modules are discussed in detail below.

 doi:10.20944/preprints202509.2260.v1

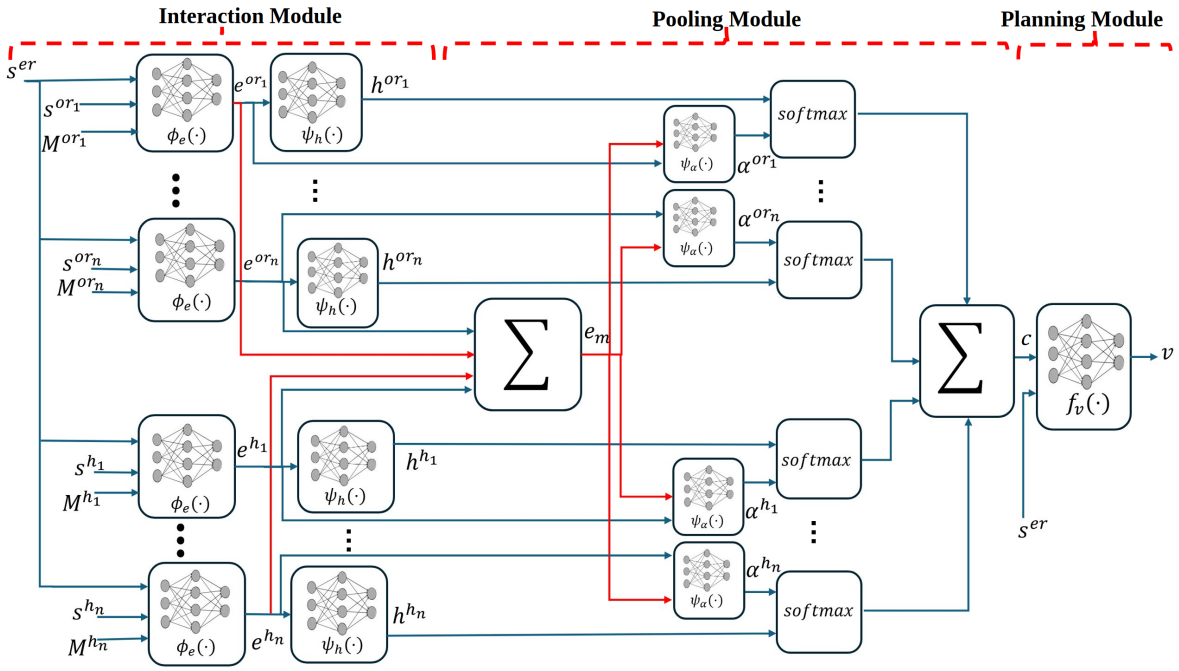

Figure 2. System architecture of the ego robot learning framework.

3.4.1. Interaction Module

All entities in the environment (human, ego-robot, other robot) have an impact on each other, and it would be too computationally expensive and to model all these interactions individually. Local maps were created to obtain representation for the Human-Human, Human-other robot, and other robot-other robot interactions, and a pairwise interaction module was used for the Human-Ego Robot and Ego Robot-other robot interactions. To capture the presence and velocities of neighbours in a neighbourhood of size L, Figure 3 shows a $L \times L \times 3$ map tensor M^{*i} centred at each agent (with $*$ representing *or* for other robot and h for human). This map is referred to as the local map and is given by

$$M^{*i}(a, b, :) = \sum_{j \in \mathcal{N}_i} \delta_{ab}\left[x_j - x_i,\, y_j - y_i\right] \xi'_j \tag{11}$$

where $\xi'_j = (v_{x_j}, v_{y_j}, 1)$ is a local state vector for agent j, and $\delta_{ab}\left[x_j - x_i,\, y_j - y_i\right]$ is an indicator function.

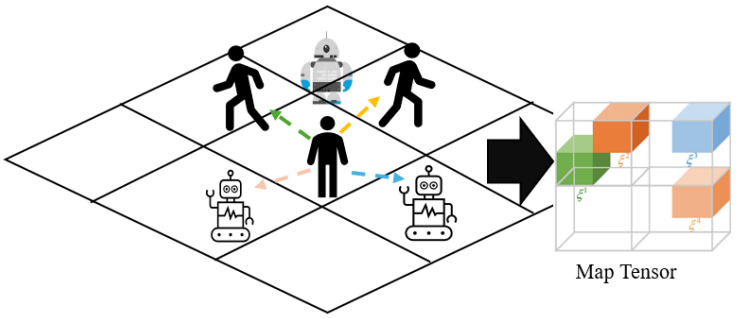

Figure 3. Creation of Local Map Tensor.

Using a Multi-Layer Perceptron (MLP), we encode the state of the ego robot, humans, other robots, and the map tensor M^{or_i} or M^{h_i} into a fixed-length vector e^{or_i} for the interaction of an 'other robot' or e^{h_i} for interaction of a human respectively. They are given as

$$e^{or_i} = \phi_e(s^{er}, s^{or_i}, M^{or_i}; W_e) \tag{12}$$

 doi:10.20944/preprints202509.2260.v1

$$e^{h_i} = \phi_e(s^{er}, s^{h_i}, M^{h_i}; W_e) \tag{13}$$

where W_e denotes the embedding weights and $\phi_e(\cdot)$ is a function with ReLU activations for embedding the states, weights, and map tensor. The interaction feature between the ego robot and a human or other robot is obtained by feeding the embedding vectors e^{or_i} and e^{h_i} to subsequent MLPs

$$h^{or_i} = \psi_h(e^{or_i}; W_h) \tag{14}$$

$$h^{h_i} = \psi_h(e^{h_i}; W_h) \tag{15}$$

where $\psi_h(\cdot)$ is another layer with ReLU activation function, and W_h denotes the weight of this MLP.

3.4.2. Pooling Module

As part of this module, we incorporate a self-attention mechanism to model the importance of each agent in the environment in relation to the ego robot. This module enhances social awareness because the nearest other robot or human is not necessarily the most important in terms of navigation, especially if its direction and speed are not in conflict with that of the ego robot. The obtained interaction vectors are pooled then operated on to obtain attention score α^{h_i} and α^{or_i} for a human and other robot respectively. They are obtained as follows

$$e_m = \frac{1}{n}\sum_{k=1}^{n} e^{or_n} + \frac{1}{n}\sum_{k=1}^{n} e^{h_n} \tag{16}$$

$$\alpha^{or_i} = \psi_\alpha(e^{or_i}, e_m; W_\alpha) \tag{17}$$

$$\alpha^{h_i} = \psi_\alpha(e^{h_i}, e_m; W_\alpha) \tag{18}$$

where W_α represents the weights, $\psi_\alpha(\cdot)$ is a Multi-Layer Perceptron with Rectified Linear Unit activations, and e_m is a vector with a bounded length that results from taking the mean after pooling all vectors of interactions.

In the final stage of crowd representation, we employ the previously obtained pairwise interaction vectors and their accompanying attention scores to obtain a linear weighted sum of all the pairs which make up the crowd representation.

$$c = \sum_{i=1}^{n} \text{softmax}(\alpha^{or_n}) h^{or_n} + \sum_{i=1}^{n} \text{softmax}(\alpha^{h_n}) h^{h_n} \tag{19}$$

3.4.3. Planning Module

This module determines the value of the network through obtaining the value of the planning state v. The basis for this is from the previously established model of the setting and crowd dynamics and it is given by

$$v = f_v(s^{er}, c; W_v) \tag{20}$$

where $f_v(\cdot)$ is a Multilayer Perceptron with ReLU activations with weights are denoted by W_v.

4. Results

4.1. Training Setup

We initialized the training of our framework with Imitation Learning (IL) as a gateway to deep reinforcement learning (DRL) to enable stable policy learning from the beginning. We simulated 2000 navigation episodes using the ORCA policy to generate the state-action pairs representing the behavior of the ego robot navigating within the environment. The generated trajectories served as data for the supervised learning, with the training objective being reduction of cross-entropy loss between predicted actions obtained through training and expert actions across all states. By using this method,

 doi:10.20944/preprints202509.2260.v1

the ego robot policy learned to mimic the conservative and reciprocal actions displayed by the ORCA expert.

In the second phase of the training with DRL, trajectories were sampled in batches, and rewards were computed according to the function in Section III-B. Updates to the policy were done with help of Adam Optimizer, and the discount factor was selected to balance between short-tern collision avoidance and long-term goal-reaching. While an exploration factor that decayed from 0.5 to 0.1 was enforced to gradually encourage convergence to stable, socially aware behaviour.

The hyperparameters summarized in Table 2 were chosen based on a survey of values chosen in literature and finetuning to select those with best performance. The learning rate of 0.001 proved gradual convergence and the batch size of 100 determined the trade-off between computational cost and the stability of gradient estimates. We chose a discount factor of 0.9 to promote long-term planning and valuing of future rewards. The hidden units listed in Table 2 define the architecture of the multilayer perceptrons used in the different modules of the framework. These sizes were selected to balance between representational capacity and training stability. The number of training episodes for the reinforcement learning phase was chosen to obtain stable training in the least amount of episodes. In the training phases of the framework, we used 5 humans and 2 Other robots setup. We tested the same configuration as well as others in the different scenarios applied in this work.

The Details for the implementation of this work including the training parameters, configuration parameters, and environment parameters are presented in Table 2. The DRL algorithm was implemented using PyTorch library [30] of Python in OpenAI Gym [31] and trained using Adam optimizer [32]. We use a square crossing simulation setup for both training and testing the navigation algorithm. In this scenario all the agents, apart from the ego robot are randomly initialized either in the left or right halves of the environment, and their goal positions similarly randomly selected at the opposite side of the start position. For the ego robot its start and goal position are predefined.

Table 2. Training and configuration parameters.

Parameter	Value
Preferred velocity	1.0 m/s
Radius of all agents	0.3 m
Discomfort distance for humans	0.2 m
Hidden units of $\phi_e(\cdot)$	150, 100
Hidden units of $\psi_h(\cdot)$	100, 50
Hidden units of $\psi_\alpha(\cdot)$	100, 100
Hidden units of $f_v(\cdot)$	150, 100, 100
IL training episodes	2000
IL epochs	50
IL learning rate	0.01
RL learning rate	0.001
Discount factor	0.9
Training batch size	100
RL training episodes	6000
Exploration rate in first 4000 episodes	0.5 to 0.1

4.2. Qualitative Results

To test the effectiveness of the developed algorithm and its generalization capabilities we have tested it in multiple simulation scenarios which include a varying number of humans and other robots and environment with static obstacles.

4.2.1. Open Space Scenario

In this scenario we simulate an open environment with no obstacles. This scenario involves the ego robot and a combination of humans and other robots each with their goals. The ego robot is initialized from position $(0, -4)$, and the target is set at $(0, 4)$ which the ego robot must navigate to in a

 doi:10.20944/preprints202509.2260.v1

socially aware manner in the presence of the humans and other robots. We test this scenario across a diverse range of human and other robots setups to show the generalization ability of the developed framework. We present this in the following subsections.

a. 5 Humans and 2 other robots

We tested the trained algorithm in a scenario of 5 Humans and 2 other robots as shown in Figure 4. The ego robot is initialized with its own self state vector which encodes its position, preferred velocity, and heading angle. The action that is selected as the outcome of the algorithm is a velocity and heading command. A scalar reward value is obtained based on the reward function, and the next state is observed after the action has been taken to continue the loop.

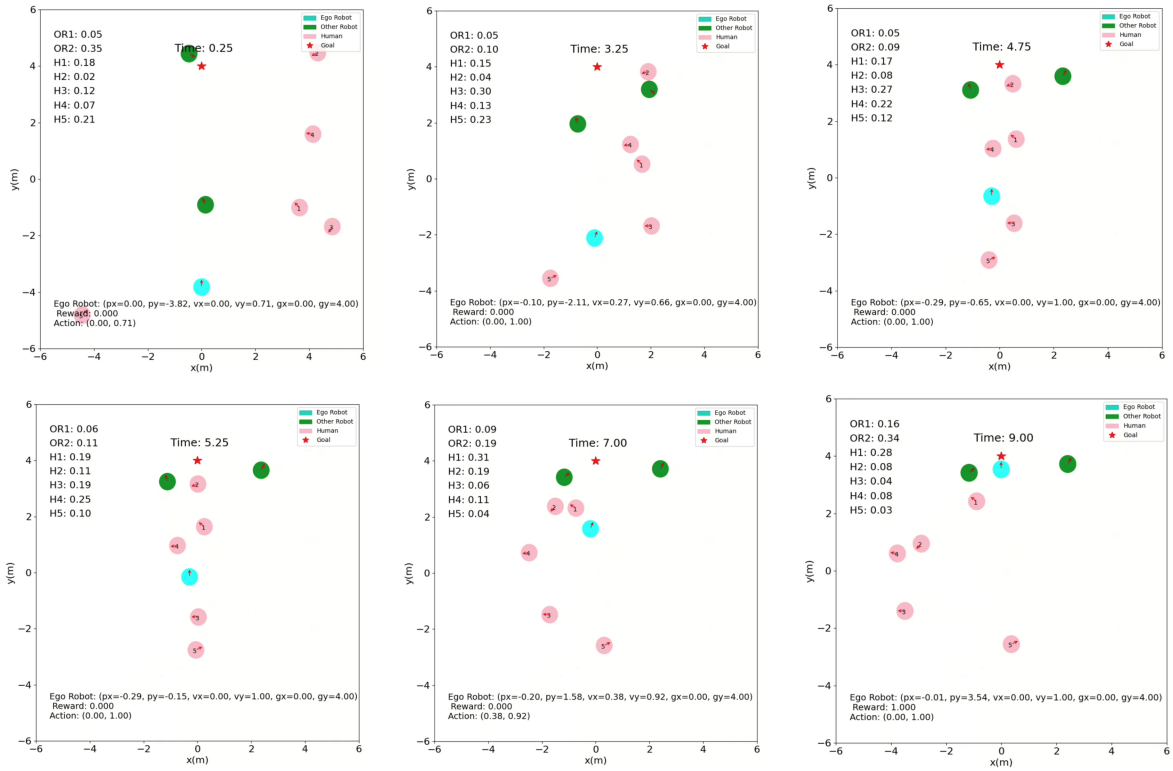

Figure 4. Simulation Scenario with 5 humans and 2 other robots at time steps of 0.25s, 3.25s, 4.75s, 5.25s, 7s, and 9s.

From the onset ($t = 0s$), the ego robot identifies a feasible trajectory that avoids early collisions with humans or other robots. As the simulation progresses ($t = 3.25s$ to $t = 4.75s$), the ego robot ventures more into the crowded area, and slows down and allows human 4 and human 1 to pass at $t = 5.25s$, and $t = 7s$ respectively. The ego robot maintains a safe space and utilizes the attention of the agents to weigh the priority of each agent. In the final timestep ($t = 9s$), the ego robot completes its path to the goal by efficiently passing between two other robots while preserving smooth and collision-free navigation.

b. 10 Humans and 3 Other robots

Figure 5 shows the scenario of 10 humans with 3 other robots. This scenario is a more densely populated and complex environment. The density of both humans and other robots imposes tighter constraints on the available free space.

 doi:10.20944/preprints202509.2260.v1

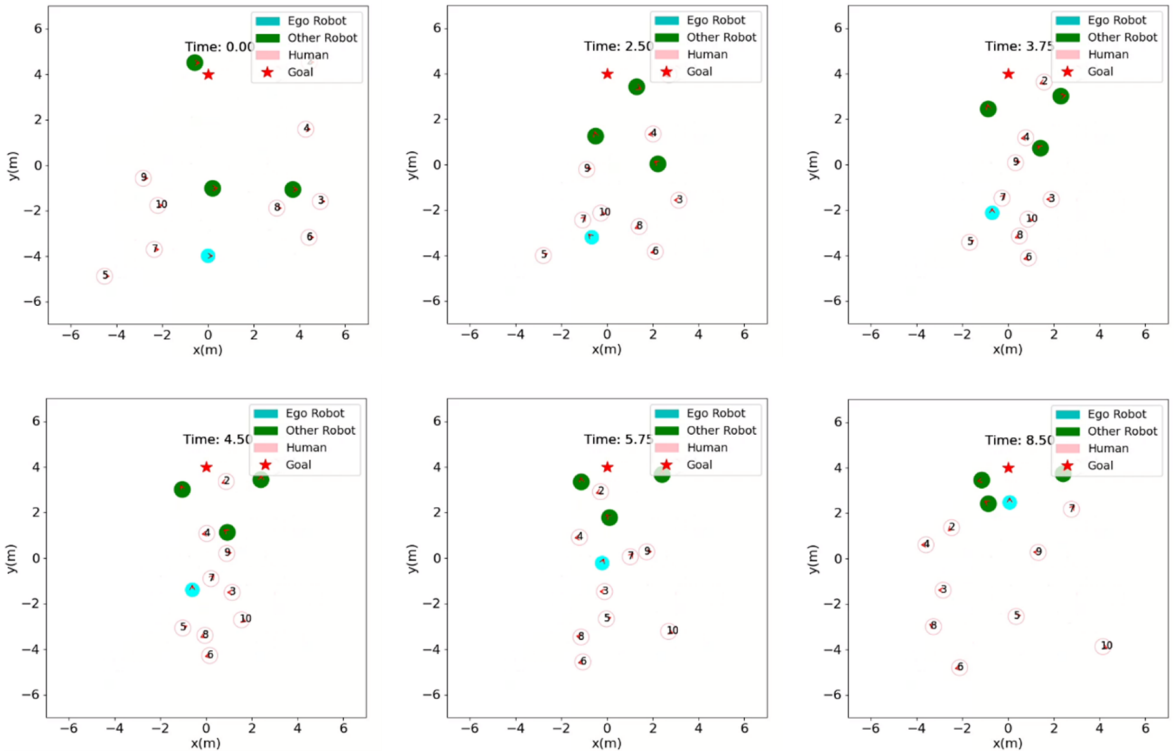

Figure 5. Simulation Scenario with 10 humans and 3 other robots at time steps of 0s, 2.5s, 3.75s, 4.5s, 5.75s, and 8.5s.

As the ego robot begins to advance ($t = 2.5s$), it dynamically adjusts its trajectory, and by $t = 3.75s$ to $t = 5.75s$, it can be seen socially avoiding humans 3, 4, and 8, as well as responding to the trajectory of an other robot approaching it head-on. Between $t = 5.75s$ to $t = 8.5s$, the robot has moved in between agents to reach toward its goal. Figure 6 shows the spatial dynamics and temporal evolution of all agents throughout this navigation scenario of 10 humans and 3 other robots.

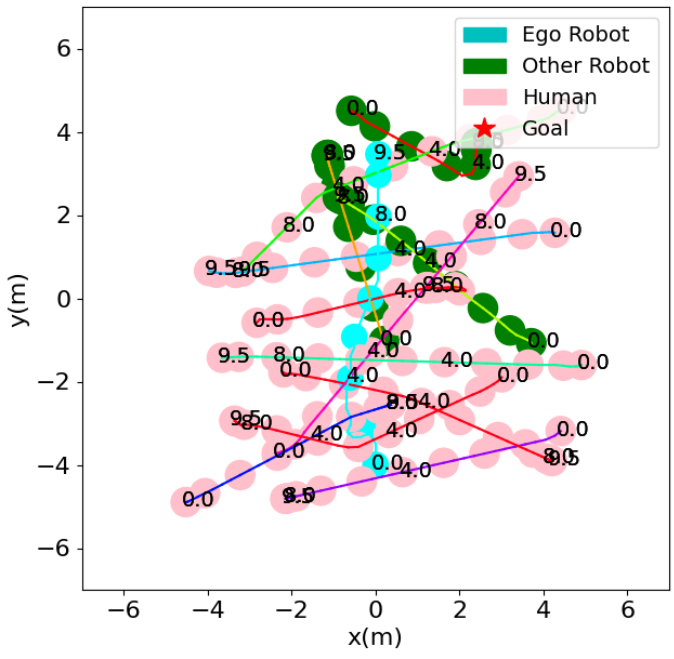

Figure 6. Trajectory plot of 10 Humans and 3 other robots scenario.

 doi:10.20944/preprints202509.2260.v1

4.2.2. Static Obstacles Scenario

We test the effectiveness of the developed navigation framework in an environment with static obstacles which more realistically depicts natural human environments.In this static obstacle scenario as shown in Figure 7, the effectiveness of the ego robot's socially aware navigation strategy is evaluated in a setting with 5 humans, 2 other robots, and several static obstacles (grey squares). This navigation case illustrates the ego robot's decision-making process as it traverses a complex environment while maintaining safe and socially acceptable distances from both moving agents and static objects.

At the onset, the ego robot begins reasoning about obstacles to avoid and starts anticipating the future positions of nearby humans and other robots. As the scenario develops ($t = 3.75s$ to $t = 4.75s$), the ego robot actively moves towards the denser part of the environment while taking precaution to venture inwards only at the right time. The successful social navigation of the ego robot can be seen as it respects comfort zones while it moves through narrow passages between moving humans and other robots ($t = 6.25s$ to $t = 7s$). By the final shown time step ($t = 8.5s$), the ego robot has approached the goal position successfully, with a trajectory that reflects a balance between comfort and efficiency.

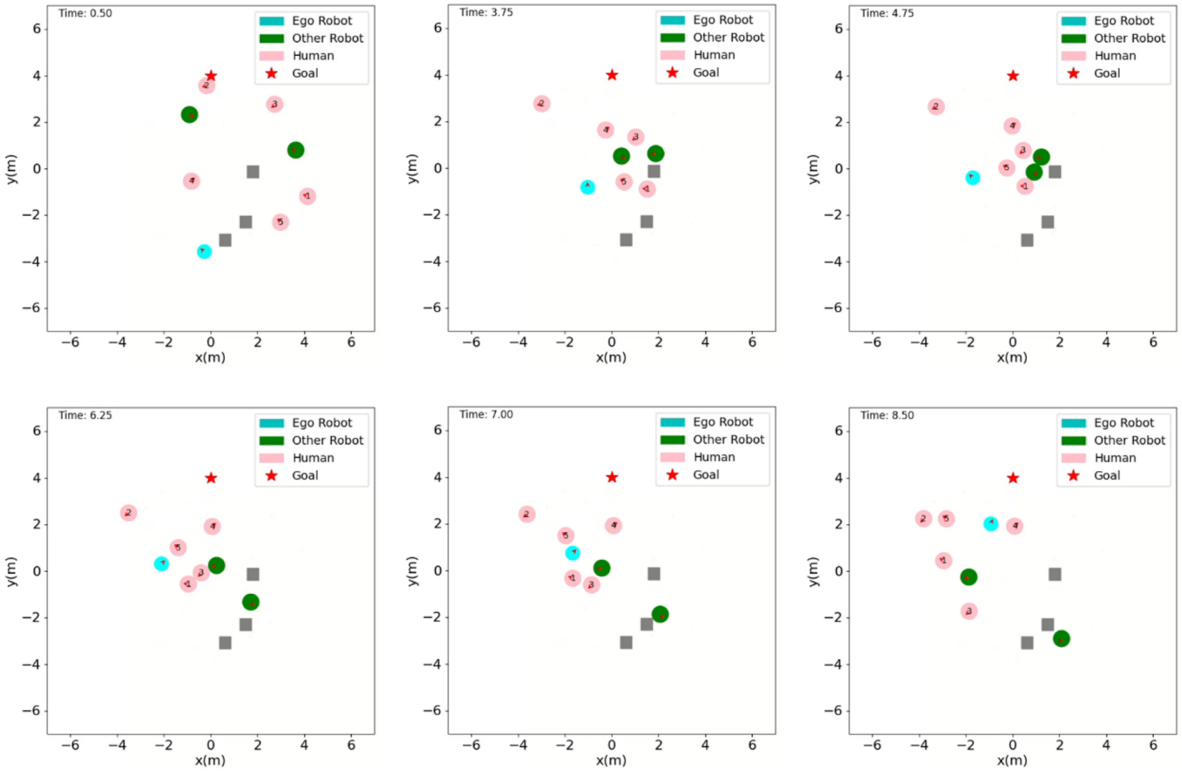

Figure 7. Simulation Scenario with static obstacles and 5 humans and 2 other robots at time steps of 0.5s, 3.75s, 4.75s, 6.25s, 7s, and 8.5s.

4.3. Quantitative Results and Analysis

We present and analyze the quantitative results obtained in this work. Table 3 presents a comparative evaluation of the developed navigation framework against HeR-DRL[29] which encodes and differentiates between different interaction types, Homogeneous Relational DRL(HoR-DRL)[29] which encodes all interactions homogeneously regardless of agent type, SARL [35], LSTM-RL [16], and ST^2 [36]. We evaluated our work using important metrics such as: Success rate, which gives the percentage of successful cases in which the ego robot arrives at the designated target location within the allotted time frame without collisions, Collision rate, which gives the rate of collisions with the other agents in the environment, Navigation time, which gives the average time for successful navigation cases, and Discomfort rate, which is the frequency of ego robot entering the discomfort region of humans.

Our developed method achieves a success rate of 89%, which, even though is slightly lower than that of most of the other frameworks, is of an acceptable value considering our approach is

 doi:10.20944/preprints202509.2260.v1

more conservative to prioritize safety and collision avoidance. A key difference is that our proposed approach achieves close to zero collision rate, highlighting its superior safety behavior. In contrast, all the other methods have significant values of collision rates. Regarding navigation time, this work achieves an average of 12.13 seconds, slightly longer than the other compared methods. This increase reflects how conservative the ego robot is in avoiding both humans and other robots. However, the trade-off results in a higher minimum separation distance of 0.16 meters, surpassing the values of all previous methods. This indicates enhanced social compliance and a lower likelihood of discomfort among surrounding agents. Additionally, this work achieves a lower discomfort rate which indicates that the ego robot comes into the discomfort zones of humans with less frequency compared to other methods. Another significant finding is the computational efficiency of our framework compared to previous works, while other methods needed approximately 10000 training episodes for convergence to an optimal policy, our framework converged at around 6000 training episodes. Overall, while the proposed method has slight trade-offs in success rate and navigation time, it significantly improves safety and social compliance. This makes it particularly reasonable for use in real-world scenarios where collisions are unacceptable and socially aware behavior is required in presence of multiple dynamic agents.

Table 3. Performance comparison across key metrics.

Works	Success Rate (%)	Collision Rate (%)	Navigation Time (s)	Discomfort Rate (%)	Avg. Min. separation distance (m)	No of Training episodes
This work	89.0	0.15	12.13	6	0.16	6000
HeR-DRL [29]	96.54	3.14	10.88	6.9	0.154	15000
HoR-DRL [29]	96.05	3.06	10.91	7.8	0.15	10000
LSTM-RL [16]	85.52	5.49	11.52	*	0.147	*
SARL [35]	93.14	4.34	10.83	*	0.154	10000
ST^2 [36]	96.46	2.99	11.08	*	0.149	1000

* Not reported

From Figure 8 we can see that the success rate improves from around 30% at the beginning to approximately 90% by the end of training, which shows that the agent is learning. The occasional dips reflect exploration or unstable policy updates, which are common during RL training. Correspondingly, the collision rate decreases sharply from an initial value of over 60% to nearly 0%. This directly indicates the policy learning to avoid humans and other robots in its path. The steep decline before episode 3000 implies early gains in safety behavior, likely guided by imitation learning and reward shaping. A downward trend in the time taken to reach the goal shows that the ego robot is learning to avoid collisions and doing so with increasing efficiency. The time drops from around 23 seconds to under 15 seconds, which suggests improved path planning and better interaction modeling. The steadily increasing cumulative reward confirms that the learning objective is being met. It is initially negative due to frequent collisions and penalties, then it transitions to positive values as the agent learns to maximize returns by reaching the goal quickly and safely. The reward curve's saturation near the end shows policy convergence.

 doi:10.20944/preprints202509.2260.v1

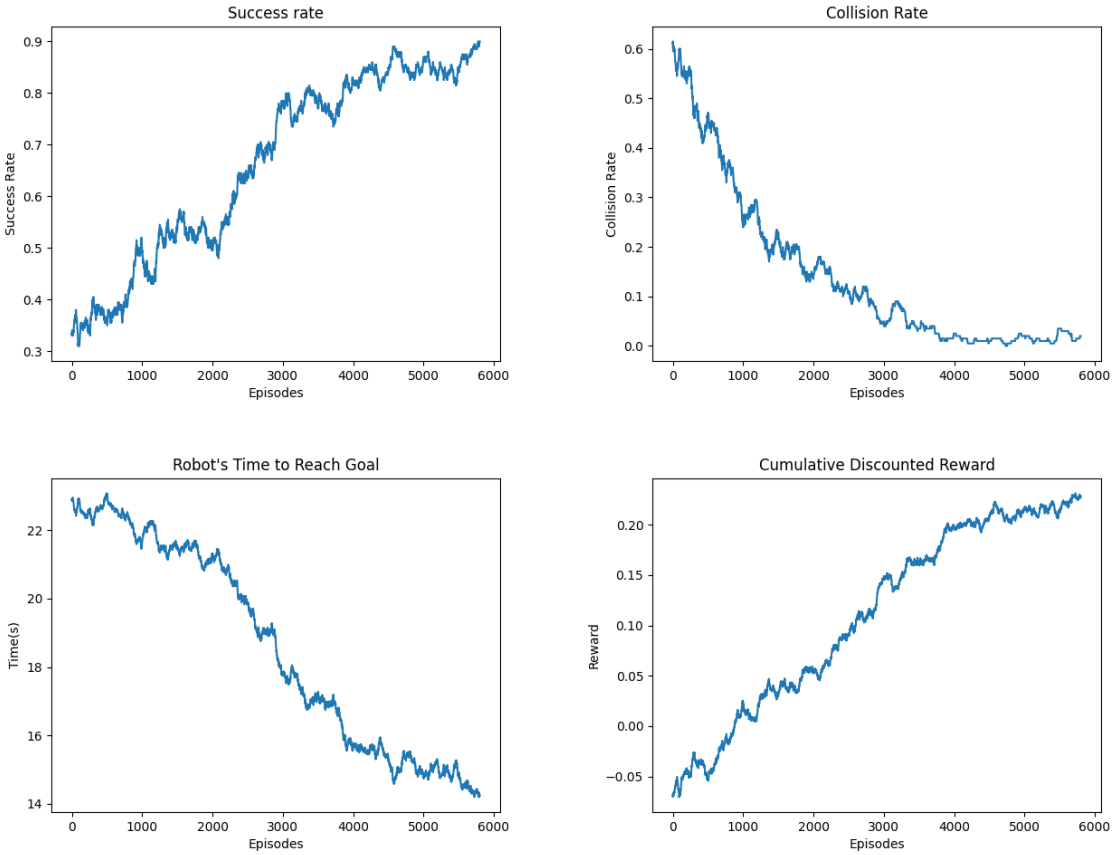

Figure 8. Performance metrics for the Navigation framework over Training episodes.

We performed a comparative experiment between two models of our framework: one trained exclusively through Deep Reinforcement Learning (DRL), and another initialized using imitation learning with ORCA-generated demonstrations prior to DRL thus quantitatively evaluating the contribution of imitation learning (IL) to the training process. This was done under same environments, reward systems, and training duration (6,000 episodes). Figure 9 shows for both settings, the training reward over episodes. With a steeper reward gradient and higher final reward, the policy initialized with IL surpasses the DRL-only policy throughout all training episodes. The graphs are shown with a ±1 standard deviation. The IL-based approach reduces the trial-and-error burden of RL, hence improving sample efficiency, stability, and performance. The first phase of the training which was done with imitation learning only had a success rate of 99% and navigation time of 7.75 seconds. This high performance is not surprising given that imitation learning replicates expert demonstrations, in this case the ORCA policy. In environments with multiple agents, where unsafe behavior could cause collisions or even harm, this is very important. These findings demonstrate the impact of imitation learning in improving training the agent towards safer and more optimal behavior.

 doi:10.20944/preprints202509.2260.v1

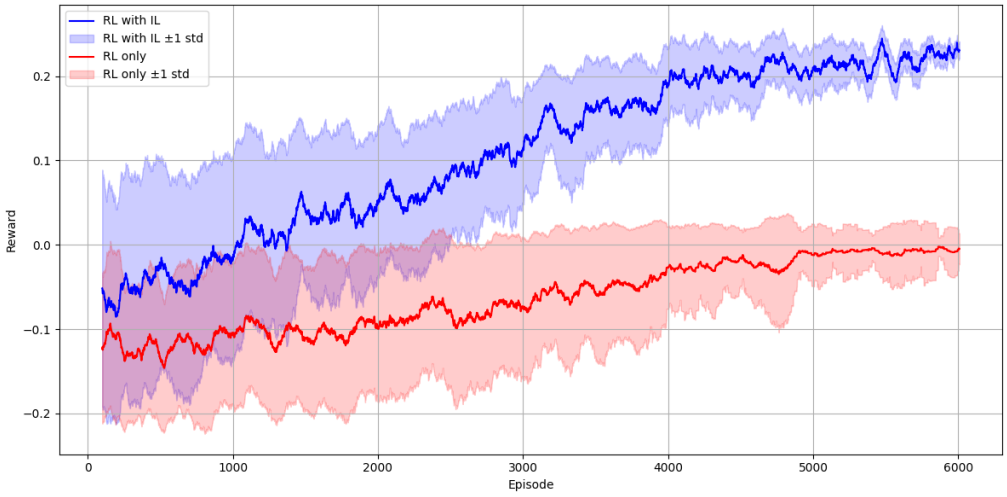

Figure 9. Reward for DRL-only and IL with DRL models, with ±1 standard deviation.

5. Conclusions

In this work, we have shown that safe and efficient socially aware robot navigation can be achieved using a single agent DRL framework which is augmented with Imitation Learning for initialization. We have explcitly modeled the interactions between the ego robot, other robots, and humans in the environment. This process enables the framework to extract socially relevant knowledge that supports different levels of social awareness and results in a conservative policy with good generalization capability. This framework has achieved enhanced socially aware metrics such as the average minimum separation distance when in a discomfort region and reduced discomfort frequency, when compared to similar works in the literature. This is in addition to a lower collision rate. The trade-off in our approach is that conservativeness of the policy and prioritization of safety result in lower success rate and average navigation time.

We have also shown the impact and benefit of using Imitation Learning for initialization of training of the agent. The developed framework was evaluated across static and dynamic obstacle layouts and varied crowd densities, showcasing its adaptability to different scenarios which reflect real human environments. We however, recommend deploying the policy on real physical robots to verify its feasibility of use in the real-world in 3D scenarios and extending the framework to multi-agent RL to test cooperative and adversarial multirobot SAN scenarios. Future work will investigate applying different human behaviour models, robots with different policies, and real-world testing.

Author Contributions: Conceptualization, I.K.K and M.F.M; methodology, I.K.K., M.F.M, and Y.I.O; software, I.K.K.; formal analysis, I.K.K., M.F.M, and A.N; investigation, I.K.K, M.F.M, Y.I.O, and A.N; resources, M.F.M; writing—original draft preparation, I.K.K.; writing—review and editing, I.K.K, M.F.M, and Y.I.O; visualization, I.K.K; supervision, M.F.M; funding acquisition, M.F.M. All authors have read and agreed to the published version of the manuscript.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors gratefully acknowledge the support of King Fahd University of Petroleum & Minerals (KFUPM), Saudi Arabia, through the Interdisciplinary Research Center for Smart Mobility and Logistics.

Conflicts of Interest: The authors declare no conflict of interest.

 doi:10.20944/preprints202509.2260.v1

References

1. Ivanov, S.; Gretzel, U.; Berezina, K.; Sigala, M.; Webster, C. Progress on robotics in hospitality and tourism: a review of the literature. *Journal of Hospitality and Tourism Technology* **2019**, *10*, (4), 489–521.
2. Personal space: An evaluative and orienting overview. Available online: https://psycnet.apa.org/record/1979-23561-001 (accessed on 9 May 2025).
3. Hall, Edward T.; Birdwhistell, Ray L.; Bock, Bernhard; Bohannan, Paul; Botkin, Daniel; Chapple, Eliot D.; Fischer, John L.; Hymes, Dell; Kimball, Solon T.; Monson, Munro S.; others. Proxemics [and comments and replies]. *Current Anthropology* **1968**, *9*, (2/3), 83–108.
4. Kruse, Thibault; Pandey, Amit Kumar; Alami, Rachid; Kirsch, Alexandra. Human-aware robot navigation: A survey. *Robotics and Autonomous Systems* **2013**, *61*, (12), 1726–1743.
5. Guillén-Ruiz, Silvia; Bandera, Juan Pedro; Hidalgo-Paniagua, Alejandro; Bandera, Antonio. Evolution of socially-aware robot navigation. *Electronics* **2023**, *12*, (7), 1570.
6. Chen, Yujing; Zhao, Fenghua; Lou, Yunjiang. Interactive model predictive control for robot navigation in dense crowds. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2021**, *52*, (4), 2289–2301.
7. Fiorini, Paolo; Shiller, Zvi. Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research* **1998**, *17*, (7), 760–772.
8. Van den Berg, Jur; Guy, Stephen J.; Lin, Ming; Manocha, Dinesh. Reciprocal n-body collision avoidance. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA), 2011; pp. 1928–1935.
9. Helbing, Dirk; Molnar, Peter. Social force model for pedestrian dynamics. *Physical Review E* **1995**, *51*, (5), 4282–4286.
10. Aoude, Georges S.; Luders, Brandon D.; Joseph, Joshua M.; Roy, Nicholas. Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. *Autonomous Robots* **2013**, *35*, 51–76.
11. Svenstrup, Mikael; Bak, Thomas; Andersen, Hans Jørgen. Trajectory planning for robots in dynamic human environments. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010; pp. 4293–4298.
12. Fulgenzi, Chiara; Tay, Christopher; Spalanzani, Anne; Laugier, Christian. Probabilistic navigation in dynamic environment using rapidly-exploring random trees and Gaussian processes. In Proceedings of the 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2008; pp. 1056–1062.
13. Trautman, Peter; Krause, Andreas. Unfreezing the robot: Navigation in dense, interacting crowds. In Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2010; pp. 797–803.
14. Chen, Yu Fan; Liu, Miao; Everett, Michael; How, Jonathan P. Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017; pp. 285–292.
15. Chen, Yu Fan; Everett, Michael; Liu, Miao; How, Jonathan P. Socially aware motion planning with deep reinforcement learning. In Proceedings of the 2017 IEEE International Conference on Intelligent Robots and Systems (IROS), 2017; pp. 1343–1350.
16. Everett, Michael; Chen, Yu Fan; How, Jonathan P. Motion planning among dynamic, decision-making agents with deep reinforcement learning. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018; pp. 3052–3059.
17. Tai, Lei; Zhang, Jingwei; Liu, Ming; Burgard, Wolfram. Socially compliant navigation through raw depth inputs with generative adversarial imitation learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018; pp. 1111–1117.
18. Pfeiffer, Mark; Schaeuble, Michael; Nieto, Juan; Siegwart, Roland; Cadena, Cesar. From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017; pp. 1527–1533.
19. Albrecht, Stefano V.; Christianos, Filippos; Schäfer, Lukas. *Multi-agent Reinforcement Learning: Foundations and Modern Approaches*; Publisher: MIT Press, 2024.
20. Chandra, Rohan; Zinage, Vrushabh; Bakolas, Efstathios; Biswas, Joydeep; Stone, Peter. Decentralized multi-robot social navigation in constrained environments via game-theoretic control barrier functions. arXiv preprint arXiv:2308.10966, 2023.
21. Escudie, Erwan; Matignon, Laetitia; Saraydaryan, Jacques. Attention graph for multi-robot social navigation with deep reinforcement learning. arXiv preprint arXiv:2401.17914, 2024.

 doi:10.20944/preprints202509.2260.v1

22. Wang, Pengyuan; Chen, Yue; He, Yu; Guo, Yuhang; Wang, Xinyu; Wang, Yijie; Wang, Jun; Wang, Chenguang. Multi-robot task assignment with deep reinforcement learning: A survey. In Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA), 2024; pp. 1–8.
23. Chen, Changan; Liu, Yuejiang; Kreiss, Sven; Alahi, Alexandre. Robot social navigation in crowded environments: A deep reinforcement learning perspective. *IEEE Robotics and Automation Magazine* **2020**, *27*, (3), 154–167.
24. Samsani, Sumanth; Maeng, JunYoung; Silva, Zachary; Faig, Jeremy; Manocha, Dinesh. Socially-aware robot navigation using deep reinforcement learning. *Robotics and Autonomous Systems* **2021**, *141*, 103757.
25. Wang, Pengyuan; Chen, Yue; He, Yu; Guo, Yuhang; Wang, Xinyu; Wang, Yijie; Wang, Jun; Wang, Chenguang. Multi-robot task assignment with deep reinforcement learning: A survey. In Proceedings of the 2024 IEEE International Conference on Robotics and Automation (ICRA), 2024; pp. 1–8.
26. Dong, Ziyi; Shi, Xijun; Huang, Jinbao; Cui, Jingzhe; Xue, Bin; Zhang, Zexi; Wang, Yang. Multi-robot cooperative target tracking via multi-agent reinforcement learning. *IEEE Robotics and Automation Letters* **2024**, *9*, (6), 5532–5539.
27. Wang, Ziyu; Chen, Yutao; Hou, Yixiao; Hou, Yue. Hypergrids: A method for multi-robot cooperative exploration. arXiv preprint arXiv:2407.00000, 2024.
28. Song, Yihan; Hu, Huimin; Kou, Xiangyang; Zhang, Xiaodong. Local target driven navigation for mobile robots in dynamic environments. Sensors, 2022.
29. Zhou, Xinyu; Piao, Songhao; Chi, Wenzheng; Chen, Liguo; Li, Wei. HeR-DRL: Human aware reinforcement learning for robot navigation. *IEEE Robotics and Automation Letters* **2025**.
30. Paszke, Adam; Gross, Sam; Chintala, Soumith; Chanan, Gregory; Yang, Edward; DeVito, Zachary; Lin, Zeming; Desmaison, Alban; Antiga, Luca; Lerer, Adam. Automatic differentiation in PyTorch. 2017.
31. Brockman, Greg; Cheung, Vicki; Pettersson, Ludvig; Schneider, Jonas; Schulman, John; Tang, Jie; Zaremba, Wojciech. OpenAI Gym. arXiv preprint arXiv:1606.01540, 2016.
32. Kingma, Diederik P. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
33. Chen, C.; Hu, S.; Nikdel, P.; Mori, G.; Savva, M. Relational graph learning for crowd navigation. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 10007–10013.
34. Zhou, Z.; Zhu, P.; Zeng, Z.; Xiao, J.; Lu, H.; Zhou, Z. Robot navigation in a crowd by integrating deep reinforcement learning and online planning. *Applied Intelligence* **2022**, *52*, (13), 15600–15616.
35. Chen, C.; Liu, Y.; Kreiss, S.; Alahi, A. Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning. In Proceedings of the 2019 IEEE International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 6015–6022.
36. Yang, Y.; Jiang, J.; Zhang, J.; Huang, J.; Gao, M. ST^2: Spatial-temporal state transformer for crowd-aware autonomous navigation. *IEEE Robotics and Automation Letters* **2023**, *8*, (2), 912–919.
37. Liu, L.; Dugas, D.; Cesari, G.; Siegwart, R.; Dubé, R. Robot navigation in crowded environments using deep reinforcement learning. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 5671–5677.