Socializing A* Algorithm for Crowd- and Socially Aware Navigation

Seif Eddine Seghiri, Noura Mansouri, Ahmed Chemori

▶ To cite this version:

Seif Eddine Seghiri, Noura Mansouri, Ahmed Chemori. Socializing A* Algorithm for Crowd- and Socially Aware Navigation. Arabian Journal for Science and Engineering, inPress, 10.1007/s13369-024-09334-6 . lirmm-04663497

HAL Id: lirmm-04663497
https://hal-lirmm.ccsd.cnrs.fr/lirmm-04663497v1

Submitted on 28 Jul 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Socializing A* Algorithm For Crowd- and Socially-Aware Navigation

Seif Eddine Seghiri[1*], Noura Mansouri[1†] and Ahmed Chemori[2†]

[1*]Department of Electronics, University Mentouri Constantine 1, Constantine, 25017, Algeria.
[2]LIRMM, University of Montpellier, Montpellier, CNRS, France.

*Corresponding author(s). E-mail(s): seifeddine.seghiri@umc.edu.dz;
Contributing authors: nor_mansouri@yahoo.fr;
Ahmed.Chemori@lirmm.fr;
†These authors contributed equally to this work.

Abstract

Today, an undeniable interest is given to the development of socially intelligent robotic systems and efficient crowd- and socially-aware navigation strategies. In this paper, we introduce a novel crowd-aware navigation algorithm that combines the A* path planner with the Double Deep Q-Network (DDQN) method. The algorithm, named Social A*, aims to allow safe navigation for mobile robots in social dynamics and crowded environments. In order to facilitate future real-world implementation, a new learning environment compatible with the Robot Operating System (ROS) is developed. This allows expert teleoperation to help train the DDQN agent and refine the reward function. We conducted extensive simulations to compare the performance of Social A* with Socially-attentive Reinforcement Learning (SARL*) and Intention Aware Robot Crowd Navigation with Attention-Based Interaction Graph (IARL). The obtained simulation results demonstrate that Social A* not only surpasses SARL* and IARL but also shows enhanced performance in handling static obstacles. These results showcase the excellent crowd-aware navigation performance, the efficiency, and the significant potential of the algorithm.

Keywords: Socially-aware navigation, Deep Reinforcement Learning, ROS, Crowd awareness, A* algorithm

1 Introduction

Nowadays, human-robot coexistence has become more widespread in our daily social interactions. This coexistence significantly improves healthcare efficiency and service industry experiences due to its adept integration of robotic assistance [1]. Therefore, the development of advanced navigation policies that are collision-free, efficient, and aware of crowd dynamics is essential, attracting considerable attention from researchers [2–5].

In the domain of robot navigation, multiple strategies are employed to enhance safety and social awareness. These strategies encompass reaction-based approaches, trajectory prediction, integration of social cues, and the adoption of learning-based methods.

In reaction-based approaches such as Optimal Reciprocal Collision Avoidance (ORCA) [6] and Social Force Model (SFM) [7], one-step interaction rules are employed. These interactions determine the optimal action for the robot. While some studies have primarily focused on collision-free navigation [8], others emphasize maintaining social distance for the robot [9]. This reflects human collision avoidance behavior and prioritizes safety and respect for pedestrians' personal space. Some researches were conducted to anticipate and represent the social behaviour using some parameters such the goal destination weight, the social weight and the obstacle weight. One of the widely used social models is the Social Force Model (SFM) and its variations [7, 10].

Another category of techniques relies on predicting the future trajectories of other agents in the environment. To safely navigate through its environment, the robot may attempt to plan a path that avoids any potential collisions with the anticipated trajectories. This approach is particularly useful in environments with a high density of agents [11]. To improve robot-human interactions, a different study [12] recommended including translation and human indicators into the existing motion planning algorithms. These human indicators are nonverbal social actions undertaken by individuals to prevent collisions.

However, with the aforementioned techniques, there is a risk of the emergence of the robot freezing problem [13]. That is why learning-based methods are introduced as alternative solutions. Several learning-based techniques rely on a set of expert demonstrations, namely imitation learning, in which the robot learns by observing the expert's demonstrations [14]. Other learning-based approaches combine supervised learning and Reinforcement Learning (RL) to form: Deep RL (DRL) [15].

The DRL-based approaches are getting more attention from researchers given their high-dimensional states and/or actions flexibility [4]. In DRL, the crowd-aware navigation problem is represented as a Partially Observable Markov Decision Process (POMDP) [16]. This improves the navigation performance by encoding the interactions between the different apparent agents. The decision-making process in the DRL framework relies on an approximation provided by a Deep Neural Network (DNN) of the state and/or the state-action value [17]. The feasibility of encoding information about the human crowd in this context has been supported by various early published DNN architectures, utilizing techniques such as pooling [3], applying a maximal operation [18] or using a long short-term memory (LSTM) [19, 20].

Other DNN architectures that facilitate the encoding of the information can be included such as Contrastive Learning [5, 21], Generative Adversarial Network (GAN) [22–24], and Structural RNN (S-RNN) [25–27]. In socially-aware robotic navigation, each of these architectures is designed to address specific challenges by incorporating some assumptions such as homogeneous crowd dynamics and consistent environmental perception accuracy. To improve generalization, DRL models typically represent the state of the robot and the crowd relative to the robot's frame by using normalized positions and velocities of nearby humans, effectively adapting learned behaviors to various environments [3, 18, 28].

In this paper, we propose a new DRL-based algorithm that combines a DRL architecture [29] with a global path planner [30], specifically the A* algorithm [31]. The A* algorithm is used as a reward function design framework. The resulting algorithm, named *Social A**, incorporates social factors, including humans' minimum comfortable distance and multiple features based on the A* path [32].

The main contributions of our work can be highlighted by the following three points:

- A novel crowd-aware navigation algorithm, called Social A*, based on DRL is proposed. It outperforms two state-of-the-art algorithms and it is effective at navigating crowded environments while considering static obstacles.
- For rapid training and to stimulate the robot to reach its destination faster, an effective reward function has been proposed. This is accomplished by incorporating the optimal velocity and a local goal into the reward value.
- A ROS-compatible learning environment is developed to facilitate future transition from simulation to real-world implementation. Furthermore, expert ROS-based teleoperation is used to enable simple adjustment of the reward function. This approach assists domain experts in determining the necessary incentive values to enhance the performance of the robot.

The remainder of this paper is organized as follows. In section 2, we will provide a problem formulation. In section 3, we provide a thorough explanation of the proposed algorithm and its key characteristics. The outcomes of the training and simulation experiments are presented in sections 4 and 5, respectively. The results demonstrate the effectiveness of the proposed algorithm for enhancing mobile robot navigation in crowded and socially complex environments.

2 Problem formulation

Robot navigation in crowd- and socially-aware spaces is an important research area. It seeks to implement algorithms and methodologies, to allow the robot to safely share the space with humans. The robot has to navigate its environment taking into account the pedestrians' movements and their social space.

To formalize the problem, we consider a 2D environment with one robot and n pedestrians, for a total of $n + 1$ agents. All agents retain partially observable states. The observable part of the state holds the position $p = [p_x, p_y]$, velocity $v = [v_x, v_y]$, and since the agents will be represented as circles, we define the circle radius r. The

unobservable state holds the agent's orientation θ, its goal position $g = [g_x, g_y]$.
In this study, we introduce a local goal to the unobservable state of the robot denoted $g^l = [g^l_x, g^l_y]$. The robot's full state at time t is defined as
$\mathbf{S}_t = [p_x, p_y, g^l_x, g^l_y, v_x, v_y, \theta, g_x, g_y, r]$, and the k-th human's observable state in the robot's coordinate frame at time t is $\mathbf{O}^k_t = [p^k_x, p^k_y, v^k_x, v^k_y, r^k]$. To generalize the state representation, the current and target positions (p, g) are substituted by the A* path length, denoted d_g. In addition, the robot's orientation θ is substituted by the term θ_{g^l} which represents the angular error between θ and the direction to its local goal g^l. The new states are then expressed as follows:

$$\begin{aligned} \mathbf{S}_t &= [d_g, \theta_{g^l}, v_x, v_y, r] \\ \mathbf{O}^k_t &= [p^k_x, p^k_y, v^k_x, v^k_y, r^k, r^k + r] \end{aligned} \tag{1}$$

By concatenating the state $\mathbf{S}_t$ with all observable human states $\mathbf{O}^k_t$, we obtain at time t a joint state of all $n+1$ agents:

$$\mathbf{J}_t = [\mathbf{S}_t, \mathbf{O}^1_t, \mathbf{O}^2_t, ..., \mathbf{O}^n_t] \tag{2}$$

In this context, $\mathbf{J}_t$ is defined within the robot's coordinate frame, enabling the model's ability to generalize across diverse environments. Furthermore, when encountering scenarios with more than n pedestrians, the model prioritizes the nearest n individuals based on proximity to the robot for consideration.

According to a specified navigation policy $\pi(\mathbf{J}_t)$, the possible actions $\mathbf{a}_t$, that a differential drive mobile robot can take, are linear and angular velocities: $\pi(\mathbf{J}_t) = \mathbf{a}_t = \mathbf{v}_t$. If we suppose that at each time t, the robot is awarded by a function $R(\mathbf{J}_t, \pi(\mathbf{J}_t))$, the optimal value of $\mathbf{J}_t$ at time t is given by:

$$V^*(\mathbf{J}_t) = \sum_{i=0}^{K} \gamma^{i.\Delta t} R(\mathbf{J}_t, \mathbf{a}^*_t) \tag{3}$$

Where $\mathbf{a}^*_t$ is the optimal action at $\mathbf{J}_t$, and it is selected according to the optimal policy $\pi^*(\mathbf{J}_t)$. The term K represents the overall amount of decision steps taken from the initial state to the last one. Δt is the decision time interval between two consecutive actions, and $\gamma \in [0, 1]$ is a discount factor.
The optimal policy $\pi^*(\mathbf{J}_t)$ is obtained by maximizing the following cumulative reward, which is as follows:

$$\begin{aligned} \pi^*(\mathbf{J}_t) = \underset{\mathbf{a}_t \in \mathbf{A}}{\arg\max}\, & R(\mathbf{J}_t, \mathbf{a}_t) + \gamma^{\Delta t} \\ & \int_{\mathbf{J}_{t+\Delta t}} P(\mathbf{J}_{t+\Delta t} \mid \mathbf{J}_t, \mathbf{a}_t) \mathbf{V}^*(\mathbf{J}_{t+\Delta t}) d\mathbf{J}_{t+\Delta t} \end{aligned} \tag{4}$$

Where $\mathbf{A}$ is the action space, $P(\mathbf{J}_{t+\Delta t} \mid \mathbf{J}_t, \mathbf{a}_t)$ is the transition probability from $\mathbf{J}_t$ to $\mathbf{J}_{t+\Delta t}$ when action $\mathbf{a}_t$ is taken.
It is reasonable to assume that pedestrians walk at a constant speed within the time interval $[t, t+\Delta t]$ since pedestrian intentions are difficult to forecast and the

length Δt is quite short. As a result, we can consider a constant velocity model to forecast humans' next states and approximate the next combined state: $\mathbf{J}_{t+\Delta t} \leftarrow propagate(\mathbf{J}_t, \Delta t, \mathbf{a}_t)$.
Eq. (5) simplifies the computation of the optimal policy π^* as follows:

$$\pi^*(\mathbf{J}_t) = \arg\max_{\mathbf{a}_t \in \mathbf{A}} R(\mathbf{J}_t, \mathbf{a}_t) + \gamma^{\Delta t} \mathbf{V}^*(\mathbf{J}_{t+\Delta t}) \tag{5}$$

To solve Eq. (5), we will employ a state-value Double Deep Q-Network (DDQN) DRL algorithm. This algorithm extends the standard Deep Q-Network (DQN) by using a dual network architecture that mitigates overestimation bias—a common issue in traditional DQNs. Specifically, one network estimates the state-action values to select actions, while a separate periodically updated network evaluates these actions to avoid feedback loops in value estimation [29]. This separation enhances the accuracy of the value function approximation, which is crucial for the efficient training of our policy π^* under the complex dynamics of pedestrian navigation modeled as a Markov Decision Process (MDP) [33]. Further technical details of the DDQN implementation are discussed in Section 3.

3 Proposed Social A* algorithm

The Social A* algorithm is based on the Double Deep Q-Network (DDQN) algorithm [29], where the A* path planning algorithm is used in its reward design step 2. The DDQN is an advanced form of the standard Deep Q-Network [15, 29] that mitigates the overestimation of action values typically seen in Q-learning. As it is described in Algorithm 1 and given Eq. (6) and Eq. (7), The DDQN employs two neural networks: the main network with weights θ and a target network with weights θ^-. These networks are used to estimate and periodically stabilize the state-value function V, based on transitions $(\mathbf{J}_t, \mathbf{a}_t, r_t, \mathbf{J}_{t+1})$ stored in a replay buffer D.

The main objective of the DDQN algorithm is to minimize the loss function:

$$L(\theta) = \mathbb{E}_{(\mathbf{J}_t, \mathbf{a}, \mathbf{r}, \mathbf{J}_{t+1}) \sim D} \left[(y - V(\mathbf{J}; \theta))^2 \right] \tag{6}$$

where

$$y = \begin{cases} r & \text{if the episode terminates at step } j+1 \\ r + \gamma \hat{V}(\mathbf{J}_{t+1}; \theta^-) & \text{otherwise} \end{cases} \tag{7}$$

Here, $\hat{V}$ is the target state-value function, and γ is the discount factor.

The A* algorithm is a heuristic search method used for finding the shortest path in a graph, making it highly effective for pathfinding in structured environments [31, 34]. It uses the evaluation function $f(n) = g(n) + h(n)$ to determine the most cost-effective path from a start node to a goal node, where $h(n)$ estimates the cost to reach the goal, and $g(n)$ represents the path cost from the start node to any node n.

Along with the A* and the DDQN employed in this work, the Social Force Model (SFM) is employed to simulate the social behavior of the crowd. This model is widely used to estimate the trajectories of individuals within a crowd, capturing the social

Algorithm 1 Double Deep Q-Network (DDQN) Using State Value Function

```
Initialize replay buffer $D$ to capacity $N$
Initialize state value function $V$ with random weights $\theta$
Initialize target state value function $\hat{V}$ with weights $\theta^- = \theta$
Initialize policy $\pi$ to select actions based on $V$
for episode = 1 to $M$ do
    Initialize state $\mathbf{J}_1$
    for $t = 1$ to $T$ do
        With probability $\epsilon$ select a random action $\mathbf{a}_t$
        Otherwise select $\mathbf{a}_t^* = \text{argmax}_a \, \pi(\mathbf{J}_t, a; \theta)$
        Execute action $\mathbf{a}_t$ and observe reward $r_t$ and new state $\mathbf{J}_{t+1}$
        Store transition $(\mathbf{J}_t, \mathbf{a}_t, r_t, \mathbf{J}_{t+1})$ in $D$
        Sample random minibatch of transitions $(\mathbf{J}_j, \mathbf{a}_j, r_j, \mathbf{J}_{j+1})$ from $D$
        Set $y_j = r_j$ if episode terminates at step $j + 1$
        Otherwise set $y_j = r_j + \gamma \hat{V}(\mathbf{J}_{j+1}; \theta^-)$
        Perform a gradient descent step on $(y_j - V(\mathbf{J}_j; \theta))^2$ with respect to $\theta$
        Every $C$ steps reset $\hat{V} = V$
    end for
end for
```

forces that influence pedestrian behaviors [7, 35–37]. The SFM posits that a pedestrian behaves as if it is subject to external attractive and repulsive forces. These include an attractive force towards a goal F_{goal}, repulsive forces to avoid collisions with other pedestrians F_{soc}, obstacles F_{obs}, and forces related to social group dynamics F_{group}. The total force influencing a pedestrian at any given moment is given by:

$$F_{total} = F_{goal} + \sum F_{soc} + \sum F_{obs} + F_{group} \tag{8}$$

In our simulations and tests, the SFM does not consider the robot's presence, focusing solely on the interactions among crowd members.

The reward function of the Social A* algorithm is defined sparsely by several social parameters along with characteristics of the A* path. A notable parameter introduced to the reward function is a local goal, which is dynamically placed along the robot's A* path at a predetermined distance from its current position, as illustrated in Fig. 1. This local goal is marked by a small red star on the A* path, facilitating the robot's agility in progressing towards its final goal, represented by a larger red star.

During learning, the robot will be strengthened to understand how to move towards it and eventually towards the overall goal.

For demonstration purposes, the local goal is placed at four different orientations $(-2, -1, 1, 2)$ rad. In Fig. 2a, the local goal is considered within a field-of-view (FOV) of $\frac{\pi}{6}$. Where, in Fig. 2b, it is set to $\frac{\pi}{3}$. The selection of the FOV depends on the desired emphasis, whether it is to emphasize rewards or punishments. To illustrate further, a decrease in the number of yellow zones results in a reduction of the robot's reward

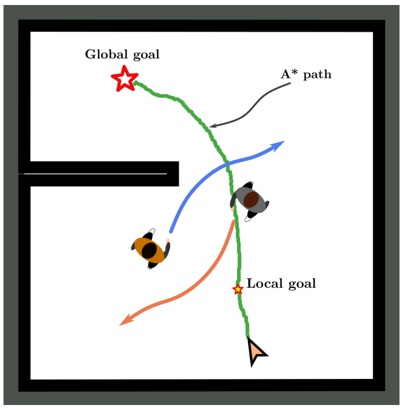


Fig. 1: Robot navigation using the proposed algorithm.

and an increase in its punishment. The reward function is described as follows:

$$\mathbf{R}(\mathbf{J}_t, \mathbf{a}_t) = \begin{cases} -400, & \text{if } d_{min} < 0; \\ -1500, & \text{if } t_{global} > T_{max}; \\ 200\,(d_{min} - d_c), & \text{if } 0 < d_{min} < d_c; \\ 1500, & \text{if } d_g < g_{off}; \\ \alpha\,(FOV - \theta_{g^l})\,T_{max}\,\Delta t\,v^*(t), & \text{otherwise.} \end{cases} \tag{9}$$

Where d_{min} represents the smallest separation distance from the robot to pedestrians during the decision time interval Δt, d_c is the least distance that humans can endure without discomfort, and t_{global} is the travel time. The offset of the goal destination d_g is set to g_{off}, which is in our case, set to $2\,r$. Furthermore, the term α indicates how much we want the local goal and the optimal velocity to stimulate the robot to reach its destination as quickly as feasible.

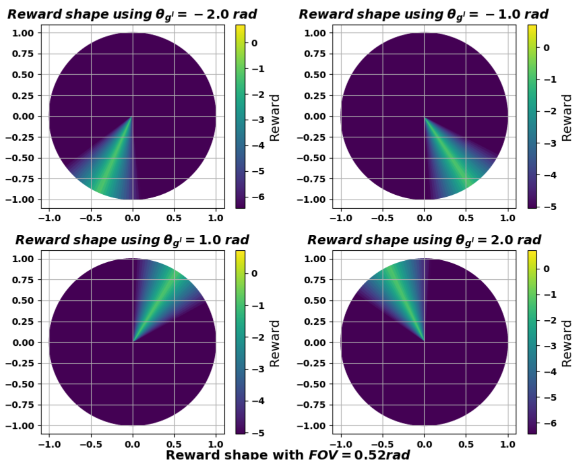
The incorporation of the optimal linear velocity, denoted as $v^*(t)$, into the design of the reward function was intended to motivate the robot's navigation system to prefer higher velocities. The reward function is structured to allocate greater rewards to actions that achieved velocities close to $v^*(t)$, thereby, encouraging the robot to operate at or near its optimal speed at any given time t.

Employing Eq. (6), Eq. (7), and Eq. (9), the proposed Social A* algorithm is therefore described by the following two equations:

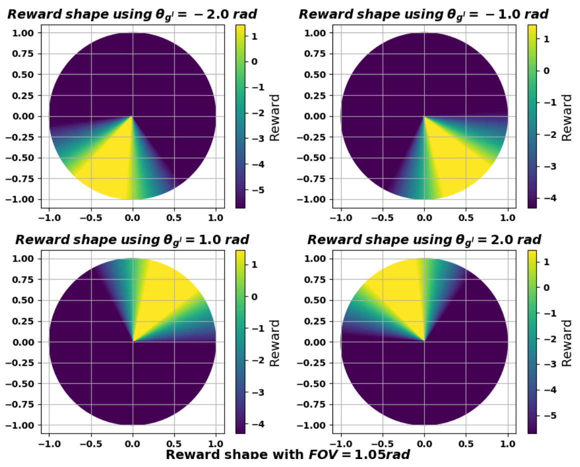
$$L_{\text{Social A}^*}(\theta) = \mathbb{E}_{(\mathbf{J}_t, \mathbf{a}_t, \mathbf{r}_t, \mathbf{J}_{t+1}) \sim D}\left[\left(y - V(\mathbf{J}_t; \theta)\right)^2\right] \tag{10}$$

where

$$y = \begin{cases} R(\mathbf{J}_t, \mathbf{a}_t) & \text{if the episode terminates at step } t+1 \\ R(\mathbf{J}_t, \mathbf{a}_t) + \gamma \hat{V}(\mathbf{J}_{t+1}; \theta^-) & \text{otherwise} \end{cases} \tag{11}$$



(a) The field-of-view (FOV) is set to $\frac{\pi}{6}$.



(b) The field-of-view (FOV) is $\frac{\pi}{3}$.

Fig. 2: Illustration of the reward distribution, and the impact of field-of-view (FOV) and local goal orientation on the robot's decision-making process.

We have adopted the attention-based DRL architecture from SARL* [3, 28] with significant adaptations in the proposed Social A* algorithm. The principal differences between SARL* and Social A* include:

- Social A* adheres to the A* path in order to socialize it, i.e. navigating socially while considering the environment's state. However, SARL* fails to take into account the environment's map [2].
- The SARL* policy depends on the ORCA policy [6], however, the Social A* is entirely self-taught. In addition, since the Social A* is based on the DDQN and

Algorithm 2 Social A*

```
while $d_g > g_{off}$ do
    Run A*
    Set a local goal falling on the A* path, and get the A* path's length
    Formulate the state $\mathbf{J}_t$ with respect to the robot frame
    Pad the state $\mathbf{J}_t$ with zeros to have a compatible input to the DDQN Algorithm 1
    Propagate the state using the DDQN
    Update the value function $V(\mathbf{J}_t)$
    Choose the best action $\mathbf{a}_t^*$
end while
```

incorporates the local goal adherence mechanism, it requires fewer episodes and less time to learn effectively.

4 Training environment and outcomes

This section outlines the simulation environment developed for training purposes and discusses the training results.

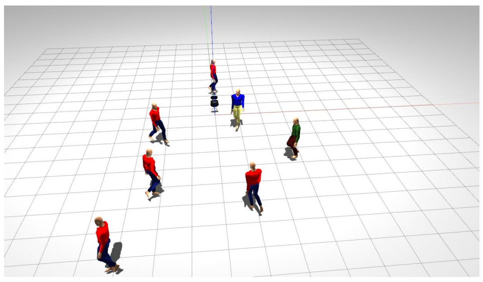
4.1 Training environment description

The training environment named `CrowdSimROS-v0` depicted in Fig. 3 is a crowd-robot coexisting Gazebo environment.

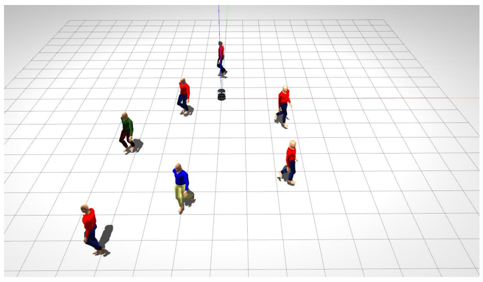
The mobile robot used in this environment is referred to as Turtlebot2. It is a differential-drive robot, comprised of a Yujin Kobuki mobile platform, various sensors (optical encoders), actuators (DC motors), and a Microsoft XBOX 360 Kinect sensor. Rather than using the Kinect, we have installed a 2D lidar sensor, for future crowd and environment state estimation. Additionally, it should be noted that this training environment provides all the necessary data e.g. velocities and sensors' measurements for the robot's learning algorithm. This environment is inherited from the `OpenAI Gym` official environment. Hence, it uses the normal training procedure of the Gym [38]. However, unlike other Gym environments, it enables the leverage of the advantageous utilities and state-of-the-art algorithms provided by ROS. The state of pedestrians is systematically retrieved from the environment to ensure an accurate representation of their positions and movements. Additionally, to realistically simulate the inherent noise associated with sensor measurements, Gaussian noise is meticulously introduced into the pedestrian state data. This approach not only enhances the robustness of the simulation by incorporating potential sensor inaccuracies but also contributes to the development of more resilient and adaptive navigation algorithms.

4.2 Training results

The robot successfully learned how to navigate its crowded environment. The reward curve depicted in Fig. 4 shows a steady increase over time, demonstrating that the agent is able to improve its performance along each episode of training.



(a)



(b)

Fig. 3: Illustrative top-view of the crowd-robot coexisting `CrowdSimROS-v0` Gazebo environment. a) view of the crowd departure states, b) view of the crowd arrival states.

Table 1 summarizes the DDQN training settings.

Table 1: Summary of hyperparameters that can affect the performance and convergence of the DDQN algorithm.

Parameter	**Value**
Number of episodes	6000
Discount factor γ	0.99
Epsilon decay ϵ_{decay}	$1.5e^{-5}$
Minimum epsilon value ϵ_{min}	0.1
Agent memory size	50000
Learning rate α	0.0001
Batch size	64

The robot maintains the navigation along the shortest path due to the use of the A* algorithm's local goal introduced in this work. As the training progressed, the agent became more efficient in its navigation, making fewer mistakes and achieving its goal in a shorter amount of time.

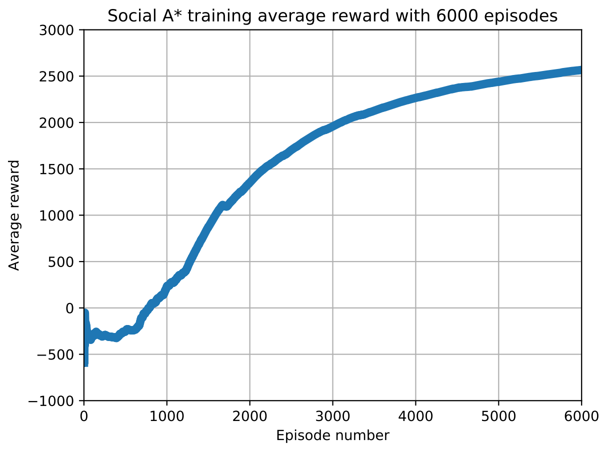


Fig. 4: Reward curve showing the improvement of the robot's social navigation learning over time using DDQN.

5 Numerical simulation results

To test the efficiency and robustness of the proposed algorithm, several tests were conducted to compare its performance with two state-of-the-art socially-aware navigation algorithms. The tests likely measured several metrics to evaluate the performance of each algorithm such as the path length, the navigation time, and the minimum distance to pedestrians.

Initially, comprehensive evaluations were conducted to assess the performance of the Social A* algorithm. These evaluations were designed to test the algorithm's accuracy across various starting and goal positions. Depending on the initial position of the robot, the observed accuracy of the algorithm varied between 71% and 96%.

Furthermore, the Social A* algorithm navigation time varied between 18 to 28 seconds, influenced by the initial positioning of the robot.

We conducted four circle-crossing simulation scenarios:

- *Scenario 1:* the objective of the robot is to navigate from an initial position $x_0 = (0.0, 0.0)$ to a specific goal destination $x_g = (0.0, -8.0)$.
- *Scenario 2:* the robot starts moving from an initial position $x_0 = (-3.0, -1.6)$ and must reach a goal destination $x_g = (3.6, -6.6)$.
- *Scenario 3:* in this scenario we test the adaptability and path efficiency of Social A* compared to SARL* and IARL navigation algorithms. The robot performs multiple trials to evaluate consistency across runs.
- *Scenario 4:* in this scenario we introduce a wall-shaped obstacle along the path to the goal to test the algorithms' capability to navigate around static obstacles. This scenario assesses how effectively each algorithm acts in the presence of static obstacles.

5.1 Scenario 1

The robot must reach the target destination taking into consideration the social space of 7 pedestrians. All pedestrians move in a way to cross the A* path of the robot from multiple directions without avoiding the robot. The evaluation of the Social A* and SARL* policies is described in Table 2. The metrics include the number of involved pedestrians, the time taken to complete the task, the distance traveled by the robot, and the minimum distance to the crowd.

Table 2: Average performance for multiple trials of Social A* and SARL* in scenario 1.

Metric	**Social A***	**SARL***
Number of humans	7	7
Starting position x_0	$(0.0, 0.0)$	$(0.0, 0.0)$
Destination x_g	$(0.0, -8.0)$	$(0.0, -8.0)$
Minimum distance to pedestrians (meters)	**1.1**	0.87
Path length (meters)	10.2	**8.79**
Navigation time (seconds)	26.0	**24.0**

Fig. 5 shows the effectiveness of both the proposed Social A* algorithm and the SARL* algorithm. With the Social A* algorithm, the robot is able to navigate socially through the crowd, maintaining a minimum distance of 1.1 meters to pedestrians, compared to only 0.87 meters with the SARL* algorithm. This sometimes leads to collisions in the SARL* algorithm due to its excessive compliance with initial learning conditions. A graphical representation of the spatial distribution of the distance to each pedestrian in the gathering for both algorithms is depicted in figures Fig. 6a-Fig. 6b.

The initial rotational motion observed in the trajectory generated by the proposed Social A* algorithm, as illustrated in Fig. 5, is attributed to the algorithm's deliberate balance between optimizing for the shortest path and adhering to social interaction norms.

By incorporating the optimal linear velocity $v^*(t)$ in the reward function, the robot managed to navigate at its maximum speed at time t. Conversely, SARL* relies on a collection of up to 3000 experiences based on the ORCA navigation policy to initialize its memory and facilitate swift navigation. This can lead to ORCA policy-like behavior, which may induce reciprocal assumptions among agents [6]. It is important to emphasize that such behavior does not accurately reflect real-life crowded scenarios that we encounter.

5.2 Scenario 2

The second scenario involves the same setup as the first one. However, the initial conditions have been modified, creating a new set of challenges for the robot. In this way, the scenario aims to test the robustness of the Social A* policy under different

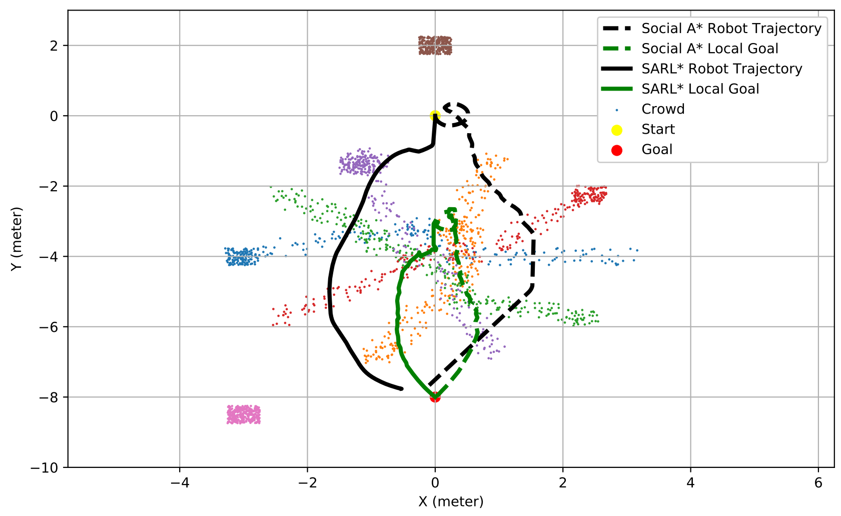
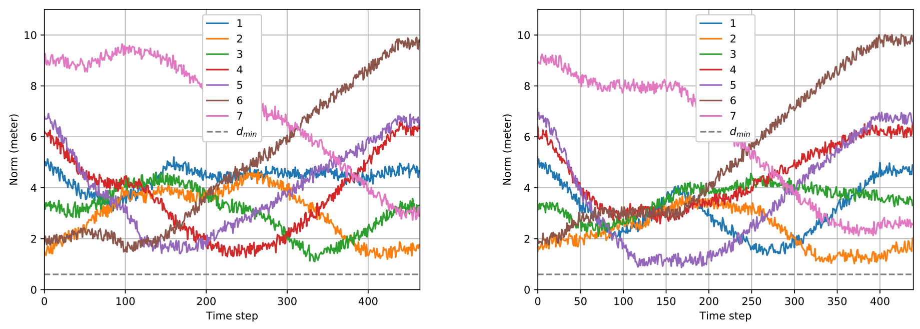


Fig. 5: Scenario 1: demonstration of the trajectories generated by Social A* and SARL* algorithms. The dashed black line and the green line represent the path and local goal of Social A*, respectively, while the solid black and the green lines represent the path and local goal for SARL*. The scattered pedestrian trajectories illustrate the added 2D noise.



(a) Social A* (b) SARL*

Fig. 6: Comparative representation of robot-crowd distances for Social A* and SARL* algorithms in Scenario 1.

operating conditions and situations. Table 3 provides an overview and evaluation of both navigation policies.

Table 3: Average performance for multiple trials of Social A* and SARL* in scenario 2.

Metric	**Social A***	**SARL***
Number of humans	7	7
Starting position x_0	$(-3.0, -1.6)$	$(-3.0, -1.6)$
Destination x_g	$(3.6, -6.6)$	$(3.6, -6.6)$
Minimum distance to pedestrians (meters)	**0.89**	0.88
Path length (meters)	9.8	**9.24**
Navigation time (seconds)	**19.75**	22.0

The evaluation of the second scenario focuses on the comparative performance analysis of the proposed Social A* algorithm and the SARL* algorithm under identical operating conditions. Fig. 7 consolidates the navigation trajectories of both algorithms, **with the Social A* algorithm represented by dashed black and lines for the trajectory and local goal path, respectively** this is fague. Conversely, the SARL* algorithm is depicted using solid black and lines for its trajectory and local goal path, respectively. This visualization highlights the effectiveness of both algorithms in navigating through crowded environments while maintaining respect for social spatial boundaries.

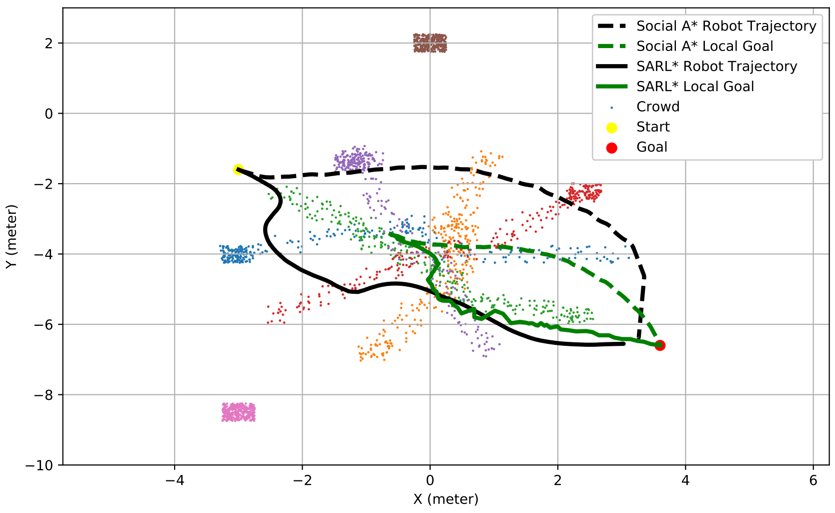
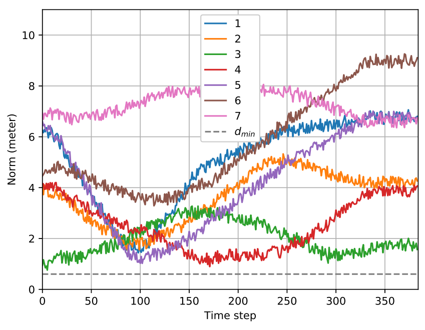
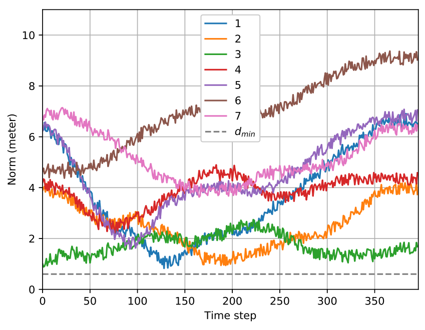


Fig. 7: Scenario 2: Comparative demonstration of the trajectories generated by the Social A* and SARL* algorithms, with distinct line styles for each algorithm's trajectory and local goal paths.

To further dissect the operational dynamics of each algorithm, Fig. 8a and Fig. 8b separately illustrate the evolution of the distance between the robot and the crowd. These figures provide a granular insight into how each algorithm modulates the robot's path in response to varying crowd densities.



(a) Social A*



(b) SARL*

Fig. 8: Comparative representation of robot-crowd distance for Social A* and SARL* algorithms in Scenario 2.

Quantitatively, the Social A* algorithm produced a path length of 9.8 meters, marginally exceeding the 9.24 meters path length by the SARL* algorithm. Nevertheless, it outperformed SARL* in navigation efficiency, completing the task in 19.75 seconds compared to 22.0 seconds by SARL*. This showcases the robust decision-making and effective trajectory-planning capabilities of Social A*. Furthermore, the minimal distance maintained from pedestrians was 0.89 meters for Social A*, slightly better than the 0.88 meters by SARL*. Although the differences in minimum distances are marginal, they underscore the consistent adherence of Social A* to the established social spatial norms throughout the navigation process.

These results substantiate the effectiveness of the Social A* algorithm. It achieves an optimal balance between path efficiency and safety, proving its potential for reliable and socially aware navigation in crowded environments.

We further evaluated our proposed Social A* algorithm by incorporating additional complex simulations that introduced variations in pedestrian dynamics and static obstacles. To comprehensively assess the adaptability and performance of Social A*, we expanded our comparative analysis to include not only the SARL* algorithm but also the Intention Aware Robot Crowd Navigation with Attention-Based Interaction Graph algorithm [39, 40], henceforth referred to as IARL in our simulations. This approach allowed us to analyze the capabilities of Social A* in crowded environments where predictive and attention-driven navigation strategies, such as those employed by IARL, play crucial roles.

5.3 Scenario 3: Multiple run trajectory analysis (5 pedestrians)

In this scenario, we simulated an environment with five pedestrians, where three moved in intersecting circular paths and two remained stationary. We executed five independent trials for each algorithm. Fig. 9 displays the trajectories from these simulations, showing the paths taken by the mobile robot under Social A*, SARL*, and IARL.

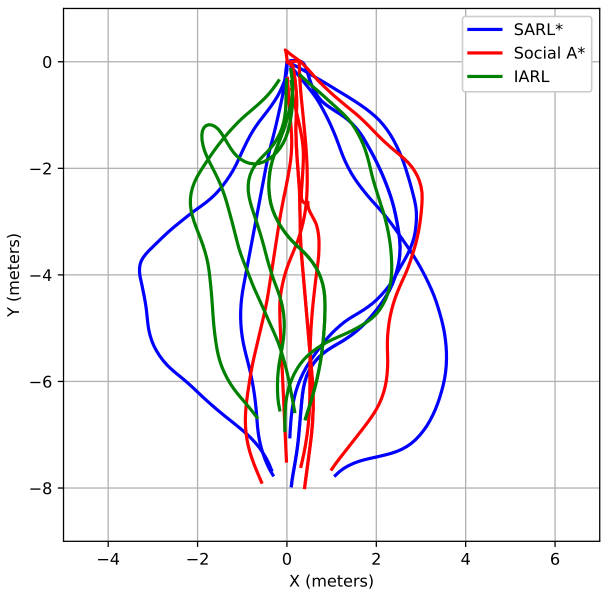


Fig. 9: Trajectories of the mobile robot in Scenario 3, under Social A*, SARL*, and IARL algorithms. The crowd trajectories are depicted in gray.

Table 4: Navigation time and path length for scenario 3

Trial	Time (seconds)			Path Length (meters)		
	Social A*	SARL*	IARL	Social A*	SARL*	IARL
1	18.90	20.97	20.12	9.11	11.05	8.10
2	17.80	19.95	16.20	10.95	8.65	8.69
3	19.21	20.30	19.05	8.55	11.19	9.78
4	20.70	19.00	16.91	8.69	11.14	10.90
5	18.76	20.90	17.42	8.71	10.42	7.85

Table 4 compares the navigation times and path lengths of Social A*, SARL*, and IARL across five trials in scenario 3.

The results show that the performances of Social A*, SARL*, and IARL are generally similar across most metrics. However, IARL shows marginally better performance in terms of navigation time and path length in several trials. This suggests that while all algorithms are effective in social environments, IARL may offer slightly improved efficiency and route optimization, though the differences are not substantial.

5.4 Scenario 4: Introduction of a static obstacle

The fourth scenario introduced a wall-shaped obstacle near the goal position to evaluate how well each algorithm navigates around static obstacles. Five trials were conducted to assess the navigation effectiveness of Social A*, SARL*, and IARL. Figure 10 shows a screenshot of the Gazebo simulation setup with the wall-shaped obstacle, providing a visual context for the experimental setup. Figure 11 illustrates the trajectories for all trials, and the recorded navigation times and path lengths are summarized in Table 5.

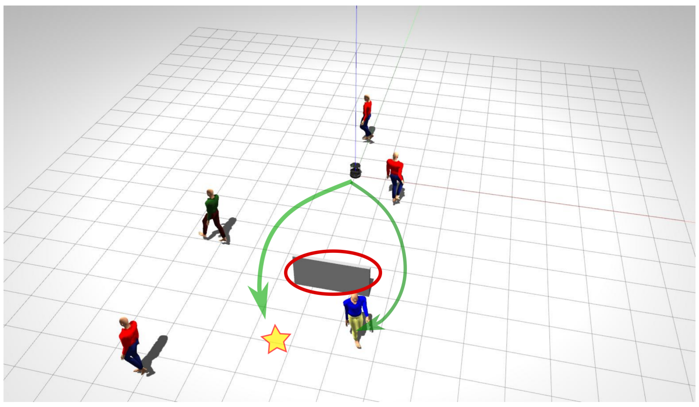


Fig. 10: Gazebo simulation screenshot showing the wall-shaped obstacle near the goal position in Scenario 4.

Table 5: Navigation time and path length for scenario 4 (with an obstacle)

Trial	Time (s)			Path Length (m)		
	Social A*	SARL*	IARL	Social A*	SARL*	IARL
1	19.20	24.10	60.52	8.01	12.03	28.95
2	18.00	32.10	46.24	7.73	8.36	18.43
3	25.70	36.80	50.30	10.01	8.42	12.28
4	19.90	33.50	37.04	8.09	12.14	14.31
5	24.50	29.70	18.74	10.78	15.79	8.16

Table 5 presents the navigation times and path lengths for Social A*, SARL*, and IARL in scenario 4.

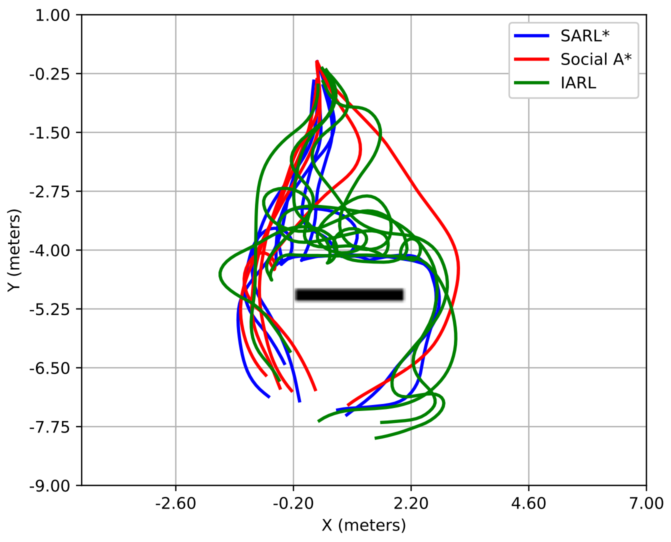


Fig. 11: Trajectories of the mobile robot in Scenario 4 with the obstacle, under Social A*, SARL*, and IARL algorithms.

The data indicates that Social A* generally achieves shorter navigation times and more efficient path lengths compared to SARL* and IARL. Specifically, Social A* demonstrates consistently quicker completion times across all trials, highlighting its effective obstacle negotiation strategies. In terms of path length, Social A* also tends to follow more direct routes than SARL*, particularly noticeable in Trials 1 and 4 where SARL* records significantly longer paths.

IARL shows varied performance, particularly prolonged times in Trials 1, 2, and 3. The excessive navigation time and path length in Trial 1 suggest difficulties in efficiently circumnavigating the obstacle, possibly due to its conservative or over-cautious approach in complex scenarios.

Overall, these results underscore Social A*'s proficiency in handling static obstacles, affirming its suitability for environments where both dynamic and static challenges are prevalent.

5.5 Comparative analysis

The results from Fig. 9 and Fig. 11, along with the data summarized in Tables 4 and 5, demonstrate that Social A* adapts well to both social dynamics challenges and static obstacles in the environment, consistently outperforming SARL* and IARL in terms of navigation efficiency. This superior performance is attributed to the integration of the A*-based reward design, which enhances the algorithm's capability to navigate complex environments effectively. While SARL* and IARL are proficient in managing crowds and social interactions, they struggle comparatively with static obstacles. This limitation arises because their strategies are primarily oriented towards dynamic interactions, lacking robust measures for static challenges. Furthermore, the reason why SARL* and IARL do not crash into the wall is that they are hardcoded to avoid collisions. They propagate the robot's state based on the selected action and test if the next state will potentially collide with the cost map; if a collision is imminent, the robot will simply rotate in place. The robustness and adaptability of Social A* suggest its significant potential for real-world applications in crowded and complex environments.

6 Conclusions and future work

The proposed algorithm uses the A* path as its basis, allowing the robot to navigate in a socially acceptable manner. By adhering to the A* path's local goal. Furthermore, the algorithm ensures the robots' maximum total cumulative reward.

In conclusion, the proposed Social A* demonstrates significant potential for navigating in crowded environments for mobile robots. This algorithm provides a fast, short, safe, and social path for the robot, ensuring that it navigates its environment without disrupting the movement of pedestrians. Additionally, the compatibility of the proposed DRL environment with ROS enables a seamless transition from simulation to real-time, thereby facilitating interoperability.

Future work may include further improvements to the performance of the Social A* algorithm. For instance, incorporating additional features, increasing the crowd size, or fine-tuning the hyperparameters of the DDQN agent. All of which may lead to better navigation in complex and dynamic environments. Moreover, estimating the states and features of dynamic obstacles and integrating these into the DDQN algorithm, along with reformulating the reward function to include their features and states, will be crucial for enhancing the algorithm's effectiveness in a dynamic environment.

Competing Interests

The authors declare that there are no competing interests associated with this research article. No potential conflict of interest was reported by the authors, and all aspects of the research were conducted with full transparency and integrity, ensuring an unbiased and objective presentation of the findings.

Data Availability

Data will be available on request from the authors.

References

[1] Cheng, C.-L., Hsu, C.-C., Saeedvand, S., Jo, J.-H.: Multi-objective crowd-aware robot navigation system using deep reinforcement learning. Applied Soft

Computing **151**, 111154 (2024)

[2] Seghiri, S.E., Mansouri, N., Chemori, A.: Implementation of sarl* algorithm for a differential drive robot in a gazebo crowded simulation environment. In: 2022 2nd International Conference on Advanced Electrical Engineering (ICAEE), pp. 1–6 (2022). IEEE

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

[4] Chen, C., Hu, S., Nikdel, P., Mori, G., Savva, M.: Relational graph learning for crowd navigation. In: IROS (2020)

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

[6] Berg, J., Lin, M., Manocha, D.: Reciprocal velocity obstacles for real-time multi-agent navigation. In: 2008 IEEE International Conference on Robotics and Automation, pp. 1928–1935 (2008). Ieee

[7] Helbing, D., Molnar, P.: Social force model for pedestrian dynamics. Physical review E **51**(5), 4282 (1995)

[8] Wang, L., Li, Z., Wen, C., He, R., Guo, F.: Reciprocal collision avoidance for nonholonomic mobile robots. In: 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV), pp. 371–376 (2018). IEEE

[9] Cheng, C.-L., Hsu, C.-C., Saeedvand, S., Jo, J.-H.: Multi-objective crowd-aware robot navigation system using deep reinforcement learning. Applied Soft Computing **151**, 111154 (2024) https://doi.org/10.1016/j.asoc.2023.111154

[10] Wu, W., Chen, M., Li, J., Liu, B., Zheng, X.: An extended social force model via pedestrian heterogeneity affecting the self-driven force. IEEE Transactions on Intelligent Transportation Systems (2021)

[11] Kuderer, M., Kretzschmar, H., Sprunk, C., Burgard, W.: Feature-based prediction of trajectories for socially compliant navigation. In: Robotics: Science and Systems (2012)

[12] Reddy, A.K., Malviya, V., Kala, R.: Social cues in the autonomous navigation of indoor mobile robots. International Journal of Social Robotics **13**(6), 1335–1358 (2021)

[13] Trautman, P., Krause, A.: Unfreezing the robot: Navigation in dense, interacting

crowds. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 797–803 (2010). IEEE

[14] Qin, L., Huang, Z., Zhang, C., Guo, H., Ang, M., Rus, D.: Deep imitation learning for autonomous navigation in dynamic pedestrian environments. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 4108–4115 (2021). IEEE

[15] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., *et al.*: Human-level control through deep reinforcement learning. nature **518**(7540), 529–533 (2015)

[16] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2nd edn. MIT Press, Cambridge, MA (2018)

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

[18] Chen, Y.F., Liu, M., Everett, M., How, J.P.: Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 285–292 (2017). IEEE

[19] Everett, M., Chen, Y.F., How, J.P.: Motion planning among dynamic, decision-making agents with deep reinforcement learning. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3052–3059 (2018). IEEE

[20] Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S.: Social lstm: Human trajectory prediction in crowded spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 961–971 (2016)

[21] Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning, pp. 1597–1607 (2020). PMLR

[22] Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A.: Social gan: Socially acceptable trajectories with generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2255–2264 (2018)

[23] Lin, K.-C., Hsu, J.-Y., Wang, H.-W., Chen, M.-Y.: Early fault prediction for wind turbines based on deep learning. Sustainable Energy Technologies and Assessments **64**, 103684 (2024)

[24] Ma'sum, M.A., Pratama, M., Lughofer, E., Ding, W., Jatmiko, W.: Assessor-guided learning for continual environments. Information Sciences **640**, 119088 (2023)

[25] de Jesús Rubio, J., Orozco, E., Cordova, D.A., Hernandez, M.A., Rosas, F.J., Pacheco, J.: Observer-based differential evolution constrained control for safe reference tracking in robots. Neural Networks **175**, 106273 (2024) https://doi.org/10.1016/j.neunet.2024.106273

[26] Jesús Rubio, J., Garcia, D., Rosas, F.J., Hernandez, M.A., Pacheco, J., Zacarias, A.: Stable convolutional neural network for economy applications. Engineering Applications of Artificial Intelligence **132**, 107998 (2024) https://doi.org/10.1016/j.engappai.2024.107998

[27] Jesús Rubio, J., Garcia, D., Sossa, H., Garcia, I., Zacarias, A., Mujica-Vargas, D.: Energy processes prediction by a convolutional radial basis function network. Energy **284**, 128470 (2023) https://doi.org/10.1016/j.energy.2023.128470

[28] Li, K., Xu, Y., Wang, J., Meng, M.Q.-H.: Sarl*: Deep reinforcement learning based human-aware navigation for mobile robot in indoor environments. In: 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 688–694 (2019). https://doi.org/10.1109/ROBIO49542.2019.8961764

[29] Van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30 (2016)

[30] Ganesan, S., Natarajan, S.K., Srinivasan, J.: A global path planning algorithm for mobile robot in cluttered environments with an improved initial cost solution and convergence rate. Arabian Journal for Science and Engineering **47**(3), 3633–3647 (2022)

[31] Ulrich, I., Borenstein, J.: Vfh/sup*: Local obstacle avoidance with look-ahead verification. In: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065), vol. 3, pp. 2505–2511 (2000). IEEE

[32] Ji, X., Feng, S., Han, Q., Yin, H., Yu, S.: Improvement and fusion of a* algorithm and dynamic window approach considering complex environmental information. Arabian Journal for Science and Engineering **46**, 7445–7459 (2021)

[33] Ebrahimi, S.H.S.: A hybrid principal label space transformation-based binary relevance support vector machine and q-learning algorithm for multi-label classification. Arabian Journal for Science and Engineering (2024) https://doi.org/10.1007/s13369-024-09034-1

[34] Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics **4**(2), 100–107 (1968)

[35] Alitaleshi, A., Jazayeriy, H., Kazemitabar, J.: Indoor pedestrian trajectory reconstruction using spatial–temporal error correction and dynamic time warping-based path matching for fingerprints map creation. Arabian Journal for Science and Engineering **48**(2), 2101–2119 (2023)

[36] Khan, S.D., Basalamah, S.: Sparse to dense scale prediction for crowd couting in high density crowds. Arabian Journal for Science and Engineering **46**(4), 3051–3065 (2021)

[37] Moussaïd, M., Helbing, D., Garnier, S., Johansson, A., Combe, M., Theraulaz, G.: Experimental study of the behavioural mechanisms underlying self-organization in human crowds. Proceedings of the Royal Society B: Biological Sciences **276**(1668), 2755–2762 (2009)

[38] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W.: Openai gym. arXiv preprint arXiv:1606.01540 (2016)

[39] Liu, S., Chang, P., Huang, Z., Chakraborty, N., Hong, K., Liang, W., Livingston McPherson, D., Geng, J., Driggs-Campbell, K.: Intention aware robot crowd navigation with attention-based interaction graph. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 12015–12021 (2023)

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