Complex & Intelligent Systems (2024) 10:1149–1166
https://doi.org/10.1007/s40747-023-01216-y

ORIGINAL ARTICLE

Dynamic warning zone and a short-distance goal for autonomous robot navigation using deep reinforcement learning

Estrella Elvia Montero[1] · Husna Mutahira[2] · Nabih Pico[3] · Mannan Saeed Muhammad[1]

Received: 6 February 2023 / Accepted: 3 August 2023 / Published online: 22 August 2023

Abstract

Robot navigation in crowded environments has recently benefited from advances in deep reinforcement learning (DRL) approaches. However, it still presents a challenge to designing socially compliant robot behavior. Avoiding collisions and the difficulty of predicting human behavior are crucial and challenging tasks while the robot navigates in a congested social environment. To address this issue, this study proposes a dynamic warning zone that creates a circular sector around humans based on the step length and speed of humans. To properly comprehend human behavior and keep a safe distance between the robot and the humans, warning zones are implemented during the robot's training using deep enforcement learning techniques. In addition, a short-distance goal is established to help the robot efficiently reach the goal through a reward function that penalizes it for going away from the goal and rewards it for advancing towards it. The proposed model is tested on three state-of-the-art methods: collision avoidance with deep reinforcement learning (CADRL) , long short-term memory (LSTM-RL), and social attention with reinforcement learning (SARL). The suggested method is tested in the Gazebo simulator and the real world with a robot operating system (ROS) in three scenarios. The first scenario involves a robot attempting to reach a goal in free space. The second scenario uses static obstacles, and the third involves humans. The experimental results demonstrate that the model performs better than previous methods and leads to safe navigation in an efficient time.

Keywords Deep reinforcement learning · Autonomous robots · Dynamic obstacle avoidance · Dynamic warning zone · Short-distance goal

List of symbols

$\bar{p}$	Position of agents (robot or humans)
$\bar{v}$	Velocity of agents
$\bar{g}$	Goal position
$J(t)$	Join state of agents
$\bar{d}_{\rho g}$	Robot's distance to goal
$\bar{d}_{\rho h}$	Robot's distance to human
$a(t)$	Robot's action
R	Reward
V^*	Optimal value function
π^*	Optimal policy
γ	Discount factor
r_{wz}	Warning zone radius
ϕ_{wz}	Warning zone angle
h_{gait}	Step length human

✉ Husna Mutahira
husna@sogang.ac.kr

✉ Mannan Saeed Muhammad
mannan@skku.edu

Estrella Elvia Montero
emontero@g.skku.edu

Nabih Pico
npico@espol.edu.ec

1 Department of Electrical and Computer Engineering, Sungkyunkwan University, Natural Sciences Campus, Suwon 16419, South Korea

2 Department of Computer Science and Engineering, Sogang University, Seoul 04107, South Korea

3 Facultad de Ingeniería en Electricidad y Computación, Escuela Superior Politécnica del Litoral, ESPOL, Campus Gustavo Galindo, 09-01-5863, Guayaquil, Ecuador

Introduction

The relevance and utility of robot navigation in diverse applications have increased in recent years [1, 2]. Robots are currently used to navigate autonomously in spaces such as hospitals, airports, malls, industries, hotels, etc. [3–6]. The main purpose of deploying autonomous robots is their ability to operate in unknown environments and navigate without human intervention or only slight intervention [7, 8]. Static environments are navigated using traditional navigation techniques, but obstacles that are dynamic still pose challenges [9]. Nevertheless, intelligent mobile robots must be capable of navigating complex environments [10, 11]. In complex scenarios, combining human motion prediction with robot motion planning can be challenging due to humans' unpredictable and dynamic nature [12]. The robots must be capable of evaluating, comprehending, and predicting future situations to avoid colliding with objects in a dynamic environment. However, effective autonomous navigation includes mapping and localization, human detection, social rules, and decision-making, making it a challenging task [13].

Prior studies have used robot path planning in conjunction with the prediction of future human trajectory to avoid overlap between human and robot path [14–16]. The methods, however, encountered some major problems, including freezing robot scenarios. In this case, the robot freezes or oscillates in dense crowds deciding that all paths are unsafe [17, 18].

Recent research has demonstrated reinforcement learning alongside deep learning [19], is an effective method of creating socially compliant robot navigation systems [20–22]. The learning-model methods present human-aware navigation as a Markov decision process (MDP) and use deep V-learning, where the agent chooses an action based on the state value approximated by the neural networks [23, 24]. By maximizing the total reward of the action, the deep reinforcement learning (DRL) technique selects the optimal policy for leading the robot through environmental interaction [25]. The robot learns the collective importance of neighboring humans to their future states. However, in the real-world scenario when the goal is located at a farther distance, these methods are ineffective since the robot made detours to find the optimal path. [26, 27]. A map-based mechanism is also introduced for the robot to take safe actions within indoor environments [28].

Most of the existing DRL-based social navigation methods assume homogeneity of agents, i.e., that humans behave similarly [21, 26, 29]. Also, they assume that the human transition states are well known and well defined. These states are, in fact, difficult to model due to the unpredictable nature of human behavior, i.e., humans move at different speeds based on their physical conditions, activities, and even their size. Therefore, it is still challenging to develop new techniques that can better account for the heterogeneity of human behavior in the design of socially compliant robot navigation systems.

To solve the aforementioned issues, the aim of this proposed work is to design a reward function that considers the safety of the surrounding environment and simplicity to lead the robot to take action for collision-free navigation. In the proposed method, dynamic warning zones are formulated to enable the robot to predict and anticipate the diversity of human behavior at the current state, thus the robot takes actions that lead it to avoid these zones for safe navigation. To lead the robot to minimize the arrival time, the reward function also evaluates the action taken by the robot at the current time and encourages it to take a route that forwards it closer to the desired goal.

Accordingly, the contribution of this study is as follows:

- Robots are trained with the dynamic warning zones around humans based on their size, speed, and the length of steps they take, to avoid collisions. The warning zone is a circular space that the robot avoids crossing to keep a comfortable distance from humans.
- A proposed reward function evaluates the distance between the robot's position and the goal to reach a short-distance goal. A positive reward is given to the robot when it gets closer to the goal, and a negative reward when it moves away from the goal.
- The proposed navigation system model is integrated into the robot operating system (ROS) for extensive evaluation in three different simulated and real scenarios: empty space, static obstacles, and dynamic obstacles.

The dynamic warning zones and short-distance goals are formulated and later trained and tested on DRL-based navigation methods to lead the robot to take collision-free actions while considering the unpredictable human behavior in densely crowded environments. The paper is structured as follows: "Related work" begins with the related work, followed by the methodology in "Methodology". The experimental setup, results, and evaluations are presented in "Results and discussion". Finally, the conclusions are provided in "Conclusions".

Related work

Crowd navigation is complex since the robot cannot directly acquire each human's implicit goal since the environment is partially observable [10, 30]. The recent research on this problem can be roughly divided into the following two categories:

Reaction-based models

Using attractive and repulsive forces, the Social Force model [31] is one of the earliest models developed for improving navigation in crowded environments. Other reaction-based models such as reciprocal velocity obstacles (RVO) [32], and optimal reciprocal collision avoidance (ORCA) [33] use optimal collision-free velocities for obstacle avoidance in multi-agent navigation. These models have the drawback of being computationally intensive since well-engineered parameters are required.

Learning-based methods

DRL algorithms are currently being used in the field of robotics to directly learn optimal control strategies from visual inputs for various real-world issues [34]. The popularity of DRL as an end-to-end approach has led to it being used to learn collaborative behavior in crowd-aware scenarios [35]. DRL is also used for socially compliant obstacle-aware avoidance [29, 36, 37]. Additionally, imitation learning can be used to discover policies based on demonstrations of desired behaviors [38, 39]. Human interactions are addressed using Collision Avoidance with DRL [23]. The socially aware DRL approach has been extended with long short-term memory (LSTM) architecture [24], which converts each human state into a fixed-length vector in reverse order of distance from the robot. All these methods use deep V-learning, where the planner calculates the value of all possible next states. Based on the agents' state transitions, the planner chooses an action that leads to the highest valued state. Supervised and reinforcement learning are combined by considering human behaviors in human–robot interaction [26, 40, 41]. To fully define the spatial–temporal interactions between pedestrians and robots for autonomous navigation in a dynamic environment, the spatial–temporal state transformer is created [42]. Despite this, all of the approaches mentioned above do not perform well when presented with complex environments that include real-time human behavior. Using the decentralized collision avoidance method, a reinforcement learning framework is proposed where each agent makes its own decisions independently without communicating with any other agent [43].

Although DRL-based methods have demonstrated improvement in performance in crowded environments, they still suffer from a number of drawbacks. These include not adapting to human behavior by considering their size and speed, or not finding a path in some situations. Therefore, limiting the application of these methods to real-world situations to a substantial extent.

This manuscript proposes a method by which a robot can learn to adapt its behavior by considering comfort and safety. To accomplish this, dynamic warning zones are created as circular sectors around humans. To find an efficient and smooth path, a short-distance goal is integrated with DRL to intelligently select waypoints, thus aiding the robot in reaching its goal by maintaining an optimized course.

Methodology

Problem formulation

In this work, the crowd-aware robot navigation problem is formulated as a sequential decision-making problem in a reinforcement learning framework. The robot interacts in an environment comprising of n number of humans, where all agents (humans and robot) move in a two-dimensional plane. The states of agents at any time t consist of position $\bar{p} = [x, y]^T$, velocity $\bar{v} = [v_x, v_y]^T$, and the radius r. They constitute *observable states* that can be observed by all the agents. On the other hand, the *unobservable states* include the goal position $\bar{g} = [g_x, g_y]^T$, preferred velocity $\bar{v}_p$ of each agent with its magnitude as v_p, and orientation θ_ρ. Here, the subscript p represents the *preferred parameters*, while subscript ρ represents the *robot parameters*.

The robot state O_ρ at any time t is given by:

$$O_\rho(t) = \left[\bar{p}_\rho, \bar{v}_\rho, r_\rho, \bar{g}_\rho, v_{p\rho}, \theta_\rho\right], \tag{1}$$

and the each human's state O_h is given by:

$$O_h(t) = [\bar{p}_h, \bar{v}_h, r_h, v_{ph}], \tag{2}$$

where h is the index for each human and has a value of $1 < h < n$. From a geometric perspective and for better parametrization, a robot-centric system is used, with the origin at its current position, and the x-axis pointing towards the goal [23]. Such a system is given by:

$$O_\rho(t) = \left[\bar{d}_{\rho g}, \bar{v}_\rho, r_\rho, v_{p\rho}\right], \tag{3}$$

where $\bar{d}_{\rho g}$ represents the distance of the robot from the goal and is given as $\bar{d}_{\rho g} = \parallel \bar{p}_\rho - \bar{g} \parallel_2$, and:

$$O_h(t) = [\bar{d}_h, \bar{p}_h, \bar{v}_h, r_h], \tag{4}$$

where $\bar{d}_{\rho h} = \parallel \bar{p}_\rho - \bar{p}_h \parallel_2$ is the robot's distance to neighbor h.

The observable robot state and human state are combined to form a Joint State J of the system at time t, which is given by:

$$J(t) = [O_\rho, O_1, ..., O_h, ..., O_n]. \tag{5}$$

At each time-step t, the robot takes an action $a(t) \in A$, where A represents the action space, according to its policy $\pi(J)$. As a result, the robot receives a reward of $R(J, a)$ for selecting that action. The optimal value function $V^*(J)$ is calculated by accumulating the expected rewards at each

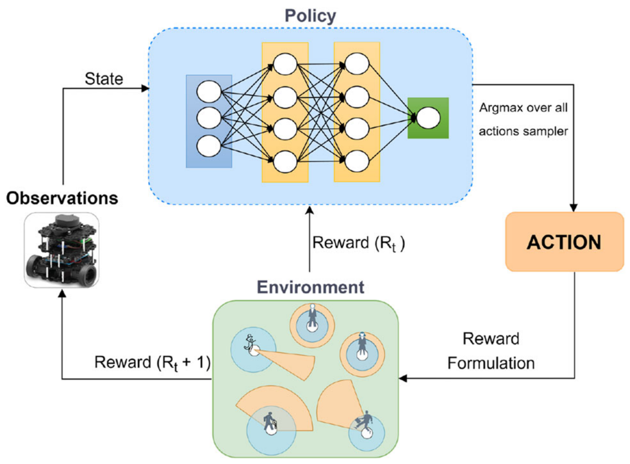


Fig. 1 The overview of deep reinforcement learning framework

time-step t given an action performed and is computed as:

$$V^*(J) = \sum_{k=o}^{T} \gamma^{k\Delta t v_p} R(J, a^*), \tag{6}$$

where γ is a discount factor and its value is given as $0 \leq \gamma \leq 1$, and a^* is the optimal action. As mentioned before v_p is the preferred velocity, and here it is used as a normalization factor for numerical reason [26]. $R(J, a^*)$ is the total accumulated reward at each time-step t.

With an optimal value function $V^*(J)$ as provided in (6), the optimal policy can be calculated by choosing the optimized value, i.e., selecting an action with maximum cumulative reward, as provided below:

$$\begin{aligned}\pi^*(J) = \arg\max_a R(J, a) \\ + \gamma^{k\Delta t v_p} \int_{J^+} P(J, a, J^+) \mathrm{d}J^+,\end{aligned} \tag{7}$$

where $P(J, a, J^+)$ is the probability of the unknown state transition from t to t^+. The integral in (7) is approximated by assuming that the human travels with a constant velocity for a short time interval t to t^+, where $t^+ = t + \Delta t$. Therefore, the next joint state can be estimated, and the optimal policy is simplified as:

$$\pi^*(J) = \arg\max_a R(J, a) + \gamma^{k\Delta t v_p} V^*(J^+). \tag{8}$$

Figure 1 shows the DRL architecture. The observations such as position and velocity are provided to the deep neural network framework. With a pre-trained value network N, a path is planned by selecting actions that maximize the cumulative reward. The proposed reward function creates an optimal path by incorporating dynamic warning zones and short-goal distances to ensure comfort and safety. Additionally, three different groups of humans are simulated in the training environment. These groups include adults over 18 years of age, adolescents between 12 and 17 years of age, and children between 5 and 12 years of age [44]. The sector's radii and speeds are assigned to each group during the training process depending on their step length and speed.

Algorithm design

A detailed description of the proposed algorithm is provided in this section.

Dynamic warning zones

To emphasize navigational safety, a realistic dynamic environment is modeled by incorporating human behavior patterns. Human behavior and speeds differ based on the characteristics of each individual. Therefore to simplify human behavior, three groups are selected: adults, adolescents, and children. During training, the diameters of the circles represent the size of the person. Thus, circles with diameters of 0.41–0.6 m are used to represent adults, 0.31–0.4 m for adolescents, and 0.2–0.3 m for children. Based on various activities, such as walking slowly or rapidly, or running, the velocity varies from 0.5 to 0.6 m/s, 0.61 to 0.8 m/s, and 0.81 to 1.8 m/s, respectively [44, 45]. To accommodate the robot's hardware limitations, the human velocity is limited to 1.8 m/s for this study.

The dynamic warning zones are circular sectors with radius r_{wz} based on human velocity v_{h} and the step length h_{gait}, given by:

$$r_{\mathrm{wz}} = 0.8 v_{\mathrm{h}} + h_{\mathrm{gait}}. \tag{9}$$

A gain of 0.8 is applied to reduce the elongation of the tangent lines with the maximum increase in velocity. The value is found through experimentation. The values assigned to h_{gait} are calculated by measurement trial. Therefore, the maximum length each individual covers in one step corresponds to 0.8 m for an adult, 0.65 m for an adolescent and 0.45 m for a child. Figure 2 shows the geometry of the dynamic warning zone.

Warning zone angle is formulated based on [46], where ϕ_{wz}, decreases exponentially with magnitude of human velocity, v_{h}, and is expressed as follows:

$$\phi_{\mathrm{wz}} = 2.2\pi e^{-1.7 v_{\mathrm{h}}} + 0.2\pi. \tag{10}$$

This has a difference with [46], in terms of the factors of 2.2, 1.7, 0.2π. These values are based on experimental trials to increase the angle of the zone, and subsequently formalize a reward that penalizes as a safety precaution. This allows the

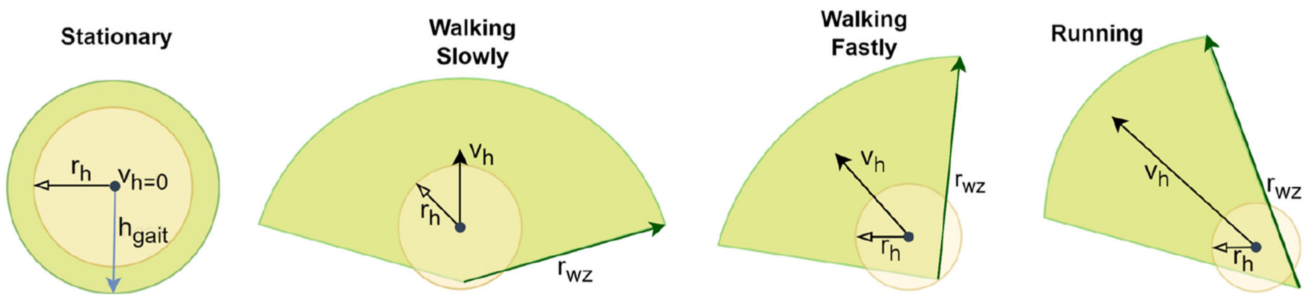


Fig. 2 Dynamic warning zones based on zero to the maximum velocity of the humans with different radii

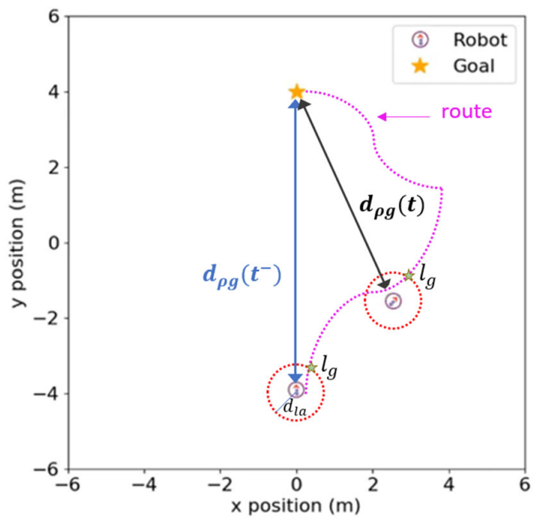


Fig. 3 A local goal l_g is selected on the global path at time t, within the range of look-ahead distance d_{la} centered at the robot's position

robot to react faster and anticipate the movement of humans from far distances, thus avoiding collisions with humans and safety.

Short-distance goal

The primary objective of robot navigation is to reach the goal *safely* and *efficiently*. Based on the Dijkstra algorithm [47], the robot follows a trajectory from the initial point to the goal. To reach the goal, the algorithm generates waypoints based on the occupancy grid map. A local short-distance goal, l_{g}, is introduced to ensure the robot follows the waypoints even if it deviates from avoiding a dynamic or static obstacle. Figure 3 shows the short-distance goal.

The robot's local goal is set to be the closest waypoint within the range of look-ahead distance d_{la} and is based on the robot's current position $\bar{p}_\rho$. This procedure updates in the joint state J and continues until the robot's local goal coincides with its final goal. The robot's trajectory is also evaluated by a reward function, if it is approaching the goal by $d_{\rho g}(t)$ and $d_{\rho g}(t^-)$, corresponding to the distance from the robot's position to the goal in current time-step t and the previous time-step (t^-). The reward function encourages the robot to move towards the goal and penalizes it if it chooses a different course that leads it away from the goal.

Reward function

The reward formulation in DRL models for social navigation penalizes only collisions or uncomfortable distances [26]. The proposed reward function takes into account the safety between humans and robots and at the same time encourages the robot to continue approaching the goal, thereby minimizing navigation time. According to the proposed reward formulation, once the robot has reached the goal position, it will receive a positive reward of 10. The robot, however, is penalized with -0.1 for the actions that lead to a collision. In the event that the robot navigates beyond the distance tolerable by humans, the robot is penalized further.

As mentioned before, the robot is penalized if it exceeds the distance that is acceptable to human safety while navigating, as:

$$0.25(d_{\rho h} - d_{\mathrm{min}}), \quad \text{if } 0 < d_{\rho h} < d_{\mathrm{min}},$$

where $d_{\rho h}$ represents the robot's distance to neighbor h, and d_{min} is discomfort factor which is set to 0.2 [35].

Another reward function is formulated based on the dynamic warning zone, where r_{wz} is the radius of the dynamic warning zone, and r_{h} is the human radius accordingly to the group of the human. The robot is penalized if it exceeds $d_{\rho h} \le r_{\mathrm{wz}} - r_{\mathrm{h}}$ as:

$$0.2\left(e^{(d_{\rho h} - r_{\mathrm{wz}} - r_h)} - 0.3\right).$$

To encourage the robot to continue moving towards the goal, another reward function is established based on a short-distance goal by calculating the difference between the current time-step t and the previous time-step t^-. This value is multiplied by a constant gain of 0.01 formulated based on experimental trials, given by:

$0.01\left(d_{\rho g}(t^{-}) - d_{\rho g}(t)\right),$

where $d_{\rho g}(t^{-})$ and $d_{\rho g}(t)$ denote the robot's distance to the goal $\bar{d}_{\rho g} = \| \bar{p}_{\rho} - \bar{g}_{\rho} \|_2$ at time t^{-} and t.

During each time-step, the robot's progress is evaluated. The robot is rewarded if it advances towards the goal position and is penalized if it diverges from its goal.

The final reward R_{wz} is calculated by combining all the aforementioned reward functions as follows:

$$R_{\text{wz}} = \begin{cases} 10, & \text{if reach the goal} \\ -0.1, & \text{if } d_{\rho h} < 0 \\ 0.25(d_{\min} - d_{\rho h}), & \text{if } 0 < d_{\rho h} < d_{\min} \\ 0.2\left(e^{(d_{\rho h} - r_{\text{wz}} - r_h)} - 0.3\right), & \text{if } d_{\rho h} \leq r_{\text{wz}} - r_h \\ 0.01\left(d_{\rho g}(t^{-}) - d_{\rho g}(t)\right), & \text{otherwise.} \end{cases} \tag{11}$$

Results and discussion

To improve clarity and organization, this section has been divided into multiple sub-sections. A comprehensive description of the training setup is provided first. A detailed analysis of both quantitative and qualitative evaluations is presented, followed by a statistical analysis of the results. Towards the end of this section, a detailed exposition of the implementation and experimental setup has been provided.

Training setup

To evaluate the approach of the proposed model, the robot is trained in an environment with five and ten simulated humans belonging to different categories as mentioned in the previous section. As part of the training and testing process, the humans navigate from random positions in circular and square crossing scenarios, where each human's goal is the opposite of his initial position. Additionally, the robot navigates in two modes: *invisible* and *visible*. When the robot is in *invisible* mode, humans react only to the movements of other humans without taking into account the movements of the robot. In the *visible* mode, the human interacts according to the robot's movement.

The model is trained for 10,000 episodes, with a learning rate of 0.001 and a discount factor γ set to 0.9. In the first 4000 episodes, the exploration rate ϵ of the greedy policy decreases linearly from 0.5 to 0.1 and then remains unchanged for the remaining episodes. The value network is reinforced using mini-batches of size 100 after every 1000 episodes. The robot is tested in 500 test scenarios with the same configuration. A comparison of the reinforcement learning parameters used in training and testing is presented in Table 1.

Table 1 Reinforcement learning parameters

Parameters	Value
Training episodes	10,000
Testing episodes	500
Batch size	100
Learning rate	0.001
Discomfort factor γ	0.9
Exploration rate ϵ	0.5–0.1

The humans are represented by circles with random diameters from 0.2 to 0.6 m in reference to children, teens, and adults sizes. Humans' speed is also random and varies between 0.5 and 1.8 m/s according to actions such as walking slowly, rapidly, and running. The time-step Δt is 0.25 s and the maximum navigation time is 25 s. The look-ahead distance, d_{la}, of the local goal, is 2.5 m, and the total distance from the start point to the endpoint is 8 m.

The proposed method is compared with reward functions over three state-of-the-art DRL methods: collision avoidance with deep reinforcement learning (CADRL), long short-term memory (LSTM-RL), and social attention with reinforcement learning (SARL). The network frameworks of all methods are kept. The proposed reward function R_{wz} is also compared with three previously proposed reward functions R_1 [26], R_2 [24], and R_3 [46] as provided in (12), (13), and (14), respectively.

$$R_1 = \begin{cases} 1, & \text{if reach the goal} \\ -0.25, & \text{else if collision} \\ -0.1 + d_{\rho h}/2, & \text{else if uncomfortable} \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

$$R_2 = \begin{cases} 1, & \text{if reach the goal} \\ -0.25, & \text{else if collision} \\ -0.1 + 0.05 d_{\rho h}, & \text{else if uncomfortable} \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

$$R_3 = \begin{cases} 1, & \text{if reach the goal} \\ -0.1, & \text{else if collision} \\ -d_{\rho h}\left(\frac{-0.1}{r_{dz} - 0.3}\right) + 0.1), & \text{else if danger zone} \\ 0, & \text{otherwise.} \end{cases} \tag{14}$$

While previous reward formulations have encouraged the robot to avoid collisions during navigation, the robot disregards safety during the course of its behavior. The reward function R_3 [46] imposes a penalty for entering a danger zone, however, it is based on the assumption that the agent is placed in an environment with dynamic obstacles of the same size, which idealizes the agent's state and its surrounding crowd. Moreover, it does not evaluate the distance between the robot and the goal, resulting in a long time to reach the target and non-smooth routes.

Table 2 Quantitative evaluation results in the invisible and visible circular crossing scenarios with 5 humans

Method	Invisible					Visible				
	SR	CR	Time	PL	DR	SR	CR	Time	PL	DR
CADRL(R_1)	0.66	0.23	12.16	9.28	0.72	0.81	0.07	11.44	9.77	0.37
CADRL(R_2)	0.68	0.22	12.02	9.99	0.64	0.86	0.04	11.34	9.15	0.35
CADRL(R_3)	0.77	0.14	12.26	9.65	0.79	0.93	0.06	11.54	9.81	0.92
CADRL(R_{wz})	0.77	0.12	10.05	8.65	0.62	0.94	0.06	10.18	8.95	0.57
LSTM-RL(R_1)	0.90	0.09	10.73	9.71	0.22	0.95	0.04	10.31	9.14	0.13
LSTM-RL(R_2)	0.84	0.12	10.96	9.87	0.30	0.94	0.06	10.51	10.02	0.12
LSTM-RL(R_3)	0.93	0.07	10.46	9.70	0.63	0.96	0.02	10.20	9.65	0.63
LSTM-RL(R_{wz})	0.95	0.07	10.20	9.57	0.44	0.97	0.02	10.14	9.09	0.28
SARL(R_1)	0.94	0.02	10.33	9.02	0.33	0.97	0.03	11.07	9.96	0.22
SARL(R_2)	0.91	0.04	10.91	9.85	0.38	0.96	0.03	12.89	9.84	0.27
SARL(R_3)	0.96	0.02	10.07	9.77	0.23	0.99	0.01	10.47	9.55	0.22
SARL(R_{wz})	0.98	0.02	9.98	9.71	0.13	1.00	0.00	10.06	9.34	0.10

Table 3 Quantitative evaluation results in the invisible and visible square crossing scenarios with 5 humans

Method	Invisible					Visible				
	SR	CR	Time	PL	DR	SR	CR	Time	PL	DR
CADRL(R_1)	0.60	0.25	12.82	10.45	0.39	0.86	0.05	10.67	9.47	0.69
CADRL(R_2)	0.63	0.23	12.08	10.17	0.37	0.86	0.04	10.16	9.44	0.64
CADRL(R_3)	0.72	0.21	12.21	9.73	0.79	0.89	0.03	10.18	9.37	0.77
CADRL(R_{wz})	0.83	0.13	11.65	9.45	0.65	0.90	0.03	10.06	9.24	0.57
LSTM-RL(R_1)	0.85	0.09	9.44	9.87	0.57	0.90	0.09	9.86	9.72	0.72
LSTM-RL(R_2)	0.84	0.09	9.64	9.98	0.63	0.90	0.08	9.89	9.78	0.60
LSTM-RL(R_3)	0.90	0.04	9.30	9.15	0.63	0.96	0.03	9.73	9.67	0.52
LSTM-RL(R_{wz})	0.92	0.04	9.27	9.06	0.44	0.98	0.02	9.68	9.55	0.51
SARL(R_1)	0.91	0.04	9.79	9.33	0.31	0.96	0.04	10.09	9.82	0.27
SARL(R_2)	0.90	0.05	9.88	9.43	0.33	0.94	0.07	10.74	9.91	0.30
SARL(R_3)	0.97	0.02	9.66	9.28	0.31	0.98	0.02	9.53	9.27	0.55
SARL(R_{wz})	0.97	0.02	9.51	9.01	0.28	0.99	0.01	9.62	9.09	0.17

Quantitative evaluations

Several metrics are used to evaluate robot navigation performance: success rate (SR), collision rate (CR), total time (T), path length (PL), and discomfort rate (DR). The SR measures the success rate without collision, whereas CR measures the rate of collision between the robot and the human. Total time (T) is the average time taken by the robot to reach its goal. PL is the average length of the path traveled by the robot in successful trials, and DR represents how often the robot crosses the warning zone in R_{wz} and R_3, or crosses the discomfort distance in R_1 and R_2. The navigation performance with the proposed reward function is evaluated by calculating the success rate, collision rate, and discomfort rate.

Navigation performance

Navigation performance is evaluated in two modes: *invisible* and *visible* modes with circular and square crossing scenarios. Tables 2 and 3 list the quantitative evaluations of experiments with five humans in both modes and scenarios, respectively. SARL, CADRL, and LSTM are evaluated with R_1, R_2, R_3 and the proposed R_{wz}.

In the *invisible* mode, the performances of CADRL(R_1) and CADRL(R_2) are lowest as shown in Tables 2 and 3. With the proposed method, CADRL(R_{wz}) and CADRL(R_3) improve their SR reaching 0.77. The lowest CR and DR are obtained by CADRL(R_{wz}).

Table 4 Quantitative evaluation results in visible circular and square crossing scenarios with 10 humans

Method	Circular					Square				
	SR	CR	Time	PL	DR	SR	CR	Time	PL	DR
CADRL(R_1)	0.62	0.16	15.28	9.87	0.56	0.62	0.12	12.70	9.16	0.83
CADRL(R_2)	0.63	0.18	16.77	9.97	0.61	0.73	0.21	11.98	9.02	0.68
CADRL(R_3)	0.80	0.13	13.38	9.61	0.90	0.75	0.13	12.81	9.13	0.91
CADRL(R_{wz})	0.80	0.11	12.83	9.15	0.37	0.77	0.13	11.81	8.77	0.82
LSTM-RL(R_1)	0.80	0.09	11.45	10.35	0.40	0.85	0.08	10.33	9.38	1.14
LSTM-RL(R_2)	0.76	0.10	10.89	10.87	0.46	0.83	0.09	10.83	9.44	1.23
LSTM-RL(R_3)	0.82	0.06	10.20	9.34	0.67	0.87	0.05	10.78	9.22	0.69
LSTM-RL(R_{wz})	0.83	0.03	9.53	9.28	0.27	0.89	0.07	9.68	9.15	0.52
SARL(R_1)	0.87	0.09	11.10	10.09	0.25	0.83	0.11	11.54	10.89	0.38
SARL(R_2)	0.86	0.09	11.22	10.12	0.27	0.80	0.15	11.56	10.92	0.45
SARL(R_3)	0.97	0.04	10.45	9.89	0.52	0.96	0.04	11.27	10.06	0.96
SARL(R_{wz})	0.99	0.01	10.37	9.01	0.16	0.98	0.02	10.39	9.71	0.14

On the other hand, the performance of LSTM-RL(R_{wz}) increases, achieving an SR of 0.96, with a low CR of 0.03 and a DR of 0.44 compared to previous LSTM-RL(R_1), and LSTM-RL(R_2) models. Unlike CADRL(R_3) and LSTM-RL (R_3), the CADRL (R_{wz}) and LSTM-RL(R_{wz}) present the lowest DR demonstrating more safety and comfort during the robot navigation.

As shown in Tables 2 and 3, the SR for the SARL baseline is the highest in circular and square scenarios. Furthermore, SARL(R_1) and SARL(R_2) are superior in terms of CR, with the lowest average of 0.02. However, in comparison with SARL(R_3), the proposed SARL(R_{wz}) has a higher SR of 0.98 and a lower DR, demonstrating that the robot is less frequently entering discomfort distance with humans. Due to the invisible mode, the robot is very likely to have collisions and close crossovers.

In the *visible* mode, the human interacts with the robot, and the SR and CR are improved in all the methods. CADRL(R_{wz}) performs better, achieving a high SR of 0.94 and a low CR of 0.06. Nevertheless, DR is still high, because CADRL does not consider complex interactions between agents, and the robot is more likely to take paths that lead it to enter a group of humans. LSTM-RL with rewards (R_1), and (R_2) show similar SR and CR. Though LSTM-RL(R_3) keeps relative success and collision rates, LSTM-RL(R_{wz}) performed better by obtaining higher safety with a low DR.

Table 4 shows the quantitative evaluation results in *visible* circular and square crossing scenarios with 10 humans. It can be seen that the higher crowd density results in reduced performance for CADRL and LSTM-RL with rewards R_1 and R_2. CADRL(R_{wz}) and LSTM-RL(R_{wz}), however, enhance SR, reducing CR and DR. In contrast, LSTM-RL(R_{wz}) shows a higher DR when compared with DR with less crowd density, as shown in the table.

In terms of SR and CR, the SARL performs better in all scenarios. Nonetheless, SARL(R_{wz}) is able to achieve higher SRs of 0.99 and 0.98, and better CRs close to 0 in both circular and square scenarios.

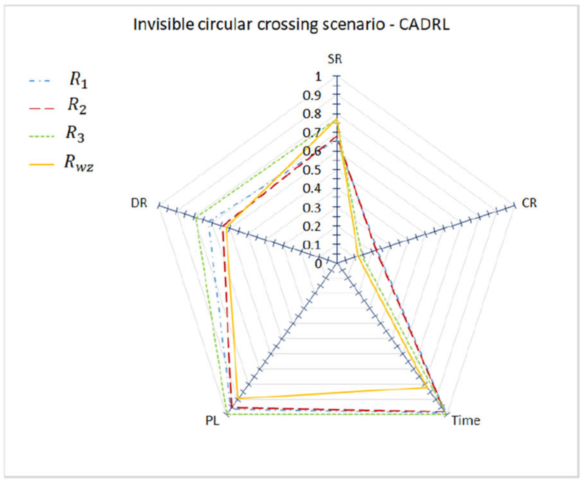
SARL(R_{wz}) also has the lowest DR, demonstrating that the robot prioritizes safety. By adapting its navigation behavior for environments with humans and their variety of speeds, SARL(R_{wz}) shows the best performance in the different scenarios and crowd densities.

Efficiency

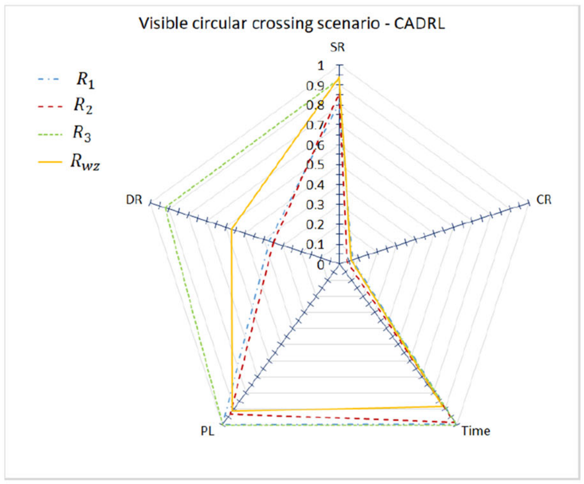
The efficiency is assessed by the average time (T) taken to reach the goal by the robot and the average length of the path traveled for successful navigation (PL), supported by SR, CR, and DR. The results are presented in the Tables 2, 3 and 4.

As the proposed reward function also evaluates the trajectory of the robot, the PL is lower than the previous methods with rewards R_1, R_2, and R_3 in all experimented scenarios, thus showing that the proposed method decreases the successful path length significantly. Furthermore, the proposed method consumes less time to reach the goal with fewer chances of collision as demonstrated by SR, CR, and DR as shown in the tables. Consequently, the proposed method requires less time and travels less distance. The efficiency of SARL(R_{wz}) is higher than all other performance metrics, due to the fact that the path maintained by SARL(R_{wz}) minimizes arrival time while achieving fewer collisions.

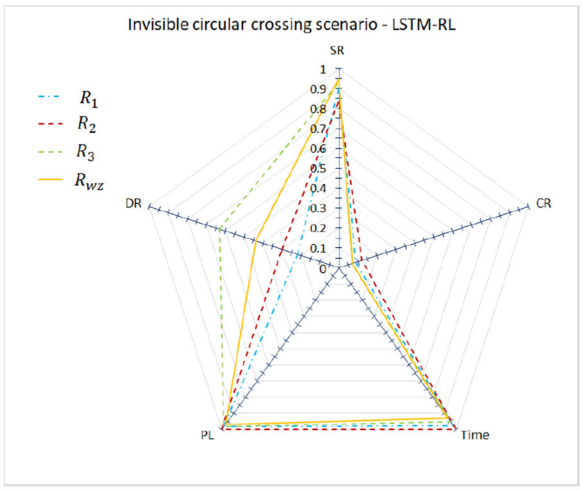
A complete overview of the performance of CADRL, LSTM-RL, and SARL in all metrics is given in Figs. 4 and 5. All metrics results are plotted in spider graphs, and the time (T) and path length (PL) are normalized for better visualization.



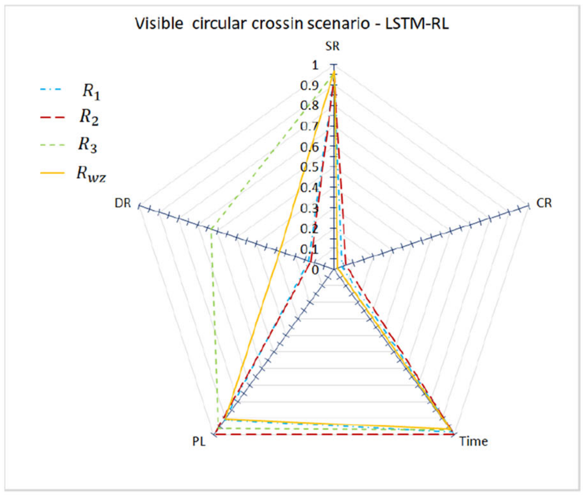
(a)



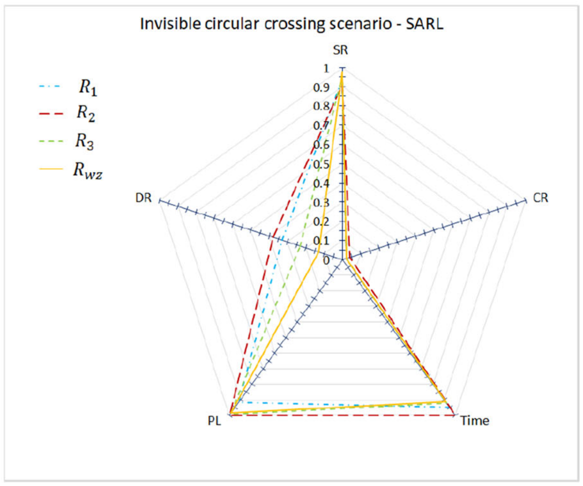
(b)



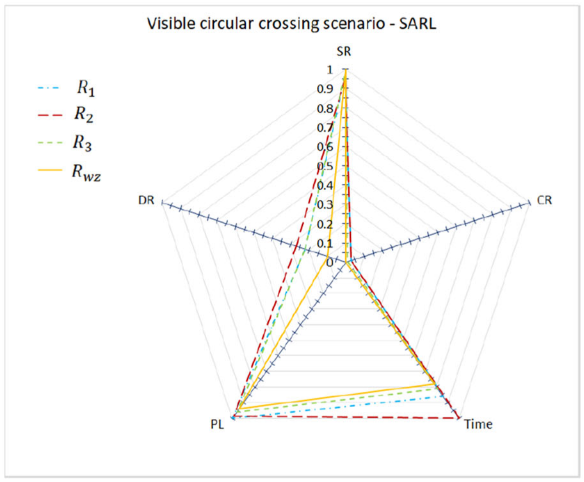
(c)



(d)

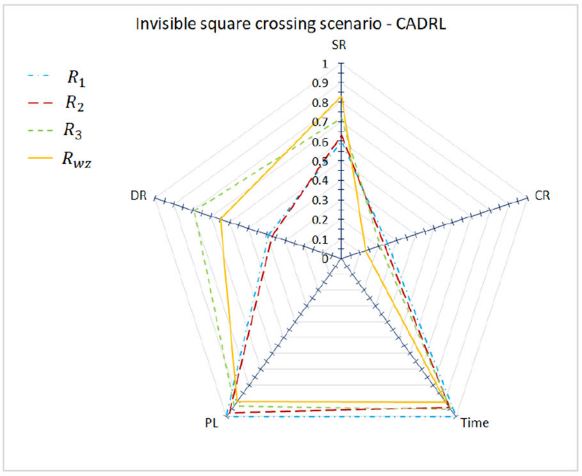


(e)

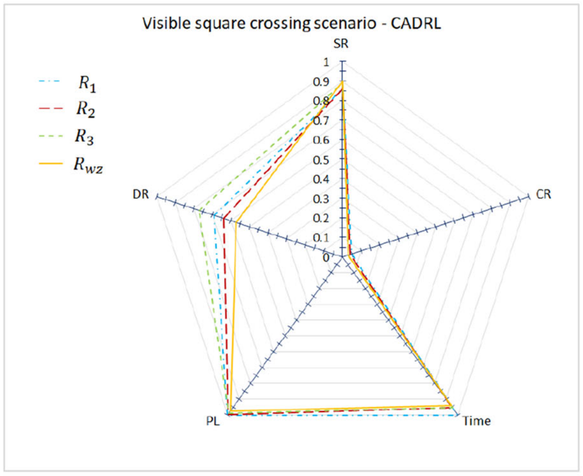


(f)

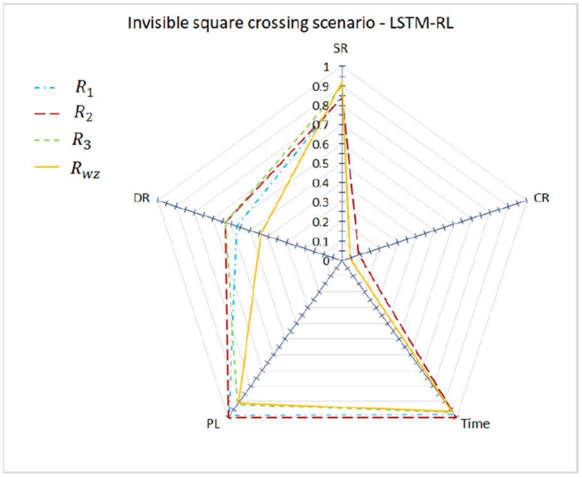
Fig. 4 Results from circular crossing scenario with all methods plotted in spider graph. **a** CADRL model results in invisible mode. **b** CADRL model results in visible mode. **c** LSTM-RL model results in invisible mode. **d** LSTM-RL model results in visible mode. **e** SARL model results in invisible mode. **f** SARL model results in visible mode



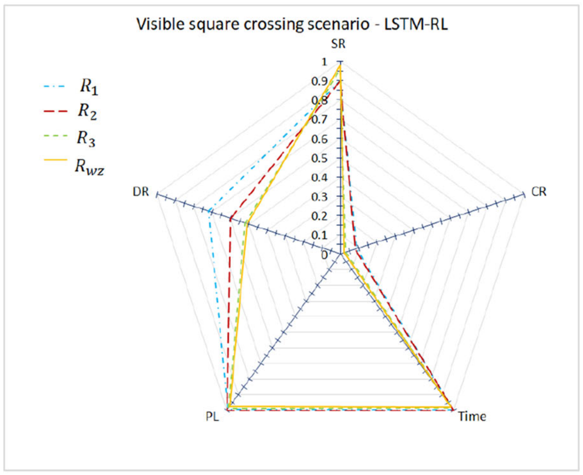
(a)



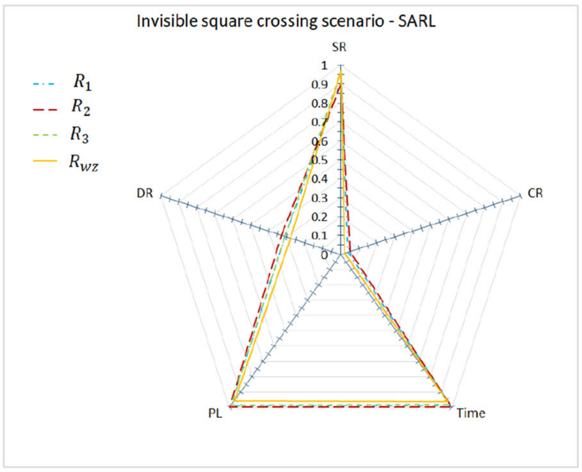
(b)



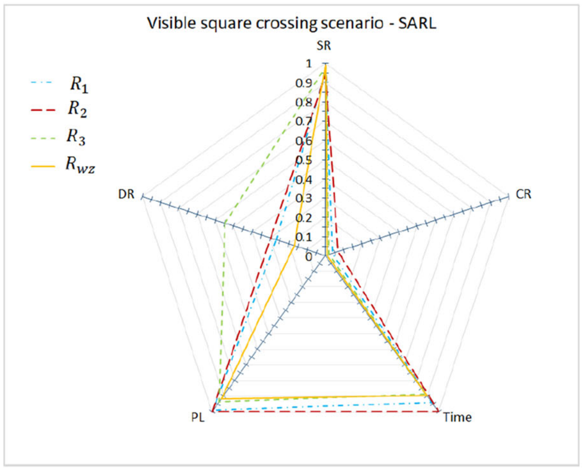
(c)



(d)

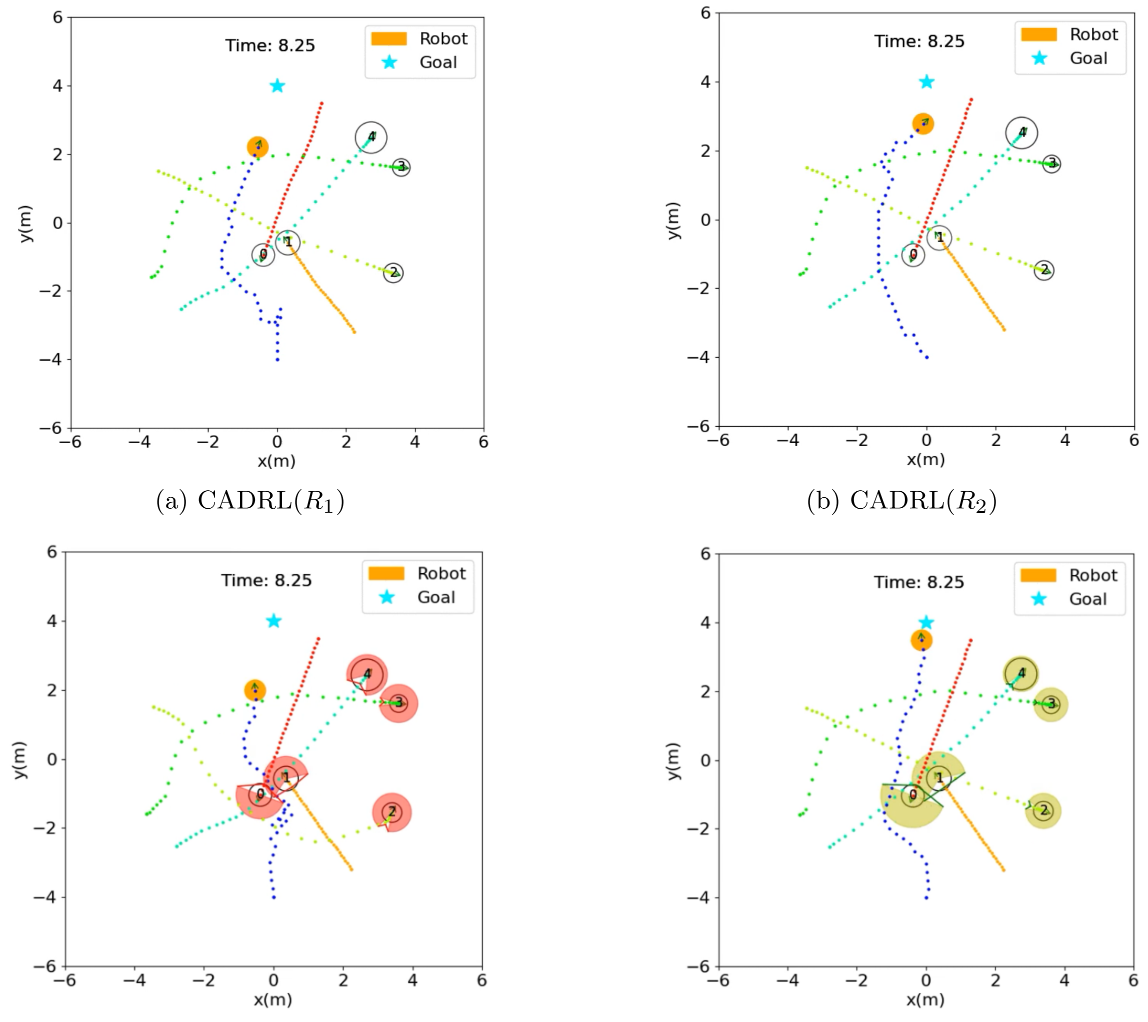


(e)



(f)

Fig. 5 Results from square crossing scenario with all methods plotted in spider graph. **a** CADRL model results in invisible mode. **b** CADRL model results in visible mode. **c** LSTM-RL model results in invisible mode. **d** LSTM-RL model results in visible mode. **e** SARL model results in invisible mode. **f** SARL model results in visible mode



(a) CADRL(R_1) (b) CADRL(R_2)

(c) CADRL(R_3) (d) CADRL(R_{wz})

Fig. 6 Local trajectory comparison with CADRL method in a random test episode with five humans in a circle crossing scenario. The experimental conditions have the same starting point, goal position, and time-step

Qualitative evaluation

For qualitative evaluation, all the methods are tested under the same experimental conditions such as the same starting point, goal position, and time-step. Figure 6 shows the CADRL performance in a random test episode with five humans in a circle crossing scenario. CADRL(R_{wz}) stops and moves backward to avoid the humans surrounding it. The robot exits without collision and avoids the warning zones around the humans, taking 8.25 s to reach the goal. CADRL(R_{wz}) tends to follow an even path in comparison to CADRL with previous formulations.

On the other hand, in LSTM-RL(R_{wz}), the robot goes backward and turns right. Later, the robot is heading to find the path to reach the goal. Figure 7 illustrates the trajectory taken by the robot with all the formulations, demonstrating that with the proposed reward formulation, the robot reaches the goal with less time and a smooth path. It can be seen that at 8.75 s, the robot reaches the goal with LSTM-RL(R_{wz}); while with other methods, the robot is still on the way towards the goal.

Figure 8a shows how SARL(R_1) makes greater deviations at first to avoid collisions with humans and therefore takes longer to find and reach the goal. During crossovers, SARL(R_2) model attempts to locate a path through the middle, causing the robot to be uncomfortably near and at a higher risk of collision, as shown in Fig. 8b. SARL(R_3) initially stops and backtracks, losing sight of the goal; finally, it chooses a path that intersects the humans, which leads it to do more maneuvers and takes more time to reach the goal, as is shown in Fig. 8c. The robot trajectory performed by SARL(R_{wz}) is shown in Fig. 8d. It is observed that SARL(R_{wz}) avoids all humans by a larger margin. The SARL(R_{wz}) behavior takes a larger detour to the left, esti-

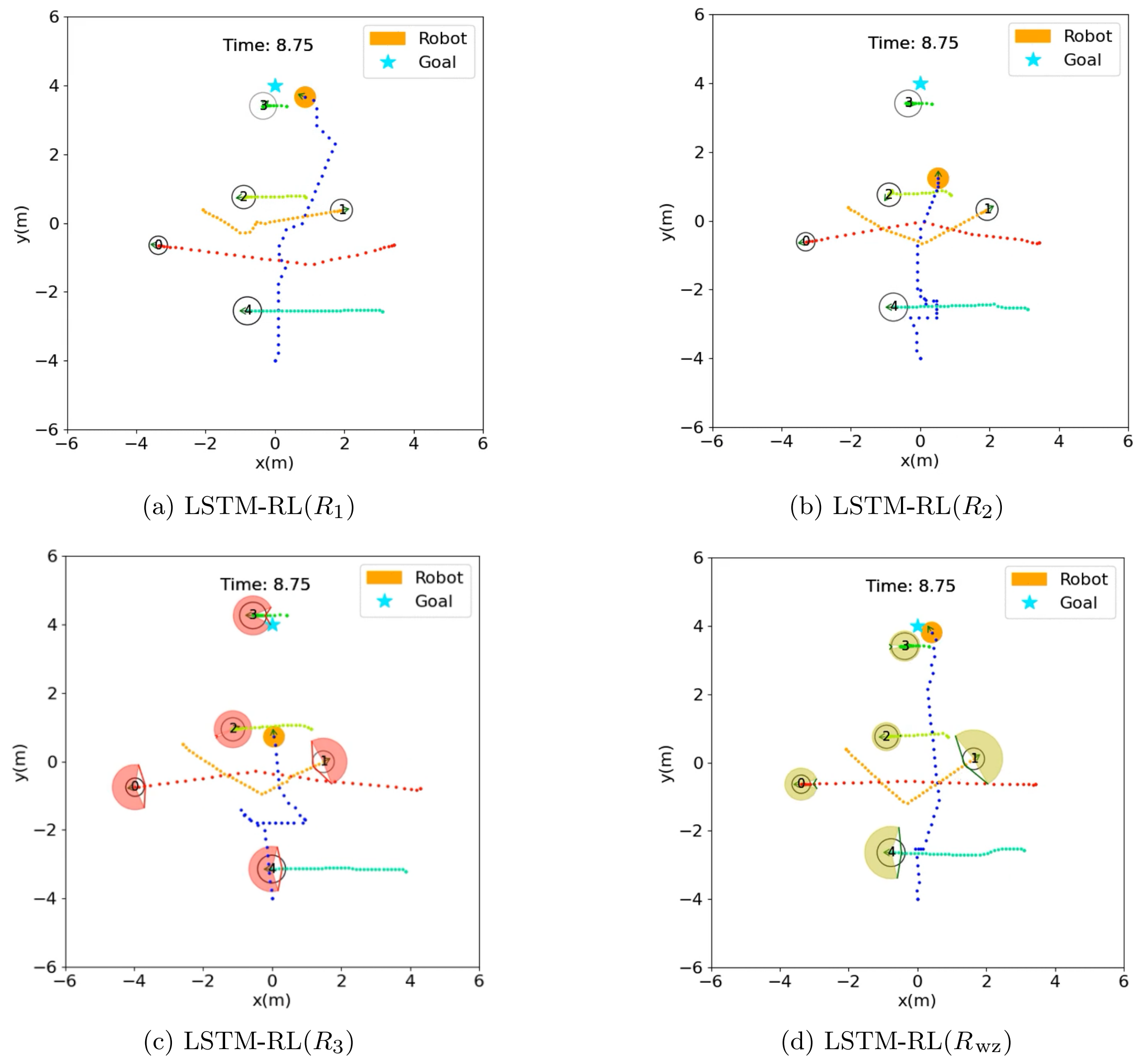


Fig. 7 Local trajectory comparison with lSTM-RL method in a random test episode with five humans in a square crossing scenario. The experimental conditions have the same starting point, goal position, and time-step

mating in advance the warning zones and managing to avoid front-on human contact.

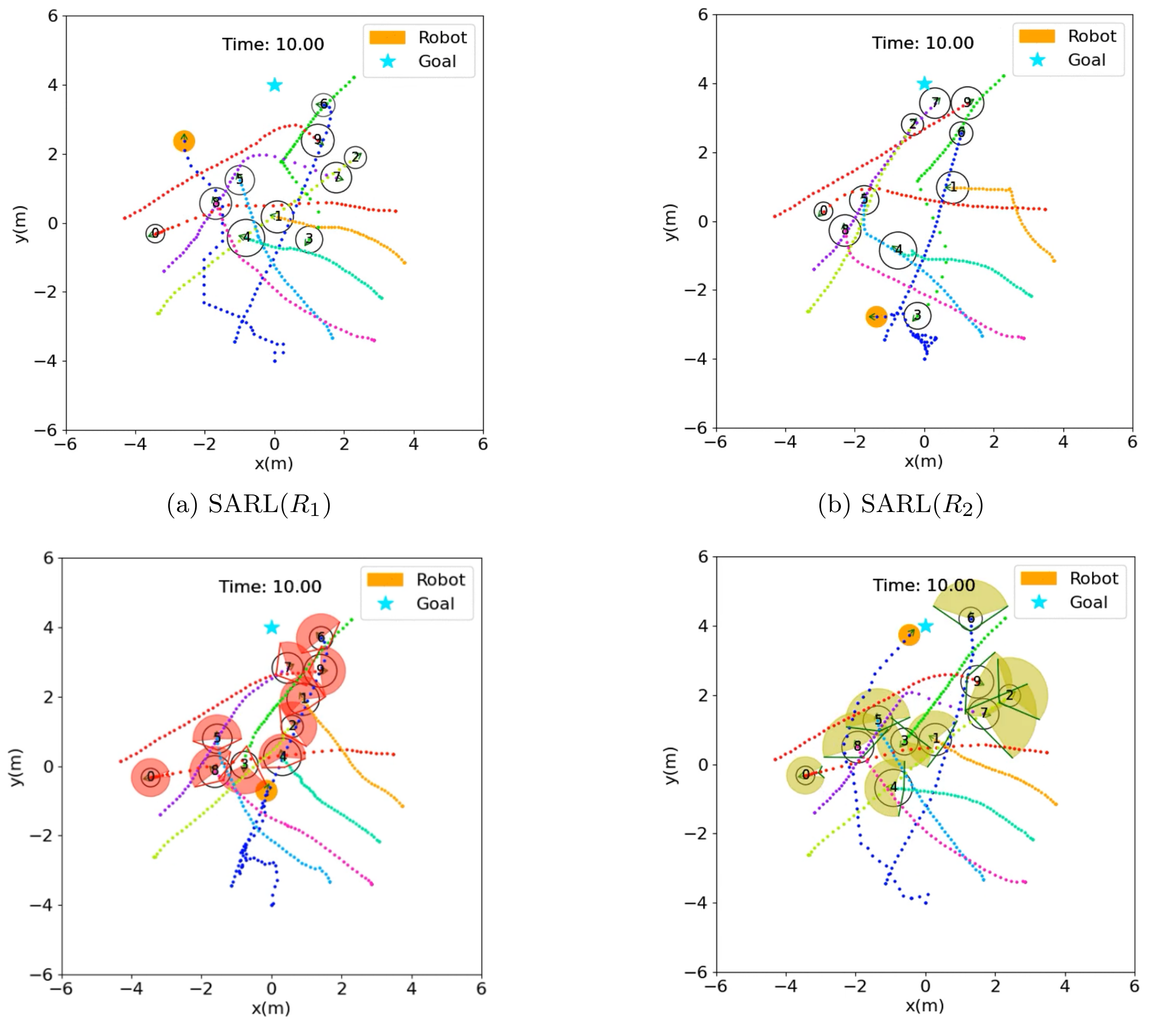
As seen from the experiments SARL shows the performance when combined with the proposed reward formulation. The robot moves away from the humans and takes open routes to guarantee safe navigation.

Statistical analysis

Statistical analysis is performed to assess the proposed method over baseline methods. To ensure reliable data comparison, the Kolmogorov–Smirnov test [48] was used to determine normality. The test's outcome revealed that several metrics had a non-Gaussian distribution, therefore, the non-parametric Wilcoxon signed-rank test is used to identify differences and significant levels. A test statistic value (i.e., z-score) is generated using Wilcoxon signed-rank, and this value is then converted into a p value. The smaller p value denotes a considerable variation between populations. The threshold of p value is 0.05, where if the p value is less than 0.05, pair 1 is less than pair 2, rejecting the null hypothesis and accepting the alternative hypothesis. The one-sided alternative hypothesis with the *less* option is taken into account. The results in Table 5 demonstrate that the success rate does not accept the null hypothesis; thus, the proposed method is more significant than other methods. In contrast, the metrics of collision rate, time, path length, and discomfort rate do not reject the null hypothesis, proving that the proposed method improves the metrics in comparison to previous methods.

Implementation and experimental setup

Since the robot considers all surrounding humans' dynamic warning zones, the risk of a robot freezing diminishes due to



(a) SARL(R_1) (b) SARL(R_2)

(c) SARL(R_3) (d) SARL(R_{wz})

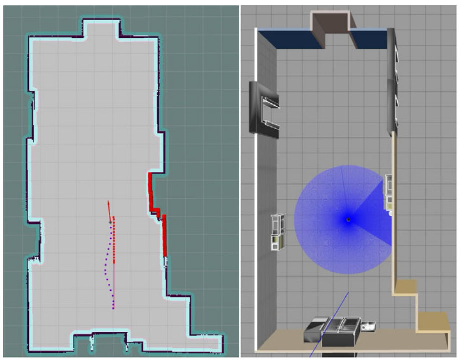
Fig. 8 Local trajectory comparison with SARL method in a random test episode with ten humans in a circle crossing scenario. The experimental conditions have the same starting point, goal position, and time-step

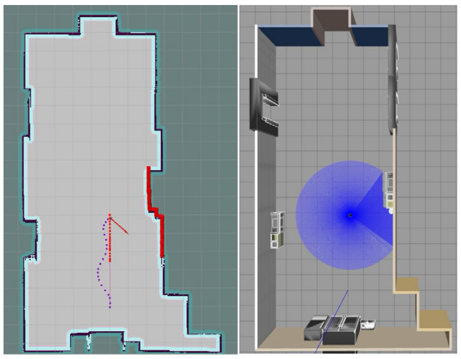
Table 5 Statistical analysis of quantitative evaluation results using Wilcoxon signed-rank test

Metrics	z-score	p value
Success rate	0	0.02847628
Collision rate	11.55	0.84790811
Time	17.05	0.93923611
Path length	16.22	0.92534722
Discomfort rate	13.33	0.74479167

the proposed reward function. Based on prior training results, superior outcomes were obtained with the SARL method given by its collective significance of the crowd. Therefore, to assess the effectiveness of SARL with our proposed method, Gazebo simulations, and real-time experiments are conducted.

The Turtlebot 3 is used for experimental tests, and it includes some hardware limitations to consider. Its limitations include the limited computational power of a Raspberry Pi 3, which constrains the computational capabilities to quickly process the actions of the robot. To address this limitation, the Raspberry Pi 3 is connected to a remote PC that processes real-time inputs related to the observable state of humans and the full state of the robot. Since the maximum velocity of the Turtlebot 3 is 0.23 m/s, the robot may be unable to take prompt action and avoid obstructions if the velocity of the human surpasses that of the robot. Therefore, to overcome this constraint, the robot was trained by considering that the maximum velocity the human can reach is 1.8 m/s. Originally, the Turtlebot 3 has a 2D laser distance sensor (LDS) with a sensor range of 3.5 m. To mitigate, the sensor range, the Turtlebot 3 was equipped with a Hokuyo LIDAR with a detection distance of 5.6 m and a 1% margin of error.


(a) SARL(R_1)


(b) SARL(R_2)

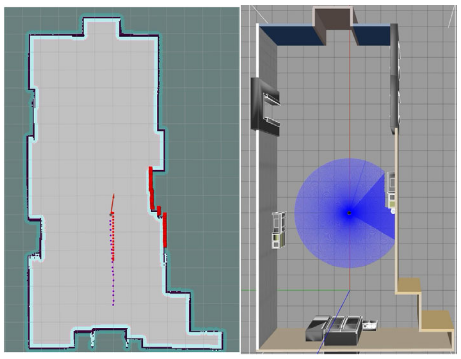

(c) SARL(R_{wz})

Fig. 9 Performance comparison of SARL(R_1), SARL(R_2), SARL(R_{wz}) in an empty scenario in Gazebo-ROS. The red dots denote the path that the robot must follow. The purple dots show the trajectory performed by the robot

Gazebo-ROS simulation

The reward functions R_{wz}, R_1 and R_2 from equations (11), (12) and (13), respectively, are evaluated in Gazebo simulations using ROS within three scenarios: empty space, static obstacles, and dynamic obstacles. The following packages are used for simulations including GMapping [49] for SLAM approach to obtain a 2D occupancy grid map of the environment by a 2D Laser Distance Sensor (LDS), *AMCL* package [50] for robot's localization, and the *leg-detector* package [51] for the position and velocity of the dynamic obstacles.

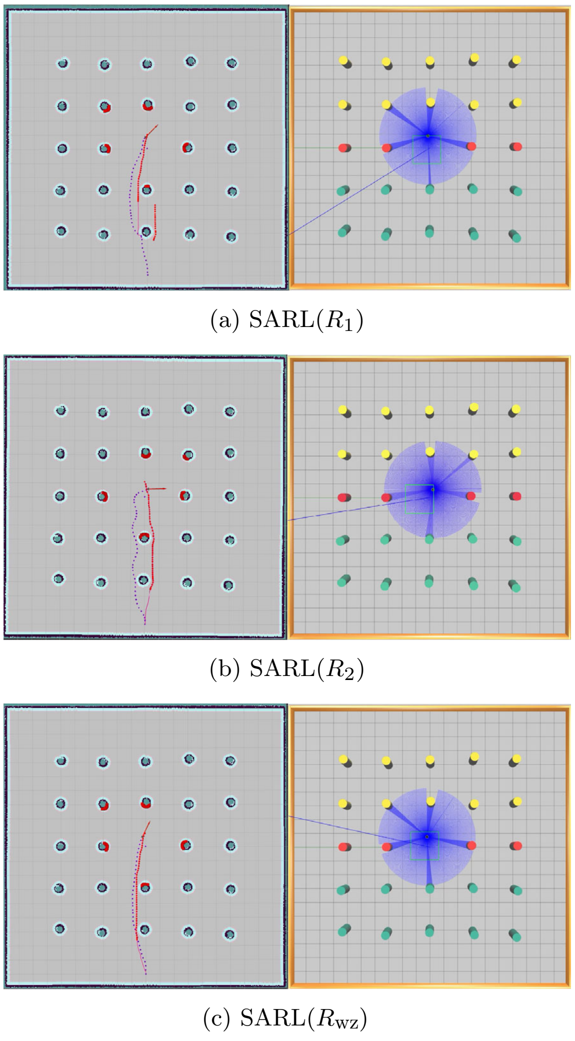
Figure 9 shows the robot navigation in an empty scenario with the goal distance of 5 m. The red dots denote the path that the robot must follow. The purple dots show the trajectory performed by the robot. Table 6 shows the results comparing the SARL approach with the reward functions R_1, R_2, and R_{wz} within 5, 10, and 15 m of goal distance. For all three models, the robot navigates at a similar speed. As the distance to the goals increased, the navigation time also increased. It appears that SARL(R_1) and SARL(R_2) are slow to follow the path. Comparatively, their courses become irregular, unlike SARL(R_{wz}), which can keep steady navigation without changing directions.

Figure 10 shows the comparison of the SARL approach with the three reward functions in a static obstacle scenario. The goal distance in this case is 10 m. The red dots denote the path that the robot must follow. The purple dots show the trajectory performed by the robot. Table 6 shows the results with 5, 10, and 15 m of goal distances. SARL(R_1) and SARL(R_2) take a longer time to reach the goal as compared to SARL(R_{wz}). The performance of SARL(R_{wz}) is better even if the goal is farther away. Figure 10 shows that SARL(R_1) and SARL(R_2) change their trajectory, indicating that the models do not follow the global planner of the Dijkstra algorithm, while the SARL(R_{wz}) follows the proposed short-distance goal method. Accordingly, SARL(R_{wz}) demonstrates high performance by achieving the most efficient results in terms of time, path planning, and obstacle avoidance.

Figure 11 shows the robot navigation with ten dynamic obstacles with 15 m of the goal distance. The red dots denote the path that the robot must follow. The purple dots show the trajectory performed by the robot. Table 6 shows the navigation results of the three scenarios for 5, 10, and 15 m of goal

Table 6 Navigation time (seconds) in empty, static obstacles, and dynamic obstacles scenario on Gazebo-ROS simulator

	Empty			Static obstacles			Dynamic obstacles		
	5 m	10 m	15 m	5 m	10 m	15 m	5 m	10 m	15 m
SARL(R_1)	10.74	20.75	31.34	11.49	21.74	45.74	13.50	39.75	81.24
SARL(R_2)	11.02	21.01	31.46	11.50	25.01	54.50	15.49	42.00	92.02
SARL(R_{wz})	10.25	20.50	30.30	10.74	20.50	34.75	11.00	26.74	63.51

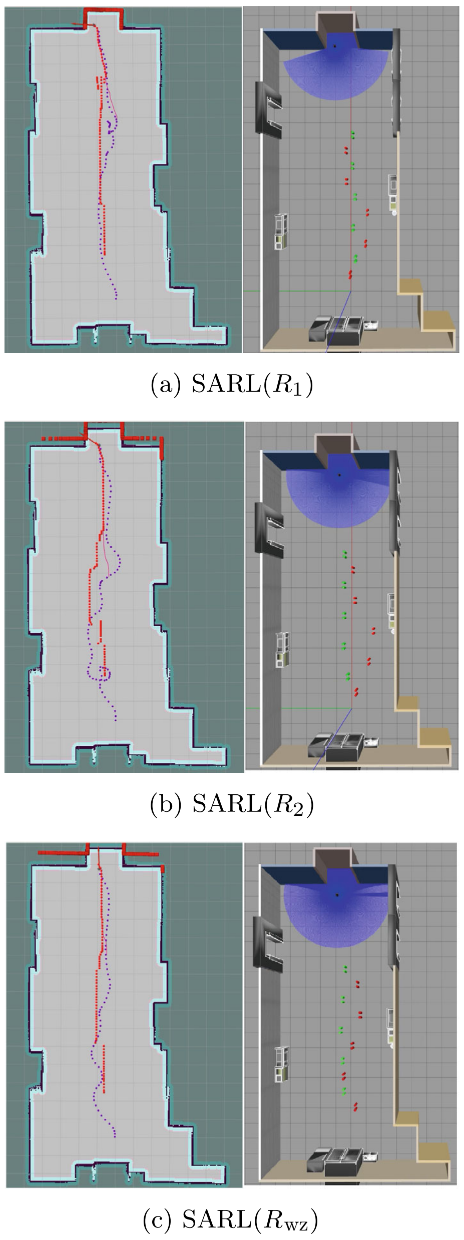


(a) SARL(R_1)

(b) SARL(R_2)

(c) SARL(R_{wz})

Fig. 10 Performance comparison of SARL(R_1), SARL(R_2), SARL(R_{wz}) in static obstacles scenario in Gazebo-ROS. The red dots denote the path that the robot must follow. The purple dots show the trajectory performed by the robot



(a) SARL(R_1)

(b) SARL(R_2)

(c) SARL(R_{wz})

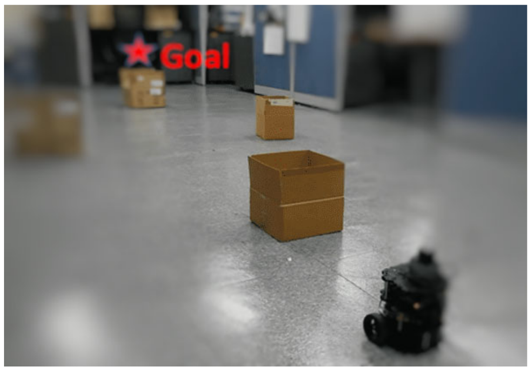
Fig. 11 Performance comparison of SARL(R_1), SARL(R_2), SARL(R_{wz}) in dynamic obstacles scenario in Gazebo-ROS. The red dots denote the path that the robot must follow. The purple dots show the trajectory performed by the robot

distance. To avoid the dynamic obstacles, SARL(R_1) turns at a large trajectory or moves continuously in the opposite direction of the goal when it perceives an approaching obstacle, as shown in Fig. 11a. SARL(R_2) model frequently freezes, requiring the robot to rotate in place to find an opportunity to escape, as shown in Fig. 11b. The R_{wz} allows the robot to react from a farther distance, preventing it from crossing over the circular sector around the dynamic obstacle. As a result, SARL(R_{wz}) demonstrates remarkable performance by avoiding dynamic obstacles and following the global path planner of the Dijkstra algorithm.

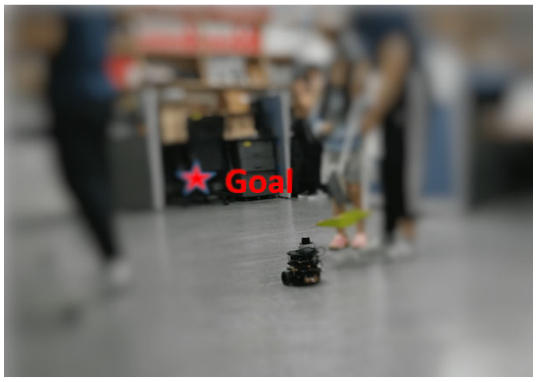
Real-time experiments

This section presents the real-time environment for the experimental tests, robot, and sensors. The information about packages used to validate the performance of autonomous navigation based on DRL is also provided. The robot tries to reach the goal in three scenarios: empty space, obstacles, and humans at a distance of 8 m. The Turtlebot 3, with a maximum velocity of 0.23 m/s, is used with a Hokuyo 2D scanner sensor with a 360-degree resolution for mapping and obstacle detection. An odometer is used to localize the robot's position. The *GMapping* package [49] is used for mapping. The *AMCL* package [50] is used in conjunction with the LiDAR and

(a) Empty scenario



(b) Scenario with obstacles



(c) Scenario with humans

Fig. 12 Real-world experiments in three scenarios: empty, static obstacles, and with humans

odometer for robot's localization, and the *obstacle-detector* package [52] is used to identify static and dynamic obstacles.

In the initial deployment, the robot with SARL(R_1) takes a long detour to the right, which causes it to veer off course and take longer to reach its destination, as shown in Fig. 12a. In comparison, using the proposed formulation SARL(R_{wz}), follows the route directly without detours, resulting in a quicker arrival time at the goal.

In the static obstacle scenario as shown in Fig. 12b, SARL(R_1) collides with the obstacles and takes longer to reach the goal. whereas, with SARL(R_{wz}), the robot avoids obstacles and takes a path that leads to its goal without collision.

Figure 12c shows the scenario with humans. The robot with SARL(R_1) is able to respect the comfortable distance and avoid colliding with people. However, on some occasions, the robot rotates in its own space and looks like it loses its path, resulting in a long time to reach the goal. Table 7 shows the experimental results in the three different scenarios, where SARL(R_{wz}) has a significantly lower arrival time, resulting in smoother trajectories and demonstrating fewer oscillations in comparison to SARL(R_1). SARL(R_{wz}) is the model that performs well in the human–robot coexistence scenario. When the robot perceives the human from the front, it stops and quickly looks in another direction to continue with the desired route. In this way, the robot anticipates collisions and navigates safely to reach its destination faster.

Conclusions

This manuscript proposes a safe navigation method for mobile robots using deep reinforcement learning. To teach the robot to maintain comfortable distances from humans, the proposed model formulates the reward function for warning zones based on the size and velocity of humans or dynamic obstacles. This allows the robot to anticipate and predict human motion while navigating in a socially acceptable manner. Furthermore, the distance between the robot's current position and the goal is evaluated and a reward system is implemented that penalizes the robot for moving away from the goal and rewards the robot for moving

Table 7 Experimental results in empty, static obstacles and human scenario

Metrics	Method	Empty scenario	Static scenario	Human scenario
Average time (s)	SARL(R_1)	34	46	52
	SARL(R_{wz})	27	38	35
Distance (m)	SARL(R_1)	7.82	10.58	11.96
	SARL(R_{wz})	6.21	8.74	8.05

closer to the goal. The robot is therefore able to reach the goal in a shorter distance, resulting in smoother trajectories and fewer oscillations. The proposed reward function is tested on three state-of-the-art methods, collision avoidance with deep reinforcement learning, long short-term memory, and social attention with reinforcement learning, and also tested in the Gazebo simulator and the real world with robot operating system (ROS) in three scenarios. The results of extensive experiments were evaluated both quantitatively and qualitatively. Compared to previous formulations, the results demonstrated better overall performance in all scenarios, increasing the robot's ability to reach the goal with the lowest collision rate, maintain safety, and minimize its arrival time. To improve the robot's navigation policy in the real world, our future study will exploit and improve the policy with more complicated settings of the environment such as dynamic human groups in a high-density crowd. We aim to encode complex interactions using memory-based neural networks to learn navigation policies that help to achieve collision-free and socially appropriate navigation in crowds of human groups.

Funding This article was funded by National Research Foundation of Korea (NRF- 2022R1G1A1012746).

Declarations

Conflict of interest The authors have no conflicts of interest to declare.

References

1. Alatise MB, Hancke GP (2020) A review on challenges of autonomous mobile robot and sensor fusion methods. IEEE Access 8:39830–39846
2. Pico N, Kim EC, Park SH et al (2022) Geometric recognition of diverse terrain in real-time for a six-wheeled robot based on laser scanning sensors. In: 2022 22nd international conference on control, automation and systems (ICCAS), pp 1924–1929. https://doi.org/10.23919/ICCAS55662.2022.10003923
3. Chiang AH, Trimi S (2020) Impacts of service robots on service quality. Serv Bus 14(3):439–459
4. Pico N, Hr Jung, Medrano J et al (2022) Climbing control of autonomous mobile robot with estimation of wheel slip and wheel-ground contact angle. J Mech Sci Technol 36(2):959–968
5. Choi Y, Choi M, Oh M et al (2020) Service robots in hotels: understanding the service quality perceptions of human–robot interaction. J Hosp Market Manag 29(6):613–635
6. Fragapane G, De Koster R, Sgarbossa F et al (2021) Planning and control of autonomous mobile robots for intralogistics: literature review and research agenda. Eur J Oper Res 294(2):405–426
7. Pico N, Park SH, Luong T et al (2022) Terrain recognition based on the wheel contact angle measurement using laser scanners for six-wheel mobile robot. In: 2022 19th international conference on ubiquitous robots (UR). IEEE, pp 23–29
8. Omeiza D, Webb H, Jirotka M et al (2021) Explanations in autonomous driving: a survey. IEEE Trans Intell Transp Syst 23(8):10142–10162
9. Liu B, Xiao X, Stone P (2021) A lifelong learning approach to mobile robot navigation. IEEE Robot Autom Lett 6(2):1090–1096
10. Xiao X, Liu B, Warnell G, Stone P (2022) Motion planning and control for mobile robot navigation using machine learning: a survey. Auton Robots 46(5):.569–597
11. Pico N, Park SH, Js Yi et al (2022) Six-wheel robot design methodology and emergency control to prevent the robot from falling down the stairs. Appl Sci 12(9):4403
12. Zhang B, Amirian J, Eberle H, Pettrè J, Holloway C, Carlson T (2021) From HRI to CRI: crowd robot interaction–understanding the effect of robots on crowd motion: Empirical Study of Pedestrian Dynamics with a Wheelchair and a Pepper Robot. Int J Soc Robotics 1–13
13. Möller R, Furnari A, Battiato S et al (2021) A survey on human-aware robot navigation. Robot Auton Syst 145(103):837
14. Rudenko A, Palmieri L, Herman M et al (2020) Human motion trajectory prediction: a survey. Int J Robot Res 39(8):895–935
15. Vemula A, Muelling K, Oh J (2018) Social attention: modeling attention in human crowds. In: 2018 IEEE international conference on robotics and automation (ICRA). IEEE, pp 4601–4607
16. Gupta A, Johnson J, Fei-Fei L et al (2018) Social GAN: socially acceptable trajectories with generative adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. IEEE, pp 2255–2264
17. Trautman P, Krause A (2010) Unfreezing the robot: navigation in dense, interacting crowds. In: 2010 IEEE/RSJ international conference on intelligent robots and systems. IEEE, pp 797–803
18. Sathyamoorthy AJ, Patel U, Guan T et al (2020) Frozone: freezing-free, pedestrian-friendly navigation in human crowds. IEEE Robot Autom Lett 5(3):4352–4359
19. Bacanin N, Zivkovic M, Bezdan T et al (2022) Modified firefly algorithm for workflow scheduling in cloud-edge environment. Neural Comput Appl 34(11):9043–9068
20. Sun H, Zhang W, Yu R et al (2021) Motion planning for mobile robots-focusing on deep reinforcement learning: a systematic review. IEEE Access 9:69061–69081
21. Luong M, Pham C (2021) Incremental learning for autonomous navigation of mobile robots based on deep reinforcement learning. J Intell Robot Syst 101(1):1
22. Cheng G, Wang Y, Dong L, Cai W, Sun C (2023) Multi-objective deep reinforcement learning for crowd-aware robot navigation with dynamic human preference. Neural Comput Appl 1–19
23. Chen YF, Everett M, Liu M et al (2017) Socially aware motion planning with deep reinforcement learning. In: 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 1343–1350
24. Everett M, Chen YF, How JP (2018) Motion planning among dynamic, decision-making agents with deep reinforcement learning. In: 2018 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 3052–3059

25. Cimurs R, Suh IH, Lee JH (2021) Goal-driven autonomous exploration through deep reinforcement learning. IEEE Robot Autom Lett 7(2):730–737
26. Chen C, Liu Y, Kreiss S et al (2019) Crowd–robot interaction: crowd-aware robot navigation with attention-based deep reinforcement learning. In: 2019 international conference on robotics and automation (ICRA). IEEE, pp 6015–6022
27. Chen Y, Liu C, Shi BE et al (2020) Robot navigation in crowds by graph convolutional networks with attention learned from human gaze. IEEE Robot Autom Lett 5(2):2754–2761
28. Li K, Xu Y, Wang J et al (2019) Sarl*: deep reinforcement learning based human-aware navigation for mobile robot in indoor environments. In: 2019 IEEE international conference on robotics and biomimetics (ROBIO). IEEE, pp 688–694
29. Francis A, Faust A, Chiang HTL et al (2020) Long-range indoor navigation with PRM-RL. IEEE Trans Rob 36(4):1115–1134
30. Shi W, Zhou Y, Zeng X et al (2022) Enhanced spatial attention graph for motion planning in crowded, partially observable environments. In: 2022 international conference on robotics and automation (ICRA). IEEE, pp 4750–4756
31. Kretz T, Lohmiller J, Sukennik P (2018) Some indications on how to calibrate the social force model of pedestrian dynamics. Transp Res Rec 2672(20):228–238
32. Van den Berg J, Lin M, Manocha D (2008) Reciprocal velocity obstacles for real-time multi-agent navigation. In: 2008 IEEE international conference on robotics and automation. IEEE, pp 1928–1935
33. Van Den Berg J, Guy SJ, Lin M et al (2011) Reciprocal n-body collision avoidance. In: Robotics research: the 14th international symposium ISRR. Springer, pp 3–19
34. Shakya AK, Pillai G, Chakrabarty S (2023) Reinforcement learning algorithms: a brief survey. Expert Syst Appl 120495
35. Cao C, Trautman P, Iba S (2019) Dynamic channel: a planning framework for crowd navigation. In: 2019 international conference on robotics and automation (ICRA). IEEE, pp 5551–5557
36. Faust A, Oslund K, Ramirez O et al (2018) PRM-RL: long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning. In: 2018 IEEE international conference on robotics and automation (ICRA). IEEE, pp 5113–5120
37. Samsani SS, Mutahira H, Muhammad MS (2023) Memory-based crowd-aware robot navigation using deep reinforcement learning. Complex Intelll Syst 9(2):2147–2158
38. Long P, Liu W, Pan J (2017) Deep-learned collision avoidance policy for distributed multiagent navigation. IEEE Robot Autom Lett 2(2):656–663
39. Tai L, Zhang J, Liu M et al (2018) Socially compliant navigation through raw depth inputs with generative adversarial imitation learning. In: 2018 IEEE international conference on robotics and automation (ICRA). IEEE, pp 1111–1117
40. Choi J, Park K, Kim M et al (2019) Deep reinforcement learning of navigation in a complex and crowded environment with a limited field of view. In: 2019 international conference on robotics and automation (ICRA). IEEE, pp 5993–6000
41. Liu L, Dugas D, Cesari G et al (2020) Robot navigation in crowded environments using deep reinforcement learning. In: 2020 IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 5671–5677
42. Yang Y, Jiang J, Zhang J, Huang J, Gao M (2023) ST^2: Spatial-temporal state transformer for crowd-aware autonomous navigation. IEEE Robotics and Autom Lett 8(2), pp.912–919
43. Liu S, Chang P, Liang W et al (2021) Decentralized structural-RNN for robot crowd navigation with deep reinforcement learning. In: 2021 IEEE international conference on robotics and automation (ICRA). IEEE, pp 3517–3524
44. Feigl T, Kram S, Woller P et al (2020) RNN-aided human velocity estimation from a single IMU. Sensors 20(13):3656
45. Craig JJ, Bruetsch AP, Huisinga JM (2019) Coordination of trunk and foot acceleration during gait is affected by walking velocity and fall history in elderly adults. Aging Clin Exp Res 31(7):943–950
46. Samsani SS, Muhammad MS (2021) Socially compliant robot navigation in crowded environment by human behavior resemblance using deep reinforcement learning. IEEE Robot Autom Lett 6(3):5223–5230
47. Pak J, Kim J, Park Y et al (2022) Field evaluation of path-planning algorithms for autonomous mobile robot in smart farms. IEEE Access 10:60253–60266
48. Demir S (2022) Comparison of normality tests in terms of sample sizes under different skewness and kurtosis coefficients. Int J Assess Tools Educ 9(2):397–409
49. Gerkey B (2021) Gmapping ROS package. https://wiki.ros.org/gmapping/. Accessed: 2022-10-17
50. Gerkey B (2022) AMCL ROS package. https://wiki.ros.org/amcl/. Accessed: 2022-10-17
51. Pantofaru C (2022) Leg detector ROS package. https://wiki.ros.org/legdetector/. Accessed: 2022-10-17
52. Przybyla M (2022) Obstacle detector ROS package. https://github.com/tysik/obstacledetector. Accessed: 2022-10-17