Received: 28 April 2023 | Revised: 5 July 2023 | Accepted: 3 August 2023

CAAI Transactions on Intelligence Technology

DOI: 10.1049/cit2.12269

ORIGINAL RESEARCH

A safe reinforcement learning approach for autonomous navigation of mobile robots in dynamic environments

Zhiqian Zhou[1] | Junkai Ren[1] | Zhiwen Zeng[1] | Junhao Xiao[1] | Xinglong Zhang[1] | Xian Guo[2] | Zongtan Zhou[1] | Huimin Lu[1]

[1]College of Intelligence Science and Technology, National University of Defense Technology, Changsha, China

[2]Institute of Robotics and Automatic Information System, College of Artificial Intelligence, NanKai University, Tianjin, China

Correspondence

Zhiwen Zeng and Junhao Xiao.
Email: zengzhiwen@nudt.edu.cn and junhao.xiao@ieee.org

Funding information

National Natural Science Foundation of China, Grant/Award Numbers: 62203460, U1913202, U22A2059

[Correction added on 14 Oct 2023, after first online publication. The below correction needs to be noted.

The affiliation of authors Zhiqian Zhou, Junkai Ren, Zhiwen Zeng, Junhao Xiao, Xinglong Zhang, Zongtan Zhou and Huimin Lu is corrected as below:

"College of Intelligence Science and Technology, National University of Defense Technology, Changsha, China"]

Abstract

When deploying mobile robots in real-world scenarios, such as airports, train stations, hospitals, and schools, collisions with pedestrians are intolerable and catastrophic. Motion safety becomes one of the most fundamental requirements for mobile robots. However, until now, efficient and safe robot navigation in such dynamic environments is still an open problem. The critical reason is that the inconsistency between navigation efficiency and motion safety is greatly intensified by the high dynamics and uncertainties of pedestrians. To face the challenge, this paper proposes a safe deep reinforcement learning algorithm named Conflict-Averse Safe Reinforcement Learning (CASRL) for autonomous robot navigation in dynamic environments. Specifically, it first separates the collision avoidance sub-task from the overall navigation task and maintains a safety critic to evaluate the safety/risk of actions. Later, it constructs two task-specific but model-agnostic policy gradients for goal-reaching and collision avoidance sub-tasks to eliminate their mutual interference. Then, it further performs a conflict-averse gradient manipulation to address the inconsistency between two sub-tasks. Finally, extensive experiments are performed to evaluate the superiority of CASRL. Simulation results show an average 8.2% performance improvement over the vanilla baseline in eight groups of dynamic environments, which is further extended to 13.4% in the most challenging group. Besides, forty real-world experiments fully illustrated that the CASRL could be successfully deployed on a real robot.

KEYWORDS

autonomous navigation, dynamic environments, gradient-based policy optimisation, mobile robots, safe reinforcement learning

1 | INTRODUCTION

With the deployment of an increasing number of robots for various service tasks in the past decades, there has been a growing interest in robotics regarding the autonomous navigation of mobile robots in real-world dynamic environments. In the problem, a crucial bottleneck is how to ensure the motion safety of mobile robots. With various dynamic obstacles (e.g., pedestrians, other robots, and vehicles), the environment is time-varying, and it becomes difficult to find time-efficient and collision-free paths. One natural idea is to expand traditional autonomous navigation algorithms with the temporal dimension. It first predicts trajectories of dynamic obstacles over the coming period and then constructs time-varying obstacle constraints for the subsequent planning sessions [1–3]. However, both trajectory predicting and optimising with time-varying constraints are time-consuming. Besides, the two-stage framework may lead to the freezing robot problem [4] in which a great number of possible trajectories fill up the space, eliminating any plausible path.

Meanwhile, benefiting from the impressive non-linear fitting capacity of deep neural networks, deep reinforcement learning (DRL) has shown impressive potential in autonomous navigation, especially in dynamic and complex environments [5–13]. They formulate the robot autonomous navigation problem as a Markov decision process (MDP), and solve it via reinforcement learning (RL). Their core idea is to continuously improve the navigation policy based on reward signals from interactions with the environment [14]. Therefore, their performance highly depends on a proper task-specific reward function.

Since autonomous robot navigation is essentially a multi-objective optimisation problem, the corresponding reward function always consists of multiple components for different objectives. Some widely-used reward components include rewards on reaching or approaching the goal [10, 15] and penalties on collisions with obstacles, violations to social norms [16], or predictable collisions [17, 18]. The former is built to navigate the robot to its goal as fast as possible, while the latter is built to prevent the robot from unsafe or high-risky states. Though these reward components can be integrated into the overall navigation task by introducing a minus to penalties, they are not always consistent and sometimes even conflicted. In simple scenarios, reward shaping via carefully weighting multiple components is enough to balance the time efficiency and motion safety [9, 15, 16]. However, in more realistic and complex dynamic environments, it is hard to design a proper reward function manually [10, 17, 18]. There exists a mutual interface among different objectives. As a result, though DRL-based approaches are capable of performing better than traditional methods in navigation efficiency, they fail to deal with the motion safety problem, and most of them are limited in laboratory environments [10, 11, 15].

Over the past few years, extensive work from safe reinforcement learning (SRL) has been proposed to improve the safety of DRL-based approaches, which coincidentally detach safety components from the original reward function. As shown in Figure 1, constraint-based approaches extend MDP into constrained MDP and explicitly construct a constraint on safety. By rescaling policy gradients with Lagrange multipliers [20–22], or introducing a safety-constraint-based penalty to the policy objective [23, 24], they can limit constraint violations under a given threshold. However, such a threshold is somewhat contrary to the collision-free requirement in autonomous robot navigation. Apart from them, some structure-based approaches maintain the safety critic to estimate the risk of actions or state-action pairs. Once the risk is over a given threshold, unsafe actions will be projected into safe zones via predefined rules [25, 26]. However, since such an action editing function is also hard to determine, the latest researches furthermore design additional task-specific policy networks [27, 28] or modules [19] as the action editor.

This work follows the idea of task decomposition from previous SRL methods [19–21] but further separates sub-tasks in the aspect of policy gradient. In particular, it first decomposes the navigation task into two sub-tasks: goal-reaching and collision avoidance by grouping reward components by

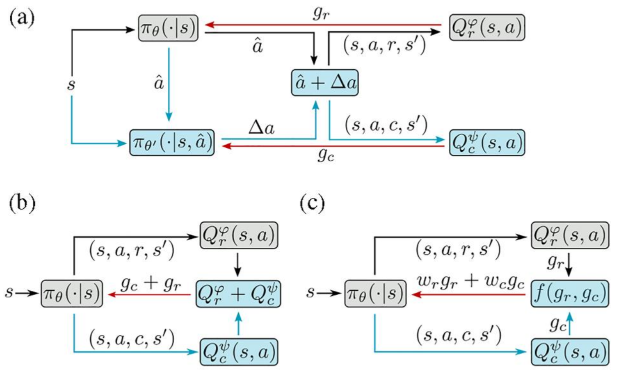


FIGURE 1 Three frameworks of SRL. Paths and blocks in cyan highlight their difference from the standard DRL. Paths in red indicate the back-forward propagation of gradients. The structure-based framework maintains task-specific modules for the collision avoidance sub-task [19]. Constraint-based solutions sum up two value functions and then generate a policy gradient for the policy network without any consideration of the inconsistency between two sub-tasks [20, 21]. The proposed gradient-based framework constructs an optimisation problem to determine the final policy gradient $f(g_r, g_c) = w_r g_r + w_c g_c$, which balances two sub-tasks by weighting policy gradients for sub-tasks. (a) Structure-based SRL. (b) Constraint-based SRL. (c) Gradient-based SRL.

sub-tasks, which is the common basis of extensive SRL methods. Later, from the grouped rewards, a critic and another safety critic can be learnt for the goal-reaching and collision avoidance sub-task, respectively. Based on them, this work further constructs two policy gradients to explicitly separate the two sub-tasks. And then, to deal with the inconsistency between the two sub-tasks, it proposes an optimisation objective on maximising the worse local policy improvement in sub-tasks. Under the guidance of the above ideas, this work presents an SRL approach for autonomous robot navigation in dynamic environments and shows its superior via extensive simulation experiments. Finally, a real-world experiment proves that the proposed algorithm can be migrated effectively to real-world applications. To sum up, the contribution of this work is threefold.

- A gradient-based SRL framework for autonomous robot navigation in dynamic environments is designed, which constructs two task-specific but model-agnostic policy gradients to eliminate the mutual interference between goal-reaching and collision avoidance sub-tasks. The proposed framework can be combined with all existing DRL-based approaches for autonomous robot navigation.
- A novel conflict-averse policy optimisation method is presented, which can effectively reconcile the inconsistency between two sub-tasks and ensure an efficient and stable training process. To our best knowledge, it is the first work to improve motion safety in autonomous robot navigation using gradient manipulations.
- A practical autonomous robot navigation algorithm named Conflict-Averse Safe Reinforcement Learning (CASRL) is proposed, which outperforms the vanilla algorithm by an

8.2% success rate and fully illustrates the effectiveness of the proposed two innovations.

This paper is structured as follows. Section 2 presents related work, and Section 3 depicts the proposed CASRL algorithm. Section 4 verifies the effectiveness of CASRl with extensive simulation and real-world experiments. Finally, Section 5 concludes the whole work.

2 | RELATED WORK

In this section, we first briefly review recent DRL-based approaches for autonomous robot navigation in dynamic environments. And then, we go further to highlight related work on SRL in robotics and gradient-based optimisation in multi-task learning, which directly inspire this work.

2.1 | DRL for robot navigation in dynamic environments

Encouraged by the superior performance in video games [29], DRL methods have been intensively studied for autonomous robot navigation in dynamic environments over the last few years. Sensor-level methods take the raw perceptual information from laserscans and cameras as inputs of networks and then learn a sensorimotor control policy by RL [30, 31]. With an accurate sensor simulation model, their policies learnt from the simulation environment can be migrated to real robots. However, due to the lack of low-level motion properties and high-level interaction models for obstacles, their decisions are inherently reactive and may lead to oscillatory motions in complex dynamic environments, such as crowd scenarios with many pedestrians. To address the problem, the authors in Refs. [8, 9, 32] unanimously utilise another network to estimate motion properties of moving objects, named world transition models, by supervised learning, which greatly enhance mobile robots' ability to avoid collisions proactively. Yet, world transition models are far from enough to model sophisticated interactions in real-world dynamic environments. Besides, since world transition models are always constructed in the forms of images or occupied maps, either the training or inference of the world transition model is time-consuming.

To achieve more proactive and foresighted avoidance of dynamic obstacles, some researchers turn their attention to the agent-level representation of surrounding environments and focus on modelling interactions among agents. These methods follow the classic two-stage framework, which detects obstacles at first and then learns collision avoidance policy from the processed agent-level representation [15, 16]. The separation of detection and navigation makes it possible to perform a higher-level inference than sensor-level methods. Representative achievements include the relational graph model in Ref. [33], the human-like gaze model in Ref. [5], and the social attention model in Ref. [10]. They enable mobile robots to assess disturbances from surrounding obstacles and identify critical threats to the robot. Based on them, some researchers furthermore introduce historical trajectory information [11, 34], semantic information (e.g. types of obstacles) [18, 35], and velocity-obstacle-based interactive information [17, 18] to help mobile robots understand scenarios better. Though the agent-level representation is manually designed, experimental results show that this high-level information, especially interactive information from traditional collision avoidance models, plays an important role in improving the motion safety of mobile robots in dynamic environments.

Although these methods achieve continuous performance improvements, they suffer from a common problem from the scalar reward function. Rewards for arriving at and approaching the goal, and penalties for collisions with obstacles and high-risk states, are all integrated into a scalar reward function. However, the navigation task is inherently multi-objective, and different objectives may be inconsistent or conflicted. A scalar reward function may result in mutual interference among different objectives. Therefore, reward shaping, via carefully fine-tuning reward weights [11, 33, 34] or introducing other so-called reasonable reward components [16–18], is a critical point for their final performance in motion safety.

To deal with the mutual interference among different objectives, this work decomposes the navigation task into two sub-tasks: goal-reaching and collision avoidance, and correspondingly groups rewards/penalties. Then two policy gradients are constructed from learnt critic and safety critic to separate the two sub-tasks. Finally, a gradient-based policy optimisation method is proposed to resolve the inherent inconsistency between goal-reaching and collision avoidance sub-tasks. In practical implementation, considering that modelling interactions among agents is the key to dealing with the collision avoidance problem in dynamic environments, a heterogeneous graph representation from Ref. [18] is used to describe navigation scenarios.

2.2 | SRL in robotic

Over the past few years, although much promising progress has been made by DRL in robot navigation, the motion safety problem of robots is still open, especially in real-world applications. To address the safety problem, many researchers are turning their attention to SRL. Their key innovations are twofold: separating cost functions (related to different safety constraints) from the scalar reward function in traditional MDP and constructing explicit constraints on costs in the optimisation objective [20–24]. The former greatly alleviates the problem of balancing safety and performance carefully with a problematic hand-designed scalar reward function. Meanwhile, the latter can theoretically prevent the optimised policy from constraint violations.

However, unlike constraints on velocity, acceleration, or energy consumption, a positive threshold for the collision avoidance constraint means a permissible possibility of collisions. On the one hand, it is hard to define the threshold of motion safety [20–23]. On the other hand, the positive threshold is contrary to the collision-free requirement of autonomous robot navigation. Meanwhile, methods using a Lagrange multiplier to balance time efficiency and motion safety, it is necessary to evaluate the safety performance online [20–22]. Finally, current benchmark environments in SRL consider only static or simple dynamic obstacles. There is rarely any interaction between obstacles in these environments, which differs from real-world navigation scenarios. These problems make it difficult to deploy these SRL algorithms in real-world autonomous robot navigation scenarios [36].

Meanwhile, other researchers concentrate on editing unsafe actions. They also separate the cost/penalty function for collision and construct a safety critic to evaluate the expected cumulative cost. In deployment, the robot can identify high-risk actions with the safety critic, and then project them into safe zones with predefined rules [25, 26]. However, predefined collision avoidance policies do not necessarily work by themselves in dynamic or complex environments. Therefore, a two-policy framework is proposed in Ref. [27], which learns a task policy to maximise the task reward under non-emergency cases and a recovery policy to minimise collision probability under emergency cases. Another similar work is the safety editor framework proposed in Ref. [19]. It also trains two policies: a utility maximiser to maximise the utility reward and a safety editor to adjust unsafe actions from the utility maximiser into safe actions. Benefiting from the explicit structural separation between the primary task and the collision avoidance task, both considerably improve safety.

Inspired by the above research work [19–21, 26], this work also divides the robot navigation task into two sub-tasks: the primary goal-reaching sub-task and the secondary collision avoidance sub-task. On this basis, we further propose a new optimisation objective to reconcile the inconsistencies between two sub-tasks. Compared with existing methods, there is no need to set an upper bound and perform an online evaluation for safety performance or introduce additional task-specific modules to edit unsafe actions.

2.3 | Gradient-based optimisation in multi-task learning

Multi-task learning (MTL) aims to simultaneously deal with multiple tasks with a single model [37–39]. Compared to learning with separate models, MTL methods achieve better data and computational efficiency by a smaller model size with sharing parameters across tasks. However, since objectives of multiple tasks are always inconsistent, a challenging optimisation problem is to harmonise multi gradients with varying directions and magnitudes [40–42].

To tackle the MTL problem, several architectural solutions have been proposed based on task-specific modules [43], attention mechanisms [38], or activating different paths along deep networks [44]. Apart from reconstructing network architectures, another branch of methods decomposes a large task into smaller but general sub-task modules and then aligns learning targets of tasks and sub-tasks modules with knowledge distillation [45].

Recently, considering that over-parameterised networks have adequate capacity to allow harmonious joint training for multiple tasks, a number of gradient-based methods focus on balancing the learning process of each task via manipulation on task-specific but model-agnostic gradients. Representative work includes MGDA-UB [37], CAGrad [41], PCGrad [40], and Meta-GF [42]. Among them, both MGDA-UB and CAGrad formulate MTL as a multi-objective optimisation problem and focus on maximising the worse local policy improvement of sub-tasks. Their difference lies in the search area of parameters. PCGrad identifies conflicting gradients with cosine similarity and then suppresses the interfering components. Meta-GF takes a meta-learnt weighted fusion policy to combine multi-task gradients. Experimental results verify that all of them can effectively improve the learning process on a series of challenging multi-task supervised, semi-supervised, and RL problems.

3 | METHODOLOGY

This section proposes a novel safe reinforcement learning approach named CASRL. As shown in Figure 2, the network consists of three components: an actor, a critic, and a safety critic. Firstly, the critic is trained with rewards from the goal-reaching sub-task, while the safety critic is trained with costs from collision cases. Then, by substituting the learnt critic and safety critic into the policy gradient equation, two task-specific but model-agnostic policy gradients are obtained for both sub-tasks. Finally, via a conflict-averse gradient manipulation of maximising the worst local policy improvement of sub-tasks, the inconsistency between two sub-tasks can then be alleviated in policy optimisation. Additionally, in practical implementation, the framework is built on the heterogeneous graph representation of states. Details of the approach are as follows.

3.1 | Safe Markov decision process

The critical point to improve motion safety is the notion of safety for target scenarios. Therefore, this work formulates the autonomous navigation problem in the form of safe MDP proposed in Refs. [21, 26]. The safe MDP can be represented with a tuple of $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \gamma_r, \mathcal{R}, \gamma_c, \mathcal{C} \rangle$. Compared with the standard MDP, the key improvement is that the single unified reward function in standard MDP is divided into two portions, the reward function $\mathcal{R}: \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ for the goal-reaching sub-task and the cost function $\mathcal{C}: \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ for the collision avoidance sub-task. Their discounted factors are denoted by γ_r and γ_c. For a clear comparison to the reward function, the penalty for unsafe states and actions is used to build the cost function $\mathcal{C}$. It means that the cost value is set positive when a collision occurs.

Due to the distinct separation of sub-tasks, the policy optimisation process can also be adapted accordingly. Firstly,

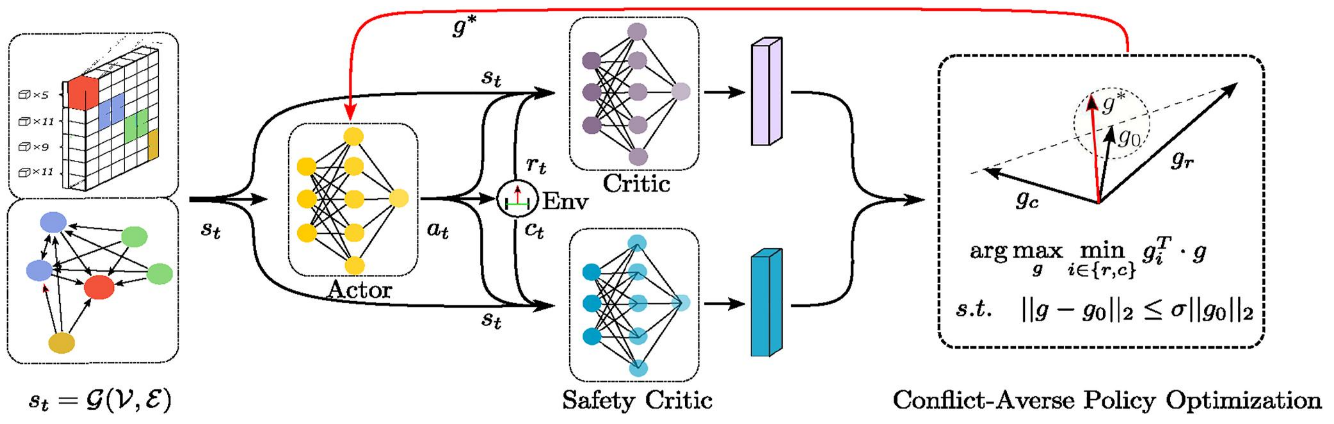

g^*
s_t
s_t
s_t
a_t
r_t
c_t
Env
Actor
Critic
Safety Critic
g^*
g_0
g_r
g_c
$\arg\max_g \min_{i\in\{r,c\}} g_i^T \cdot g$
$s.t. \quad ||g - g_0||_2 \le \sigma ||g_0||_2$
$s_t = \mathcal{G}(\mathcal{V}, \mathcal{E})$
Conflict-Averse Policy Optimization

FIGURE 2 The overall framework of CASRL.

the optimisation objective can be explicitly divided into two components: maximising the expected cumulative returns $J_r^{\pi_\theta}$ while minimising the expected cumulative costs $J_c^{\pi_\theta}$.

$$\begin{aligned} J_r^{\pi_\theta} &= E_\tau \sum_{t=0}^{T} \left[\gamma_r^t \mathcal{R}(s_t, a_t, s_{t+1})\right] \\ J_c^{\pi_\theta} &= E_\tau \sum_{t=0}^{T} \left[\gamma_c^t \mathcal{C}(s_t, a_t, s_{t+1})\right] \end{aligned} \tag{1}$$

where $\tau = s_0, a_0, \ldots, a_{t-1}, s_t$ is a trajectory following policy π_θ, and a_t is the action sampled from policy $\pi_\theta(\cdot\,|\,s_t)$.

The two optimisation objectives are built for the goal-reaching sub-task and collision avoidance sub-task. Then, it is available to differentiate different sub-tasks in the aspect of policy gradient. For the goal-reaching sub-task, its deterministic policy gradient is

$$\nabla_\theta J_r^{\pi_\theta} = E_\tau\left[\nabla_a Q_r^{\pi_\theta}(s,a)\nabla_\theta \pi_\theta(s)|a = \pi_\theta(\cdot|s)\right] \tag{2}$$

with

$$Q_r^{\pi_\theta}(s,a) = E_\tau \sum_{t=0}^{T}\left[\gamma_r^t \mathcal{R}(s_t, a_t, s_{t+1})|s_0 = s, a_0 = a\right]. \tag{3}$$

where ∇ indicates the partial derivative, and $Q_r^{\pi_\theta}(s,a)$ is the expected sum of rewards for the goal-reaching sub-task.

For the collision avoidance sub-task, its deterministic policy gradient $\nabla_\theta J_c^{\pi_\theta}$ replaces the reward function and discounted factor in Equation (2) with the cost function $\mathcal{C}$ and cost discounted factor γ_c, which is depicted as follows:

$$\nabla_\theta J_c^{\pi_\theta} = -E_\tau\left[\nabla_a Q_c^{\pi_\theta}(s,a)\nabla_\theta \pi_\theta(s)|a = \pi_\theta(\cdot|s)\right] \tag{4}$$

with

$$Q_c^{\pi_\theta}(s,a) = E_\tau \sum_{t=0}^{T}\left[\gamma_c^t \mathcal{C}(s_t, a_t, s_{t+1})|s_0 = s, a_0 = a\right]. \tag{5}$$

Here, $Q_c^{\pi_\theta}(s,a)$ is the expected sum of costs. Note that the deterministic policy gradient for the collision avoidance sub-task introduces an additional minus. It is caused by the fact that the optimisation objective for the collision avoidance sub-task is to minimise expected cumulative costs.

3.2 | Training of critic and safety critic

Since the optimal critic $Q_r^{\pi_\theta}(s,a)$ and the optimal safety critic of $Q_c^{\pi_\theta}(s,a)$ are unavailable, two neural networks are parameterised by φ and ψ to estimate them. By *temporal-difference* (TD) learning, $Q_r^{\varphi}(s,a)$ and $Q_c^{\psi}(s,a)$ are regarded as approximate estimates of the optimal ones and can be used to replace them in policy gradient.

$$\begin{aligned} Q_r^{\varphi}(s,a) &\to Q_r^{\pi_\theta}(s,a) \\ Q_c^{\psi}(s,a) &\to Q_c^{\pi_\theta}(s,a). \end{aligned} \tag{6}$$

Specifically, the critic is updated by the clipped double Q-Learning proposed in Ref. [46], which maintains two separate value networks to resolve the overestimation problem caused by the critic. Meanwhile, a separate target network is built for stable off-policy training. It is the same as the online network except that its parameters, $\widehat{\theta}$, $\widehat{\varphi}$, and $\widehat{\psi}$, are learnt via soft updates. Finally, the target value for the two value networks in a critic is rewritten as

$$y_r = r + \gamma_r \min_{j=1,2} Q_r^{\widehat{\varphi}_j}\left(s', \pi_{\widehat{\theta}}(s')\right) \tag{7}$$

where $s' = P(s, a)$ is the subsequent state from after taking action a on state s, and $r = R(s, a, s')$ is the immediate reward for the state transition.

As for the safety critic, considering the collision avoidance sub-task only dominates the navigation task in high-risk cases, the overestimation problem no longer happens. Therefore, only one value network is used, and its target value is

$$y_c = c + \gamma_c \max\left\{ Q_c^{\widehat{\psi}}\left(s', \pi_{\widehat{\theta}}(s')\right), 0 \right\}. \tag{8}$$

where $c = \mathcal{C}(s, a, s')$ is the immediate cost for the state transition. Considering that the cost value is always non-negative, a ReLU operator is used to avoid underestimating collision risk.

Then, by replacing optimal critic $Q_r^{\pi_\theta}(s, a)$ and safety critic $Q_c^{\pi_\theta}(s, a)$ in policy gradient equation (Equations 2 and 4) with the learnt critics $Q_r^{\varphi_1}(s, a)$ and safety critic $Q_c^{\psi}(s, a)$, policy gradients for sub-tasks are estimated by

$$\begin{aligned} \nabla_\theta J_r^{\pi_\theta} &= E_\tau\left[\nabla_a Q_r^{\varphi_1}(s, a) \nabla_\theta \pi_\theta(s) | a = \pi_\theta(\cdot|s)\right] \\ \nabla_\theta J_c^{\pi_\theta} &= -E_\tau\left[\nabla_a Q_c^{\psi}(s, a) \nabla_\theta \pi_\theta(s) | a = \pi_\theta(\cdot|s)\right] \end{aligned} \tag{9}$$

3.3 | Conflict-averse policy optimisation

Finally, the key problem is updating the policy network with policy gradients for sub-tasks. Considering that the minimisation objective can be transformed into the maximisation objective by introducing the minus, an intuitive and simple combination for optimisation objectives can be described by

$$\begin{aligned} &\arg\max_\theta J_{r,c}^{\pi_\theta} \\ &= \arg\max_\theta J_r^{\pi_\theta} - J_c^{\pi_\theta} \\ &= \arg\max_\theta E_\tau \sum_{t=0}^{T} \left(\gamma_r^t \mathcal{R}(s_t, a_t, s_{t+1}) - \gamma_c^t \mathcal{C}(s_t, a_t, s_{t+1})\right) \end{aligned} \tag{10}$$

Then, according to the additional rules of derivatives, the parameterised policy network can be updated by the following gradient:

$$\nabla_\theta J_{r,c}^{\pi_\theta} = \nabla_\theta J_r^{\pi_\theta} + \nabla_\theta J_c^{\pi_\theta} \tag{11}$$

Even though the optimisation objective in Equation (10) can combine different components, it ignores the inherent inconsistency, even conflict, between two sub-tasks. For example, in the autonomous robot navigation problem, the goal-reaching sub-task aims to navigate the robot along time-efficient paths, while the collision avoidance sub-task drives the robot from high collision-risk regions. When an obstacle lies between the robot and its goal, an obvious conflict exists between goal-reaching and collision avoidance sub-tasks. To deal with the inconsistency between them, this work further proposed a conflict-averse policy optimisation algorithm via gradient manipulation.

In particular, to mitigate the inconsistency between the two sub-tasks, the optimisation objective for each policy update is constructed based on the minimum performance improvement on sub-tasks, as subsequently defined.

$$\arg\max_\theta\left(\min\left\{\Delta J_r^{\pi_\theta}, -\Delta J_c^{\pi_\theta}\right\}\right) \tag{12}$$

where $\Delta J_r^{\pi_\theta}$ and $-\Delta J_c^{\pi_\theta}$ are the local policy improvement of expected returns and costs at each update step. When there is only a slight inconsistency between the goal-reaching and collision avoidance sub-tasks, such an optimisation objective aims to simultaneously improve both performances of goal-reaching and collision avoidance sub-tasks. When there exists a clear conflict between the two sub-tasks, the optimisation objective focuses on minimising performance degradation on sub-tasks. The following is a detailed solution to this optimisation problem.

Given a small step size of α and an update vector of $g \in \mathbb{R}^m$, the updated policy network is determined by $\theta' = \theta + \alpha * g$. By using the first-order Taylor approximation, Equation (12) can be represented by

$$\begin{aligned} &\arg\max_{g \in \mathbb{R}^m} \min_{i \in \{r,c\}} g_i^{\mathsf{T}} g \\ &s.t. \quad \|g - g_0\| \le \sigma \|g_0\| \end{aligned} \tag{13}$$

where T is the transpose operation for vectors and $\|\cdot\|$ denotes the L2 norm. Policy gradients are flatten to be gradient vectors: g_r for $\nabla_\theta J_r^{\pi_\theta}$ and g_c for $\nabla_\theta J_c^{\pi_\theta}$. The inequality constraints the best policy update vector in a local ball centred at the averaged gradient vector $g_0 = (g_r + g_c)/2$, with a radius controlled by a hyper-parameter of $\sigma > 0$.

Considering $\min_{i \in \{r,c\}} \left(g_i^{\mathsf{T}} g\right) = \min_{\omega_r, \omega_c} \sum_{i \in r,c} \left(\omega_i g_i^{\mathsf{T}} g\right)$, the Lagrangian objective of Equation (13) can be written as follows:

$$\max_{g \in \mathbb{R}^m} \min_{\lambda > 0, \omega \in \mathbb{W}} \left(g_w^{\mathsf{T}} g - \frac{\lambda}{2}\|g - g_0\|^2 + \frac{\lambda}{2}\sigma\|g_0\|^2\right) \tag{14}$$

where $\omega = [\omega_r, \omega_c] \in \mathbb{W}$ is the normalised weight and $g_\omega = \omega_r g_r + \omega_c g_c$ is the weighted sum of two policy gradient vectors. For the convex optimisation problem, both the Slater's condition and strong duality hold when $\sigma > 0$. Therefore, it is allowed to exchange the min and max:

$$\min_{\lambda > 0, \omega \in \mathbb{W}} \max_{g \in \mathbb{R}^m} \left(g_w^{\mathsf{T}} g - \frac{\lambda}{2}\|g - g_0\|^2 + \frac{\lambda}{2}\sigma\|g_0\|^2\right). \tag{15}$$

With fixing λ and ω, the optimal gradient vector $g_{\omega,\lambda}^* = g_0 + g_w/\lambda$. Substituting it into Equation (15), another dual optimisation problem is

$$\min_{\lambda > 0, \omega \in \mathbb{W}} \left(g_w^{\mathsf{T}} g_0 + \frac{1}{2\lambda}\|g_w\|^2 + \frac{\lambda}{2}\sigma\|g_0\|^2\right). \tag{16}$$

Further, let $\lambda^* = \frac{1}{\sigma^{1/2}}\|g_w\|/\|g_0\|$, the optimisation problem becomes

$$\min_\omega\left(g_\omega^{\mathsf{T}} g_0 + \sigma\|g_\omega\|_2\|g_0\|_2\right). \tag{17}$$

This optimisation problem is equal to Equation (12), but more time-efficient to be solved with only two optimisation parameters. After several steps of iterative optimisation on Equation (17), the optimal weight vector is approximated by ω^*. Substituting ω^* into $g^*_{\omega,\lambda}$, the optimal policy gradient vector for Equation (13) is $g^* = \omega^*_r g_r + \omega^*_c g_c$.

The proposed CASRL is summarised in Algorithm 1, which is built based on the Twin Delayed Deep Deterministic policy gradient algorithm (TD3) [46]. It maintains a single actor, a single safety critic, and a pair of critics. For each episode, the critics are updated towards the minimum target reward value (line 14), while the safety critic is updated to approximate the target cost value (line 15). Every f iterations, the policy is updated along the optimal conflict-averse gradient vector g^* with the step size of α (line 17–20). Meanwhile, target networks are updated with τ (line 21).

Algorithm 1: Conflict-Averse Safe Reinforcement Learning

```
   Input: exploration noise $\sigma$, update frequency $f$, policy noise
          $\tilde{\sigma}$, noise clip $b$, and update rate $\tau$
1  Initialize networks $\pi_\theta, Q_r^{\varphi_1}, Q_r^{\varphi_2}, Q_c^{\psi}$
2  Initialize target networks with $\hat{\theta}, \hat{\varphi}_1, \hat{\varphi}_2, \hat{\psi} \leftarrow \theta, \varphi_1, \varphi_2, \psi$
3  Initialize the replay buffer $\mathbb{B}$
4  for $\underline{episode = 1, \ldots, N_{eps}}$ do
5      Set $t$ to 0 and initialize state $s_0$
6      while $\underline{\textit{no Arrival, Collision or Timeout}}$ do
7          Select an action with exploration noise:
             $a_t \leftarrow \pi_\theta(s_t) + \epsilon, \epsilon \sim \mathcal{N}(0, \sigma)$
8          Execute $a_t$ and obtain $r_t$, $c_t$ and $s_{t+1}$
9          Assimilate tuple $(s_t, a_t, r_t, c_t, s_{t+1})$ into $\mathbb{B}$
10         $t \leftarrow t + 1$
11     end
12     for $\underline{k = 1, \ldots, N_{batch}}$ do
13         Sample $m$ experiences $(s, a, r, c, s')$ from $\mathbb{B}$
14         Update critics with sampled experiences:
             $\tilde{a} \leftarrow \pi_\theta(s') + \epsilon, \epsilon \sim clip(\mathcal{N}(0, \tilde{\sigma}, -b, b))$
             $y_r \leftarrow r + \gamma_r \min_{i=1,2} Q_r^{\hat{\varphi}_i}(s', \tilde{a})$
             $\varphi_i \leftarrow \arg\min \frac{1}{m} \sum (y_r - Q_r^{\varphi_i}(s, a))$
15         Update the safety critic with sampled experiences:
             $\tilde{a} \leftarrow \pi_\theta(s') + \epsilon, \epsilon \sim clip(\mathcal{N}(0, \tilde{\sigma}, -b, b))$
             $y_c \leftarrow c + \gamma_c Q_c^{\hat{\psi}}(s', \tilde{a})$
             $\psi \leftarrow \arg\min \frac{1}{m} \sum (y_c - Q_\psi(s, a))$
16         if $\underline{k \bmod f}$ then
17             Compute policy gradients $\nabla_\theta J_r^{\varphi_1}$ and $\nabla_\theta J_c^{\psi}$
18             Flatten policy gradients:
                 $g_r \leftarrow flatten(\nabla_\theta J_r^{\varphi_1})$
                 $g_c \leftarrow flatten(\nabla_\theta J_c^{\psi})$
19             Conflict-averse policy optimization $g^*$:
                 $g^* = \arg\max_{g \in \mathbb{R}^m} \min_{i \in \{r,c\}} \left(g_i^\top g\right)$
                 $s.t.\quad ||g - g_0||_2 \le \sigma ||g_0||_2$
20             Update actor with the policy gradient from with
                 $g^*$: $\theta' = \theta + \alpha * g^*$
21             Update target networks:
                 $\hat{\theta} \leftarrow \tau\theta + (1 - \tau)\hat{\theta}$,
                 $\hat{\varphi}_i \leftarrow \tau\varphi_i + (1 - \tau)\hat{\varphi}$
                 $\hat{\psi} \leftarrow \tau\psi + (1 - \tau)\hat{\psi}$
22         end
23     end
24 end
```

3.4 | Feature vector construction with HGAT

To help the robot navigate in dynamic environments, for example, crowd scenario as shown in Figure 3, this work follows the representation from Ref. [18], which constructs a feature vector with heterogeneous graph attention network (HGAT). After simplification, it describes various navigation scenarios with four types of objects: the robot, dynamic circular obstacles, static circular obstacles, and static line obstacles. Their observable configurations are defined by

$$o_i = \begin{cases} [d, \mathbf{v}_0, v_m, \theta] \in \mathbb{R}^5, & \text{if} \quad label(i) = 0 \\ [\mathbf{p}_i, \tilde{\rho}_i, \mathbf{v}_i, \mathbf{v}_{li}, \mathbf{v}_{ri}, \mu_i, \xi_i] \in \mathbb{R}^{11}, & \text{if} \quad label(i) = 1 \\ [\mathbf{p}_i, \tilde{\rho}_i, \mathbf{v}_{li}, \mathbf{v}_{ri}, \mu_i, \xi_i] \in \mathbb{R}^{9}, & \text{if} \quad label(i) = 2 \\ [\mathbf{p}_{si}, \mathbf{p}_{ei}, \rho_0, \mathbf{v}_{li}, \mathbf{v}_{ri}, \mu_i, \xi_i] \in \mathbb{R}^{11}, & \text{if} \quad label(i) = 3 \end{cases} \tag{18}$$

where all objects are numbered with subscript $i (i > 0)$: 0 for the robot and others for obstacles. The function $label(\cdot)$ marks types of objects. In the robot's configuration, d and θ describe the Euclidean distance and orientation from its current position to the given goal. Its velocity is donated by $\mathbf{v}_0$, limited by the maximum speed of v_m. For circular obstacles, both dynamic and static, their positions are presented by $\mathbf{p}_i = [p_{xi}, p_{yi}]$. Their radii are $\tilde{\rho}_i = \rho_i + \rho_0$, enlarged by the robot's radius ρ_0. For line obstacles, for example, boundaries, they are located by their start points $\mathbf{p}_{si}$ and end points $\mathbf{p}_{ei}$. Meanwhile, they are also inflated by ρ_0. Besides, for all obstacles, a VO-inspired interactive configuration is introduced to enhance the state representation. Assuming an obstacle maintains its velocity of $\mathbf{v}_i$, the robot will collide with it only if the robot's velocity lies in the VO cone as shown in Figure 4. The VO cone can be briefly described with a vertex $\mathbf{v}_i$ and two normalised direction

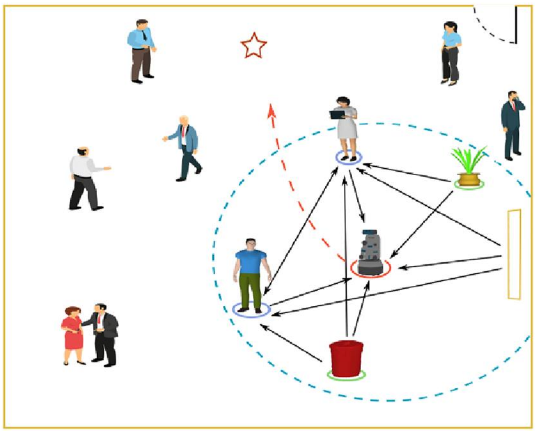


FIGURE 3 Indoor crowd scenario. During navigation, the robot can only observe obstacles, such as pedestrians, bins, and walls, within its sensor range. Its goal is marked by a red pentagram. Black arrows depict interactions among objects.

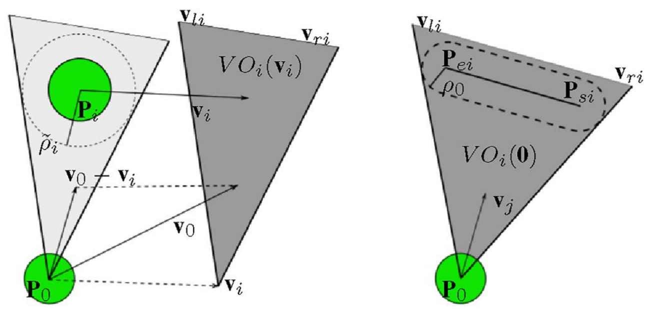


FIGURE 4 An illustration of the velocity obstacle for circular (left) and static line obstacles (right).

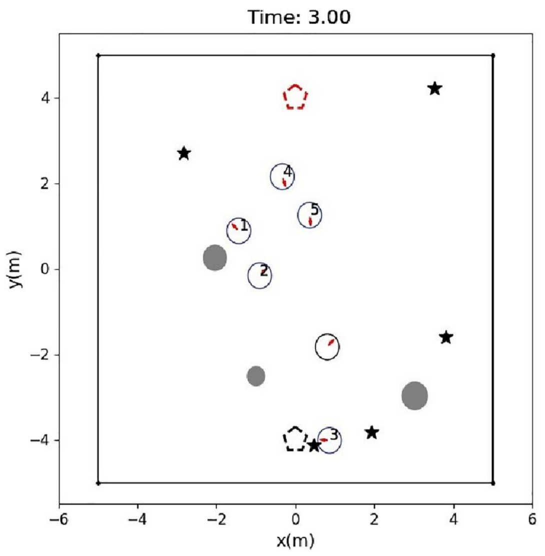


FIGURE 5 Simulation environment. Both the robot and dynamic circular obstacles are represented by hollow circles, with a red arrow indicating the heading. Dynamic obstacles are painted with numbers to distinguish them from the robot. Except for hollow circles, solid grey circles and black lines are painted for static circular obstacles and boundaries, respectively. Additionally, two pentagons are used to mark the start (black) and the goal (red) position of the robot. Meanwhile, goal positions for dynamic obstacles are marked with black stars.

vectors, $\mathbf{v}_{li}$ and $\mathbf{v}_{ri}$ for the left and right rays [47]. Furthermore, to better distinguish the emergency of obstacle avoidance, the minimum separation distance μ_i and expected collision time ς_i are introduced to represent the configuration of the obstacle. However, since ς_i is infinite when the robot's velocity lies outside the VO cone, another equivalent variable $\xi_i = 1/(\varsigma_i + 0.5)$ is used to replace ς_i. Finally, the vertex components of static obstacles are ignored since they have zero velocity.

Then, state s_t can be represented as a heterogeneous graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V}$ for objects and edge $\mathcal{E}$ for interactions among objects. An edge from node i to node j means that there exists an interaction from object i to object j. In other words, object j is required to avoid collision with object i, actively.

Based on the heterogeneous graph representation of states, the proposed method utilises two layers of HGATs to extract the feature vector. It has two main advantages: scalability in the number of nodes and attention-based message passing among nodes. A detailed setting and description for HGAT can refer to Ref. [18]. Then, the feature vector is fed into subsequent Multi-Layer Perceptions (MLPs) to construct the actor π_θ, critic Q_r^φ, and safety-critic Q_c^ψ.

4 | CASE STUDY

4.1 | Simulation platform and settings

The simulation environment is adapted from CrowdNav [48]. As shown in Figure 5, there exists a robot, dynamic and static circular obstacles, and static linear obstacles from the boundary. The width and length of the room are 10.0 m. All dynamic obstacles are navigated by ORCA [49], a widely used reciprocal collision avoidance algorithm for multi-agent simulation. The dynamic obstacles have a radius of 0.3 m and a maximum velocity of 1 m/s. To interact with the robot frequently, some dynamic obstacles are initialised to simulate the circle-crossing phenomenon. Their initial and goal positions are roughly symmetrically placed on the central circle with a radius of 4.0 m. Moreover, random noises uniformly distributed between −0.5 and 0.5 m are added to their positions. Apart from circle-crossing dynamic obstacles, the remaining are initialised with random initial and goal positions in the room to mimic the square-crossing phenomenon. Once a dynamic obstacle reaches its goal, a new random goal is generated within the room. Static circular obstacles, with random radii ranging from 0.1 to 0.4 m, are randomly placed in the room.

During the training process, the number of circular obstacles is randomly generated, with 3–10 for dynamic obstacles and 3–5 for static obstacles. As for the robot, its maximum velocity $v_{\max}$ and acceleration $a_{\max}$ are set to 1.0 m/s and 1.0 m/s^2. At the beginning of each training episode, the robot is set with the initial and goal position of (0.0, −4.0) and (0.0, 4.0). Additionally, the robot is assumed to have a limited observation range of 6.0 m. Therefore, only observed circular obstacles or segments of static linear obstacles, rather than global obstacle information, are considered in navigation. Other detailed settings for the training process are followed.

4.1.1 | Reward function

The reward function $\mathcal{R}$ is a sum of three components: r^g, r^d, and r^θ. Among them, r^d and r^θ are the dense rewards designed to navigate the robot approaching and steering towards its goal, and r^g is the sparse immediate reward for reaching its goal. Namely,

$$r_t = r_t^g + r_t^t + r_t^\theta. \tag{19}$$

where r_t^g, r_t^d, and r_t^θ are given by

$$
\begin{aligned}
r_t^g &= \begin{cases} \kappa_1, & \text{if} \quad d_t < \eta_1 \\ 0, & \text{otherwise} \end{cases} \\
r_t^d &= \kappa_2(d_{t-1} - d_t), \\
r_t^\theta &= \kappa_3(\cos(\theta_t) - 1)/(d_t + \eta_2)),
\end{aligned} \tag{20}
$$

where d_t and θ_t are the distance and the orientation from the robot's position to the robot's goal at step t, respectively. The goal tolerance of the robot η_1 is set to 0.2 m. The hyper-parameter η_2 is used to smooth and limit the value of r_t^θ, with its value of 5.0 m. Parameters κ_1, κ_2, and κ_3 are used to balance different reward components, and their values are 25.0, 1.0, and 1.0.

4.1.2 | Cost function

The cost function $\mathcal{C}$ only includes the sparse immediate cost of $\kappa_4 = 25.0$ on collisions, which is given by

$$
c_t = \begin{cases} \kappa_4, & \min\limits_{i>0} \mu_i \leq 0 \\ 0, & \text{otherwise} \end{cases} \tag{21}
$$

where μ_i is the minimal distance from the robot to the object i.

4.1.3 | Action space

The robot is assumed with a common two-wheeled differential drive model, and its action $a_t = \left(a_t^l, a_t^r\right)$ donates the acceleration of wheels (a_t^l for the left wheel and a_t^r for the right one). Considering the acceleration constraint, both of a_t^l and a_t^r are limited in $[-a_{\max}, a_{\max}]$, where $a_{\max} = 1.0$ is the maximal acceleration.

4.2 | Simulation results and discussions

Four methods, MPC (Model Predictive Control) from Ref. [50], TEB (Time-Elastic-Band) from Ref. [51], HGAT-DRL from Ref. [18], and HGAT-DRL-Lag adapted from Ref. [20] are implemented as baseline algorithms. Among them, both MPC and TEB are traditional optimisation-based approaches, while HGAT-DRL is a recent DRL-based approach. HGAT-DRL-Lag is a variant of HGAT-DRL, which introduces a Lagrange multiplier λ_l to balance the two sub-tasks adaptively. The Lagrange multiplier is updated by $-\lambda_l(\phi - \phi')$, where ϕ and ϕ' are the current collision rate and the threshold of the collision rate. In the implementation, ϕ is estimated on the last 500 episodes, and ϕ' is set to 10%.

This work proposes three approaches: NGSRL, PCSRL, and CASRL, which differ in how they update the policy network. Specifically, NGSRL updates the policy network with the gradient vector of $g_0 = (g_\mathcal{R} + g_\mathcal{C})/2$, without considering the consistency between goal-reaching and collision avoidance sub-tasks. PCSRL modifies PCGrad [40] into the proposed SRL framework. Specifically, the optimal policy gradient vector is defined by

$$
g^* = \begin{cases} 0.5(g_r + g_c), & \text{if} \quad g_r^T g_c \leq 0 \\ 0.5\left(g_c + g_r - \dfrac{g_r^T g_c}{\|g_r\|} g_r\right), & \text{otherwise} \end{cases} \tag{22}
$$

CASRL is a further development of PCSRL. It aims to maximise the worse local policy improvement of sub-tasks to deal with the inherent inconsistency between the two sub-tasks.

Five DRL-based approaches are training five times with different random seeds. In the training process, all network parameters are trained by the Adam optimiser with a learning rate of 0.001. Both discount factors for expected cumulative rewards and costs, γ_r and γ_c, are set to 0.987. Since this work is adapted from the TD3 algorithm, hyper-parameters for training include the exploration noise, policy noise, noise clip, update rate, and update frequency, with values of 0.2, 0.2, 0.2, 0.005, and 8, respectively. With these hyper-parameters and a 13th-generation i7 CPU from Intel, and the training of HGAT-DRL, NGSRL, and CASRL spent 11,659s, 13,569s, and 13,936s, respectively. The separation of two sub-tasks only requires an additional 1,910s, and the proposed conflict-averse policy optimisation only increased the training time by 367s. Therefore, there is no need to worry about an excessive computational burden caused by the proposed conflict-averse policy optimisation.

For a comprehensive and fair comparison, all methods are evaluated with 12,000 random test cases. These cases are grouped based on the number of dynamic circular obstacles (ranging from 3 to 10) and static circular obstacles (ranging from 3 to 5), with 500 cases for each combination. Eight metrics are used to evaluate different algorithms quantitatively, including *"Suc. Rate"*, *"Col. Rate"*, *"Nav. Time"*, *"Col. Time"*, *"Dis. Number"*, *"Avg. Return"*, *"Avg. Cost"*, and *Over. Perf.* They describe the success rate, collision rate, navigation time of success cases, time response of collision cases, number of violations to the safety zone of dynamic obstacles, discounted cumulative return over steps, discounted cumulative cost over steps, and overall performance, respectively.

4.2.1 | Performance comparison

Statistic results are shown in Table 1 and training curves from DRL-based methods are depicted in Figure 6. Since the random seed does not affect TEB and MPC, their standard deviations are always 0. As expected, TEB and MPC achieve a terrible performance in the simulation environment. Especially, since MPC does not take into account the motion property of obstacles, its performance is catastrophic, with a success rate of only 17.5%. All five DRL-based methods perform better than

TABLE 1 Statistical results of different algorithms.

Methods	Suc. Rate ↑	Col. Rate ↓	Nav. Time (s) ↓	Col. Time (s) ↑	Dis. Number ↓	Avg. Return ↑	Avg. Cost ↓	Over. Perf.
MPC [50]	0.132 ± 0.000	0.867 ± 0.000	18.88 ± 0.00	4.53 ± 0.00	1819.3 ± 0.0	3.45 ± 0.00	19.93 ± 0.00	−16.49 ± 0.00
TEB [52]	0.756 ± 0.000	0.236 ± 0.000	13.79 ± 0.00	5.21 ± 0.00	1358.8 ± 0.0	16.59 ± 0.00	4.96 ± 0.00	11.63 ± 0.00
HGAT-DRL [18]	0.767 ± 0.037	0.233 ± 0.38	**10.97 ± 0.49**	5.80 ± 0.43	1061.5 ± 165.9	17.78 ± 0.61	5.08 ± 0.87	12.69 ± 1.47
HGAT-DRL-Lag [20]	**0.833 ± 0.021**	**0.138 ± 0.023**	15.82 ± 1.03	**10.61 ± 0.71**	**929.4 ± 75.0**	17.37 ± 0.75	**2.68 ± 0.48**	14.69 ± 1.07
NGSRL(ours)	0.809 ± 0.018	0.190 ± 0.018	**11.72 ± 0.34**	6.72 ± 0.43	1010.1 ± 117.7	18.38 ± 0.33	4.05 ± 0.40	14.33 ± 0.71
PCSRL(ours)	0.829 ± 0.017	0.170 ± 0.017	12.19 ± 0.16	7.34 ± 0.30	1066.6 ± 82.5	**18.61 ± 0.52**	3.56 ± 0.37	**15.05 ± 0.88**
CASRL(ours)	**0.849 ± 0.010**	**0.148 ± 0.009**	12.76 ± 0.40	**7.67 ± 0.28**	**968.6 ± 95.0**	**18.82 ± 0.25**	**3.11 ± 0.19**	**15.71 ± 0.42**

Note: ± is used to indicate the standard deviation. All DRL-based algorithms are tested with five models from a different random seed. The upward/downward arrow indicates that the higher/lower the metric is, the better the algorithm. In each metric, the best two are bolded, and the best one is marked with another wave line.

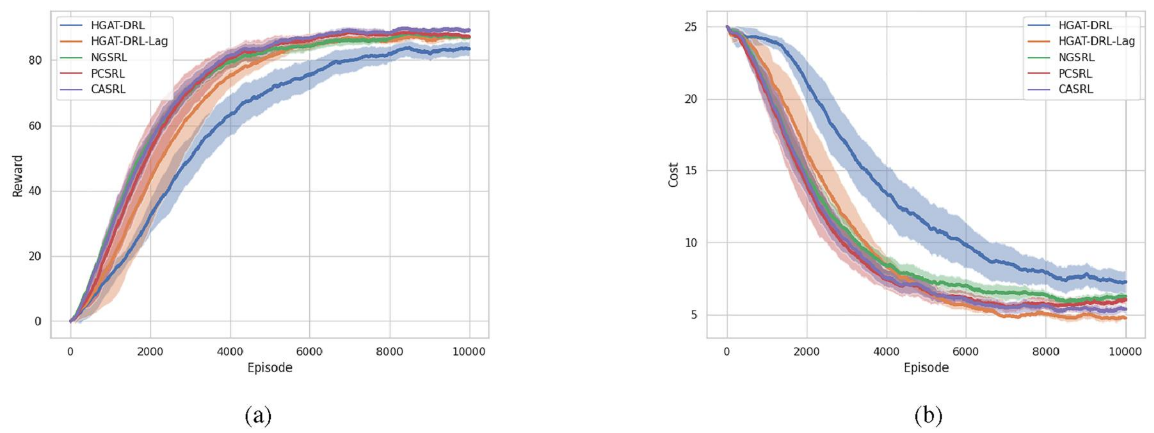


FIGURE 6 Training curves from HGAT-DRL, HGAT-DRL-Lag, NGSRL, PCSRL, and CASRL, which are coloured blue, orange, green, red, and purple, respectively. (a) Training curve of episodic reward. (b) Training curve of episodic cost.

the two optimised-based methods, which shows the ability of DRL to solve complex optimisation problems.

In the further comparison among five DRL-based algorithms, it is easy to find that CASRL achieves the top two in almost all metrics, except for navigation time. Specifically, CASRL achieves the highest success rate of 84.9%, the most average discounted return of 18.84, and the best overall performance of 15.71. In other metrics on motion safety, including *Col. Rate*, *Col. Time*, *Avg. Cost,* and *Dis. Number*, it also achieves the second-best performance, slightly worse than HGAT-DRL-Lag. Compared to the vanilla algorithm of HGAT-DRL, CASRL greatly delays the occurrence of collisions in high-risk cases by about 1.94s and reduces violations to safety zones of obstacles (with a distance less than 0.2 m) by 11.1%. HGAT-DRL-Lag is the safest algorithm, with a *Col. Rate* of only 13.8%. However, since it does not take into account the conflicting policy gradient from two sub-tasks, its time efficiency is greatly reduced, speeding 23.7% more navigation time than CASRL. Therefore, it performs terribly in *Avg. Return* and ranks only third in *Over. Perf.*

4.2.2 | Ablation study of gradient manipulation

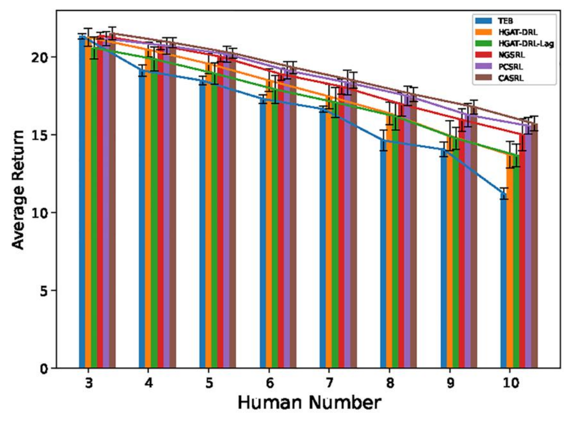
NGSRL is implemented for ablation experiments on the proposed two innovations: decomposition of the overall navigation task and gradient-based conflict-averse policy optimisation. Benefiting from the first innovation, NGSRL outperforms HGAT-DRL in terms of safety, showing a 0.92s (15.7%) increase in *Col. Time*, as well as a 4.3%, 4.9%, and 18.4% drop in *Col. Rate*, *Dis. Number*, and *Avg. Cost*, respectively. Benefiting from the second innovation, CASRL also shows a further improvement on the basis of NGSRL in terms of safety, increasing/decreasing by 1.02s (15.1%), 4.0%, 6.6%, and 11.9% in *Col. Rate*, *Col. Time*, *Dis. Number*, and *Avg. Cost*, respectively.

PCSRL is also implemented as a comparison algorithm to NGSRL and CASRL, which performs gradient surgery according to Equation (22). Due to the projection operation of conflicted gradients, PCSRL achieves better motion safety than NGSRL. However, PCSRL focuses on the conflicted cases, ignoring the more common cases of slight inconsistency. Therefore, its motion safety is worse than CASRL.

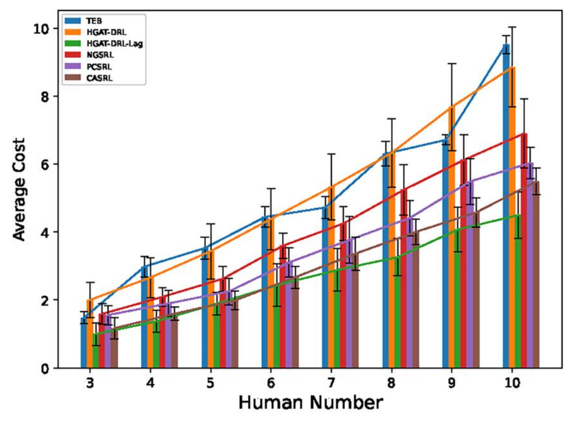
4.2.3 | Performance in different environments

To further validate the effectiveness of CASRL and demonstrate the proposed two innovations, a comprehensive comparative analysis is conducted on a more detailed set of experimental results. Here, 12,000 cases are divided into eight groups according to the number of dynamic circular obstacles ranging from 3 to 10. Each group comprises 1500 test cases with static circular obstacles ranging from 3 to 5. Six methods, TEB, HGAT-DRL, HGAT-DRL-Lag, NGSRL, PCSRL, and CASRL, are tested on four core metrics, including *Ave. Return*, *Suc. Rate*, *Nav. Time*, and *Ave. Cost*.

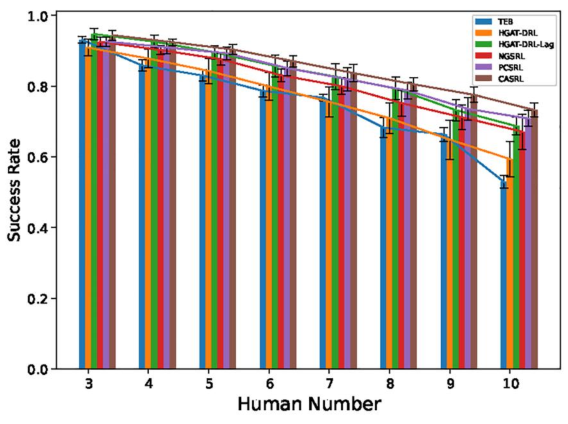
Experimental results are shown in Figure 7. As the number of dynamic obstacles increases, the difficulty of collision avoidance increases, so the performance of all six algorithms is detelriorating. Nevertheless, CASRL always keeps the best performance in *Suc. Rate*, and *Ave. Return*. TEB experiences the most dramatic decline in performance, from the third best in the simplest scenarios to the worst in others. Furthermore, it is worth noting that as the difficulty level increases, the gaps among HGAT-DRL, NGSRL, and CASRL also widen significantly. Specifically, in the simplest cases with only three dynamic circular obstacles, their success rates are 90.9%, 92.5%, and 94.9%, with gaps of 1.6% and 2.4%. When the number of dynamic obstacles becomes 6, their success rates become 79.9%, 83.0%, and 87.1%, with widened gaps of 3.1% and 4.1%. Yet in the most difficult cases with 10 dynamic circular obstacles, their success rates are 59.3%, 67.1%, and 72.7%, with further widened gaps of 7.8% and 5.6%. This phenomenon is in line with the expectations of the proposed innovation. In simple scenarios, the conflict between goal-reaching and collision avoidance sub-tasks is not so slight that the neglect of conflict causes only a slight performance degradation. However, as the number of dynamic obstacles increases, obstacle avoidance becomes more challenging, and the conflict between two sub-tasks intensifies. Then, either separating the collision avoidance sub-task from the overall task or eliminating conflicts between the two sub-tasks can effectively



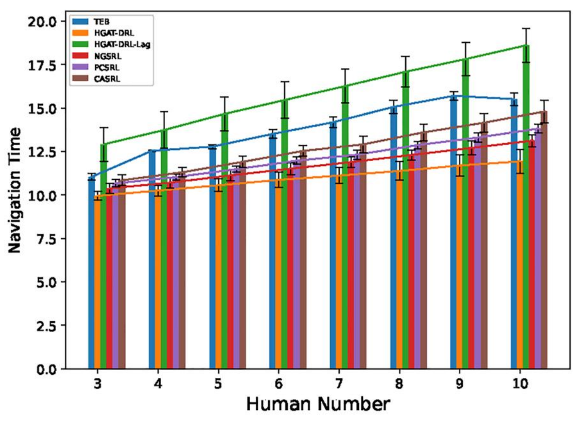
(a)



(b)



(c)



(d)

FIGURE 7 Experimental comparisons of TEB, HGAT-DRL, NGSRL, and CASRL in different environments. They depict the average return, average cost, success rate, and navigation time, respectively. (a) Average Return. (b) Average Cost. (c) Success Rate. (d) Navigation Time.

improve the motion safety of mobile robots. Another notable point is that though HGAT-DRL-Lag achieves the best performance in motion safety, it spends the most navigation time in each group of environments. The result is consistent with the result mentioned above.

4.2.4 | Experiment in interactive environments

The crowd is one of the typical dynamic environments that mobile robots encounter. In real-world crowd scenarios, a key difference is that pedestrians can see and react to the robot. Therefore, we further evaluate DRL-based policies from Sec. 4.2.1 in interactive scenarios, where the root is visible to dynamic circular obstacles. The statistical results are shown in Table 2.

As expected, in interactive environments, all five DRL-based algorithms achieve a visible improvement in the four metrics for motion safety. It shows that the proactive collision avoidance policy learnt by an invisible robot can be directly deployed in real-world visible robots. The performance improvement can be attributed to pedestrians' proactive avoidance behaviours. Even if there is no communication network between the robot and pedestrians, they can still achieve collaborative collision avoidance. A possible problem is the increased violations and navigation time. It is caused by the lack of a communication mechanism between the robot and pedestrians. Actually, even for pedestrians, it is also difficult to realise optimal reciprocal avoidance in communication-less environments. Therefore, the robot and pedestrians sometimes hider each other, resulting in increased navigation time and violations of the safety zone. Finally, their *Over. Perf* increase by an average of 6.6%.

4.3 | Real-world experiments

Apart from simulation experiments, we also deploy CASRL on a Fetch robot. Its diameter, maximum velocity, and maximum wheel acceleration are 0.56 m, 1.0 m/s, and 1.0 m/s^2, respectively. Though the policy is trained with a simulation timestep of 0.25s, it runs at a frequency of 20 Hz for quicker reactions to possible collisions. Besides, since only the robot provides the control interface with linear and angular velocity, planned actions on accelerations are transformed to control commands at a higher frequency of 100 Hz. The whole experimental field is 8.0 and 18.0 m, and a motion capture system is utilised to record the positions of all objects. To better verify the algorithm's effectiveness, the Fetch robot must patrol between point (0, −4.0) and (0,4.0) rather than a goal-reaching navigation task in the simulation experiment. Once the distance to the current goal is less than 0.3 m, the robot is considered to have completed the current individual navigation task, and its goal will be reset for the next navigation. Its free space is also limited in a 10.0 m × 10.0 m square. As shown in Figure 8, three suitcases are randomly placed as static circular obstacles. Except for them, four

TABLE 2 Statistical results in interactive environments. ± is used to indicate the standard deviation. The upward/downward arrow indicates that the item is larger/smaller than the corresponding one in Table 1.

Methods	Suc. Rate ↑	Col. Rate ↓	Nav. Time (s) ↓	Col. Time (s) ↑	Dis. Number ↓	Avg. Return ↑	Avg. Cost ↓	Ove. Perf. ↑
HGAT-DRL [18]	0.796 ± 0.049 ↑	0.203 ± 0.049 ↓	**11.68** ± **0.54** ↑	6.85 ± 0.43 ↑	1469.7 ± 154.1 ↑	18.07 ± 0.88 ↑	4.33 ± 1.09 ↓	13.70 ± 1.95 ↑
HGAT-DRL-*Lag* [20]	**0.865** ± **0.018** ↑	**0.096** ± **0.022** ↓	16.46 ± 0.92 ↑	**11.96** ± **1.00** ↑	**1028.4 ± 73.5** ↑	17.69 ± 0.62 ↑	**1.80** ± **0.44** ↓	15.89 ± 0.84 ↑
NGSRL (ours)	0.826 ± 0.028 ↑	0.173 ± 0.029 ↓	**12.47** ± **0.31** ↑	7.57 ± 0.39 ↑	1301.4 ± 183.6 ↑	18.35 ± 0.52 ↑	3.61 ± 0.62 ↓	14.74 ± 1.13 ↑
PCSRL (ours)	0.864 ± 0.045 ↑	0.133 ± 0.047 ↓	12.89 ± 0.24 ↑	8.67 ± 0.24 ↑	1370.7 ± 83.5 ↑	**18.97** ± **1.12** ↑	2.71 ± 0.96 ↓	**16.26** ± **2.08** ↑
CASRL (ours)	**0.880** ± **0.017** ↑	**0.116** ± **0.016** ↓	13.40 ± 0.26 ↑	**8.72** ± **0.21** ↑	**1213.3** ± **108.6** ↑	**19.08** ± **0.28** ↑	**2.41** ± **0.32** ↓	**16.66** ± **0.58** ↑

Note: In each metric, the best two are bolded, and the best one is marked with another wave line.

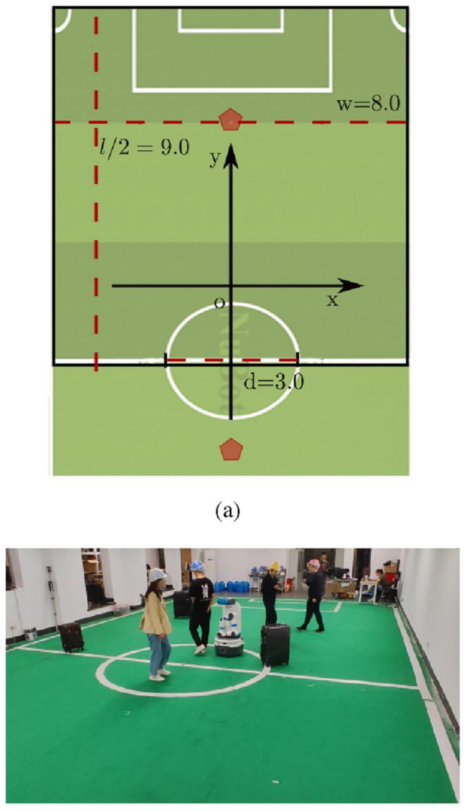


FIGURE 8 An illustration for the real-world experiment filed. (a) The field coordinate and some parameters. (b) A snapshot from real-world experiments.

TABLE 3 Statistical results from hardware experiments.

Methods	Suc. Rate	Nav. Time (s)	Ave. Speed. (m/s)
CASRL	1.00	14.36 ± 3.19	0.64 ± 0.06

volunteers join in the hardware experiment to interact with the robot.

Statistical results of the hardware experiment are shown in Table 3, which comes from 40 individual navigation tasks. One surprising result is that CASRL achieves a 100% success rate and an average speed of 0.643 m/s, without any collision with volunteers, suitcases, or walls. However, in the real-world experiment, *Nav. Time* increases to 14.36 s, 3.14 s higher than that from the simulation experiment with four dynamic and three static circular obstacles. There are three possible reasons for the difference. At first, in real-world environments, volunteers exhibit more diverse and changeable behaviour, rather than the ideal ORCA algorithm in the simulation environment. Secondly, there exists an unavoidable noise in state estimation for volunteers caused by relative perception modules, which can also weaken the time efficiency of the navigation policy. Finally, the patrol task in real-world experiments differs from the goal-reaching task in the simulation environment, where the Fetch robot has to make a U-turn at the beginning of each run. Figure 9 shows two sample trajectory diagrams and corresponding velocity curves. Their navigation times are 15.35 and 15.60 s, respectively. In Figure 9(a), noticeable collision avoidance behaviours of the robot include (1) turning right to avoid the oncoming cyan volunteer coloured at the time of about 2.0 s; (2) turning left and giving way for magenta and orange volunteers at t = 5.0 s; and (3) turning towards its goal after passing by the magenta volunteer human at t = 10.0 s. In Figure 9b, noticeable collision avoidance behaviours include (1) rotating towards the goal first, (2) turning right to avoid the oncoming cyan volunteer, (3) keeping its velocity to pass quickly in front of the magenta volunteer, (4) slowing down at t = 6.25 s, turning left at t = 7.0 s, and staying parallel with the orange volunteer until t = 10.0 s. In the comparison of the robot's behaviours in two tests, an interesting thing is its reaction to the magenta volunteer. In the test shown in Figure 9(a), since the robot is close to the magenta volunteer after avoiding the cyan one, it decides to not only give way to the volunteer but also turn left to enlarge its distance. While in the test shown in Figure 9(b), there exists enough passable space in front of the magenta volunteer, and the robot performs a quick pass ahead of the volunteer. As for the velocity curves, slight derivations exist between velocity commands and actual velocities, which fully illustrate that the planned actions of accelerations are dynamically feasible for the Fetch robot.

5 | CONCLUSION

This paper proposes an SRL framework for autonomous robot navigation, along with a safe navigation algorithm for mobile robots in dynamic environments. It first decomposes the overall navigation task into two sub-tasks, one for goal-reaching and another for collision avoidance, and introduces a safety critic to estimate the motion safety/risk of actions. On the basis of critic and safety critic, it further constructs two task-specific but model-agnostic policy gradients, eliminating the mutual interference between two sub-tasks. And then, it proposes a conflict-averse manipulation of policy gradients. By maximising the worst local policy improvement of both sub-tasks, the inconsistency between the two sub-tasks can be greatly eliminated. Additionally, the conflict-averse policy update ensures an efficient and stable learning process. Furthermore, this work proposes a practical safe navigation algorithm named CASRL by combining the SRL framework with the heterogeneous graph representation of dynamic environments from Ref. [18].

As expected, the proposed algorithm achieves a high success rate of over 84.9% in a challenging simulated environment, outperforming the vanilla algorithm of HGAT-DRL by 8.2%. Especially in the most difficult environment with 10 dynamic circular obstacles, where the inconsistency between

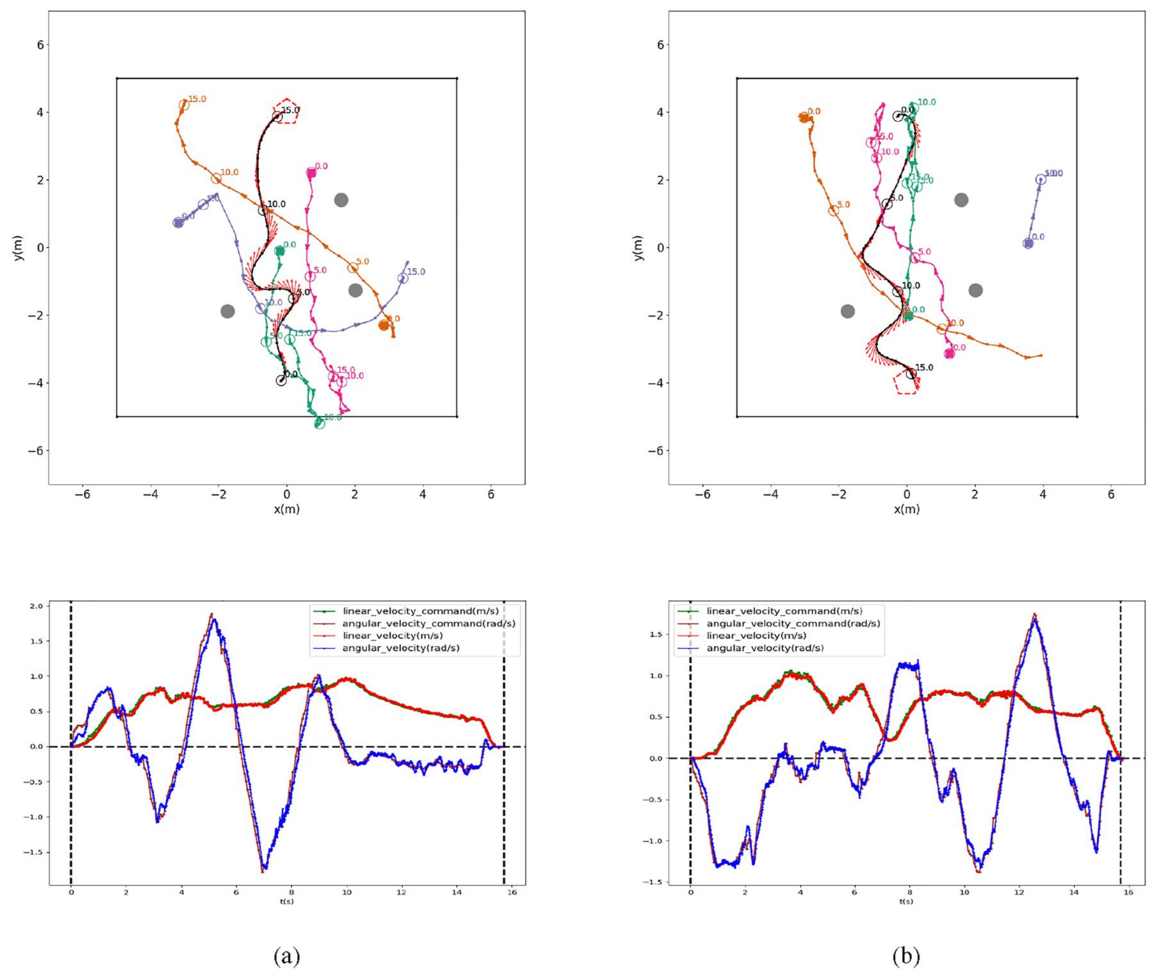


FIGURE 9 Trajectory diagrams and velocity curves in two sample scenarios. In trajectory diagrams (the top line), grey discs represent static circular obstacles. Coloured curves depict the trajectories of the robot (black) and volunteers (other colours). Black lines represent the boundaries of the robot's free space. Coloured numbers on trajectories indicate the time, and the interval between two consecutive nodes is 0.5s. Small arrows on humans' trajectories indicate the forward directions, while red arrows on the robot's trajectory show its heading angles. The robot's goal is marked with a red pentagon. The figures' bottom line shows the robot's velocity curves, consisting of the velocity commands and the actual velocities of the Fetch robot. In addition, two dashed black lines are used to mark the start time and end time of the navigation task. (a) Scenario 1. (b) Scenario 2.

two sub-tasks intensifies, the proposed algorithm can further extend its lead over HGAT-DRL to 13.4%. The superiority of the proposed algorithm to the baseline fully illustrates the effectiveness of the proposed two innovations. Besides, extensive hardware experiments also demonstrate that the proposed algorithm can be directly deployed in real-world dynamic environments.

CONFLICT OF INTEREST STATEMENT

The authors declare no conflicts of interest.

DATA AVAILABILITY STATEMENT

Research data are not shared.

ORCID

Zhiqian Zhou https://orcid.org/0000-0002-9407-075X
Junkai Ren https://orcid.org/0000-0002-6199-7452
Xinglong Zhang https://orcid.org/0000-0002-0587-2487

REFERENCES

1. Ferrer, G., Sanfeliu, A.: Proactive kinodynamic planning using the extended social force model and human motion prediction in urban environments. In: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1730–1735 (2014)
2. Chen, Y., Zhao, F., Lou, Y.: Interactive model predictive control for robot navigation in dense crowds. IEEE Trans. Syst. Man Cybernetics, 1–13 (2021)
3. Lin, S., et al.: Receding horizon control with trajectron++: navigating mobile robots in the crowd. In: 2022 41st Chinese Control Conference (CCC), pp. 3871–3877 (2022)
4. Trautman, P., Krause, A.: Unfreezing the robot: navigation in dense, interacting crowds. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 797–803 (2010)
5. Chen, Y., et al.: Robot navigation in crowds by graph convolutional networks with attention learned from human gaze. IEEE Rob. Autom. Lett. 5(2), 2754–2761 (2020). https://doi.org/10.1109/lra.2020.2972868
6. Yan, C., Xiang, X., Wang, C.: Towards real-time path planning through deep reinforcement learning for a uav in dynamic environments. J. Intell. Rob. Syst. 98(2), 297–309 (2020). https://doi.org/10.1007/s10846-019-01073-3

7. Li, G., Hildre, H.P., Zhang, H.: Toward time-optimal trajectory planning for autonomous ship maneuvering in close-range encounters. IEEE J. Ocean. Eng. 45(4), 1219–1234 (2020). https://doi.org/10.1109/joe.2019.2926822
8. Cui, Y., et al.: Learning world transition model for socially aware robot navigation. In: 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 9262–9268 (2021)
9. Xie, Z., Xin, P., Dames, P.: Towards safe navigation through crowded dynamic environments. In: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4934–4940 (2021)
10. Zhou, Z., et al.: Robot navigation in a crowd by integrating deep reinforcement learning and online planning. Appl. Intell. 52(13), 15600--15616 (2022). https://doi.org/10.1007/s10489-022-03191-2
11. Liu, S., et al.: Decentralized structural-rnn for robot crowd navigation with deep reinforcement learning. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 3517–3524 (2021)
12. Yan, C., et al.: Deep reinforcement learning of collision-free flocking policies for multiple fixed-wing uavs using local situation maps. IEEE Trans. Ind. Inf. 18(2), 1260–1270 (2022). https://doi.org/10.1109/tii.2021.3094207
13. Zhao, L., et al.: Toward an online decision support system to improve collision risk assessment at sea. IEEE Intell. Transp. Syst. Mag. 15(2), 137–148 (2023). https://doi.org/10.1109/mits.2022.3190965
14. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2nd ed. The MIT Press (2018)
15. Chen, Y.F., et al.: Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 285–292 (2017)
16. Chen, Y.F., et al.: Socially aware motion planning with deep reinforcement learning. In: : 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1343–1350 (2017)
17. Han, R., et al.: Reinforcement learned distributed multi-robot navigation with reciprocal velocity obstacle shaped rewards. IEEE Rob. Autom. Lett. 7(3), 5896–5903 (2022). https://doi.org/10.1109/lra.2022.3161699
18. Zhou, Z., et al.: Navigating robots in dynamic environment with deep reinforcement learning. IEEE Trans. Intell. Transport. Syst. 23(12), 25201–25211 (2022). https://doi.org/10.1109/tits.2022.3213604
19. Yu, H., Xu, W., Zhang, H.: Towards safe reinforcement learning with a safety editor policy. In: Advances in Neural Information Processing Systems (2022)
20. Ha, S., et al.: Learning to walk in the real world with minimal human effort. In: Kober, J., Ramos, F., Tomlin, C. (eds.) Proceedings of the 2020 Conference on Robot Learning. Vol. 155 of Proceedings of Machine Learning Research, pp. 1110–1120 (2021)
21. Achiam, J., Amodei, D.: Benchmarking Safe Exploration in Deep Reinforcement Learning. arXiv (2019)
22. Stooke, A., Achiam, J., Abbeel, P.: Responsive safety in reinforcement learning by pid Lagrangian methods. In: Proceedings of the 37th International Conference on Machine Learning (2020)
23. Achiam, J., et al.: Constrained policy optimization. In: Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 22–31 (2017)
24. Yang, L., et al.: Constrained update projection approach to safe policy optimization. In: Advances in Neural Information Processing Systems, vol. 35, pp. 9111–9124. Curran Associates, Inc. (2022)
25. Dalal, G., et al.: Safe Exploration in Continuous Action Spaces. arXiv (2018)
26. Srinivasan, K., et al.: Learning to Be Safe: Deep RL with a Safety Critic. arXiv (2020)
27. Thananjeyan, B., et al.: Recovery rl: safe reinforcement learning with learned recovery zones. IEEE Rob. Autom. Lett. 6(3), 4915–4922 (2021). https://doi.org/10.1109/lra.2021.3070252
28. Berliac, F., Basu, Y., Saac, D.: Safe reinforcement learning as an adversarial game of actor-critics. In: The Multi-Disciplinary Conference on Reinforcement Learning and Decision Making (2022)
29. Mnih, V., et al.: Human-level control through deep reinforcement learning. Nature 518(7540), 529–533 (2015). https://doi.org/10.1038/nature14236
30. Tai, L., Paolo, G., Liu, M.: Virtual-to-real deep reinforcement learning: continuous control of mobile robots for mapless navigation. In: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 31–36. IROS) (2017)
31. Long, P., et al.: Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 6252–6259 (2018)
32. Sathyamoorthy, A.J., et al.: Densecavoid: real-time navigation in dense crowds using anticipatory behaviors. In: 2020 IEEE International Conference on Robotics and Automation, pp. 11345–11352. ICRA) (2020)
33. Chen, C., et al.: Relational graph learning for crowd navigation. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 10007–10013 (2020)
34. Liu, S., et al.: Intention aware robot crowd navigation with attention-based interaction graph. In: IEEE International Conference on Robotics and Automation. ICRA) (2023)
35. Kästner, L., et al.: Enhancing navigational safety in crowded environments using semantic-deep-reinforcement-learning-based navigation. In: 2022 IEEE International Symposium on Safety, Security, and Rescue Robotics, pp. 87–93. SSRR) (2022)
36. Lv, S., et al.: A deep safe reinforcement learning approach for mapless navigation. In: 2021 IEEE International Conference on Robotics and Biomimetics, pp. 1520–1525. ROBIO) (2021)
37. Sener, O., Koltun, V.: Multi-task learning as multi-objective optimization. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 525–536. Curran Associates Inc., NY (2018)
38. Liu, S., Johns, E., Davison, A.J.: End-to-end multi-task learning with attention. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1871–1880. CVPR) (2019)
39. Zhang, Y., Yang, Q.: A survey on multi-task learning. IEEE Trans. Knowl. Data Eng. 34(12), 5586–5609 (2022). https://doi.org/10.1109/tkde.2021.3070203
40. Yu, T., et al.: Gradient surgery for multi-task learning. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. Curran Associates Inc., NY (2020)
41. Liu, B., et al.: Conflict-averse gradient descent for multi-task learning. In: Ranzato, M., et al. (eds.) Advances in Neural Information Processing Systems, vol. 34, pp. 18878–18890. Curran Associates, Inc. (2021)
42. Sun, Y., Li, J., Xu, X.: Meta-gf: training dynamic-depth neural networks harmoniously. In: Computer Vision – ECCV 2022, pp. 691–708. Springer-Verlag, Berlin (2022)
43. Misra, I., et al.: Cross-stitch networks for multi-task learning. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3994–4003 (2016)
44. Rosenbaum, C., Klinger, T., Riemer, M.: Routing networks: adaptive selection of non-linear functions for multi-task learning. In: International Conference on Learning Representations (2018)
45. Wang, X., Li, Y.: Harmonized dense knowledge distillation training for multi-Exit architectures. Proc. AAAI Conf. Artif. Intell. 35(11), 10218–10226 (2021). https://doi.org/10.1609/aaai.v35i11.17225
46. Fujimoto, S., van Hoof, H., Meger, D.: Addressing function approximation error in actor-critic methods. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning, pp. 1587–1596 (2018)
47. Fiorini, P., Shiller, Z.: Motion planning in dynamic environments using velocity obstacles. Int. J. Robot Res. 17(7), 760–772 (1998). https://doi.org/10.1177/027836499801700706
48. Chen, C., et al.: Crowd-robot interaction: crowd-aware robot navigation with attention-based deep reinforcement learning. In: 2019 International Conference on Robotics and Automation, pp. 6015–6022 (2019)
49. Van den Berg, J., et al.: Reciprocal n-body collision avoidance. In: The International Journal of Robotics Research, pp. 3–19 (2011)
50. Rösmann, C., et al.: Exploiting sparse structures in nonlinear model predictive control with hypergraphs. In: 2018 IEEE/ASME

International Conference on Advanced Intelligent Mechatronics, pp. 1332–1337. AIM) (2018)

51. Rösmann, C., Hoffmann, F., Bertram, T.: Timed-elastic-bands for time-optimal point-to-point nonlinear model predictive control. In: 2015 European Control Conference, pp. 3352–3357. ECC) (2015)
52. Rösmann, C., Hoffmann, F., Bertram, T.: Integrated online trajectory planning and optimization in distinctive topologies. Robot. Autonom. Syst. 88, 142–153 (2017). https://doi.org/10.1016/j.robot.2016.11.007

How to cite this article: Zhou, Z., et al.: A safe reinforcement learning approach for autonomous navigation of mobile robots in dynamic environments. CAAI Trans. Intell. Technol. 1–16 (2023). https://doi.org/10.1049/cit2.12269