IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2023, London, UK. Cite as RA-L paper.

Collision Avoidance Among Dense Heterogeneous Agents Using Deep Reinforcement Learning

Kai Zhu[1], Bin Li[1], Wenming Zhe[2], and Tao Zhang[1,3], *Senior Member, IEEE*

***Abstract*—Navigating in a complex congested social environment without collision is a crucial and challenging task. Recent studies have demonstrated the considerable success of Deep Reinforcement Learning (DRL) in multi-agent collision avoidance. However, the assumption of these studies that agents are homogeneous circles deviates from reality, leading to performance deterioration in congested scenarios. The current work extends the DRL-based approaches to develop a collision avoidance method for congested scenarios wherein the heterogeneity of agents can no longer be disregarded. Considering shape heterogeneity, we use the Orientated Bounding Capsule (OBC) to model the agents and transform the interactive state space of Robot-Obstacle agent pair. For speed heterogeneity, we design a velocity-related collision risk function to shape the behavior of the robot. Experimental results demonstrate that our proposed method outperforms state-of-the-art DRL-based approaches in terms of success rate and safety. It also exhibits desired collision avoidance behavior.**

***Index Terms*—Collision Avoidance, Reinforcement Learning, Autonomous Agents.**

I. Introduction

COLLISION avoidance is an active research area in robot navigation. Future mobile robot systems are expected to integrate naturally into human society and navigate crowded environments, such as shopping malls, airports, and intersections, in a manner similar to that of people. In these scenarios, a mobile robot must avoid multiple dynamic obstacles and reach the goal point. Robot collision avoidance in crowded environments, which we referred to as *Social Navigation*, has elicited the interests of researchers who have achieved different forms of progress.

Robot navigation in a crowded environment faces the Freezing Robot Problem (FRP) [1], which has not yet been solved effectively. The unknown target of other agents and the high uncertainty of dynamic movement are the major causes of FRP [2]. Several traditional velocity-based methods, such as Reciprocal Velocity Obstacles (RVO) [3], are suitable for multi-robot cooperative collision avoidance and crowd simulation but perform poorly in social navigation. Early studies established the Social Force Model (SFM) [4], [5] or the human-robot interaction model based on the Gaussian Process (GP) [2], [6] to explore pedestrian behavior and social navigation. These modeling methods require numerous parameters, such as *responsibility* for reciprocal collision avoidance, which would be difficult to identify for other agents in reality. In addition, some trajectory prediction methods [7] have achieved great success, and their explicit integration and joint optimization with planning algorithms are desired but challenging. Deep Reinforcement Learning (DRL) provides another way to implicitly integrate interactive prediction and planning, which effectively offloads the expensive online computation to an offline training procedure. DRL-based methods describe navigation as a Partially Observable Markov Decision Process (POMDP) and learn a policy which maximizes the expected return [8]. Most of these methods [9]–[12] use the circumscribed circle of the agent contour to model a robot and its surrounding crowd, thereby idealizing the agents' states.

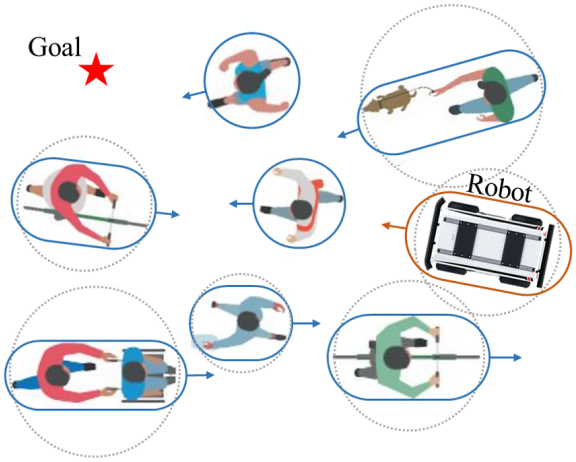


Fig. 1. Robot navigation among dense heterogeneous agents. Dotted line: Existing methods that typically model the robot and obstacle agents with circles consume considerable free space. Solid line: The proposed method models agents with capsules to alleviate FRP in dense heterogeneous scenarios.

In this paper, we advocate that the existing methods of modeling the agent with circles are not conducive to alleviating FRP, particularly in an environment with dense heterogeneous obstacles. Fig. 1 shows that social navigation involves not only pedestrians but also mobile agents with various contours and speeds, such as cyclists and wheelchair users. The typically circular homogeneity assumption increases the space occupied by agents and wastes the free space for moving (refer to the

Manuscript received May 12, 2022; Revised September 19, 2022; Accepted November 8, 2022. This paper was recommended for publication by Associate Editor A. Bera and Editor T. Asfour upon evaluation of the reviewers' comments. This work was supported by the Scientific and Technological Innovation 2030 of China under Grant 2021ZD0110900. (*Corresponding author: Tao Zhang.)*

[1]Kai Zhu, Bin Li, and Tao Zhang are with the Department of Automation, Tsinghua University, Beijing, China {zhuk19@mails., binli626@mail., taozhang@}tsinghua.edu.cn

[2]Wenming Zhe is with JD Logistics, Beijing, China zhewenming@jd.com

[3]Tao Zhang is also with the Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, China.

dotted line in Fig. 1). Some works on reciprocal collision avoidance and trajectory prediction address heterogeneity with convex polygon modeling [7], [13]. However, to the best of our knowledge, existing DRL-based social navigation methods have not yet considered the heterogeneous nature of agents.

The main contribution of the current study is that it extends the DRL-based navigation methods to collision avoidance scenarios with heterogeneous decision-making agents, such as pedestrians, cyclists, and other robots. To solve the aforementioned problems, this work considers the heterogeneous characteristics of the agents' shape and speed:

- Shape heterogeneity: We use the Orientated Bounding Capsule (OBC) to model the agents. Then, the compatibility of the multi-agent shape heterogeneity in a DRL-based navigation method is achieved by calculating the minimum separating distance and state rotation.
- Speed heterogeneity: We propose a velocity-related collision risk metric for OBC. Then, the reward function is designed based on collision risk to enable the robot to perform better when interacting with low-speed and high-speed obstacle agents.

An extensive set of simulation experiments demonstrates that the performance of several state-of-the-art DRL-based collision avoidance algorithms has been improved signicantly after considering the heterogeneity of agents. As the density of moving obstacles increases, the performance improvement brought by our approach becomes more evident.

II. Related Work

In this section, we provide a brief overview of prior work on collision avoidance for social navigation and some related studies on FRP.

A. Model-based Collision Avoidance

Velocity-based algorithms, such as VO [14], RVO [3], and Optimal Reciprocal Collision Avoidance (ORCA) [15], are widely used for multi-robot and crowd simulations. Some RVO variants consider the robot's non-holonomic constraints [16] and heterogeneity [13] for different types of agents, and achieve a satisfactory heterogeneous multi-robot collision avoidance performance. A key challenge in these methods is that they rely heavily on parameters specified by human experts. Moreover, no accurate description of pedestrian behavior is provided. By contrast, SFM, which was proposed by Helbing *et al.*, provides a theoretical tool for social navigation [4]. Other researchers have considered "force" or "potential" as a measure of the internal motivation of individuals to perform specific actions or movements [5], [17], [18]. Trautman *et al.* proposed a nonlinear interaction potential term to couple separate GP trajectories of each agent in [1] and mixed GPs interpolating between waypoints in [2]. Although their methods model Human-Robot cooperation, which is important for social navigation, these methods are limited to specific dynamic environmental assumptions. In addition, the mobile robot may cause stress and discomfort to humans during their interaction.

B. Learning-based Collision Avoidance

Recently, machine learning has been rapidly developed for robot navigation in dense, uncertain environments. As a supervised learning technology, imitation learning methods obtain navigation policies by directly imitating expert demonstrations [19], [20]. However, it is much easier to obtain the trajectory data of social agents than that of the robot itself. Researchers have established a number of data-driven trajectory prediction architectures to understand the nature of human-human interaction mechanisms [21]–[23]. Luo *et al.* proposed GAMMA [7] for trajectory prediction, using Bayesian inference to estimate the key parameters of ORCA. These trajectories are then used to support downstream algorithms such as belief-space planning in [7] to output action for the ego-agent.

As an alternative, DRL [24]–[26], which learns by trial and error, exhibits high potential in dealing with decision-making problems under uncertainty. Many researchers have described navigation as POMDP and selected sensor-level or agent-level observations as states [8]. Among them, sensor-level policies are typically trained end-to-end, such as in [27], [28], learning to choose actions directly from raw sensor readings (laser scans or images). By contrast, agent-level methods such as CADRL [9] extract agent-level data (e.g., position, velocity, and radius) as state vectors. CADRL employs a fully connected deep neural network to parametrize the value function. The value network is trained by a Deep V-learning algorithm, and the policy is derived from the maximal value action in the discretized action space. CADRL was further extended by introducing variants with reward shaping and LSTM to achieve social compliance in pedestrian-rich environments [10], [11]. Although these algorithms outperform most traditional methods such as ORCA, social navigation remains a challenging task as crowd size and density increase. Then SARL [12] was proposed by Chen *et al.* with a novel neural network encoding Crowd-Robot interaction rather than Human-Robot interaction. They designed a social attentive pooling module to learn the relative importance of each neighbor and the collective impact of the crowd. The network is trained by a Deep V-learning algorithm similar to CADRL. Furthermore, the attention weights can be learned from gaze data collected from humans [29].

C. Freezing Robot in Congested Environment

FRP occurs when the planner decides that all forward paths are unsafe and the only option for the robot is to freeze in place [1], [6]. The work of [30] constructed a potential freezing zone in DRL to solve FRP. [31] proposed an unfrozen and unlost mechanism after a robot freezes. In addition, a recent study noted the effect of non-circular contours on collision avoidance in crowded scenes [32]. Their Reactive Driving Support (RDS) method constructs VO between each obstacle and the nearest subpart of the robot, and limits its speed accordingly to avoid local collisions. The primary difference between the RDS method and our work is as follows: RDS is based on the traditional VO method and does not consider the heterogeneity of obstacle agents.

III. Problem Formulation

A. Deep Reinforcement Learning for Navigation

A robot is supposed to move toward a goal through n heterogeneous agents without any collision. Following the work of [9] and [12], the task is formulated as a sequential decision-making problem in a reinforcement learning framework. Two types of agents exist in the environment: a single robot agent and n obstacle agents. At time step t, the state of the robot agent can be fully observed. It is denoted as $\mathbf{s}_t$, including the robot's position $\mathbf{p}_t$, velocity $\mathbf{v}_t$, goal position $\mathbf{g}_t$, preferred speed v_{pref}, and shape parameters (details are provided in Section III-B). The state of the i-th obstacle agent can only be partially observed. It is denoted as $\mathbf{o}_t^i$, including position $\mathbf{p}_t^i$, velocity $\mathbf{v}_t^i$, and shape parameters. By concatenating the observable state of all the $(n+1)$ agents, the joint state can be defined as $\tilde{\mathbf{s}}_t = [\mathbf{s}_t, \mathbf{o}_t^1, \mathbf{o}_t^2, ..., \mathbf{o}_t^n]$.

The robot's navigation policy π is trainable, whereas the i-th obstacle agent's policy π^i is unknown and untrainable, and it is given by the simulator or the real individual. Let $\mathbf{a}_t$ denote a robot's action at time t decided by policy π, i.e., $\mathbf{a}_t = \pi(\tilde{\mathbf{s}}_t)$. An assumption is made that the robot can change its velocity immediately in accordance with the action command $\mathbf{a}_t$. The objective of reinforcement learning is to obtain an optimal policy π^* that maximizes the cumulative discount return. This objective can be achieved by finding the optimal value function V^* that encodes an estimate of the expected return as follows:

$$V^*(\tilde{\mathbf{s}}_t) = \mathbb{E}\left[\sum_{t'=t}^{T} \gamma^{t' \cdot v_{pref}} R\left(\tilde{\mathbf{s}}_{t'}, \pi^*\left(\tilde{\mathbf{s}}_{t'}\right)\right) \mid \tilde{\mathbf{s}}_t\right], \quad (1)$$

where T is the terminal time of an episode (terminated with collision, timeout, or reaching goal), $\gamma \in [0, 1)$ is a discount factor, and $R\left(\tilde{\mathbf{s}}_{t'}, \pi^*\left(\tilde{\mathbf{s}}_{t'}\right)\right)$ is the reward received at time t'. The preferred velocity v_{pref} is a weighted parameter to control the discounted factor and temporal field of view during deployment, rather than a manually fixed discounted factor [9]. Let $V\left(\tilde{\mathbf{s}}_t; \theta_i\right)$ be an approximate value network with parameters θ. The updates to θ can be derived from a variety of reinforcement learning algorithms. In accordance with the Bellman equation, the value function can be updated through the Temporal Difference (TD) learning method to approach the optimal gradually. We use a one-step TD error, which is the difference in estimated return before and after the time-step Δt. The loss function of training the value network is given by:

$$L(\theta_i) = \mathbb{E}\left[R_{t+\Delta t} + \gamma^{\Delta t \cdot v_{pref}} V\left(\tilde{\mathbf{s}}_{t+\Delta t}; \theta_{i-1}\right) - V\left(\tilde{\mathbf{s}}_t; \theta_i\right)\right]^2. \quad (2)$$

The optimal policy for maximizing the expected return can be derived from the optimal value function, i.e.,

$$\pi^*(\tilde{\mathbf{s}}_t) = \underset{\mathbf{a}_t \in \mathcal{A}}{\operatorname{argmax}} R(\tilde{\mathbf{s}}_t, \mathbf{a}_t) + \gamma^{\Delta t \cdot v_{pref}} \int_{\tilde{\mathbf{s}}_{t+\Delta t}} P\left(\tilde{\mathbf{s}}_{t+\Delta t} \mid \tilde{\mathbf{s}}_t, \mathbf{a}_t\right) V^*\left(\tilde{\mathbf{s}}_{t+\Delta t}\right) d\tilde{\mathbf{s}}_{t+\Delta t}, \quad (3)$$

where $\mathcal{A}$ is the discreted action space, and $P\left(\tilde{\mathbf{s}}_{t+\Delta t} \mid \tilde{\mathbf{s}}_t, \mathbf{a}_t\right)$ is the transition probability from $\tilde{\mathbf{s}}_t$ to $\tilde{\mathbf{s}}_{t+\Delta t}$ when taking action $\mathbf{a}_t$. We follow the implementation of SARL [12], and the next state $\tilde{\mathbf{s}}_{t+\Delta t}$ is obtained by querying the environment for the true value in training. During deployment, the transition probability is approximated with a linear motion model for simplicity, which can also be estimated by a trajectory prediction model [7], [21]–[23]. Thus, $R\left(\tilde{\mathbf{s}}_t, \mathbf{a}_t\right)$ can be calculated analytically through simple geometry and the best action is chosen from a set of permissible velocity vectors.

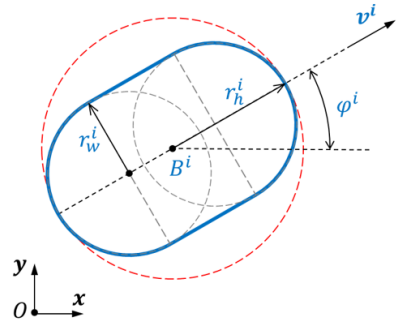


Fig. 2. OBC model of the i-th agent. The shape features include the length, width and orientation of the capsule body. It is assumed that the direction of velocity is consistent with the long axis.

B. Modeling Heterogeneous Agents

As mentioned earlier, the agents involved in collision avoidance are heterogeneous, and the assumption of circular homogeneity exacerbates FRP. We observed that the projections of robots and dynamic obstacles on a 2D plane are mainly circular, rectangular, or long convex. In practice, the Orientated Bounding Box (OBB) is widely used to represent convex contours in collision detection tasks [33]. Inspired by OBB, we proposed Orientated Bounding Capsule (OBC) to model the agent. In this manner, the DRL method can be compatible with agents of different shapes.

Each agent modeled by OBC has three key parameters regarding shape heterogeneity, as shown in Fig. 2. For the i-th obstacle agent, let $\varphi^i \in [0, 2\pi]$ be the orientation of the OBC's long axis in the global coordinate system $\{\mathbf{O}\}$. Let $r_h^i \in \mathbb{R}_+$ denote the radius of the OBC's circumscribed circle, which is also half of the length. Let $r_w^i \in \mathbb{R}_+$ denote the radius of the OBC's inscribed circle, which is also half of the width ($r_w^i \leq r_h^i$). Similarly, let φ, r_h, r_w be those parameters of the robot agent. We assume that all the agents participating in collision avoidance are non-holonomic and the direction of velocity $\mathbf{v}^i$ is consistent with φ^i, which indicates $\varphi^i = \arctan(v_x^i, v_y^i)$. In addition, when $r_h^i = r_w^i$, the OBC degenerates into a circle; when $r_h^i > r_w^i$, the OBC can envelop different convex contours with a small gap.

IV. Methodology

The following section presents the state space and reward function for the DRL-based collision avoidance method, which considers the heterogeneous characteristics of agents' shape and speed.

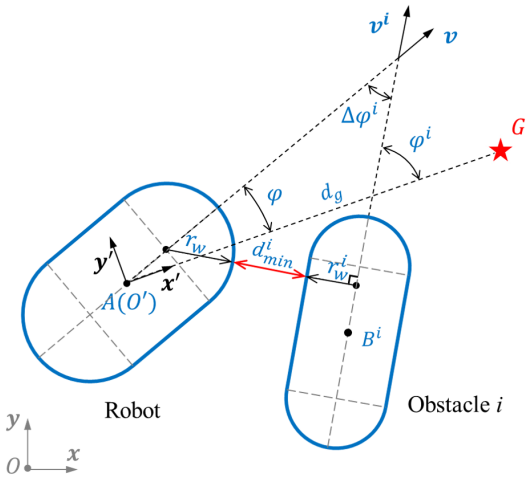


Fig. 3. Interactive states between two OBCs. We transform all state variables into the local coordinate space to reduce redundant parameters. The robot agent and each obstacle agent are matched separately, and the angle and minimum separating distance between each pair of OBCs are regarded as essential interactive state values.

A. Interactive State Space

With the notations provided in Section III, the fully observable state of the robot agent $\mathbf{s}$ and the partially observable state of the i-th obstacle agent $\mathbf{o}^i$ in the global coordinate system are defined as:

$$\mathbf{s} = [\mathbf{p}, \mathbf{v}, \varphi, r_h, r_w, \mathbf{g}, v_{pref}], \tag{4}$$

$$\mathbf{o}^i = [\mathbf{p}^i, \mathbf{v}^i, \varphi^i, r^i_h, r^i_w], \tag{5}$$

where $\mathbf{p} = [p_x, p_y]$ is the location of the robot agent, $\mathbf{p}^i = [p^i_x, p^i_y]$ is the location of the i-th obstacle agent, and $\mathbf{g} = [g_x, g_y]$ is the robot's goal position.

Note that the original state vectors in (4) and (5) have redundant parameters and cannot intuitively reflect the interaction between the two heterogeneous agents. Therefore, a better state representation is desirable. First, the state vectors are transformed from the global coordinate system into the local coordinate system $\{\mathbf{O}'\}$, where the robot is located at the origin and the x-axis points toward the robot's goal (as shown in Fig. 3). Then, the orientation difference between the obstacle and the robot, the minimum separating distance (described in Section IV-B), and other variables are added to the state vector to enhance representativeness. The states of the robot and the i-th obstacle agent in $\{\mathbf{O}'\}$ after transformation are given by:

$$\mathbf{s}' = [v'_x, v'_y, \varphi, r_h, r_w, d_g, v_{pref}], \tag{6}$$

$$\mathbf{o}^{i\prime} = [p^{i\prime}_x, p^{i\prime}_y, v^{i\prime}_x, v^{i\prime}_y, \varphi^i, r^i_h, r^i_w, d^i_m, \Delta\varphi^i, \Sigma r^i_h, \Sigma r^i_w], \tag{7}$$

where d_g represents the distance between the robot and its goal point; d^i_m represents the minimum separating distance between the robot and the i-th obstacle agent; and $\Delta\varphi^i = \varphi^i - \varphi$ denotes the orientation difference between the obstacle and the robot, which describes the relative posture between two OBCs. In addition, $\Sigma r^i_h = r^i_h + r_h$ and $\Sigma r^i_w = r^i_w + r_w$ describe the relative space margin between two OBCs.

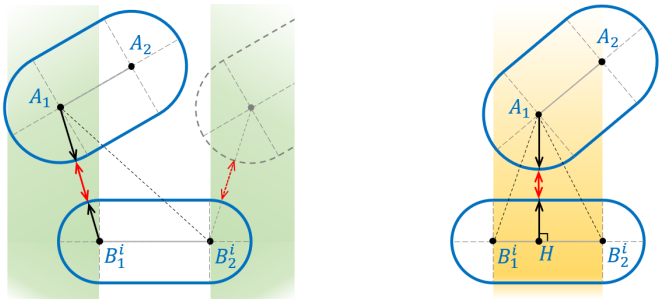


Fig. 4. Geometry of minimum separating distance between two OBCs. (a) The closest distance occurs between the end point and the end point. (b) The closest distance occurs between the end point and the vertical point.

B. Geometry of Minimum Separating Distance

In the previous section, the minimum separating distance d^i_m between the i-th obstacle agent and the robot agent is adopted as part of the state parameters. Moreover, d^i_m is crucial for collision detection and measurement of collision risk. In the circular hypothesis, d^i_m can be expressed as the distance between the two circles' centers minus the radii of the two circles due to isotropy. The amount of usable space that cannot be accessed by a navigation method based on circular representation is significant, because the circular contour is too conservative in the width direction (perpendicular to the moving direction). For example, when an agent moves forward along a linear trajectory, the swept area can be approximated as a rectangle with the width of $2r_h$ using circular representation, and $2r_w$ using capsule representation. The proportion of the missing space is about $(r_h - r_w)/r_w$. The larger the aspect ratio of the agent, the more usable space capsule representation (or convex polygon) can provide during movement and alleviates FRP. In the assumption of convex polygons such as rectangles, the minimum separating distance is related to the center point and contour direction. The separating axis theorem, commonly used in collision detection tasks, determines whether an overlap occurs by searching for disjoint projection axes. However, it cannot provide the distance. In our OBC model, each capsule can be regarded as an infinite number of circles with a center point on a line segment, and the smallest distance between two line segments occurs at the end-end or end-vertical point (as shown in Fig. 4).

Next, we explain how to estimate the minimum separating distance using Fig. 4. Suppose that the minimum separating distance $d(A_1)$ occurs at A_1. With the change in the relative poses of $\overrightarrow{A_1A_2}$ and $\overrightarrow{B^i_1B^i_2}$, two situations are possible:

1) In the end-end situation as shown in Fig. 4(a), i.e., if $\overrightarrow{B^i_1A_1} \cdot \overrightarrow{B^i_1B^i_2} \leq 0$ or $\overrightarrow{B^i_1A_1} \cdot \overrightarrow{B^i_1B^i_2} \geq \left\|\overrightarrow{B^i_1B^i_2}\right\|_2$:

$$d(A_1) = \min\left(\left\|\overrightarrow{A_1B^i_1}\right\|_2, \left\|\overrightarrow{A_1B^i_2}\right\|_2\right) - r_w - r^i_w. \tag{8}$$

2) In the end-vertical situation as shown in Fig. 4(b), i.e.,

if $\overrightarrow{B_1^iA_1} \cdot \overrightarrow{B_1^iB_2^i} > 0$ and $\overrightarrow{B_1^iA_1} \cdot \overrightarrow{B_1^iB_2^i} < \left\|\overrightarrow{B_1^iB_2^i}\right\|_2$:

$$d(A_1) = \frac{\left\|\overrightarrow{B_1^iA_1} \times \overrightarrow{B_1^iB_2^i}\right\|_2}{\left\|\overrightarrow{B_1^iB_2^i}\right\|_2} - r_w - r_w^i. \tag{9}$$

Then the three other key values $d(A_2), d(B_1^i), d(B_2^i)$ between the two OBC agents can be solved using formulas similar to (8) and (9), and the minimum separating distance is set as $d_m^i = \min[d(A_1), d(A_2), d(B_1^i), d(B_2^i)]$.

C. Velocity-related Collision Risk

Inspired by the elliptical potential field of SFM [4], we believe that the robot has a higher probability of collision in the obstacle's moving direction, particularly in front of the obstacle instead of behind it. To measure this risk caused by obstacle movement, we set a risk capsule for each obstacle agent (see the yellow capsule in Fig. 5). From the assumption of non-holonomic constraints, the agent will maintain a nearly constant speed $\left\|\mathbf{v}^i\right\|_2$ in the current direction within the reaction time δt. Therefore, compared with the original OBC (the blue capsule in Fig. 5), the risk capsule is lengthened forward by $\Delta L^i = \left\|\mathbf{v}^i\right\|_2 \delta t$. As the speed of the obstacle agent increases, ΔL^i also becomes larger, reflecting the prediction of future collision risks.

The robot should try to stay away from the risk capsule and not just the obstacle itself. With this insight, a velocity-related collision risk metric for OBC is designed as:

$$risk^i = \begin{cases} kD_{risk} & \text{if } d_{mr}^i < 0, \\ k(D_{risk} - d_{mr}^i) & \text{elseif } d_{mr}^i \le D_{risk}, \\ 0 & \text{otherwise }, \end{cases} \tag{10}$$

where D_{risk} is the desired safety distance, d_{mr}^i represents the minimum separating distance between the robot and the risk capsule of the i-th obstacle, and k is the risk coefficient used to control the value range of $risk^i$.

Then, the reward function can be shaped by the aforementioned velocity-related collision risk as follows:

$$R\left(\tilde{\mathbf{s}}_t, \mathbf{a}_t\right) = \begin{cases} 1 & \text{if } \mathbf{p}_t = \mathbf{g}_t, \\ -0.25 & \text{elseif } d_m^i \le 0, \\ -risk & \text{otherwise,} \end{cases} \tag{11}$$

where $risk = \underset{i \in \{0,1,\ldots,n\}}{\operatorname{argmax}} risk^i$ originates from the obstacle agent that poses the highest threat to the robot. At time step t, the robot receives a positive reward upon reaching the goal, a negative reward for collision, or a penalty (negative reward) in accordance with the collision risk. Through this reward function, the robot can be trained to avoid collisions with dynamic obstacles of various speeds.

V. Experiments and Results

A. Experimental Setup

We built a simulation environment in Python following [12] to support robot navigation among heterogeneous moving obstacles. The simulated obstacle agents are controlled by ORCA [15]. The robot adopts two existing state-of-the-art DRL-based algorithms, namely, CADRL [9] and SARL [12], as baseline methods. We refer to our full enhanced version designed for the dense heterogeneous scenario as DH-CADRL and DH-SARL and the version without a changing reward function (i.e., $\Delta L^i \leftarrow 0$) as DH-CADRL* and DH-SARL* for ablation experiments.

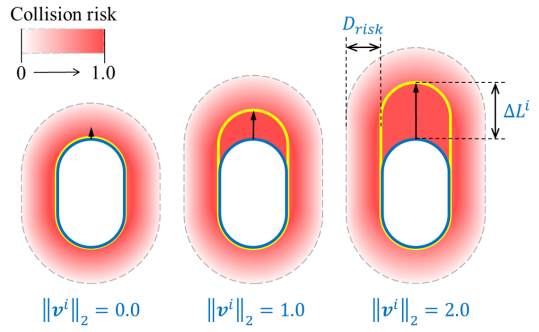


Fig. 5. Velocity-related collision risk. The blue capsule is the i-th obstacle agent, and the yellow capsule is the risk capsule extended by the former in accordance with moving speed. A darker red color indicates a higher risk value of the robot at that position.

1) Simulation Setup: To better simulate a real scene, we set up several types of OBC as obstacles and randomly sampled heterogeneous parameters including r_h, r_w, v_{pref} from the uniform distribution. The robot's OBC parameters are set as follows: $r_h = 0.5$m, $r_w = 0.2$m, and $v_{pref} = 1$m/s. In addition, for all baseline algorithms that adopt the assumption of circular homogeneity, the circle radius parameter is set as $r = r_h$. The robot's velocity $\mathbf{v}_t$ is discretized into an 81-dimensional action space, which is composed of 16 angular velocities in $[-\pi/4, \pi/4]$rad and 5 discrete linear velocities in $(0, v_{pref}]$m/s. The initial position and goal point of the robot are set as $[0, -3]$ and $[0, 3]$, while those of the obstacles are randomly sampled from an area of 8m×8m square.

2) Training Process: In general, imitation learning technology can be used to initialize a neural network and improve the efficiency of policy optimization. In the initialization step, we use 3,000 episodes demonstrated by ORCA to train the value network in the Pytorch framework for 50 epochs with a learning rate of 0.01. It could be better to initialize by the real-world pedestrian trajectory data. Then the second training step refines the policy for 10,000 episodes through RL with a learning rate of 0.001, a discount factor $\gamma = 0.9$, and a batch size of 100. After each episode, the obstacle's position and goal point are resampled randomly. The number of obstacles we used in training is $N = 5$. During training, obstacle agents can observe each other, while robots are invisible to obstacle agents, which means the obstacle agents do not actively avoid the robot.

3) Evaluation Metrics: Notably, the random seed used in the training and test cases is the same to keep the environments of different algorithms completely comparable. We highlight the following five metrics used to evaluate the algorithm in 500 random test cases. "Success Rate"(SR): the rate of a robot reaching its goal without a collision. "Collision Rate"(CR): the

TABLE I
QUANTITATIVE RESULTS IN THE INVISIBLE AND VISIBLE SETTINGS. EACH ENTRY REPORTS $N = 5/N = 10$. BOLD INDICATES BEST.

	Invisible					Visible				
Methods	SR	CR	MT	RR	RD	SR	CR	MT	RR	RD
CADRL [9]	0.57/0.20	0.26/0.63	9.14/11.24	0.20/0.33	0.09/0.08	0.71/0.38	0.15/0.41	8.95/**10.60**	0.20/0.36	0.10/0.09
DH-CADRL*(Ours)	0.62/0.31	**0.14/0.38**	**8.95**/11.43	**0.06/0.13**	**0.14/0.12**	0.77/0.57	**0.00/0.00**	8.99/11.59	**0.03/0.06**	**0.15/0.15**
DH-CADRL(Ours)	**0.69/0.37**	0.19/0.44	9.01/**10.75**	0.10/0.21	0.13/0.11	**0.89/0.78**	**0.00/0.00**	**8.66**/10.91	0.04/0.08	**0.15**/0.14
SARL [12]	0.87/0.53	0.03/0.26	9.61/11.30	0.08/0.19	0.13/0.11	0.92/0.71	0.02/0.15	9.23/9.51	0.08/0.23	0.14/0.11
DH-SARL*(Ours)	**0.98**/0.74	**0.01**/0.25	**8.44/9.05**	0.04/0.16	0.14/0.13	0.99/0.98	**0.00/0.00**	**8.08/7.94**	**0.02/0.08**	**0.16/0.16**
DH-SARL(Ours)	0.96/**0.82**	0.02/**0.15**	8.63/9.86	**0.04/0.13**	**0.15/0.14**	**0.99/0.99**	**0.00/0.00**	8.33/8.69	**0.02/0.08**	**0.16/0.16**

rate of a robot colliding with other agents. "Mean Time"(MT): the average time taken by a robot to reach its goal (in seconds). "Risk Rate"(RR): the rate of the minimum separating distance between a robot and other agents is less than D_{risk} but has not yet collided. "Risk Distance"(RD): the average robot minimum separating distance (in meters) to the nearest obstacle agent in risk cases.

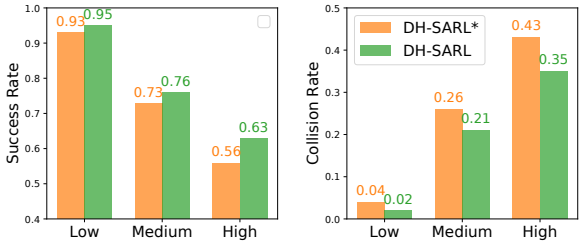


Fig. 6. Effect of speed heterogeneity. We compare our approach (DH-SARL) against the ablation one (DH-SARL*) under various speed-heterogeneity settings.

B. Quantitative Evaluation

1) Effect of Shape Heterogeneity: We first evaluate the algorithm in invisible test scenarios with the same obstacle density as the training (i.e., $N = 5$), and the quantitative results are summarized in Table I where the best ones are marked in bold. As expected, the method proposed in the current work provides more improvement to the DRL-based baselines in all five indicators. Compared with CADRL, the success rate of DH-CADRL is increased by 0.12, while the collision rate is reduced by 0.07. The success rate of DH-SARL is 0.09 higher than that of SARL. Recall that we set up the DH-CADRL* and DH-SARL* algorithms without risk-based reward shaping for the ablation experiments. And their performance on the five indicators is also satisfactory.

We also look into the visible test scenarios, where the obstacle agents can actively avoid the robot. Benefiting from the reciprocal collision avoidance, all policies are improved, and our proposed algorithms achieve the highest success rate and even zero collision results. These results indicate the advantages of the proposed OBC modeling and interactive state space settings in terms of compatibility with shape heterogeneity and reducing failure probability.

2) Effect of Speed Heterogeneity: Then, we generalize the trained models to more congested test scenarios (i.e., $N = 10$) without fine-tuning. As shown in Table I, DH-SARL is superior to DH-SARL* in terms of success rate, collision rate, risk rate, and risk distance. This result shows that our new reward function based on velocity-related collision risk is helpful in collision avoidance. But DH-CADRL is not better than DH-CADRL* on some metrics in the invisible scenarios. One possible explanation is that CADRL uses a maximin scheme for multi-agent scenarios, which can only take a single pair of interaction into account and possibly select an action that drives towards a third agent. For that reason, this work focuses the comparison against SARL which has better crowd interactions properties.

One reasonable question is whether accounting for velocity-

TABLE II
QUANTITATIVE RESULTS IN VARIOUS N SETTINGS.

Methods	N	SR	CR	MT	RR	RD
SARL [12]	5	0.87	0.03	9.61	0.08	0.13
	6	0.84	0.04	10.03	0.11	0.13
	7	0.75	0.08	10.66	0.13	0.12
	8	0.67	0.14	10.87	0.15	0.12
	9	0.58	0.20	11.06	0.19	0.11
	10	0.53	0.26	11.30	0.19	0.11
DH-SARL*(Ours)	5	**0.98**	**0.01**	**8.44**	**0.04**	0.14
	6	0.94	0.05	**8.44**	0.06	0.14
	7	0.91	0.09	**8.64**	0.09	0.14
	8	0.87	0.12	**8.83**	0.10	0.13
	9	0.82	0.17	**9.06**	0.13	0.13
	10	0.74	0.25	**9.05**	0.16	0.13
DH-SARL(Ours)	5	0.96	0.02	8.63	**0.04**	**0.15**
	6	**0.96**	**0.03**	8.94	**0.05**	**0.15**
	7	**0.95**	**0.03**	9.17	**0.07**	**0.14**
	8	**0.90**	**0.08**	9.33	**0.08**	**0.14**
	9	**0.87**	**0.10**	9.76	**0.11**	**0.14**
	10	**0.82**	**0.15**	9.86	**0.13**	**0.14**

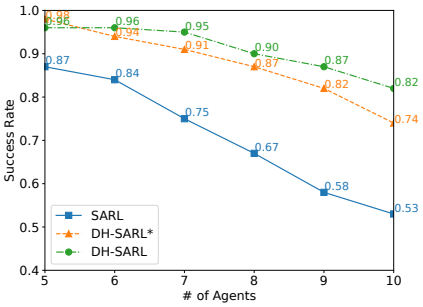
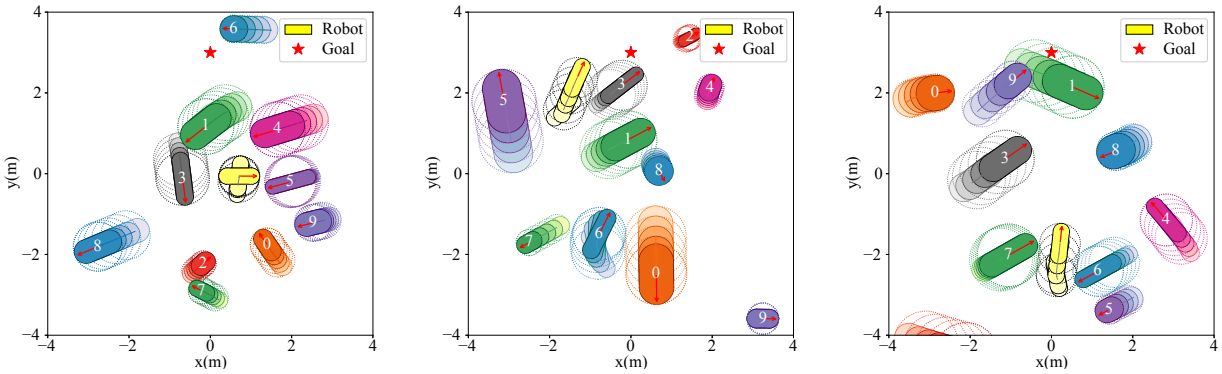
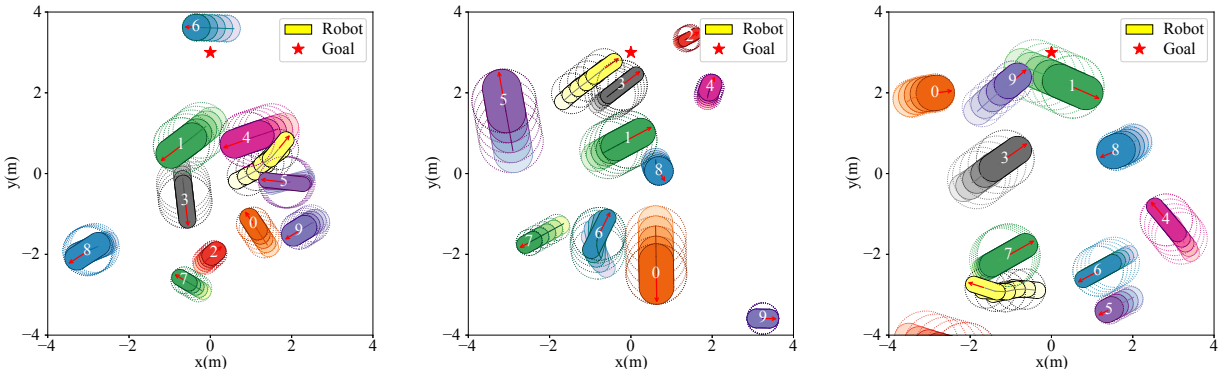


Fig. 7. Effect of obstacle density. The horizontal axis is the number of agents generated in the scenarios with the same size, which is proportional to the obstacle density. And the vertical axis represents the success rate of the algorithms in 500 random test cases.



(a) SARL: Case 1 (b) SARL: Case 2 (c) SARL: Case 3



(d) DH-SARL: Case 1 (e) DH-SARL: Case 2 (f) DH-SARL: Case 3

Fig. 8. Local trajectory comparison in several test cases. The top row shows the SARL robot using the yellow capsule, while the bottom row shows the DH-SARL robot. And the capsules in other colors represent simulated ORCA obstacle agents. Dotted circles represent the circumscribed circle of the capsules. Each picture is superimposed with 5 frames of images at 0.25s intervals, encoding time in light-dark. The white numerical index obstacle agents in the environment. DH-SARL framework generates desirable behaviors.

related collision risk improves performance due to the adaptation for speed-heterogeneity, or simply because of adding collision penalties for all obstacles. To answer this question, we test our ablation approach in three settings with different speed-heterogeneity. In the low speed-heterogeneity setting, multiple agents have the same v_{pref}, while the maximal v_{pref} is thrice the minimal v_{pref} in the high speed-heterogeneity setting, and twice in the medium speed-heterogeneity setting. Fig. 6 gives major results. DH-SARL performs comparably to DH-SARL* for low speed-heterogeneity and outperforms DH-SARL* significantly for high speed-heterogeneity, which demonstrates the effectiveness of our reward function design for addressing speed heterogeneity.

3) Effect of Obstacle Density: To fully evaluate the effectiveness of the proposed method, we examine test scenarios with different obstacle densities. For simplicity, we take the success rate of the SARL series algorithm as an example (Fig. 7) with other metrics shown in Table II. We have not performed fine-tuning, and thus, the performance of each algorithm decreases as the number of obstacle agents increases. Notably, the success rate reduction of the SARL algorithm is 0.34, while those of our proposed DH-SARL* and DH-SARL are 0.24 and 0.14, respectively. Although DH-SARL is slightly weaker than DH-SARL* when $N = 5$, it achieves the best results and minimal performance degradation in denser scenarios with $N = 6, 7, 8, 9, 10$. These results verify that the heterogeneity of agents can no longer be disregarded as the density of dynamic obstacles in the environment increases. Our method allows robots to deal with the considerable challenges posed by dense heterogeneous environments.

C. Qualitative Evaluation

We further investigate the effectiveness of our proposed method through qualitative analysis. Fig. 8 compares the local trajectories of DH-SARL and SARL baseline policy. In Test Case 1, the SARL robot is frozen between Obstacles 1, 3, 4, and 5, and thus, it fails to reach the goal point as shown in Fig. 8(a). Meanwhile, the DH-SARL robot fully utilizes the free space between Obstacles 4 and 5, passing through it as indicated in Fig. 8(d). In Test Case 2, the SARL robot passively evades Obstacle 3 and moves deviated from the target point as shown in Fig. 8(b), resulting in a longer navigation time. The DH-SARL robot adopts a more active strategy, exhibiting the intelligent behavior of moving parallel, accelerating and surpassing Obstacle 3 as shown in Fig. 8(e), and thus, improving navigation efficiency. In Test Case 3, the SARL robot tries to pass in front of Obstacle 7, causing a collision as depicted in Fig. 8(c). Meanwhile, the DH-SARL robot chooses to go around behind Obstacle 7, learning a safer collision avoidance policy, as illustrated in Fig. 8(f). We are

pleased to see that such intelligent behaviors is exactly what we have intended to achieve in collision avoidance among dense heterogeneous agents.

VI. Conclusion

This work develops a method for applying DRL-based algorithms to avoid collisions with dense heterogeneous agents. Our approach is general and can be used to enhance nearly all agent-level DRL-based navigation algorithms. We first use Orientated Bounding Capsules to model the agents and transform their interactive state space. Then, we design a new reward function via the velocity-related collision risk. Our quantitative results show that the proposed method outperforms state-of-the-art DRL-based social navigation algorithms in terms of success rate and safety, by accessing more usable space. Qualitatively, we demonstrate that our policy generates the desired collision avoidance behavior.

In the future, we will consider the uncertainty of human-robot interactive impact and environmental conditions. And we plan to train and test in a more advanced simulation environment (e.g., SUMMIT [34]), which is conducive to extending our method to the real world.

References

[1] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 797–803.

[2] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: the case for cooperation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 2153–2160.

[3] J. van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2008, pp. 1928–1935.

[4] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Phys. Rev. E*, vol. 51, no. 5, p. 4282, 1995.

[5] G. Ferrer, A. Garrell, and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 1688–1694.

[6] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of human-robot cooperation," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 335–356, 2015.

[7] Y. Luo, P. Cai, Y. Lee, and D. Hsu, "Gamma: A general agent motion model for autonomous driving," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 3499–3506, 2022.

[8] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Sci. Technol.*, vol. 26, no. 5, pp. 674–691, 2021.

[9] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 285–292.

[10] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 1343–1350.

[11] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 3052–3059.

[12] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 6015–6022.

[13] D. Bareiss and J. v. d. Berg, "Generalized reciprocal collision avoidance," *Int. J. Robot. Res.*, vol. 34, no. 12, pp. 1501–1514, 2015.

[14] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, 1998.

[15] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*. Berlin, Germany: Springer, 2011, pp. 3–19.

[16] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart, "Optimal reciprocal collision avoidance for multiple non-holonomic robots," *Proc. Int. Symp. Distrib. Auton. Robot. Syst.*, pp. 203–216, 2013.

[17] M. Svenstrup, T. Bak, and H. J. Andersen, "Trajectory planning for robots in dynamic human environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2010, pp. 4293–4298.

[18] X.-T. Truong and T. D. Ngo, "Toward socially aware robot navigation in dynamic and crowded environments: A proactive social motion model," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 4, pp. 1743–1760, 2017.

[19] P. Long, W. Liu, and J. Pan, "Deep-learned collision avoidance policy for distributed multiagent navigation," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 656–663, 2017.

[20] P. Henry, C. Vollmer, B. Ferris, and D. Fox, "Learning to navigate through crowded environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2010, pp. 981–986.

[21] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 4601–4607.

[22] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions," in *Proc. IEEE/CVF Conf. Comput. Vis. PatternRecognit.*, 2019, pp. 8475–8484.

[23] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, "Multi-agent tensor fusion for contextual trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. PatternRecognit.*, 2019, pp. 12 118–12 126.

[24] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[25] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv:1509.02971*, Sep. 2015.

[26] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv:1707.06347*, Jul. 2017.

[27] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 6252–6259.

[28] T. Fan, P. Long, W. Liu, and J. Pan, "Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios," *Int. J. Robot. Res.*, 2020.

[29] Y. Chen, C. Liu, B. E. Shi, and M. Liu, "Robot navigation in crowds by graph convolutional networks with attention learned from human gaze," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 2754–2761, 2020.

[30] A. J. Sathyamoorthy, U. Patel, T. Guan, and D. Manocha, "Frozone: Freezing-free, pedestrian-friendly navigation in human crowds," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 4352–4359, 2020.

[31] T. Fan, X. Cheng, J. Pan, P. Long, W. Liu, R. Yang, and D. Manocha, "Getting robots unfrozen and unlost in dense pedestrian crowds," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1178–1185, 2019.

[32] D. J. Gonon, D. Paez-Granados, and A. Billard, "Reactive navigation in crowds for non-holonomic robots with convex bounding shape," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4728–4735, 2021.

[33] S. Gottschalk, M. C. Lin, and D. Manocha, "Obbtree: A hierarchical structure for rapid interference detection," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Techn.*, 1996, pp. 171–180.

[34] P. Cai, Y. Lee, Y. Luo, and D. Hsu, "Summit: A simulator for urban driving in massive mixed traffic," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 4023–4029.