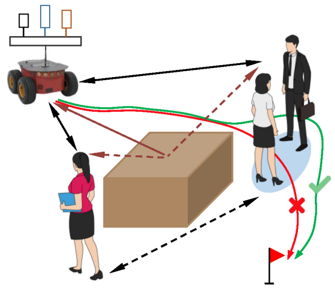
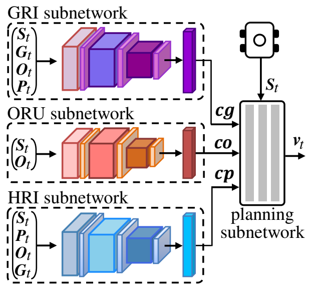
All Aware Robot Navigation in Human Environments Using Deep Reinforcement Learning

Xiaojun Lu[1], Angela Faragasso[1], Atsushi Yamashita[2], and Hajime Asama[1]

Abstract— **Mobile robots functioning in human environments should behave with a secure and socially-compliant manner. Although many studies have revealed the effectiveness of Deep Reinforcement Learning (DRL) in robot navigation, most of them can only handle the presence of human as independent individuals. Failing to consider groups may lead to the robot getting stuck or behaving rudely, and omitting to separately handle obstacles from pedestrians will cause low efficiency. In this work, we present a novel all-aware neural network that utilizes DRL to process groups, obstacles, and individuals simultaneously. The proposed solution employs a new Group–Robot Interaction (GRI) subnetwork to encode the mutual effects between groups and the robot, and a modified Obstacle–Robot Unilateral interaction (ORU) subnetwork is presented to avoid obstacle collisions caused by sensing noises or motion uncertainties. In addition, the influences of a pedestrian, obstacle, and group on other pedestrians or groups, that indirectly affect the robot, are also integrated into the Human–Robot Interaction (HRI) subnetwork or GRI subnetwork respectively by using map tensors. Finally, the GRI, ORU, and HRI subnetworks are aggregated into a planning subnetwork to train and derive an all-aware robot navigation policy based on DRL. Evaluation results in both real-world and simulation experiments show that the proposed approach outperforms the current cutting-edge methods.**



(a) Method overview



(b) All-aware network structure

Fig. 1. All-aware robot navigation method by jointly modeling GRI, ORU, and HRI. (a) The robot can successfully reach its goal by navigating in a socially-compliant manner, while avoiding intruding into groups and colliding with agents. The dashed and solid lines respectively indicate indirect and direct effects of agents on the robot. The bidirectional arrow represents the interaction; and the unidirectional arrow refers to the unilateral interaction, which denotes the one-way effect of an obstacle. (b) Proposed network containing the GRI, ORU, and HRI subnetworks: all agents aware robot navigation policy is learned through encoding both interactions and unilateral interactions between the robot and each agent.

I. INTRODUCTION

The application of mobile robots in human environments is rising with AI advancement [1]. Traditional methods for robot navigation usually consider humans to be obstacles, simply concentrating on the next action step, which result in shortsighted behaviors [2]. To achieve long-term navigation, researchers attempted to predict human trajectories before planning [3]. However, the separation of prediction and planning causes the "freezing robot problem". To address this challenge, several studies proposed cooperative navigation policies that take into account all agents jointly [4]. Nevertheless, these methods suffer from high computation costs and are hard to work in real time. To overcome this problem, recent studies have explored Deep Reinforcement Learning (DRL) for robot navigation, where a value network is trained to select action commands. Although significant progress has been achieved in DRL-based methods [5]–[8], pedestrians are only considered as independent individuals. However, up to 70% of pedestrians tend to walk in groups among commercial environments [9], [10]. Failing to consider groups may lead to the robot getting stuck, such as not knowing how to overtake or follow a group, and even intruding into groups which is not socially compliant. Another weakness of DRL-based methods [11]–[13] is they only model Human–Robot Interaction (HRI) and Human–Human Interaction (HHI), while ignoring obstacles' unilateral interactions. The unilateral interaction denotes the one-way effect of an obstacle on other agents, such as Obstacle–Robot Unilateral interaction (ORU) and Obstacle–Human Unilateral interaction (OHU). Omitting to separately consider unilateral interactions will cause low efficiency, as obstacles are different from humans in terms of unresponsiveness and lack of social norms.

In this work, we address above problems by considering individuals, obstacles, and groups simultaneously as shown in Fig. 1. First, we propose a novel Group–Robot Interaction (GRI) subnetwork to extract and aggregate the features of mutual effects between each group and the robot. Map tensors are calculated to encode indirect effects of other agents that influence the robot through this group. These tensors and the states of the group and the robot are then input to the GRI subnetwork to extract the features of GRIs. Second, we modify our prior ORU subnetwork [14] to avoid obstacle collisions arising from sensing noises or motion uncertainties by considering observation disturbances and safety redundancy. Third, we add Group–Human Interactions (GHIs) into the HRI subnetwork [14] to encode the indirect effects of an group on the robot through this human. Finally, the outputs of the three subnetworks, representing the expected features of HRIs, ORUs, and GRIs, are aggregated into a planning subnetwork to learn an all-aware navigation policy based on DRL. A new reward function is also designed to restrain the robot from approaching obstacles too close and intruding

[1]Department of Precision Engineering, The University of Tokyo, Tokyo, Japan, ({luxiaojun, faragasso, asama}@robot.t.u-tokyo.ac.jp).

[2]Department of Human and Engineered Environmental Studies, The University of Tokyo, Chiba, Japan, (yamashita@robot.t.u-tokyo.ac.jp).

into groups. Evaluations of both simulation and real-world experiments validate that the proposed method can enable the robot to follow or overtake a group appropriately, while avoiding collisions with agents and intruding into groups, outperforming the state-of-the-art methods.

II. BACKGROUND

A. Related Works

Numerous traditional approaches have been presented to improve the robot's social-awareness in navigation. One of the widely-used approaches is social forces model (SFM) [15], [16], which has proven to be effective. Other methods, such as optimal reciprocal collision avoidance [2] and reciprocal velocity obstacles [17], seek to identify mutually safe velocities. Despite their utility, these approaches need laborious parameter selection and has poor generality. In recent years, DRL has been intensively studied and applied to various fields [18]. Furthermore, DRL based algorithms have also been widely used to enable the robot to learn control policies through trial and error while interacting with the environment. Some recent studies have utilized DRL to develop navigation methods by directly using raw sensor data as input [5]. However, one challenge of these works is to model crowds with variable sizes [19]. To solve this problem, Everett et al. [20] used an LSTM module to transform the states of a crowd with varying sizes into a fixed-length vector. Based on pedestrians' proximities to the robot, this module processes the state of each pedestrian in descending order. However, prioritizing importance based on their distance from the robot is not reasonable. To overcome this challenge, attention-based DRL approaches have been proposed [11]–[13], which utilize an attention mechanism to assign varying degrees of importance to each pedestrian.

One shortcoming of above DRL based methods is obstacles are not separately considered from pedestrians, which leads to inefficiency [11]. To solve this issue, Liu et al. [7] integrated an occupancy grid map of obstacles into DRL. However, the map needs to be pre-built for each scenario, which results in low generality. In prior work [14], we attempted to encode obstacles and pedestrians using ORU and HRI subnetworks. However, we did not model motion uncertainties and sensing noises, which causes risky behaviors when the robot passes obstacles. Moreover, the method's applicability remains to be verified, as only simulations were conducted. In this work, we improve the ORU subnetwork by including observation disturbances during learning to simulate sensing noises in reality, and adding a closeness penalty term to the reward function to provide safety redundancy.

Another shortage of current DRL based methods is that pedestrians are simply treated as independent individuals. There is only one work that learns a Group-Aware robot navigation Policy (GAP) [21]. However, GAP only considers dynamic groups, simply modeling them as polygons without any group states, and merely trains the robot to avoid intruding into groups, which has not learned the interactions between the group and other agents (e.g., how to overtake or follow a group). Additionally, the agents' states are assumed to be known in simulation and provided by a motion capture system in real-world tests. Moreover, their training is conducted in a "circle crossing scenario" [8], where all agents move simultaneously towards a central point, which is impractical. In contrast, the new method proposed here has modeled the interactions between groups and other agents. We obtain all states from first-robot-view with sensors installed on the robot. Furthermore, we assign agents' start and goal positions randomly, making our approach more applicable. To the best of our knowledge, this is the first work that incorporates interaction and unilateral interaction between the robot, pedestrian, obstacle, and group for robot navigation based on DRL.

B. Problem Formulation

1) Preliminaries: The navigation task we study involves a robot moving to its goal in a complex setting with I pedestrians (including group members), K obstacles, and M groups, which are uniformly referred to as agents. Based on DRL framework, this task could be expressed as a sequential decision-making problem [5]. In our work, the agents' states are obtained through the sensors installed on the robot and represented in the robot-centric coordinates, with the x-axis originating at the robot's current position and extending towards its goal. Denote robot's position as $\boldsymbol{p}_t = [0, 0]$, orientation as θ_t, velocity as $\boldsymbol{v}_t = [v_x, v_y]$, radius as r, preferred speed as v_{pref}, and goal position as d^t_{gl}. The robot's state can be defined as $\boldsymbol{S}_t = [\boldsymbol{p}_t, \theta_t, \boldsymbol{v}_t, r, v_{\text{pref}}, d^t_{\text{gl}}]$.

Two basic types of obstacles, rectangular and circular, are considered. The obstacle's state is denoted as $\boldsymbol{O}^k = [\boldsymbol{p}^k, \theta^k, w^k, l^k]$ or $[\boldsymbol{p}^k, r^k]$, where $\boldsymbol{p}^k = [p^k_x, p^k_y]$ represents the position; θ^k denotes the orientation; w^k and l^k are the width and length, respectively; and r^k signifies the radius. For complex obstacles, such as L-shaped or U-shaped, they can be decomposed into combinations of basic types.

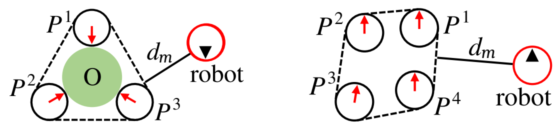
Each pedestrian has a similar state as the robot, however, the goal position and preferred speed are unobservable. Additionally, each pedestrian has a group ID, denoted as $g^i_t \in \{0,1,...,m,...,M\}$, where 0 represents the pedestrian is an individual and m implies it belongs to the m-th group. We define the pedestrian's state as $\boldsymbol{P}^i_t = [\boldsymbol{p}^i_t, \theta^i_t, \boldsymbol{v}^i_t, r^i, g^i_t]$.

2) Deep Reinforcement Learning for Robot Navigation: The joint states of (1+K+I+M) agents are denoted as $\boldsymbol{J}_t = [\boldsymbol{S}_t,...,\boldsymbol{O}^k,...,\boldsymbol{P}^i_t,...,\boldsymbol{G}^m_t,...]$, where $\boldsymbol{G}^m_t$ is the group's state that will be discussed in III-A. Assuming the $\boldsymbol{v}_t$ can be changed instantly by the robot in accordance with the navigation policy $\boldsymbol{v}_t = \boldsymbol{a}_t = \boldsymbol{\pi}(\boldsymbol{J}_t)$.

The optimal policy, $\boldsymbol{\pi}^* : \boldsymbol{J}_t \mapsto \boldsymbol{a}_t$, is formulated as

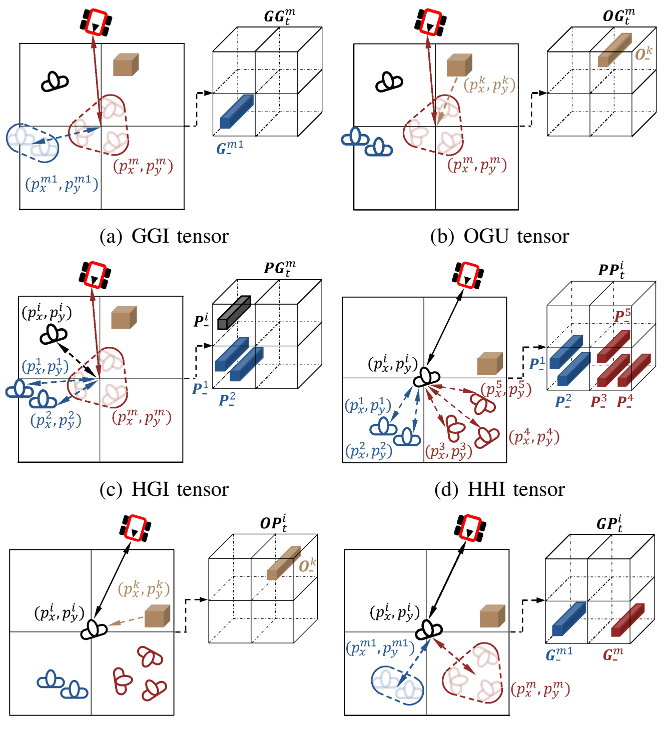
$$\begin{aligned}\boldsymbol{\pi}^*(\boldsymbol{J}_t) = \underset{\boldsymbol{a}_t \in \boldsymbol{A}}{\arg\max} \Big[& R(\boldsymbol{J}_t, \boldsymbol{a}_t) + \gamma^{\Delta t \cdot v_{\text{pref}}} \\ & \cdot \int P(\boldsymbol{J}_{t+\Delta t} \mid \boldsymbol{J}_t, \boldsymbol{a}_t) \cdot V^*(\boldsymbol{J}_{t+\Delta t}) \, \mathrm{d}\boldsymbol{J}_{t+\Delta t} \Big],\end{aligned} \tag{1}$$

where γ is a discount factor; $\boldsymbol{A}$ denotes an action space; Δt is the time interval; V^* represents the optimal value function; P is the transition probability; and $R(\boldsymbol{J}_t, \boldsymbol{a}_t)$ means the reward function, which punishes discomfort, intrusion, and collision, as well as awards task completion. A well-designed reward



(a) Space of static group (b) Space of dynamic group

Fig. 2. The spaces of static and dynamic groups are represented by convex hulls with dashed line. The red arrow means the member's orientation. d_m is the distance between the robot and a group. (a) denotes static group members face inward towards the shared o-space. (b) shows dynamic group members have similar walking speed, coincident orientation, and close proximity.



(a) GGI tensor (b) OGU tensor

(c) HGI tensor (d) HHI tensor

(e) OHU tensor (f) GHI tensor

Fig. 3. The features of GGI, OGU, and HGI, which have indirect effects on the robot through the m-th group, are encoded by tensors $\boldsymbol{PG}_t^m$, $\boldsymbol{OG}_t^m$, and $\boldsymbol{GG}_t^m$, respectively. The features of HHI, OHU, and GHI, which indirectly affect the robot through the i-th pedestrian, are modeled by tensors $\boldsymbol{PP}_t^i$, $\boldsymbol{OP}_t^i$, and $\boldsymbol{GP}_t^i$, separately.

function will guide the robot to learn faster and better [22], which will be fully discussed in III-B.

III. APPROACH

This section, firstly, introduces how to detect groups and construct their spaces. Then, the newly designed reward function, which contains a novel group reward and a new closeness penalty term for the obstacle reward, is presented. Finally, the architectures of GRI, ORU, HRI, and planning subnetworks are detailed as shown in Fig. 1(b).

A. Group Space

The F-formation method [23] is used here to detect static groups, as members in a static group maintain close proximity and face inward towards the shared o-space, as shown in Fig. 2(a). Meanwhile, a dynamic group comprises members with similar walking speed, coincident orientation, and close proximity, which can be detected by the DBSCAN method [24], as illustrated in Fig. 2(b). A group is defined as $\boldsymbol{\mathcal{G}}_t^m = \{i \in \{1,...,I\} \,|\, g_t^i = m\}$, where g_t^i is a label function indicating the i-th pedestrian's group ID. Given a pedestrian's space, $\mathbb{P}^i$, defined as a circle with center $\boldsymbol{p}_t^i$ and radius r^i, the group space, $\mathbb{G}_t^m$, is extracted as a convex hull:

$$\mathbb{G}_t^m = \text{Convexhull}(\{\mathbb{P}^i \mid i \in \boldsymbol{\mathcal{G}}_t^m\}), \tag{2}$$

where $\mathbb{G}_t^m$ is encoded by a vector containing a set of points' positions. An example of groups' spaces is shown in Fig. 2. The group state is defined as $\boldsymbol{G}_t^m = [\boldsymbol{p}_t^m, \boldsymbol{v}_t^m, \mathbb{G}_t^m, n_t^m]$, where $\boldsymbol{p}_t^m$ is the gravity center position of the convex hull; $\boldsymbol{v}_t^m$ represents the mean value of group members' velocities; and n_t^m is the number of group members.

B. Reward Function

In order to restrain the robot from intruding into groups, the reward function $R(\boldsymbol{J}_t, \boldsymbol{a}_t)$ is redesigned by proposing a novel group reward, R_{grp}. Meanwhile, to avoid obstacle collision arising from motion uncertainties, the obstacle reward, R_{obs}, is modified to take safety redundancy into account. $R(\boldsymbol{J}_t, \boldsymbol{a}_t)$, containing four parts, is defined as

$$R(\boldsymbol{J}_t, \boldsymbol{a}_t) = R_{\text{obs}} + R_{\text{prox}} + R_{\text{grp}} + R_{\text{gl}}, \tag{3}$$

The R_{obs}, penalizing robot's collision with obstacles or approaching them too close, is formulated as

$$R_{\text{obs}} = \begin{cases} -0.2 & \text{if } \exists d_k \leq 0 \\ -0.2 + \frac{1}{K}\sum_{k=1}^{K} d_k & \text{if } 0 < d_k \leq 0.2 \\ 0 & \text{otherwise} \end{cases}, \tag{4}$$

where d_k denotes the distance from an obstacle to the robot. Unlike prior methods [7], [14] that punish the robot only for $d_k \leq 0$, which is risky in reality as the motion uncertainties (e.g., response delays) may cause collisions even when $d_k > 0$, a new closeness penalty term is added into (4) in this work.

The proximity reward, R_{prox}, penalizing the robot for colliding with a pedestrian or intruding into a pedestrian's personal space, is designed as

$$R_{\text{prox}} = \begin{cases} -0.25 & \text{if } \exists d_i \leq 0 \\ \frac{-0.25}{I}\sum_{i=1}^{I} \frac{d_{\text{u}}^{0.2} - d_i^{0.2}}{d_{\text{u}}^{0.2}} & \text{if } 0 < d_i \leq d_u \\ 0 & \text{otherwise} \end{cases}, \tag{5}$$

where d_i is the distance from a pedestrian to the robot; and d_{u} denotes the threshold for uncomfortableness [8].

Although it is risky when the robot is close to a group, this behavior has already been punished in R_{prox} as the group's periphery is composed of pedestrians. Therefore, R_{grp} only contains penalty for robot's intruding into groups as

$$R_{\text{grp}} = \begin{cases} -0.25 & \text{if } \exists d_m \leq 0 \\ 0 & \text{otherwise} \end{cases}, \tag{6}$$

where d_m denotes the distance from a group to the robot.

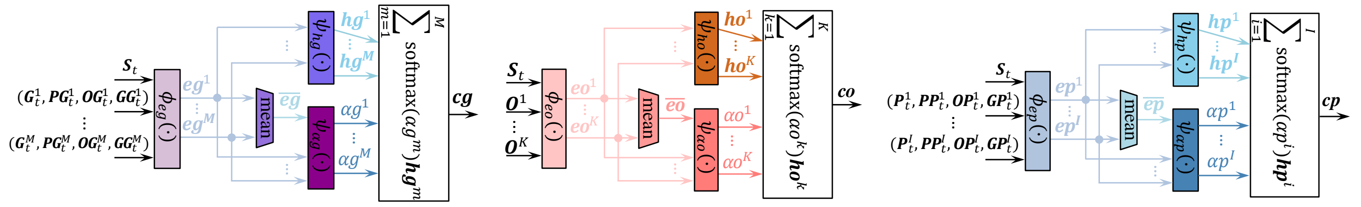
The goal reward, R_{gl}, following our prior work [14], is formulated as

$$R_{\text{gl}} = \begin{cases} 1 - 0.05 \cdot v_{\text{pref}} \cdot t_{\text{gl}}/d_{\text{gl}}^0 & \text{if } d_{\text{gl}}^t = 0 \\ -0.1 \cdot (d_{\text{gl}}^t - d_{\text{gl}}^{t-1})/d_{\text{gl}}^0 & \text{otherwise} \end{cases}, \tag{7}$$

where t_{gl} is the navigation time from the start to the goal.

C. Group–Robot Interaction Subnetwork

The robot is directly affected by a group, meanwhile is also indirectly influenced by other groups, obstacles, and



(a) GRI subnetwork (b) ORU subnetwork (c) HRI subnetwork

Fig. 4. Architectures of the GRI, ORU, and HRI subnetworks. (a) The GRI subnetwork extracts the features of GRIs ($\boldsymbol{cg}$), representing mutual effects between the robot and all groups. (b) The one-way effects of all obstacles on the robot, represented by the ORU features ($\boldsymbol{co}$), are calculated by the ORU subnetwork. (c) The mutual effects between the robot and all pedestrians, encoded by the HRI features ($\boldsymbol{cp}$), are derived with the HRI subnetwork.

pedestrians through this group. The features of GRIs are extracted by a GRI subnetwork, while Group–Group Interaction (GGI), Obstacle–Group Unilateral interaction (OGU), and Human–Group Interaction (HGI) are respectively encoded by map tensors $\boldsymbol{GG}_t^m$, $\boldsymbol{OG}_t^m$, and $\boldsymbol{PG}_t^m$, as shown in Fig. 3(a)–3(c). Given the m-th group has a neighboring group set with size NG_m, a map tensor $\boldsymbol{GG}_t^m$ centered at the m-th group is built to encode GGI:

$$\boldsymbol{GG}_t^m(\delta_x, \delta_y, :) = \sum\nolimits_{n \in NG_m} 1_{\delta_x \delta_y}(\boldsymbol{p}_t^n - \boldsymbol{p}_t^m)\boldsymbol{G}_{\_}^n, \quad (8)$$

where $\boldsymbol{G}_{\_}^n = [\boldsymbol{v}_t^n, \mathbb{G}_t^n, n_t^n]$ is the n-th group's partial state and $1_{\delta_x \delta_y}(\Delta_x, \Delta_y)$ checks whether (Δ_x, Δ_y) is in (δ_x, δ_y) or not.

Similarly to how $\boldsymbol{GG}_t^m$ is constructed, tensors $\boldsymbol{OG}_t^m$ and $\boldsymbol{PG}_t^m$ are built to encode OGU and HGI, respectively:

$$\boldsymbol{OG}_t^m(\delta_x, \delta_y, :) = \sum\nolimits_{k \in NO_m} 1_{\delta_x \delta_y}(\boldsymbol{p}^k - \boldsymbol{p}_t^m)\boldsymbol{O}_{\_}^k, \quad (9)$$

$$\boldsymbol{PG}_t^m(\delta_x, \delta_y, :) = \sum\nolimits_{i \in NP_m} 1_{\delta_x \delta_y}(\boldsymbol{p}_t^i - \boldsymbol{p}_t^m)\boldsymbol{P}_{\_}^i, \quad (10)$$

where NO_m and NP_m are sizes of the m-th group's neighboring obstacle and pedestrian sets, respectively; $\boldsymbol{O}_{\_}^k = [\theta^k, w^k, l^k]$ or $[r^k]$ is the k-th obstacle's partial state; and $\boldsymbol{P}_{\_}^i = [\theta^i, \boldsymbol{v}^i, g_t^i]$ is the i-th pedestrian's partial state.

The architecture of GRI subnetwork is shown in Fig. 4(a). First, we embed $\boldsymbol{S}_t$, $\boldsymbol{G}_t^m$, $\boldsymbol{GG}_t^m$, $\boldsymbol{OG}_t^m$, and $\boldsymbol{PG}_t^m$ into fixed-length vectors $\boldsymbol{eg}^m$. Then, the pairwise GRI features, $\boldsymbol{hg}^m$, are obtained by a Multi-Layer Perceptron (MLP) with the input of $\boldsymbol{eg}^m$ as

$$\boldsymbol{eg}^m = \phi_{eg}(\boldsymbol{S}_t, \boldsymbol{G}_t^m, \boldsymbol{GG}_t^m, \boldsymbol{OG}_t^m, \boldsymbol{PG}_t^m), \quad (11)$$

$$\boldsymbol{hg}^m = \phi_{hg}(\boldsymbol{eg}^m), \quad (12)$$

where $\phi_{eg}(\cdot)$ and $\phi_{hg}(\cdot)$ are embedding functions, using ReLU as activations. Simultaneously, another MLP is utilized to calculate the attention score of each group, based on the input of $\boldsymbol{eg}^m$, denoting their significance to the robot as

$$\alpha g^m = \psi_{\alpha g}(\boldsymbol{eg}^m, \overline{\boldsymbol{eg}}), \quad (13)$$

where $\psi_{\alpha g}(\cdot)$ is an MLP with an ReLU activation; and $\overline{\boldsymbol{eg}}$ is the mean pooling operation of all $\boldsymbol{eg}^m$, resulting in a fixed-length embedding vector. In the end, the effects of all neighbouring groups on the robot are calculated by combining αg^m and $\boldsymbol{hg}^m$ as

$$\boldsymbol{cg} = \sum\nolimits_{m=1}^{M} \mathrm{softmax}(\alpha g^m)\boldsymbol{hg}^m. \quad (14)$$

D. Obstacle–Robot Unilateral Interaction Subnetwork

An obstacle has a one-way effect on the robot but its state is unaffected by any other agents. Therefore, the ORU features can be derived with the ORU subnetwork, only taking robot's and obstacles' states as input. The ORU subnetwork, as shown in Fig. 4(b), contains following layers:

$$\boldsymbol{eo}^k = \phi_{eo}(\boldsymbol{S}_t, \boldsymbol{O}_t^k), \quad (15)$$

$$\boldsymbol{ho}^k = \phi_{ho}(\boldsymbol{eo}^k), \quad (16)$$

$$\alpha o^k = \psi_{\alpha o}(\boldsymbol{eo}^k, \overline{\boldsymbol{eo}}), \quad (17)$$

$$\boldsymbol{co} = \sum\nolimits_{k=1}^{K} \mathrm{softmax}(\alpha o^k)\boldsymbol{ho}^k, \quad (18)$$

where $\phi_{eo}(\cdot)$, $\phi_{ho}(\cdot)$, and $\psi_{\alpha o}(\cdot)$ are MLPs with ReLUs; $\boldsymbol{eo}^k$ is a fixed length vector; $\boldsymbol{ho}^k$ is the feature of unilateral interaction between the k-th obstacle and the robot; αo^k is the k-th obstacle's attention score; $\overline{\boldsymbol{eo}}$ is the mean pooling of all $\boldsymbol{eo}^k$; $\boldsymbol{co}$ is a weighted aggregation of all ORU features.

E. Human–Robot Interaction Subnetwork

The robot is directly affected by a pedestrian, meanwhile is also indirectly influenced by other pedestrians, obstacles, and groups through this pedestrian. The features of HRIs are extracted by a HRI subnetwork, while HHI, OHU, and GHI are respectively encoded with map tensors $\boldsymbol{PP}_t^i$, $\boldsymbol{OP}_t^i$, and $\boldsymbol{GP}_t^i$, as shown in Fig. 3(d)–3(f). Given the i-th pedestrian has a neighboring pedestrian set with size NP_i, a tensor centered at the i-th pedestrian is built to encode HHI:

$$\boldsymbol{PP}_t^i(\delta_x, \delta_y, :) = \sum\nolimits_{j \in NP_i} 1_{\delta_x \delta_y}(\boldsymbol{p}_t^j - \boldsymbol{p}_t^i)\boldsymbol{P}_{\_}^j. \quad (19)$$

Similarly, $\boldsymbol{OP}_t^i$ and $\boldsymbol{GP}_t^i$ are built to encode OHU and GHI:

$$\boldsymbol{OP}_t^i(\delta_x, \delta_y, :) = \sum\nolimits_{k \in NO_i} 1_{\delta_x \delta_y}(\boldsymbol{p}^k - \boldsymbol{p}_t^i)\boldsymbol{O}_{\_}^k, \quad (20)$$

$$\boldsymbol{GP}_t^i(\delta_x, \delta_y, :) = \sum\nolimits_{m \in NG_i} 1_{\delta_x \delta_y}(\boldsymbol{p}_t^m - \boldsymbol{p}_t^i)\boldsymbol{G}_{\_}^m, \quad (21)$$

where NO_i and NG_i are sizes of the i-th pedestrian's neighboring obstacle and group sets, respectively.

The HRI subnetwork architecture is depicted in Fig. 4(c). First, $\boldsymbol{S}_t$, $\boldsymbol{P}_t^i$, $\boldsymbol{PP}_t^i$, $\boldsymbol{OP}_t^i$, and $\boldsymbol{GP}_t^i$ are embedded into a fixed-length vector $\boldsymbol{ep}^i$. Then, $\boldsymbol{ep}^i$ is fed separately into two MLPs to obtain the pairwise HRI features $\boldsymbol{hp}^i$ and the attention score of each pedestrian αp^i. Finally, all $\boldsymbol{hp}^i$ and αp^i are combined as $\boldsymbol{cp}$ to represent all pedestrians' effects on the robot. This introduces the following recurrence:

$$\boldsymbol{ep}^i = \phi_{ep}(\boldsymbol{S}_t, \boldsymbol{P}_t^i, \boldsymbol{PP}_t^i, \boldsymbol{OP}_t^i, \boldsymbol{GP}_t^i), \quad (22)$$

$$\boldsymbol{hp}^i = \phi_{hp}(\boldsymbol{ep}^i), \quad (23)$$

$$\alpha p^i = \psi_{\alpha p}(\boldsymbol{ep}^i, \overline{\boldsymbol{ep}}), \quad (24)$$

$$\boldsymbol{cp} = \sum\nolimits_{i=1}^{I} \mathrm{softmax}(\alpha p^i)\boldsymbol{hp}^i, \quad (25)$$

where $\phi_{ep}(\cdot)$, $\phi_{hp}(\cdot)$, and $\psi_{\alpha p}(\cdot)$ are MLPs with ReLU activations; and $\overline{\boldsymbol{ep}}$ is the mean pooling of all $\boldsymbol{ep}^i$.

F. Planning Network

The features $\boldsymbol{cg}$, $\boldsymbol{co}$, and $\boldsymbol{cp}$ are concatenated with $\boldsymbol{S}_t$ and input to the planning subnetwork $f_v(\cdot)$, an MLP with an ReLU activation, as depicted in Fig. 1(b). The final output of $f_v(\cdot)$, $\boldsymbol{v}_t$, denotes the estimated action command for robot.

IV. EXPERIMENTS

A. Simulation Setup and Network Training Process

We leverage the robo-gym simulation environment [25], which is an integration of OpenAI Gym, Gazebo and ROS, to train and evaluate our method. It allows using sensor plugins, such as camera and Lidar, to simulate a perception system with sensing noises, and actuator plugins to emulate motion uncertainties, such as friction and response delays. This is one of the main features of the proposed method compared to prior work [19]–[21] that assume agents' states are known and presume no motion uncertainties. A custom implementation of the extended SFM [10] is applied to control the motion of groups. The start and goal (if any) positions of each pedestrian, obstacle, and group are randomly assigned inside a square with a width of 10 m. The robot's start and goal positions are (0, 1) and (-2, 9), respectively.

The dimensions of the hidden layers in $\phi_{eg}(\cdot)$, $\psi_{hg}(\cdot)$, $\psi_{\alpha g}(\cdot)$, $\phi_{eo}(\cdot)$, $\psi_{ho}(\cdot)$, $\psi_{\alpha o}(\cdot)$, $\phi_{ep}(\cdot)$, $\psi_{hp}(\cdot)$, $\psi_{\alpha p}(\cdot)$, and $f_v(\cdot)$ are set to (200, 150), (150, 100), (200, 200), (100, 50), (50, 25), (50, 50), (200, 150), (150, 100), (200, 200), and (400, 200, 150), respectively. The radius r of the robot, v_{pref}, Δt, and t_{limit} are set to 0.25m, 1m/s, 0.25s, and 25s, respectively. The temporal-difference method is used to train our network, based on fixed target network techniques and standard experience replay [18]. Imitation learning (lines 1–3) is used to initialize the network and then the DRL (lines 4–14) is utilized to optimize it, as shown in Algorithm 1. Details of the training can be found in our prior work [8].

B. Simulation Experiments

1) Ablation Study for GRI: To verify the effectiveness of our GRI subnetwork in group-awareness, ablation studies are conducted, where the model contains only GRI and HRI subnetworks, referred to as GADRL. We compare GADRL to GAP [21], which uses a similar HRI mechanism to ours but only models groups as polygons without considering group states, as discussed in II-A. We examine two scenarios, SG1: contains six individuals, a static group (two members), and a dynamic group (three members); and SG2: consists of eight individuals, two static groups (two and three members), and two dynamic groups (three and two members). GADRL and GAP are trained separately in their own simulation environment with SG1, followed by 500 random test episodes conducted in robo-gym environment with both SG1 and SG2.

Algorithm 1 Deep reinforcement learning.

```
1:  Use demonstration $D$ to initialize the replay memory $E$;
2:  Train the value network $V$ with $E$ by imitation learning;
3:  Use $V$ to initialize the target network $\widehat{V}$;
4:  for episodes from 1 to NUM do
5:      Randomly initialize $\boldsymbol{J}_{t=0}$;
6:      while not not collide, timeout, or reach the goal do
7:          $\boldsymbol{a}_t \leftarrow \arg\max_{\boldsymbol{a}_t \in \boldsymbol{A}} [R(\boldsymbol{J}_t, \boldsymbol{a}_t) + \gamma^{\Delta t \cdot v_{\text{pref}}} \cdot V(\boldsymbol{J}_{t+\Delta t})]$;
8:          $value \leftarrow R(\boldsymbol{J}_t, \boldsymbol{a}_t) + \gamma^{\Delta t \cdot v_{\text{pref}}} \cdot \widehat{V}(\boldsymbol{J}_{t+\Delta t})$;
9:          Update $E$ with tuple [$value$, $\boldsymbol{J}_{t+\Delta t}$, $\boldsymbol{a}_t$, $\boldsymbol{J}_t$,];
10:         Update $V$ with random minibatch tuples from $E$;
11:     end while
12:     Update $\widehat{V}$ with $V$;
13: end for
14: return $V$
```

TABLE I: Test results of ablation studies for GRI.

Scenario	Method	r_{s}	r_{c}	$t_{\text{gl\_a}}$(s)	n_{i}	n_{o}	n_{f}
SG1	GAP	0.95	**0.00**	11.71	**0**/500	3/500	2/500
	GADRL*	**1.00**	**0.00**	**10.23**	**0**/500	**48**/500	**29**/500
SG2	GAP	0.79	0.09	23.48	19/500	5/500	7/500
	GADRL*	**0.97**	**0.00**	**19.22**	**2**/500	**91**/500	**63**/500

Notes: (*) refers to our method with only GRI and HRI subnetworks. The r_{s}, r_{c}, $t_{\text{gl\_a}}$, n_{i}, n_{o}, and n_{f} denote the success rate, collision rate, average navigation time of successful episodes, total number of intruding, overtaking, and following group cases, respectively.

Table I reports the success rate r_{s}, collision rate r_{c}, average navigation time of successful episodes $t_{\text{gl\_a}}$, total number of intruding n_{i}, overtaking n_{o}, and following group cases n_{f}, respectively. In SG1, GAP and GADRL exhibit comparable results in r_{c} and n_{i}, while GADRL outperforming GAP slightly in r_{s} and $t_{\text{gl\_a}}$. As the scenario complexity increases, such as in SG2, the advantages of GADRL over GAP become obvious, particularly in r_{s} and $t_{\text{gl\_a}}$. This is because GADRL has learned both how to avoid intruding into groups and how to overtake and follow a group when necessary. In contrast, GAP has only learned to avoid intruding into groups without considering group states or encoding GRIs, which can be revealed by its n_{o} and n_{f}. As a result, GAP often gets stuck and has a deterioration in r_{s} and $t_{\text{gl\_a}}$. Furthermore, the r_{c} of GAP also worsens a little due to the robot's intruding into groups when it becomes stuck. The GRI ablation studies demonstrate the essentiality and benefits of considering group states and modeling GRIs.

2) Ablation Study for ORU: To assess the effectiveness of our modified ORU subnetwork in obstacle avoidance, ablation studies are executed with the model consisting of only ORU and HRI subnetworks, referred to as OADRL. OADRL is compared to the state-of-the-art GCNRL [11], SOADRL [7], and OUDRL [14], which use a similar HRI mechanism but no or different ORU mechanisms, as discussed in II-A. For a fair comparison, we feed obstacles into GCNRL, which initially contains no obstacles, as multiple standing pedestrians that have same sizes as the obstacles. We examine two scenarios, SO1: "6 obstacles & 5 individuals", and SO2: "10 obstacles & 8 individuals". The training and testing procedures are the same as described in IV-B.1.

Table II presents the r_{s}, r_{c}, $t_{\text{gl\_a}}$, and average minimum distance between the robot and obstacles of successful episodes d_{k_n}. GCNRL treats all agents as pedestrians without distinguishing obstacles, which causes GCNRL to

TABLE II: Test results of ablation studies for ORU.

Method	6 obstacles & 5 individuals				10 obstacles & 8 individuals			
	r_{s}	r_{c}	$t_{\mathrm{gl\_a}}$(s)	d_{k_n}(m)	r_{s}	r_{c}	$t_{\mathrm{gl\_a}}$(s)	d_{k_n}(m)
GCNRL	0.69	**0.00**	22.73	**0.26**	0.52	0.08	24.91	**0.24**
SOADRL	0.86	0.08	14.31	0.19	0.21	0.65	23.72	0.02
OUDRL	0.91	0.07	11.57	0.08	0.77	0.18	20.43	0.04
OADRL*	**1.00**	**0.00**	**9.15**	0.25	**0.98**	**0.00**	**18.87**	0.22

Notes: (*) refers to our method with only ORU and HRI subnetworks and without GRI subnetwork. The d_{k_n} denotes the average minimum distance between the robot and obstacles of successful episodes.

behave conservatively. As a result, it has a low r_{c} but the highest timeout rate (1–r_{s}–r_{c}) and longest $t_{\mathrm{gl\_a}}$. Nonetheless, GCNRL exhibits best performance in d_{k_n}, due to it assumes the obstacles possess social norms that the robot should follow. SOADRL behaves better than GCNRL in r_{s} and $t_{\mathrm{gl\_a}}$ with SO1, but its efficiency declines drastically with SO2. This is because the occupancy grid map is utilized by SOADRL to model obstacles, and it can merely achieve good performance if the map utilized in testing is similar to the one used in training. In other words, SOADRL is sensitive to environment changes and has a low generality. OUDRL encodes obstacles and pedestrians with various subnetworks, which enables it to outperform GCNRL and SOADRL in r_{s} and $t_{\mathrm{gl\_a}}$. However, OUDRL only penalizes the robot when obstacle collision happens, which makes it aggressive and underperforms GCNRL in r_{c} and d_{k_n}. Our OADRL outperforms the baseline methods almost in all metrics. The increase in r_{s} and decrease in $t_{\mathrm{gl\_a}}$ compared to GCNRL highlight the necessity of encoding pedestrians and obstacles separately. The decrease in r_{c} and increase in d_{k_n} compared to both SOADRL and OUDRL reveal the advantages of incorporating sensing noises and motion uncertainties, as well as providing safety redundancy for obstacle avoidance.

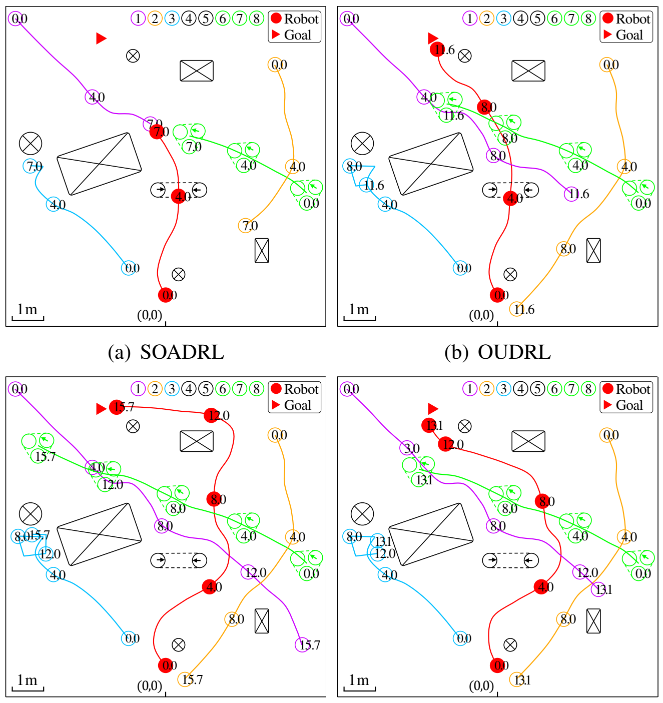
3) Quantitative Evaluation of the Full Model: Our full model, referred to as A2DRL, is compared against SOADRL, OUDRL, and GAP. To ensure a fair comparison, groups are added to SOADRL and OUDRL, controlled by the same policy as ours, and obstacles are fed into GAP following the approach described in IV-B.2. We examine two scenarios, SGO1: contains three individuals, six obstacles, a static group (two members), and a dynamic group (three members); and SGO2: consists of five individuals, eight obstacles, two static group (two and three members), and two dynamic group (three and two members). The training and testing procedures are the same as described in IV-B.1.

The full model method's test results are displayed in Table III. SOADRL and OUDRL perform poorly in n_{i}, which is not socially compliant, as they do not consider groups. The frequent intrusion into groups shows the need for a policy that factors in groups. By learning how to avoid intruding into groups, GAP achieves a better n_{i} than SOADRL and OUDRL. However, it still suffers from performance degradation when the environment becomes dense, such as a dramatic reduction in r_{s} and a large increase in both r_{c} and n_{i} with SGO2. This is due to the fact that GAP only considers the group's geometry, ignoring the interaction or unilateral interaction between the group and other agents, and

TABLE III: Test results of the full model in simulation.

	Method	r_{s}	r_{c}	$t_{\mathrm{gl\_a}}$(s)	n_{i}	n_{o}	n_{f}	d_{k_n}(m)
SGO1	SOADRL	0.79	0.12	14.52	59/500	1/500	2/500	0.17
	OUDRL	0.86	0.10	**12.02**	46/500	0/500	3/500	0.05
	GAP	0.91	0.03	15.06	7/500	2/500	3/500	**0.25**
	A2DRL*	**1.00**	**0.00**	12.81	**0**/500	**45**/500	**34**/500	0.23
SGO2	SOADRL	0.18	0.67	24.97	99/500	0/500	0/500	0.01
	OUDRL	0.72	0.22	22.76	87/500	0/500	1/500	0.03
	GAP	0.74	0.12	24.69	26/500	3/500	5/500	**0.23**
	A2DRL*	**0.96**	**0.01**	**20.75**	**3**/500	**84**/500	**70**/500	0.21

Notes: (*) refers to our method with GRI, ORU, and HRI subnetworks.



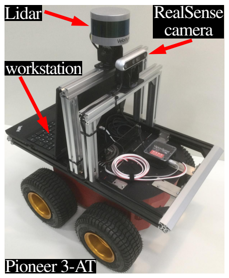
(a) SOADRL (b) OUDRL (c) GAP (d) A2DRL(ours)

Fig. 5. Example test trajectories of SOADRL, OUDRL, GAP, and A2DRL in SGO1. Rectangles and circles bearing a cross indicate obstacles, while hollow circles represent pedestrians. Groups are outlined by convex hulls and their members are highlighted in the same color. Pedestrians' positions are labeled with time and their IDs are displayed on the right-top corner. The coordinate origin is marked by (0, 0).

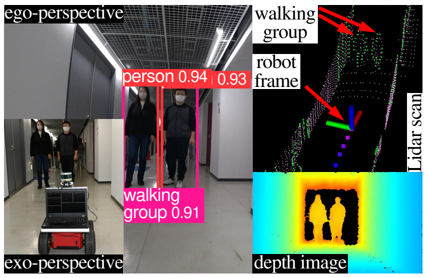
has not learned how to follow or overtake a group, as shown by its n_{o} and n_{f}. Our A2DRL outperforms other methods, achieving the highest r_{s}, lowest r_{c}, least n_{i}, and nearly best results in $t_{\mathrm{gl\_a}}$ and d_{k_n}. These results demonstrate the advantages of our novel GRI subnetwork in group awareness and improving navigation efficiency, as well as the benefits of our modified ORU subnetwork in reducing obstacle collision by considering noises and uncertainties.

Notably, it is found that OUDRL has the shortest $t_{\mathrm{gl\_a}}$ in SGO1 but not in SGO2. This may be due to that OUDRL reaches its goal successfully by intruding into groups in a sparse scenario, however, intruding causes collision in a dense scenario. We also observe GAP has the best performance in d_{k_n} because it treats obstacles as pedestrians.

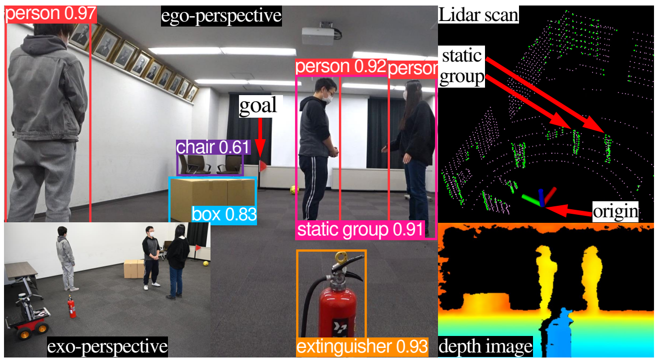
4) Qualitative Evaluation of the Full Model: The effectiveness of our method are further investigated by qualitative evaluation. Test trajectory examples of SOADRL, OUDRL, GAP, and A2DRL with SGO1 are shown in Fig. 5. The



(a) Hardware

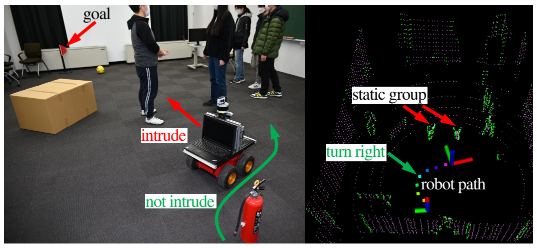


(b) Experiment in corridor

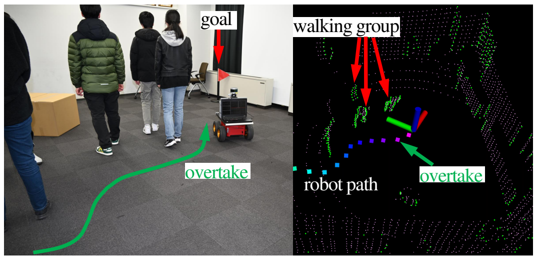


(c) Experiment in room

Fig. 6. Real-world experiments. (a) is the experiment hardware. (b) and (c) show snapshots of real-time experiments pictured from the robot ego- and exo- perspectives, 3D Lidar data, and depth images.



(a) Bypassing a static group



(b) Overtaking a walking group

Fig. 7. Results of room real-world experiment. The left of (a) and (b) are snapshots of real-time experiments, while the right are pictured paths chosen by the robot based on A2DRL. (a) The robot bypasses a static group that occludes the robot's straight route to its goal, choosing a more socially compliant path to avoid intruding. (b) The robot overtakes a walking group that is moving slowly, implementing an efficient navigation policy to minimize time costs.

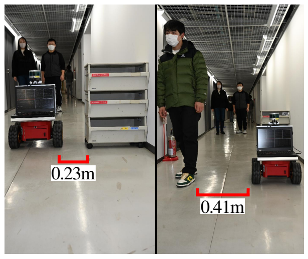
robot trained by SOADRL chooses to traverse through the static group to obtain a short route to its goal. However, it collides with the first pedestrian at around 7.0s, due to its inability to anticipate the pedestrian's direction change when encountering an obstacle. Similar to SOADRL, OUDRL also intrudes into the static group to reduce $t_{\mathrm{gl\_a}}$ and successfully avoids collisions with the first pedestrian owing to its modeling of obstacle unilateral interaction. Unlike SOADRL and OUDRL, GAP can avoid intruding into groups, but it simply treats them as geometric spaces without accounting for group interaction with other agents or considering group states. Therefore, when encountering a group at approximately 7.5s, GAP does not know how to follow or overtake the group, resulting in a detour path and a large $t_{\mathrm{gl\_a}}$. In contrast, our A2DRL overtakes the group from 8.0s to 12.0s while maintaining a suitable distance from it, which leads to a shorter route and a smaller $t_{\mathrm{gl\_a}}$ compared to GAP.

C. Real-world Experiments

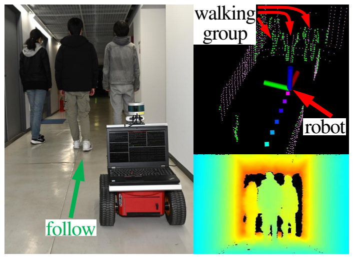
In addition to the simulation experiments described in the above subsections, the trained policy is further evaluated in reality, using a Pioneer 3-AT robot equipped with a RealSense camera D455 and a 3D Lidar as shown in Fig. 6(a). First YOLOv5 [26] is applied on the real-time RGB images to detect objects and Kalman filter is utilized to obtain objects' positions and sizes by processing the aligned depth images and Lidar scan [28] as illustrated in Fig. 6(b) and 6(c). Then, pedestrians' velocities are estimated with optical flow [27] and groups are detected based on F-Formantions and DBSCAN. A mobile workstation, composed of an Intel Core i9-13900K CPU with 32 cores and 2 threads per core, is utilized for carrying out high-level computations, such as pedestrian detection and network inference. The motion command is updated at 10Hz. To validate the applicability of our method from the training environment to real-world settings and its ability to function across diverse scenarios, we performed two experiments in a room and a corridor, as shown in Fig. 7 and 8.

1) Experiment in a Room: To verify the applicability of A2DRL, experiment is first conducted in a rectangular room (8m $*$ 9m), which is comparable to the environment in simulation in terms of size and agent composition. The world coordinate's origin is set at the midpoint of the bottom edge, and the robot's start and goal positions are (0, 0.5) and (-2, 8), respectively. Benefiting from A2DRL, the robot is able to bypass a static group who is standing on the front of its path, disregarding to the shorter route and avoiding intrusion, as shown in Fig. 7(a). Moreover, the robot successfully overtake a slow walking group to save its navigation time as illustrated in Fig. 7(b). These results verify the capability of the proposed GRI subnetwork in being group aware, able to not intrude into groups and overtake a group when necessary.

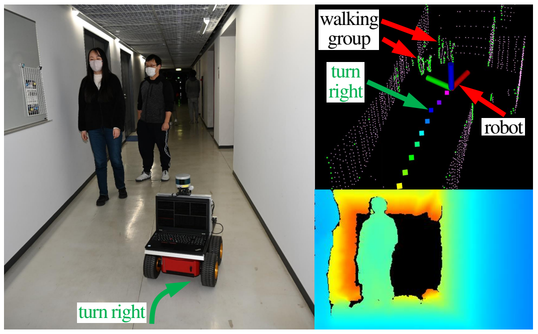
2) Experiment in a Corridor: The primary aim of corridor experiment is to examine the generality of our proposed method, as the corridor setting is greatly different from the training environments. The corridor is shaped as a rectangle (2.1m $*$ 29m), three times longer and one-fifth width of the training environment. The robot starts from the midpoint of the short edge and ends on the opposite side. As shown in Fig. 8(a), the robot takes variant strategies and distances to pass an obstacle and a pedestrian, respectively guaranteeing social norms and safety, which verifies the effectiveness of our modified ORU subnetwork in avoiding obstacle collision. Moreover, the robot decelerates to follow a walking group when the path is obstructed, as well as slows down and



(a) Passing obstacle vs. pedestrian



(b) Following a walking group



(c) Passing a walking group

Fig. 8. Results of corridor real-world experiment. (a) The robot utilizes distinct strategies and distances to pass an obstacle and a pedestrian. (b) The robot decelerates to follow a walking group, when the path is obstructed, as opposed to stopping. (c) The robot slows down, turns right, and interactively avoids collision with a walking group, passing it safely and efficiently in the narrow and crowded passage.

turns to pass a walking group instead of stopping, as shown in Fig. 8(b) and 8(c). These result proves that, based the proposed method, the robot has successfully learned how to interact with groups.

V. CONCLUSION

A novel DRL architecture for all aware robot navigation in human coexisting environments, A2DRL, has been developed in this work. There are three main contributions: First, a new GRI subnetwork, which models the mutual effects between the groups and the robot, has been proposed. Second, our prior ORU subnetwork is modified to avoid obstacle collision arising from sensing noises and motion uncertainties, as well as to provide safety redundancy by designing a new reward function for DRL. Third, the results of simulation experiments demonstrate that the proposed approach outperforms the current cutting-edge methods regarding the group awareness, collision rate, success rate, and navigation time. The real-world experiments demonstrate our approach's applicability, which can be transferred from learning environment to reality, and its robust generality to work in different scenarios. For future work, we plan to develop an "active path clearing" policy, such as asking for passage with a speaker and moving away an obstacle (e.g., a chair) with an equipped manipulator, to improve navigation efficiency in denser and more complex environments.

REFERENCES

[1] H.A. Pierson and M.S. Gashler, "Deep learning in robotics: A review of recent research," Adv. Robot., 2017, vol. 31, no. 16, pp. 821–835.
[2] J. Van den Berg, S.J. Guy, and et al., "Reciprocal n-body collision avoidance," Robot. Res., Springer, 2011, no. 70, pp. 3–19.
[3] V.V. Unhelkar, C. Perez-D'Arpino, and et al., "Human-robot co-navigation using anticipatory indicators of human walking motion," IEEE Int. Conf. Robot. Autom., 2015, pp. 6183–6190.
[4] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," IEEE/RSJ Int. Conf. Intell. Robots Syst., 2010, pp. 797–803.
[5] Y.F. Chen, M. Liu, and et al., "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," IEEE Int. Conf. Robot. Autom., 2017, pp. 285–292.
[6] C. Chen, Y. Liu, and et al., "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," IEEE Int. Conf. Robot. Autom., 2019, pp. 6015–6022.
[7] L. Liu, D. Dugas, and et al., "Robot navigation in crowded environments using deep reinforcement learning," IEEE/RSJ Int. Conf. Intell. Robots Syst., 2020, pp. 5671–5677.
[8] X. Lu, H. Woo, and et al., "Socially aware robot navigation in crowds via deep reinforcement learning with resilient reward functions," Adv. Robot., 2022, vol. 36, no. 8, pp. 388–403.
[9] A.F. Aveni, "The not-so-lonely crowd: Friendship groups in collective behavior," Sociometry, 1977, vol. 40, no. 1, pp. 96–99.
[10] M. Moussaïd, N. Perozo, and et al., "The walking behaviour of pedestrian social groups and its impact on crowd dynamics," PloS one, 2010, vol. 5, no. 4, e10047.
[11] Y. Chen, C. Liu, and et al., "Robot navigation in crowds by graph convolutional networks with attention learned from human gaze," IEEE Robot. Autom. Lett., 2020, vol. 5, no. 2, pp. 2754–2761.
[12] C. Chen, S. Hu, and et al., "Relational graph learning for crowd navigation," IEEE/RSJ Int. Conf. Intell. Robots Syst., 2020, pp. 10007–10013.
[13] W. Shi, Y. Zhou, and et al., "Enhanced spatial attention graph for motion planning in crowded, partially observable environments," IEEE Int. Conf. Robot. Autom., 2022, pp. 4750–4756.
[14] X. Lu, H. Woo, and et al., "Robot navigation in crowds via deep reinforcement learning with modeling of obstacle uni-action," Adv. Robot., 2023, vol. 37, no. 4, pp. 257–269.
[15] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," Phys. Rev. E, 1995, vol. 51, no. 5, pp. 4282.
[16] G. Ferrer and A. Sanfeliu, "Behavior estimation for a complete framework for human motion prediction in crowded environments," IEEE Int. Conf. Robot. Autom., 2014, pp. 5940–5945.
[17] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," IEEE Int. Conf. Robot. Autom., 2008, pp. 1928–1935.
[18] V. Mnih, K. Kavukcuoglu, and et al., "Human-level control through deep reinforcement learning," Nature, 2015, vol. 518, no. 7540, pp. 529–533.
[19] Y.F. Chen, M. Everett, and et al., "Socially aware motion planning with deep reinforcement learning," IEEE/RSJ Int. Conf. Intell. Robots Syst., 2017, pp. 1343–1350.
[20] M. Everett, Y.F. Chen and J.P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," IEEE/RSJ Int. Conf. Intell. Robots Syst., 2018, pp. 3052–3059.
[21] K. Katyal, Y. Gao, and et al., "Learning a group-aware policy for robot navigation," IEEE/RSJ Int. Conf. Intell. Robots Syst., 2022, pp. 11328–11335.
[22] G. Lample and D.S. Chaplot, "Playing FPS games with deep reinforcement learning," AAAI Conf. Artif. Intell., 2017, vol. 31, no. 1.
[23] F. Setti, C. Russell, and et al., "F-formation detection: Individuating free-standing conversational groups in images." PloS one, 2015, vol. 10, no. 5, e0123783.
[24] A. Wang, C. Mavrogiannis, and A. Steinfeld, "Group-based motion prediction for navigation in crowded environments," Conf. Robot Learn., PMLR, 2022, pp. 871–882.
[25] M. Lucchi, F. Zindler, and et al., "robo-gym: An open source toolkit for distributed deep reinforcement learning on real and simulated robots," IEEE/RSJ Int. Conf. Intell. Robots Syst., 2020, pp. 5364–5371.
[26] G. Jocher, A. Stoken, and et al., "ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations," 2021, Available: https://doi.org/10.5281/zenodo.4679653
[27] D. Fortun, P. Bouthemy, and C. Kervrann, "Optical flow modeling and computation: A survey," Comput. Vis. Image Und., 2015, vol. 134 pp. 1–21.
[28] T. Shan, and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," IEEE/RSJ Int. Conf. Intell. Robots Syst., 2018, pp. 4758–4765.