Autonomous Navigation System in Pedestrian Scenarios Using a Dreamer-Based Motion Planner

Wei Zhu and Mitsuhiro Hayashibe, *Senior Member, IEEE*

***Abstract*—Navigation among pedestrians is a crucial capability of service robots; however, it is a challenge to manage time-varying environments stably. Recent deep reinforcement learning (DRL)-based approaches to crowd navigation have yielded numerous promising applications. However, they rely heavily on initial imitation learning and colossal positive datasets. Moreover, the difficulties in accurately localizing robots, detecting and tracking humans, representing and generalizing reciprocal human relationships restrict their deployment in real-world problems. We propose a Dreamer-based motion planner for collision-free navigation in diverse pedestrian scenarios. Our RL framework can completely learn from zero experience via a model-based DRL. The robot and humans are first projected onto a map, which is subsequently decoded into low-dimensional latent state. A predictive dynamic model in the latent space is jointly created to efficiently optimize the navigation policy. Additionally, we leverage the techniques of system identification, domain randomization, clustering and LiDAR SLAM for practical deployment. Simulation ablations and real implementations demonstrate that our motion planner outperforms state-of-the-art methods, and that the navigation system can be physically implemented in the real world.**

***Index Terms*—Autonomous vehicle navigation, human-aware motion planning, reinforcement learning.**

I. Introduction

Autonomous driving systems are becoming prevalent in human society because of their promising prospects of high efficiency, safety, and intelligence. Additionally, an aging society, labor shortages, and noncontact services during the pandemic promoted the research and development of autonomous mobile robots in hospitals, restaurants, hotels, etc. [1]. However, socially aware robot navigation is a highly complex task because it involves mapping and localization, human detection and behavior analysis, social rules, and decision and planning [2].

On one hand, the modules of perception and motion planning are separately studied by autonomous driving companies and research institutes [3]; thus, human–vehicle interaction is not considered. Moreover, the surrounding humans are individually detected and tracked, and their motions are independently analyzed; therefore, the reciprocal relationships among pedestrians are excluded. On the other hand, state-of-the-art algorithms proposed by academia assume that states such as human number, position, and speed are fully known and simply focus on motion planning, which limits their generalizability and hinders sim-to-real transfer [4], [5], [6], [7].

In the earlier stages, rule-based motion planners played a dominant role in crowd navigation. Two pioneering approaches include optimal reciprocal collision avoidance (ORCA) [8] and the social force model (SFM) [9]. However, their one-step planning framework resulted in short-sighted, unsafe, and unnatural behaviors [10]. To plan motion over a long horizon, an intuitive strategy is to first predict human trajectories based on a model and subsequently select an optimal path for robots [11], [12]. Nevertheless, human motion models generally focus on individuals, ignoring the social relationships among pedestrians.

Recently, crowd navigation has switched to learning-based methods because of their prominent capability of representing the latent features of human–human and human–robot interactions and planning optimal navigation paths at a long time-scale. The collision avoidance with deep reinforcement learning (CADRL) algorithm [13] and its extension, socially aware CADRL (SA-CADRL), are the main algorithms used in the field of learning-based crowd navigation. These approaches feed all human states, such as positions and speeds, into deep neural networks (DNNs) to extract the implicit reciprocal motion features of humans, which are further fed into deep reinforcement learning (DRL)-based policy neural networks to learn an optimal motion planner. However, these two methods are not generalized to human numbers; thus, DNNs need to be redefined and retrained when the human number changes. Therefore, a subsequent algorithm, long short-term memory reinforcement learning (LSTM _ RL), leverages LSTM to represent the relational motion features of humans to allow arbitrary human numbers [14], [15]. Because the inputs of LSTM neural networks are sorted by the distance between the robot and each human in descending order, the relationship among humans is not reciprocal.

To comprehensively describe the reciprocal relationship among pedestrians, the attention mechanism and graph convolutional network (GCN) are broadly embedded in DRL-based crowd navigation [5], [6], [10], [16]. However, the aforementioned navigation algorithms assume fully known human states, including human number, position, and speed, and rely heavily on imitation learning and colossal positive datasets collected by rule-based methods such as ORCA, which may result in

Manuscript received 8 January 2023; accepted 24 April 2023. Date of publication 8 May 2023; date of current version 17 May 2023. This letter was recommended for publication by Associate Editor S. Ha and Editor J. Kober upon evaluation of the reviewers' comments. This work was supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant JP20KK0256. *(Corresponding author: Wei Zhu.)*

The authors are with the Department of Robotics, Graduate School of Engineering, Tohoku University, Sendai 980-8579, Japan (e-mail: zhu.wei.r5@dc.tohoku.ac.jp; mitsuhiro.hayashibe@inria.fr).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2023.3273514, provided by the authors.

Digital Object Identifier 10.1109/LRA.2023.3273514

an early sub-optimum. Moreover, precisely tracking humans and estimating their speeds is difficult in the real world owing to various uncertainties; thus, sim-to-real transfer is a crucial challenge for these methods.

To let go of the assumption of fully known human information in simulations, directly using raw sensor data is a promising alternative [17], [18], [19], [20], [21]. The reciprocal relationship among pedestrians is extracted from consecutive raw sensor data such as high-dimensional LiDAR scans with DNNs. However, direct handling of raw sensor data is inefficient. Consequently, imitation learning and colossal positive datasets are required to initialize DNNs. In addition, the lack of open-source solutions limits further development and comprehensive ablation.

Our study addresses these shortcomings in a comprehensive manner. First, we only detect humans and obtain their positions while excluding human tracking and speed estimation. After localizing the robot and humans, we create an RGB map that can be maturely processed using autoencoder algorithms to represent the instant relationship between humans and the robot. Accordingly, our algorithm can free the assumption of fully known environments and improve generalizability with respect to the human number and speed. In addition, inspired by the Dreamer approach, a model-based RL to address long-horizon tasks from images purely by latent imagination [22], we create a dynamic model with recurrent neural networks (RNNs) to accumulate history information and predict future states over a long horizon, thus reducing the probability of local optima. Moreover, the dynamic model can facilitate policy learning via the model-based DRL framework; thus, we can completely learn an optimal navigation policy without any imitation learning or a massive dataset. The contributions of this study are summarized as follows.

- A complete and publicly accessible autonomous navigation system among pedestrians is developed. We precisely obtain the robot pose using a LiDAR SLAM algorithm, and extract humans via a clustering approach. Moreover, we plan robot motion using a model-based DRL framework to avoid pedestrians and reach a target with a high success rate and navigation efficiency.
- We propose a Dreamer-based motion planning algorithm that can efficiently obtain an optimal motion planner and be generalized to arbitrary human number, variable human speed, and complex human relationships.
- We reproduce several state-of-the-art algorithms for more comprehensive ablation and ensure they are open-sourced. Additionally, sufficient sim-to-real experiments are implemented using domain randomization and system identification techniques.

The code of the whole project is publicly available at https://github.com/zw199502/navigation_among_pedestrians and the video is shown at https://youtu.be/KM2WPpQBfrI.

II. Related Work

A. Navigation System

A complete navigation system integrates perception, decision, planning, and control modules, which are broadly researched and developed for autonomous driving vehicles [3], [23]. However, these modules are studied separately in both industry and academia. Although there are several mature and open-source solutions, such as Apollo[1] and Autoware,[2] publicly accessible navigation systems among pedestrians are rare in the community of service robots. In this study, we construct a complete navigation system, in which the perception and control modules are derived from mature methods and decision and planning are achieved by a novel Dreamer-based motion planner.

B. Motion Planning Among Pedestrians

In the early stage, rule-based motion planners constituted mainstream crowd navigation, such as ORCA [8], SFM [9], and trajectory-prediction-based path optimization [11], [12]. With the rapid development of deep learning, researchers and engineers are focusing on learning-based methods, wherein DRL-based navigation algorithms are attractive because of their promising representation and optimization capabilities [5], [6], [10], [13], [14], [15], [16], [18], [19], [20], [24], [25], [26]. CADRL [13] is a pioneering study in the use of DRL for social navigation. However, its value function neglects the social relationships among pedestrians, as it only considers the robot's full state and one pedestrian's observable state. LSTM _ RL [14] improves upon CADRL by leveraging LSTM to represent pairs of the robot's state and all pedestrians' states, but its ability to capture reciprocal relationships is limited since pairs are ordered by distance and then fed into LSTM networks. Socially aware RL (SARL) [10] and relational graph learning (RGL) [16] represent state-of-the-art extensions to CADRL and LSTM _ RL by using self-attention mechanisms and graph convolutional networks, respectively, to capture interactions and reason about relations between agents. However, these methods assume fully known pedestrian information and require massive positive datasets for learning, leading to degraded performance in real-time settings. EGO [18] and LSTM _ EGO [21] offer a more direct mapping of raw sensor data to navigation actions, but open-source solutions for replicating their results are scarce. We propose an open-source Dreamer-based motion planner that can learn completely from zero experience and release the ideal assumptions of fully known environments.

C. Dreamer

The Dreamer algorithm is a reinforcement learning agent that addresses long-horizon tasks from images purely by latent imagination [22], which yields a large number of achievements in simulated environments, such as Atari games and MuJoCo robots [22], [27], [28], [29]. Conversely, we focus more on real implementations of collision-free and socially aware robot navigation by leveraging the key idea of the Dreamer algorithm. A map is created to represent complex scenarios with variable human numbers and random initial states. A dynamic model with a map as a unique observation is learned to represent social relationships among humans. The learned model can facilitate learning complex behaviors, thereby enabling the robot to learn an optimal navigation policy without any prior experience.

[1] [Online]. Available: https://github.com/chrislgarry/Apollo-11

[2] [Online]. Available: https://github.com/autowarefoundation/autoware

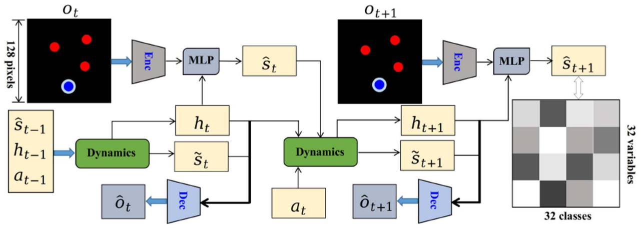

Fig. 1. Framework of Dreamer-based motion planner with image observation. The multi-layer perceptron (MLP) is used for learning and inference. The encoder network (Enc) comprises convolutional neural networks (CNNs), while the decoder network (Dec) is constructed using transposed CNNs. To propagate historical information, recurrent neural networks (RNNs) are employed in the dynamics module.

III. Approach

We first illustrate the formulation of the problem of navigation among pedestrians with model-based RL and subsequently describe the details of the model creation and navigation policy learning using the Dreamer algorithm. In addition, we introduce a completely autonomous navigation system with perception, planning, and control modules. Fig. 1 shows the overview of the Dreamer-based motion planner.

A. Problem Formulation

We formulate the problem of crowd navigation as a partially observable Markov decision process (POMDP) defined by a tuple $(\mathcal{S}, \mathcal{H}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{O}, \gamma)$, where $\mathcal{S}$ represents the stochastic state space, $\mathcal{H}$ is the deterministic history feature space, $\mathcal{A}$ denotes the action space, $\mathcal{P}$ is the state transition model, $\mathcal{R}$ represents the instant reward space, $\mathcal{O}$ is the observation space, and γ is a discount factor.

The unique observations are the sequenced RGB images $(o_t, o_{t+1}, \ldots, o_{t+K})$, each of which illustrates the instantaneous positions of humans and the robot. We decode the high-dimensional image $o_{t+\tau}$ into the stochastic latent state $s_{t+\tau}$ using considerably fewer variables. In addition, historical state information is accumulated as deterministic feature $h_{t+\tau}$. Given action $a_{t+\tau}$, stochastic state $s_{t+\tau}$, and hidden information $h_{t+\tau}$, the next state $s_{t+\tau+1}$ and accumulated feature $h_{t+\tau+1}$ can be derived from the state transition model $\mathcal{P}$. Given policy π, we can represent the expectation of the value function starting from state $s_{t+\tau}$ as follows:

$$v_\pi(s_{t+\tau}) = \mathbb{E}_\pi \left[\sum_{i=1}^{\infty} \gamma^{i-1} r_{t+\tau+i} \right], \tag{1}$$

where $r_{t+\tau+i}$ is the instant reward in state $s_{t+\tau+i}$. The goal is to determine policy π^* to maximize the value function.

B. World Model

1) Observation: Instead of directly using fully known pedestrian states, including the human number, position, and speed [5], [6], [10], [13], [16], or clumsily dealing with raw sensor data [18], [19], we create a map using only the position information of humans and the robot. Therefore, we can reduce the uncertainties of human tracking, speed estimation, and trajectory prediction; generalize our method; and maturely process high-dimensional image observations via representation learning. Fig. 2 depicts the image observation with the shape $(128, 128, 3)$. Let the size of the motion area be $L \times L$. Thus, the image resolution is $I_r = L/128$.

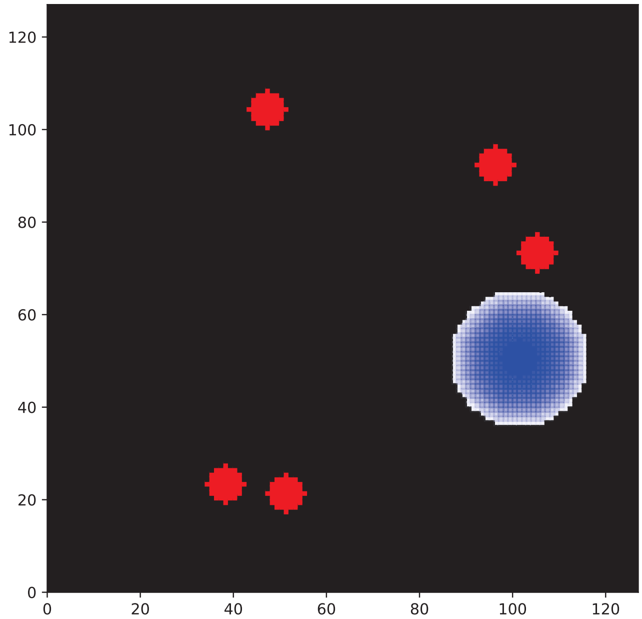

Fig. 2. Image observation with the position information of humans and the robot.

The image has three RGB channels, with the R channel representing humans and the B channel depicting the robot. We assume that the human is a circle with radius r_h, and the human circle is projected as deep red pixels with the value $p_r = 255$, as shown in Fig. 2. The pixel value p_r is zero when the pixel is not occupied by a human. Similarly, the robot is assumed to be a circle with radius r_r, which is represented by deep blue pixels with a value $p_b = 255$. By contrast, we inflate the robot circle to define an uncomfortable zone between the robot and humans. The blue pixel value p_b decreases when the pixel is far from the robot rim, and becomes zero when the pixel is outside the uncomfortable area. Note that the G channel is not used to represent the goal position in our current work because we fix the robot goal and randomize the robot's initial position, which is reasonable because the positional relation between the robot and the goal is relative.

2) Reward and Discount Factor: The reward function is defined according to the image observation. We first add the R and B channels together to yield a new 2D matrix m_a. If any value of m_a is equal to 510, a collision occurs, the reward is the minimum $r_c = -0.6$, and the discount factor γ is zero. In addition, the reward becomes $r_o = -0.1$ when the robot rim is outside the motion area, and γ remains zero. Another case of zero γ is when the goal distance d_g from the robot is less than the threshold d_r, that is, when the robot reaches the target. The reward is the maximum $r_g = 1.0$. If the robot does not collide with humans, reach the target, or stay outside the motion area, γ is $\bar{\gamma} = 0.99$, and the reward is a combination of the goal distance and the uncomfortable index. The maximum in m_a is $m_a^{\max}$ and the uncomfortable index is defined as $d_u = (m_a^{\max} - 255)/255$. In summary, the reward function and discount factor are defined as follows:

$$(r, \gamma) = \begin{cases} (-0.6, 0) & \text{if collision,} \\ (-0.1, 0) & \text{else if outside,} \\ (1.0, 0) & \text{else if reaching,} \\ (0.8 \cdot d_g/L - 0.6 \cdot d_u, 0.99) & \text{else.} \end{cases} \tag{2}$$

3) Action: We utilize a quadruped robot, which is an omnidirectional mobile robot. Accordingly, we define the action as two orthogonal velocities v_x and v_y. Additionally, v_x and v_y are continuous instead of discrete, as defined in state-of-the-art approaches [5], [6], [10], [13], [16]; therefore, we can generate smoother motions in the physical world.

4) Autoencoder: We leverage autoencoder technology [30] to reduce the dimension of the image observation and extract the motion features of humans and the robot from sequenced observations. As illustrated in Fig. 1, we first encode the image observation o_t with convolutional neural networks (CNNs) to extract the feature q_t which is an intermediate vector. Next, we project the combination of q_t and deterministic history information h_t onto the posterior latent state $\hat{s}_t$ with a multilayer preceptor (MLP). Subsequently, we concatenate h_t and $\hat{s}_t$ and decode the concatenation to restore the observation image $\hat{o}_t$ with transposed CNNs. The autoencoder is summarized as follows:

$$q_t = E_\theta(o_t), \tag{3a}$$

$$\hat{s}_t \sim p_\theta^{\hat{s}}(q_t, h_t), \tag{3b}$$

$$\hat{o}_t \sim p_\theta^{\hat{o}}(\hat{s}_t, h_t), \tag{3c}$$

where θ denotes the weights of the world model network. The loss function of autoencoder can be derived from the likelihood probability represented as below:

$$\mathcal{L}_{AE}^{(t+\tau)} \doteq \log(o_{t+\tau} | \hat{o}_{t+\tau}). \tag{4}$$

Because motion planning is executed in the latent state space, while the reward r_t and discount factor γ_t are defined in the original observation space, we need another two networks composed by MLP to predict $\hat{r}_t$ and $\hat{\gamma}_t$ from the concatenated h_t and $\hat{s}_t$:

$$\hat{r}_t \sim p_\theta^{\hat{r}}(\hat{s}_t, h_t), \tag{5a}$$

$$\hat{\gamma}_t \sim p_\theta^{\hat{\gamma}}(\hat{s}_t, h_t). \tag{5b}$$

Similar to the loss function (4), another two additional loss functions can be obtained as follows:

$$\mathcal{L}_{R}^{(t+\tau)} \doteq \log(r_{t+\tau} | \hat{r}_{t+\tau}), \tag{6a}$$

$$\mathcal{L}_{D}^{(t+\tau)} \doteq \log(\gamma_{t+\tau} | \hat{\gamma}_{t+\tau}). \tag{6b}$$

5) State Transition Model: It is prohibitively challenging to construct a state transition model in the original observation space with high dimensions; therefore, we model the motions of humans and the robot in the latent state space. We utilize the categorical latent variables to represent the latent state with 32 classes multiplied by 32 variables (shown in Fig. 1), based on the fact that categorical distributions can naturally capture multi-modal uncertainty of stochastic state transition [31]. Given the posterior latent state $\hat{s}_t$, action a_t, and history motion information h_t, the next latent state $\tilde{s}_{t+1}$ and the next hidden history information h_{t+1} can be predicted using a gated recurrent unit (GRU) neural network as shown in Fig. 1:

$$h_{t+1} = f_\theta(\hat{s}_t, a_t, h_t), \tag{7a}$$

$$\tilde{s}_{t+1} \sim p_\theta^{\tilde{s}}(h_{t+1}). \tag{7b}$$

where f_θ which is composed by a GRU and $p_\theta^{\tilde{s}}$ constructed by a MLP correspond to the dynamics shown in Fig. 1. The prior distribution $\tilde{s}_{t+\tau}$ is required to be similar to the posterior distribution $\hat{s}_{t+\tau}$ derived from the autoencoder model; thus, the fourth loss function can be defined as the Kullback–Leibler (KL) divergence:

$$\mathcal{L}_{KL}^{(t+\tau)} \doteq - \beta \text{KL}(\hat{s}_{t+\tau} || \tilde{s}_{t+\tau}), \tag{8}$$

where β is a constant factor weighing the KL divergence loss.

6) Overall Loss Function: Given an episode obtained from the interaction between humans and the robot, we select a sequence starting from time step t and ending at $t + K$, where K is a constant. We fill zero to elongate the episode when its length is less than $K + 1$ because of early collision, outside motion, or target reaching. The overall loss function along the sequence is represented as follows:

$$\mathcal{L}_{ALL} \doteq \mathbb{E}_{p_\theta} \left[\sum_{\tau=1}^{K} \left[\mathcal{L}_{AE}^{(t+\tau)} + \mathcal{L}_{R}^{(t+\tau)} + \mathcal{L}_{D}^{(t+\tau)} + \mathcal{L}_{KL}^{(t+\tau)} \right] \right]. \tag{9}$$

All world model networks, including (3a), (3b), (3c), (5a), (5b), (7a) and (7b), are jointly updated using this single overall loss function.

C. Motion Planner

The state transition model in compact latent space enables trajectory prediction in the long horizon without high-dimensional image observation, which results in a low memory footprint and speedy predictions of thousands of imagined trajectories in parallel [22]. As shown in Fig. 1, starting from the latent state $\hat{s}_{t+\tau}$ and history information $h_{t+\tau}$, a considerable number of episodes with H horizon can be swiftly generated. Consequently, we can efficiently leverage imagined episodes to optimize the navigation policy.

For the imagination process, we create an actor network and a critic network using MLP to map the current latent state and historic motion information into the action and value function, respectively:

$$\bar{a}_{t+\tau} \sim p_\phi^{\bar{a}}(\bar{s}_{t+\tau}, \bar{h}_{t+\tau}), \tag{10a}$$

$$\bar{v}_{t+\tau} \sim p_\psi^{\bar{v}}(\bar{s}_{t+\tau}, \bar{h}_{t+\tau}), \tag{10b}$$

where ϕ and ψ denote the weights of the actor and critic networks, respectively. Additionally, the reward $\bar{r}_{t+\tau}$ and discount factor $\bar{\gamma}_{t+\tau}$ are predicted using (5a) and (5b), respectively. With the imagined episode having a long horizon, we can evaluate the value function with a multi-step RL framework because it yields a better unbiased estimation than the one-step RL algorithm [32]:

$$\begin{aligned} v_i(s_\kappa) &\doteq \mathbb{E}_{(p_\theta^{\hat{r}}, p_\theta^{\hat{\gamma}}, p_\theta^{\tilde{s}})}\left[\sum_{n=\kappa}^{h-1}(\bar{\gamma}_n^{n-\kappa}\bar{r}_n) + \bar{\gamma}_h^{h-\kappa}\bar{v}_\psi(s_h)\right], \\ &i = 1, 2, \ldots, H, \\ &h = \min(\kappa + i, t + \tau + H), \\ v_\lambda(s_\kappa) &\doteq (1-\lambda)\sum_{n=1}^{H-1}(\lambda^{n-1}v_n(s_\kappa)) + \lambda^{H-1}v_H(s_\kappa), \end{aligned} \tag{11}$$

where λ is another discount factor that weighs the value function, and κ ranges from $t+\tau$ to $t+\tau+H$. We have two objectives: to determine a policy to maximize the value function, and to minimize the error between the estimated value function and the predicted value from the critic network. Therefore, the weights of the critic and actor networks can be updated as follows:

$$\min_\psi \mathbb{E}_{(p_\theta, p_\phi)}\left(\sum_{\kappa=t+\tau}^{t+\tau+H}\frac{1}{2}\|\bar{v}_\kappa^{mean} - v_\lambda(s_\kappa)\|\right), \tag{12a}$$

$$\max_\phi \mathbb{E}_{(p_\theta, p_\phi)}\left(\sum_{\kappa=t+\tau}^{t+\tau+H} v_\lambda(s_\kappa)\right), \tag{12b}$$

where $\bar{v}_n^{mean}$ is the mean of the distribution $\bar{v}_\kappa$.

D. Algorithm Summary

In the simulation, we assume that the human motion is generated by ORCA [8]. Additionally, the robot is assumed to be invisible to humans; otherwise, it will be difficult to differentiate whether our motion planner is effective or whether humans avoid the robot. The simulation is *reset* when collision, outside motion, target reaching, or timeout occurs. When the episode length is greater than $t_{\max}$, it is terminated as a timeout. We use the *step* function to update the positions of humans and the robot and obtain the instant reward and discount factor. Simulation interaction samples are collected to update the world model. We subsequently utilize the world model to imagine episodes in the latent space, which are used to update the motion planner. Next, a motion planner is used to generate new simulation episodes. We alternately update the world model and motion planner until a stable navigation policy is acquired.

E. Complete Navigation System

As stated in existing studies, wheeled mobile robots can localize themselves with the wheel odometer [5], [6], [10], [13], [16], which however drifts heavily with the increase in motion time. External motion capture systems are commonly used to precisely obtain the robot position and orientation [15], [17]. Nevertheless, such systems are expensive and impractical in a real human society. A navigation system without any accurate and internal localization module is incomplete and cannot be applied to society. Additionally, these studies have not focused on publicly accessible real-world implementations, such as human detection, speed estimation, and trajectory prediction. Conversely, we leverage the LiDAR odometry and mapping (LOAM) SLAM algorithm [33] to accurately localize the robot. In addition, we extract humans using a clustering approach [34]. Because we can obtain the sequenced position information of humans and the robot, we can extract the latent motion feature using our algorithm without individually estimating the human speed and future trajectory. Because our algorithm directly outputs two orthogonal velocities v_x and v_y, we need to further match v_x and v_y to the speed command of the quadruped robot used for practical implementations. We found that the actual speed generated by the official controller was significantly different from that of the desired command. Therefore, we use the system identification technique to calibrate the desired speed command to be consistent with the actual speed [35]. We made our system open-source for easier deployment in the autonomous navigation community.

IV. Simulation and Real Implementations

We used three simulation scenarios: Simulation one is for a comprehensive comparison, Simulation two is designed to verify the generalizability of our method, and Simulation three is constructed for sim-to-real transfer. Subsequently, we directly deployed the policy learned in Simulation three into various real scenarios without any retraining or fine-tuning.

A. Simulation

1) Training for Comparison: The side length of the motion area is $L = 10$ m. Similar to the original settings of RGL [16], we assume a human number fixed at 5, human radius $r_h = 0.3$ m, and each human is randomly initialized around a circle with a radius of 4 m. The initial position is $(x_0^{(i)}, y_0^{(i)})$, and the goal is $(-x_0^{(i)}, -y_0^{(i)})$. The i-th human moved back and forth from these two positions with a preference speed of 1 m/s. Additionally, reciprocal motion among humans is generated by ORCA [8] with the robot being invisible. The robot radius was $r_r = 0.3$ m and the maximum values of v_x and v_y were both 1 m/s. For a fair comparison, we assumed that the robot moves from (−4.0, 0.0) m to (4.0, 0.0) m. We reproduced seven popular baselines: one is a non-learning method namely ORCA [8], whereas the other six are learning-based approaches, called CADRL [13], LSTM _ RL [14], SARL [10], RGL [16], EGO [18], and LSTM _ EGO [21] respectively.

The ORCA algorithm assumes that the agent's states, including shape, size, position, and speed, are fully known. Based on this information, it generates an optimal collision-free action in one step. However, this one-step planning approach may result in short-sighted, unsafe, and unnatural behaviors. In contrast, DRL-based crowd navigation algorithms, which utilize a value function that can represent accumulated return over a long horizon, have become increasingly popular due to their ability to address these issues. For example, CADRL was a pioneering study in this area, but its value function only considered the pair of the robot and one human, making it unable to represent

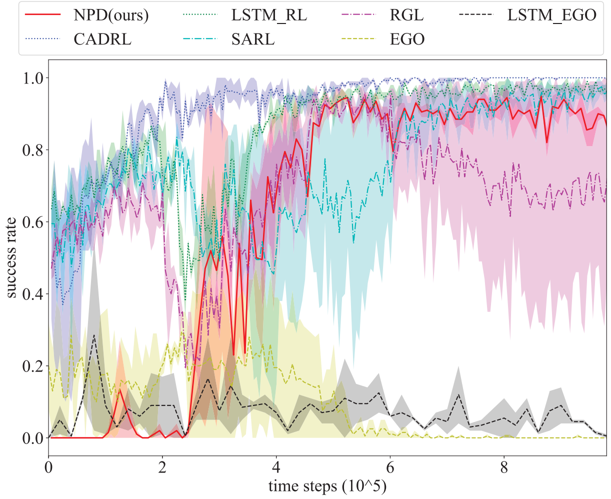

Fig. 3. Learning ablations. At certain time step, we evaluate the policy 100 times and calculate the corresponding success rate of collision-free and target-reaching navigation. CADRL and LSTM _ RL have a stable learning process while the training of another two baselines SARL and RGL vibrates intensely. EGO and LSTM _ EGO can not reach a high success rate of crowd navigation and their learning is notably unstable.

relational interactions among humans. As a result, LSTM _ RL was developed to pair the robot with all humans, but it still only captures partial interactions because it sorts the pairs by distance before feeding them into the LSTM networks. To more comprehensively represent social interactions among humans, SARL utilizes a self-attention mechanism to capture interactions within pedestrians, while RGL embeds a graph convolutional network to reason about relations between agents and compute interactions between them. However, these methods all require fully known human states. In contrast, EGO and LSTM _ EGO can handle both static and dynamic obstacles of different shapes, sizes, and numbers, as they directly map raw sensor data to navigation actions. EGO uses continuous LiDAR scans, while LSTM _ EGO uses one frame of LiDAR scan and embeds LSTM to deal with sequential scans. However, a downside of these methods is the difficulty of efficiently learning a feasible navigation policy. While the other four have open-source solutions, publicly accessible resources for EGO and LSTM _ EGO are rare. Therefore, we specially created a simulator that could generate 2D LiDAR scans and constructed neural networks for policy learning. We named our method navigation among pedestrians with a Dreamer-based motion planner (NPD). The learning processes of the six methods are illustrated in Fig. 3, and the final evaluation is shown in Table I. Please note that Fig. 3 displays the success rate of 100 evaluation episodes. Each episode may result in success, collision, overtime, or outside motion. Our analysis revealed that when the success rate was at or above 90%, the number of collision cases decreased to less than 5, sometimes even to 1, whereas most of the cases were overtime. Notably, in overtime cases, the robot was very close to the target. We hypothesized that the image resolution may have contributed to this phenomenon. As the robot was approaching the target, we reclassified the overtime cases as success in the final 500 tests.

TABLE I
FINAL EVALUATION

Method	SR	NT (s)	AT (s)
*ORCA	0.47	10.83	**5e-5**
*CADRL	0.80	12.40	0.035
*LSTM_RL	0.97	10.95	0.067
*SARL	0.98	**10.75**	0.060
*RGL	0.97	11.22	0.067
EGO	0.54	12.67	1.5e-3
LSTM_EGO	0.52	15.08	7.5e-4
NPD(ours)	**0.99**	11.15	3.3e-3

*These methods require fully known human information.
SR: success rate; NT: navigation time; AT: action time

500 random tests are executed with the best neural networks saved during the training.

Because baselines CADRL, LSTM _ RL, SARL, and RGL initialized the neural networks with imitation learning and fill the experience pool with a large number of positive samples ahead of the training, they could swiftly learn a feasible navigation policy. Although EGO's initialization was same, its learning was notably unstable and it could only reach a success rate of 0.54. The initial data settings of LSTM _ EGO were same as ours, however, its success rate was 0.52, significantly lower than our method's. In addition, LSTM _ EGO required the longest navigation time, averaging 15.08 s. We found that the success rates of all the baselines which require prior initialization became zero at all times when we removed imitation learning and the massive positive dataset. Conversely, our method could completely learn from zero experience and yielded a stable convergence and high success rate, which indicates that our method does improve learning efficiency. As shown in Table I, our approach quantitatively either outperformed or behaved similar as the other six learning baselines with respect to both the success rate and average navigation time. We found that the failure of our method initially occurred when two of the humans closely surrounded the robot. Additionally, the collision occurred within five time steps of 1 s. We believe that this short sequence results in the most of the remaining 1% failures. Although ORCA requires a short navigation time, it produced the lowest success rate, whereas our method was able to adequately balance the navigation time and success rate. Because our final goal is to deploy the navigation policy on real robot platforms, real-time performance should be another evaluation factor. The time used to generate an action when given observations is referred to action time. ORCA's action time was only tens of microseconds because ORCA is a non-learning approach. However, CADRL, LSTM _ RL, SARL, and RGL required tens of milliseconds to derive an action because they only have a value network and need to inquire each action choice to obtain the best one. On the contrary, EGO, LSTM _ EGO, and NPD(ours) have both an actor network and a critic network, therefore, they are able to quickly access an optimal action from the actor network. Because the planning frequency in the real world is 5 Hz, our method's action time (3.3e-3) is acceptable for real implementations. The aforementioned test assumes that human

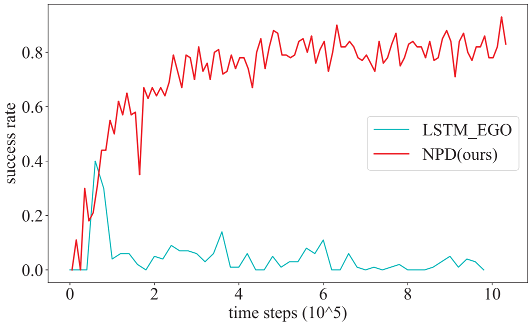

Fig. 4. Training in complex environments.

motions follow ORCA rules. We also deviated the human action originally generated using ORCA by adding uniform noises with the bound of 0.2 m/s. The success rate of our method could still reach 94%, which indicates the generalizability of the trained policy with respect to human motion modals.

2) Training for Generalizability Verification: We trained our model in a more challenging navigation task to further verify the generalizability of our approach. Similar to the scenario shown in Fig. 2, the motion area is 10 × 10 m. Differently, the obstacle number in the revised environment is up to 7, which makes the training scenario denser and more complicated. Moreover, the number of moving humans changes from 1 to 4. Additionally, we add rectangular static obstacles whose number is variable from 1 to 3 and side length ranges from 0.3 m to 0.4 m. Baselines CADRL, LSTM _ RL, SARL, and RGL assume that the obstacles are circles and their number is fixed during the whole training, whereas baselines EGO and LSTM _ EGO are independent of obstacle number and shape. We choose LSTM _ EGO as the ablation simply because it is a more recent study. The training process is depicted in Fig. 4. Although the environment becomes complex, our method can still reach a 93% success rate, significantly outperforming LSTM _ EGO whose success rate is below 40% at all times.

3) Training for Policy Transfer: To enable sim-to-real transfer, we leveraged the domain randomization technique to improve the generalizability of the simulated navigation policy. Because the maximum sideward speed of the real quadruped robot was 0.27 m/s, we first constrain the maximum v_y as 0.27 m/s and the maximum v_x 0.3 m/s. Note that we did not enlarge the forward speed v_x for slow and stable motion in our small real scenarios with a size 3×3 m. Different from the simulation configurations considered for comparison, we narrowed the motion area to $L = 6$ m, and the human number changed from 1 to 3. Additionally, we randomized the initial robot position while keeping the goal fixed at $(1.0, 0.0)$. Moreover, the initial human position was randomized over the entire motion area, while the human goal was distributed around the margin of the motion area. The preference speed of humans ranges from 0.15 m/s to 0.3 m/s. For RL-based baselines, obtaining a feasible navigation policy is challenging if the environment significantly changes. Moreover, certain baselines, such as CADRL and SARL, do not allow variable human numbers during training. Although we introduced a large number of stochastics, our algorithm could produce stable policy optimization while LSTM _ EGO failed in obtaining a feasible navigation policy, as shown in Fig. 5. The trained policy achieved a 95% success rate in the final evaluation with 500 random settings.

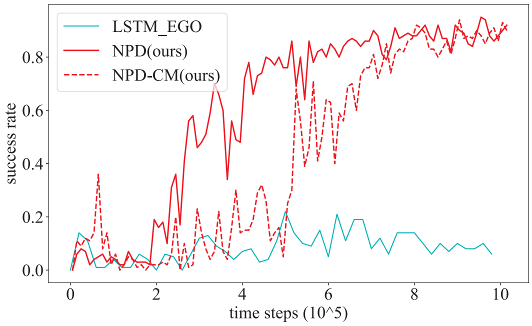

Fig. 5. Learning process with stochastic configurations in simulation. Different from the training for comparison, this training is executed in stochastic environments with variable human numbers and distributions to learn a more generalized navigation policy. NPD considers the robot's collision margin as a circular shape, whereas NPD-CM uses a more accurate collision margin that is a circumscribed rectangle around the physical robot. RGL's learning is omitted because its success rate is zero at all times.

In addition to comparing with the learning-based baselines LSTM _ EGO and RGL, we conducted ORCA as another ablation. We found that ORCA could significantly improve navigation success rate from 0.47 to 0.98 when human number was decreased from 5 to 3. Although ORCA (0.98) outperformed our method (0.95) in simulation, we found that the robot was inclined to move outside of the specific area in real scenarios. The possible reason may be inaccurate human tracking and speed estimation. Conversely, our method only needs human positions, without considering the errors of human tracking and speed estimation, therefore, our approach is able to proficiently deal with various real-world scenarios.

To align the robot collision margin with the actual robot platform, we replaced the inflated circular margin with a rectangular one. This modification enabled us to match the collision margin of the simulated robot with that of the real robot platform. The collision margin has a length and width of 0.5 m and 0.3 m, respectively, identical to the collision margin of our real robot platform. The corresponding learning is illustrated in Fig. 5, which indicates that our method can deal with different robot collision margins.

B. Real Implementations

We deployed the policy learned from the simulation on a quadruped robotic platform equipped with a Velodyne VLP-16 LiDAR. The tests are shown in the attached videos. Although the human number changed from 1 to 3 and human motions are diversified, our simulated navigation policy could be directly transferred into real scenarios without any retraining or fine-tuning, which shows the potential of our method to model complex reciprocal human relations and navigate robots among pedestrians in the real world.

V. Conclusion

This letter presented an autonomous navigation system with a Dreamer-based motion planner. We let go of the assumption of fully known human states and only utilized the human position information. The human positions and robot location were projected onto a image. From the sequenced image observations, we extracted reciprocal relationships among pedestrians through representation learning. In addition, we created a state transition model using the extracted latent information to imagine episodes for reinforcement learning. Sufficient simulation ablations demonstrated that our method could learn from zero experience with high efficiency and outperformed state-of-the-art algorithms. In addition, we leveraged the techniques of system identification, domain randomization, clustering, and LiDAR SLAM to enable sim-to-real transfer. Adequate real implementations illustrated the potential of our method to model complex reciprocal human relations and navigate the robot among pedestrians in the physical world. Our future study will focus on accurate human detection, precise robot localization, and universal navigation policy.

References

[1] S. Kim, J. Kim, F. B. Baiden, M. Giroux, and Y. Choi, “Preference for robot service or human service in hotels? Impacts of the COVID-19 pandemic,” *Int. J. Hospitality Manage.*, vol. 93, Feb. 2021, Art. no. 102795.

[2] R. Moller, A. Furnari, S. Battiato, A. Harma, and G. M. Farinella, “A survey on human-aware robot navigation,” *Robot. Auton. Syst.*, vol. 145, 2021, Art. no. 103837.

[3] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, “A survey of autonomous driving: Common practices and emerging technologies,” *IEEE Access*, vol. 8, pp. 58443–58469, 2020.

[4] K. D. Katyal, G. D. Hager, and C. M. Huang, “Intent-aware pedestrian prediction for adaptive crowd navigation,” in *Proc. Int. Conf. Robot. Automat.*, 2020, pp. 3277–3283.

[5] 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, Apr. 2020.

[6] S. S. Samsani and M. S. Muhammad, “Socially compliant robot navigation in crowded environment by human behavior resemblance using deep reinforcement learning,” *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5223–5230, Jul. 2021.

[7] Q. Li, W. Lin, Z. Liu, and A. Prorok, “Message-aware graph attention networks for large-scale multi-robot path planning,” *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5533–5540, Jul. 2021.

[8] J. Berg, S. J. Guy, M. Lin, and D. Manocha, “Reciprocal *n*-body collision avoidance,” in *Robotics Research*, vol. 70. Berlin, Germany: Springer, 2011, pp. 3–19.

[9] 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.

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

[11] S. Paris, J. Pettre, and S. Donikian, “Pedestrian reactive navigation for crowd simulation: A predictive approach,” *Comput. Graph. Forum*, vol. 26, no. 3, pp. 665–674, 2007.

[12] M. Kuderer, H. Kretzschmar, C. Sprunk, and W. Burgard, “Feature-based prediction of trajectories for socially compliant navigation,” in *Proc. Robot. Sci. Syst.*, 2012.

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

[14] 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.

[15] M. Everett, Y. F. Chen, and J. P. How, “Collision avoidance in pedestrian-rich environments with deep reinforcement learning,” *IEEE Access*, vol. 9, pp. 10357–10377, 2021.

[16] C. Chen, S. Hu, P. Nikdel, G. Mori, and M. Savva, “Relational graph learning for crowd navigation,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 10007–10013.

[17] W. Zhu and M. Hayashibe, “A hierarchical deep reinforcement learning framework with high efficiency and generalization for fast and safe navigation,” *IEEE Trans. Ind. Electron.*, vol. 70, no. 5, pp. 4962–4971, May 2023.

[18] J. Jin, N. M. Nguyen, N. Sakib, D. Graves, H. Yao, and M. Jagersand, “Mapless navigation among dynamics with social-safety-awareness: A reinforcement learning approach from 2D laser scans,” in *Proc. Int. Conf. Robot. Automat.*, 2020, pp. 6979–6985.

[19] 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.*, vol. 39, no. 7, pp. 856–892, 2020.

[20] X. Huang, H. Deng, W. Zhang, R. Song, and Y. Li, “Towards multi-modal perception-based navigation: A deep reinforcement learning method,” *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4986–4993, Jul. 2021.

[21] N. Yokoyama, Q. Luo, D. Batra, and S. Ha, “Benchmarking augmentation methods for learning robust navigation agents: The winning entry of the 2021 iGibson challenge,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 1748–1755.

[22] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, “Dream to control: Learning behaviors by latent imagination,” in *Proc. Int. Conf. Learn. Represent.*, 2020.

[23] D. Omeiza, H. Webb, M. Jirotka, and L. Kunze, “Explanations in autonomous driving: A survey,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 10142–10162, Aug. 2022.

[24] C. Arpino, C. Liu, P. Goebel, R. Martin, and S. Savarese, “Robot navigation in constrained pedestrian environments using reinforcement learning,” in *Proc. Int. Conf. Robot. Automat.*, 2021, pp. 1140–1146.

[25] U. Patel, N. K. S. Kumar, A. J. Sathyamoorthy, and D. Manocha, “DWA-RL: Dynamically feasible deep reinforcement learning policy for robot navigation among mobile obstacles,” in *Proc. Int. Conf. Robot. Automat.*, 2021, pp. 6057–6063.

[26] 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, Jul. 2020.

[27] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, “Mastering Atari with discrete world models,” in *Proc. Int. Conf. Learn. Represent.*, 2021.

[28] M. Okada and T. Taniguchi, “Dreaming: Model-based reinforcement learning by latent imagination without reconstruction,” in *Proc. Int. Conf. Robot. Automat.*, 2021, pp. 4209–4215.

[29] F. Deng, I. Jang, and S. Ahn, “DreamerPro: Reconstruction-free model-based reinforcement learning with prototypical representations,” in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 4956–4975.

[30] Y. Wang, H. Yao, and S. Zhao, “Auto-encoder based dimensionality reduction,” *Neurocomputing*, vol. 184, pp. 232–242, 2016.

[31] M. Okada and T. Taniguchi, “DreamingV2: Reinforcement learning with discrete world models without reconstruction,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 985–991.

[32] R. S. Sutton and G. B. Andrew, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.

[33] J. Zhang and S. Singh, “LOAM: Lidar odometry and mapping in real-time,” in *Proc. Robot. Sci. Syst.*, 2014, pp. 1–9.

[34] D. Matti, H. K. Ekenel, and J. P. Thiran, “Combining LiDAR space clustering and convolutional neural networks for pedestrian detection,” in *Proc. IEEE Int. Conf. Adv. Video Signal Based Surveill.*, 2017, pp. 1–6.

[35] W. Zhu, X. Guo, D. Owaki, K. Kutsuzawa, and M. Hayashibe, “A survey of sim-to-real transfer techniques applied to reinforcement learning for bioinspired robots,” *IEEE Trans. Neural Netw. Learn. Syst.*, early access, doi: 10.1109/TNNLS.2021.3112718.