Risk-Sensitive Mobile Robot Navigation in Crowded Environment via Offline Reinforcement Learning

Jiaxu Wu[1], Yusheng Wang[1], Hajime Asama[1], Qi An[2], Atsushi Yamashita[2]

***Abstract*— Mobile robot navigation in a human-populated environment has been of great interest to the research community in recent years, referred to as crowd navigation. Currently, offline reinforcement learning (RL)-based method has been introduced to this domain, for its ability to alleviate the sim2real gap brought by online RL which relies on simulators to execute training, and its scalability to use the same dataset to train for differently customized rewards. However, the performance of the navigation policy suffered from the distributional shift between the training data and the input during deployment, since when it gets an input out of the training data distribution, the learned policy has the risk of choosing an erroneous action that leads to catastrophic failure such as colliding with a human. To realize risk sensitivity and improve the safety of the offline RL agent during deployment, this work proposes a multipolicy control framework that combines offline RL navigation policy with a risk detector and a force-based risk-avoiding policy. In particular, a Lyapunov density model is learned using the latent feature of the offline RL policy and works as a risk detector to switch the control to the risk-avoiding policy when the robot has a tendency to go out of the area supported by the training data. Experimental results showed that the proposed method was able to learn navigation in a crowded scene from the offline trajectory dataset and the risk detector substantially reduces the collision rate of the vanilla offline RL agent while maintaining the navigation efficiency outperforming the state-of-the-art methods.**

I. Introduction

Robot navigation in human-populated environments has gained wide interest in recent years. Applications using mobile robots such as delivery, unmanned patrolling, and floor cleaning are growing rapidly. For applications in spaces shared with other pedestrians, it is important to realize safe and efficient robot navigation in crowded environments which is referred to as crowd navigation [1].

Recently, Deep reinforcement learning (RL)-based methods [1], [2] have made great progress compared to rule-based methods [3], [4] of the early time, since they can generalize to various situations in a crowd through trial and error. However, realistic simulation is necessary for such a routine. Due to that the human reaction to the robot movement is still an open problem [5], there is always a gap between the simulation and the real-world crowd, which brings a risk of failure to the learned navigation policy during deployment.

[1] J. Wu, Y. Wang, H. Asama: Department of Precision Engineering, Graduate School of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan, wujiaxu@robot.t.u-tokyo.ac.jp

[2] A. Yamashita: Department of Human and Engineered Environmental Studies, Graduate School of Frontier Sciences, The University of Tokyo, 5-1-5 Kashiwanoha, Kashiwa, Chiba 277–8563, Japan yamashita@robot.t.u-tokyo.ac.jp

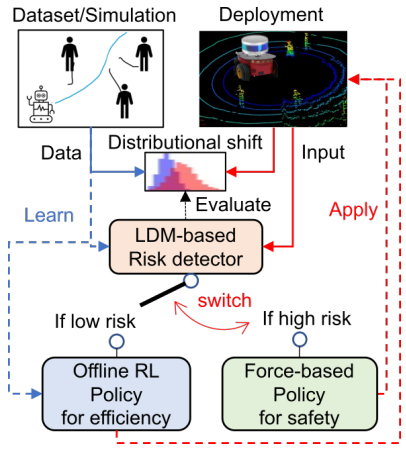


Fig. 1. The risk-sensitive mobile robot navigation is realized as a multipolicy control framework, in which an LDM-based risk detector that considered the distributional shift between the training data and the runtime input controls the switch between an efficiency-prior deep reinforcement learning offline reinforcement learning (RL)-based policy and a safety-prior force-based policy and prevents the mobile robot going into states that is without data support.

The gap is indeed the distributional shift of the state-action pairs seen during the training and during the deployment.

To alleviate the gap, a straightforward idea is to train the robot directly in the real world. However, moving robots in public space is costly considering the low sample efficiency of the RL-based methods, and sometimes dangerous [6]. On the contrary, learning with an offline dataset involving human-robot interactions may be desirable [7]. Offline RL-based methods could train a navigation policy with only offline datasets [8]. After offline training, the robot could directly exploit the current policy or start further adaptation to the environment smoothly. However, the distributional shift still exists during the deployment due to the limited diversity of the dataset [7]. The robot has a risk to move to a state that is out of the training data distribution, in which the navigation policy may make erroneous actions and cause catastrophic failures such as collisions with other people [9]. To avoid the risk during deployment, an additional module along with the offline RL policy is required that can sense the distributional shift and foresee those failures.

Model predictive control (MPC)-based methods can handle the risk more easily, since they explicitly formularize the state constraint according to the predicted trajectory of the pedestrian [10]. However, the performance of MPCs is degraded when there are high compounding errors in the pedestrian model, especially when the robot faces a novel

state. Moreover, trajectory sampling from deep networks introduces a high computational burden, compared to RL-based methods that directly map from state to action at the decision time [11].

This work tackles mobile robot navigation that is sensitive to the risk of erroneous action caused by the distributional shift by taking advantage of both RL-based methods and explicit safety constraints. A multi-policy control framework based on offline RL is proposed (Fig. 1), in which conservative Q learning (CQL) is introduced to learn crowd-aware navigation using offline datasets [8]. The sense of risk is achieved by a Lyapunov density model (LDM)-based risk detector, which models the probability of moving out of the data support. When the robot has the potential to encounter a state out of the distribution of the training dataset, the risk detector switches the control to a cautious risk-avoiding movement that was known to be safe, such as an emergency stop. Compared to the previous multipolicy methods which relied on either online trial and error or used ad-hoc switching without considering the risk [12], [13], this is the first work in crowd navigation that is able to train with an offline dataset and tackles risk caused by the distributional shift.

The rest of the paper is organized as follows. Section II introduces some related works. Sction III describes the objective and detailed problem setting. Section IV describes the proposed method. Section V presents the experimental tests and the related results and discussion. Finally, the conclusions and future work are drawn in Section VI.

II. Related works

A. Learning-based robot navigation methods

Many online RL-based methods have been proposed. They mainly focused on representational learning, which targets a generalizable representation of the robot-crowd joint state [1], [2]. Some works tackled the design of rewards to inform the RL agent with the social norm [14], [15]. Although they greatly improve the generality of navigation in the simulator, they deployed the model to the real world without considering distributional shift and the risk of catastrophic failure. In contrast, Liu *et al* tackled the problem ill-distributed training data, and introduced a data augmentation that can alleviate the distributional shift [16]. They conducted a test on offline RL and, to the best of the author's knowledge, this is the only previous work training crowd navigation under an offline setting. However, they have not considered the risk of failure. Shah *et al.* proposed an offline RL-based visual navigation and demonstrated the robot was able to learn navigation to distant goals using only offline training [17]. However, they only considered static scenes.

B. Risk sensitive motion planning

MPC-based methods could explicitly consider the risk constraint. Many risk-sensitive path planning researches adopt Conditional Value-at-Risk (CVaR) to measure the risk of collision brought by the moving obstacle or the unstructured environment [18]. However, CVaR is difficult to calculate in real-time. The previous study [10] developed a real-time risk-sensitive method based on multimodal human trajectory prediction. However, these methods relied on learned deep neural networks (DNN)-based pedestrian models [19] which uncertainty is difficult to analyze. For data-driven methods, when the model is highly uncertain of a novel input, there is a risk that the model takes action that leads to catastrophic failure. Hoque *et al.* trained an ensemble of policies to measure novelty by the standard deviation of the current output of the ensemble [20]. However, sequential decision problems have non-i.i.d nature in which a seemingly safe decision in the current may have a risk to lead the system to fail in the future. Kang *et al.* proposed a farsighted risk measurement using a generalization of control Lyapunov functions and density models to ensure safety in the future [9]. This paper leverages the power of the Lyapunov density model to realize risk-sensitive navigation in crowded scenarios.

III. Objective

This work aims to address risk-sensitive robot navigation in a 2D scenario with a crowd of N humans, where a single robot navigates through the crowd to its goal as safely and efficiently as possible. In addition, the proposed system aims to learn a robot navigation policy from a fixed offline dataset, which is able to sense the risk of going out of data support and avoid making erroneous actions caused by distributional shift.

IV. Method

A. Risk-sensitive multi-policy control framework

This section will provide an overview of the proposed system and then describe each of its modules. The system consists of a deep reinforcement learning (DRL) agent with a state encoder, a Lyapunov density model (LDM)-based risk detector, and a flow-based density model that learns the distribution of the training data (Fig. 2). Both the DRL agent, the state encoder, and the LDM are learned using the given offline dataset using conservative Q-learning (CQL). During training, the DRL agent learns the navigation policy. Then, the encoded latent feature of the robot joint state is fed to the flow-based network to learn the probability density of the training data distribution. The resulting log-probability of each latent feature-action pair will then be used to train the LDM.

B. Reinforcement learning framework

The navigation problem is formalized by the Markov decision process (MDP) following previous works [1], [2]. The state space S is composed of an observable part O and an unobservable part H for both the robot and human. The observable state includes position $\mathbf{p} = [p_x, p_y]$, velocity $\mathbf{v} = [v_x, v_y]$ and the agent's size r. The unobservable part includes the agent's goal position $\mathbf{p}_g$ and the preferred speed v_{pref} and direction θ. The robot input state is defined as a joint state $s_t^{jn} = [s_t^r, \mathbf{o}_t] \in S$ concatenated from the robot state s_t^r and the observable states of humans $\mathbf{o}_t = [o_t^i]_{i=1}^N$, where

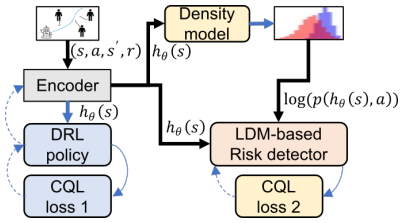


Fig. 2. Flow chat of learning navigation policy and risk detector. The black arrow denotes data and labels feeding. The blue arrow denotes forward processing and the blue dashed arrow denotes the bask propagation. The state encoder and the navigation DRL policy is learned first. Secondly, a Flow-based network is trained to model the distribution of the hidden state of the DRL policy. Finally, the Lyapunov density model is trained using the minus log probability estimated for each hidden state as a label. Both the navigation policy and the LDM is trained in CQL fashion.

t denotes the time step, i denotes the human index, and N denotes the total number of humans in the scenario. The agent's actions a are defined as its current velocity and are assumed to be instantaneous. The optimal policy $\pi^*: s_t^{jn} \mapsto a_t^r$ maps the joint state S_t^{jn} to the robot action a_t^r at time t, which is to maximize the expected return $V^*(s_t^{jn})$:

$$\begin{aligned}\pi^*(s_t^{jn}) &= \arg\max_{a_t^r} R(s_t^{jn}, a_t^r) \\ &+ \gamma^{\Delta t v_{pref}} \int_{s^{jn}} P(s_{t+\Delta t}^{jn} | a_t^r, s_t^{jn}) V(s_{t+\Delta t}^{jn}) ds^{jn}, \quad (1)\\ V^*(s_t^{jn}) &= \Sigma_{k=t}^{T} \gamma^{k v_{pref}} R(s_k^{jn}, \pi^*(s_k^{jn})), \quad (2)\end{aligned}$$

where $P(s_{t+\Delta t}^{jn} | a_t^r, s_t^{jn})$ denotes the unknown transition of the environment and γ is the discount factor. The reward function $R(s_t^{jn}, a_t^r)$ is defined following the previous work [1].

$$R(s_t^{jn}, a_t^r) = \begin{cases} 1 & \text{(reached goal)} \\ -0.25 & \text{(collided)} \\ \frac{1}{2}(d_{min} - 0.2)\Delta t & (d_{min} < 0.2) \\ 0 & \text{(else)} \end{cases}, \quad (3)$$

where d_{min} represents the minimum distance between the robot and its neighboring humans during the state transition and 0.2 is the threshold to judge whether the robot entered the discomfort zone of its neighboring humans.

C. Learning navigation via offline reinforcement learning

Though there are multiple alternatives for offline RL [21], [22], the proposed system adopts CQL [8] because it directly addresses the distributional shift and obtains state-of-the-art results in public benchmark environments. In this paper, a discrete control for mobile robot navigation using CQL is developed. The action space with holonomic kinematics is quantized along the radial direction and the circumferential direction following previous work [1], [2], which results in 81 discrete actions.

$$\begin{aligned}[v_x, v_y] \in \{ [&\frac{e^{(\frac{i+1}{5})} - 1}{e-1} \cos(j\pi/8), \frac{e^{(\frac{i+1}{5})} - 1}{e-1} \sin(j\pi/8)] \\ &i \in \{0,1,2,3,4\}; j \in \{0,1,...,15\}; \end{aligned} \quad (4)$$

A deep neural network with 81 outputs is used to approximate the Q-value of each action and is trained via CQL. The network is referred to as Deep Q Network (DQN). The DQN contains an encoder that embeds the joint state in the latent feature.

$$h_t = f_\theta(s_t^{jn}), \quad (5)$$

where θ denotes the encoder parameters. Two types of encoder are adopted from previous state-of-the-art research: Attention-based encoder [1] that transformed the observed joint state into ego-centric robot coordinates with the x-axis directly to the target and embedded human states with an attention layer; Graph Convolutional Network-based encoder [2]. Then, a multi-layer perceptron (MLP) maps the latent feature to the Q-values of all the actions ($Q(s_t^{jn}, a_t^r) = f_\phi(h_t)$), and the optimal policy π^* is to choose the action with the maximum Q-value. Based on the Q-value estimation, the network is trained using CQL loss.

$$\begin{aligned}loss_1 = \quad &\alpha E_{s^{jn} \sim D}[\log \Sigma_i^{81} \exp(Q(s^{jn}, a_i^r)) \\ &- E_{a^r \sim \pi}[Q(s^{jn}, a^r)]] + loss_{td}, \end{aligned} \quad (6)$$

where the first term weighted by α is used to address the overestimation problem on out-of-distribution action, which enables RL training in offline. The second term is normal temporal differential (TD)-error.

D. Lyapunov density model-based risk detector

The risk detector learns LDM model $G(s,a)$ from the offline data to estimate the probability that a navigation trajectory (sequence of state-action pairs) will remain within the distribution of the training dataset when following the navigation policy. The idea of LDM is to set

$$G(s_t, a_t) = \max\{-\log P(s_t, a_t), \min_{a_{t+1}}\{G(s_{t+1}, a_{t+1})\}\}, \quad (7)$$

so that if the system executes action a_t at time t such that $G(s_t, a_t) < \delta$, the Lyapunov stability guarantees that for all future time steps $t' > t$, $\log P(s_{t'}, a_{t'}) > \varepsilon$, where $\log P(s_t, a_t)$ denotes the log-probability of the state-action data distribution. This means that if navigation starts from an in-distribution state, $G(s_t, a_t)$ with a certain threshold δ can be used to decide whether an action will keep the future trajectory within the area with data support and thus makes risk detection far-sighted. Unlike the work proposed in [9], this paper learns LDM from the latent feature vector of the joint state of the robot. This is because previous researches [1], [2] have already shown that different representations of the state result in different generalities. That implies different distributions of the latent feature and different ranges of feasible actions that will keep the trajectory inside the data support. Therefore, the risk detector can be formulated as

$$\begin{aligned}G(f_\theta(s_t^{jn}), a_t^r) = \max\{ &\\ &-\log P(f_\theta(s_t^{jn}), a_t^r), \\ &\min_{a_{t+1}}\{G(f_\theta(s_{t+1}^{jn}), a_{t+1}^r\} \\ &\}, \end{aligned} \quad (8)$$

$$a_t^r = \begin{cases} \pi(s_t^{jn}) & (G(f_\theta(s_t^{jn}), a_t^r) < \delta) \\ \text{risk-avoiding} & (G(f_\theta(s_t^{jn}), a_t^r) >= \delta) \end{cases}, \quad (9)$$

Note that defining δ is not trivial as it trades off safety and efficiency. How to decide the threshold is still an open problem, a research similar to the LDM chose to tune the threshold online [6].

The LDM can be trained by only offline datasets using a CQL styled loss function $loss_2$ composed by a modified TD-error L_{ldm} with CQL term weighted by a constant β (more detailed proof can be found in [9]).

$$L_{ldm} = (G_t - B(G_t))^2 \quad (10)$$

$$B(G_t) = \max\{E(f_\theta(s_t^{jn}), a_t^r), G_{t+1}\}. \quad (11)$$

A flow-based probability density model [23] is learned to estimate the minus log probability density $-\log P(f_\theta(s_t^{jn}), a_t^r)$ as the label E of latent feature-action pairs in the offline dataset.

E. Force-based risk-avoiding behavior

Social Force Model (SFM) [3] is adapted for performing cautious movement when the robot is in risky states. SFM was known for its high local collision avoidance performance, since it generates an artificial repulsive force between agents and forces them to separate [12]. The adapted SFM policy consists of the repulsive force between robot and humans and an attractive force between the robot and its goal, and the output action (velocity) is proportional to the sum of these forces. To achieve a cautious movement, the attractive force is set relatively lower and the repulsive force is set to be high. Since the upper bound of safety performance depends on those settings, careful tuning is needed to adapt to different scenarios, but in this paper, as a proof of the concept, those parameters are left fixed for all experiments.

F. Implementation detail

The implementation was based on the open source provided by previous works [1], [2], [23]. The conservative weights α and β were set to 0.1. The training and evaluation was using a desktop PC with RTX2080Ti graphic card and Core i-9 10980EX cpu (this work used cpu only).

V. Experiments

The experiments were conducted on a simulator. An offline dataset was gathered by a rule-based policy (OCRA [24]) at first and the proposed system was trained with this dataset and tested under different conditions. The details were described as follows.

A. Simulation environment

A 2D simulation environment was adapted from [1]. In the simulation, there were always several humans and the robot moving from random initial positions to their goal. 3 different scenarios were used to simulate the real-world.

- Circle crossing scenario: agents move from a random initial position on a circle with a radius of 4 meters to the opposite side of the circle, which simulated open space.
- Street crossing scenario: agents start from a random initial position on a line and move to a random position on a line parallel to the starting line with distant of 8 meters.
- Circle crossing scenario with static human: the robot's random initial position and goal was on a 4 meters circle, but humans' were on a 2 meters circle in the middle of the 4 meters circle. Those humans have 50% probability to stay static.

In all scenarios, random perturbation was added to the x, y coordinates of the initial and goal positions. For all agents, holonomic kinematics was assumed. Human movement was controlled by ORCA, and their movement will be affected by the robot. The size and preferred speed of all agents were set at 0.3 m and 1.0 m/s, respectively, and uniform noise $[-0.5, 0.5]$ was added to the preferred speed to increase randomness.

B. Training and testing setup

All variants of the proposed method and the baseline methods were learned using simulated demonstration data of the circle crossing scenario containing 4 pedestrians. ORCA was used as an expert policy to generate 500 episodes of demonstration data. To add suboptimalness to the human data, during the demonstration, the human action was quantized in the same way as that of the robot's discrete action space.

As variants of the proposed method, the aforementioned two types of encoders (Attention-based encoder [1], Graph Convolutional Network-based encoder [2]) were trained and tested to investigate the effect of the encoder. For both encoders, the original hyperparameters proposed in previous work were used.

As a baseline risk detection method, a novelty estimator proposed in ThrifyDAgger [20] was implemented. An ensemble of CQL navigation policies ($\# = 4$) was trained under various random seeds and the novelty was estimated as the standard deviation of the ensemble's outputs at runtime. Since the upstream process was different, the final output selection rule was modified as the action with the maximum advantage ($Q(s_t^{jn}, a_t^r) - E_\pi[Q(s_t^{jn}, a_t^r])$ was selected.

Those methods were evaluated under 3 scenarios with different agent densities (4, 6, 8 humans), and various risk detection thresholds. 500 episodes with random initials and goal for each evaluation. The proposed LDM-based risk detector used threshold $\varepsilon \in [m - 1.5\sigma, m + 1.5\sigma]$ where m denotes the mean value of the minus log-probability of the training data and σ denotes its standard deviation. The novelty detector used threshold $[0, 0.61]$. The lower the threshold, the more restrictive the risk detection was performed, which invokes more intervention in the risk-avoiding policy and reduces efficiency.

C. Results and evaluation

1) Quantitative evaluation: Quantitative evaluation results used metrics that include the success rate (SR), timeout rate (TR), average robot time (RT), collision rate (CR), and average calculation time per decision (CT). The results of the navigation policy using the attention-based encoder were

TABLE I

QUANTITATIVE EVALUATION RESULT ON 4 HUMAN CIRCLE-CROSSING ($\delta = m - 0.5\sigma$)

Metrics	SR ↑	TR↓	CR ↓	RT [s]↓	CT [ms]↓
Att-CQL	0.78	0.02	0.18	13.4	1.3
Att-CQL-ESM	0.85	0.03	0.12	13.17	5.7
Att-CQL-LDM	0.92	0.03	0.05	14.98	1.8
Force-based	1.0	0.0	0.0	18.40	0.2

TABLE II

QUANTITATIVE EVALUATION RESULT ON 4 HUMAN CIRCLE-CROSSING

Metrics	SR↑	TR↓	CR ↓	RT [s] ↓	CT [ms] ↓
GCN-CQL	0.33	0.0	0.67	8.32	0.3
GCN-CQL-LDM ($\delta = m - 0.5\sigma$)	0.78	0.01	0.21	19.43	0.7
GCN-CQL-LDM ($\delta = m - 3.5\sigma$)	0.98	0.01	0.01	18.93	0.7

shown in Table I, in which the up and down arrows in the table represent the bigger the better and the smaller the better, respectively. "Att" denoted the attention-based encoder, and "LDM" denoted the using risk detector. The pure ensemble of policy was denoted as "ESM" and referred to as "ESM-NE" when the novelty estimator was used. The table suggested pure offline RL was able to achieve safety and efficiency to some extent and the risk detector-aided system reduced the collision rate by 72.3%. On the other hand, since the risk detection will sometimes trigger the conservative force-based policy, the navigation time increased. The ensemble of policies improved the system performance while requiring extra calculation time. However, the calculation of all variants was lower than 10ms and was considered capable of running in real-time [10].

2) System performance in novel scenario: The evaluation results of the proposed methods under all 9 setting (scenarios: circle crossing, street crossing, circle crossing with static humans, human number: 4, 6, 8) was shown in Fig. 3. 4 human circle crossing scenario was learned and the rest were novel to the robot. In all testing settings, the proposed Att-CQL-LDM substantially reduced the collision rate, while achieving more efficient navigation compared to the force-based risk-avoiding policy (0.55s quicker in 6 human circle crossing with static human scenario and 3.03s quicker in 4 human street crossing scenario). This means the proposed multipolicy control framework could take advantage from both the efficient-prior RL-based method and the safety-prior force-based method by switching the control between them using the LDM-based risk detector. By comparing the results across different scenario, the figure suggested that circle crossing with static humans caused a lot of time out case. That may be because both CQL and force-based policy were difficult to handle the situation when there were several people stand in front and did not move. Incorporating a proactive policy that can help robot clean its path may be desired for this situation. By comparing the results across different numbers of human, the proposed method was almost always able to achieve the same collision rate of the force-based policy, though the force-based policy had degradation when the number of humans increased.

3) Safety-efficiency trade off compared to the previous method: The circle crossing scenario was picked up to investigate the safety-efficiency trade off of both the proposed method and the baseline that using novelty estimation with ensemble of policies. Under 4 human scenario, the system performance under varying risk detection thresholds is shown in Fig. 4. The safety-efficiency trade-off curve of the proposed risk detector and the novelty estimator build upon the ensemble of policies were compared. The figure suggested that though the ensemble novelty estimator got higher efficiency when $CR > 2\%$, the proposed system got higher efficiency when higher safety level is required ($CR < 2\%$). Since safety was the main purpose of the extra risk detection module, the proposed LDM-based detector was considered a better choice compared to the previous method.

The proposed system was further tested in scenarios containing 6 and 8 pedestrians. The results in Fig. 5 suggested that the proposed LDM-based approach outperformed the state-of-the-art method with large gaps in scenarios where human density differs from the training data. In the 6-human scenario, the performance to reduce the collision rate of both methods was saturated at 2%, but the proposed method used less navigation time. In the 8-human scenario, the proposed method got a much lower collision rate at a similar navigation time compared to the ESM-NE and was able to achieve a lower collision rate by using more time. The reason was considered that when navigating in scenarios largely different from the training data, the performance of all subpolicies in the ensemble degraded and failed to leverage the advantage of numbers. On the contrary, detecting out-of-distribution input basically depended on the training data of the LDM, and thus the proposed method was more robust against changes in human density.

4) System performance with different state representation: To investigate the performance of the system with different types of encoders, a graph convolutional network (GCN)-based alternative was implemented. The results in Table II and I suggested that system performance varied under different choices of encoders. While the attention-based encoder led to safety but relatively conservative navigation, the GCN-based encoder led to aggressive navigation. The results also suggested that the risk detector could help the system to reduce the collision rate. However, the time for multi-policies navigation was longer than simply using force-based policies, and under more restricted risk detection the navigation time became closer to the time of the force-based model. The reason for attention-based encoder getting a better result was considered as ego-centric representation reduced the dimensions of the input data and got milder distributional shift compared to GCN, and thus was more favored by offline RL.

5) Qualitative evaluation: A collided case of vanilla Att-CQL was picked up, and the agents' trajectories and the corresponding trajectories in the latent space of the policy

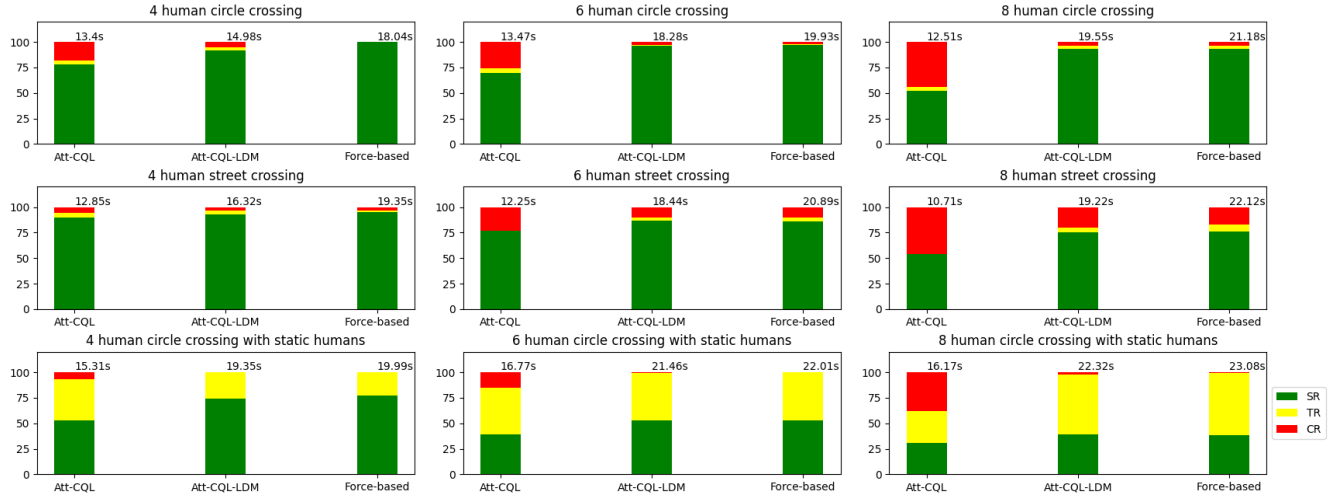


Fig. 3. The evaluation results of the proposed methods under all 9 setting (scenarios: circle crossing, street crossing, circle crossing with static humans, human number: 4, 6, 8). The vanilla Att-CQL, Att-CQL with LDM-based risk detector and the force-based pure risk avoiding policy were compared. Green: success rate, Yellow: time out rate, Red: collision rate. (The results corresponded to $\delta = m - 0.5\sigma$.)

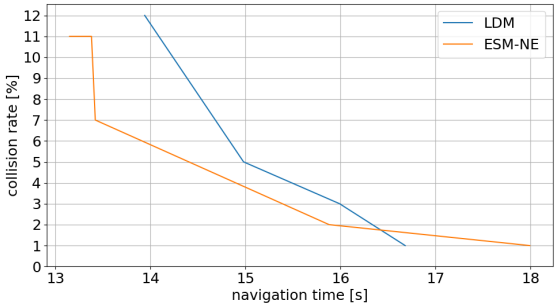


Fig. 4. The safety-efficiency trade-off curve of the proposed LDM-based risk detector (blue) and the novelty estimator built upon the ensemble of policies (orange) were compared. ($\delta \in [m-1.5\sigma, m+\sigma]$, novelty estimation threshold $\in [0.1, 0.5]$)

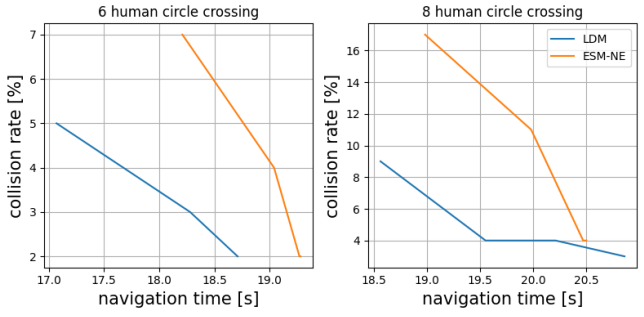


Fig. 5. The safety-efficiency trade-off curve of the proposed LDM-based risk detector (blue) and the novelty estimator built upon the ensemble of policies (orange) compared under scenarios with human density different from the training data.($\delta \in [m-1.5\sigma, m]$, novelty estimation threshold $\in [0.0, 0.08]$). Force-based policy: 6-human: $CR = 2\%$, $RT = 19.93s$; 8-human: $CR = 4\%$, $RT = 21.18s$.

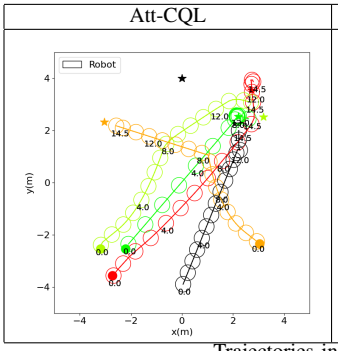
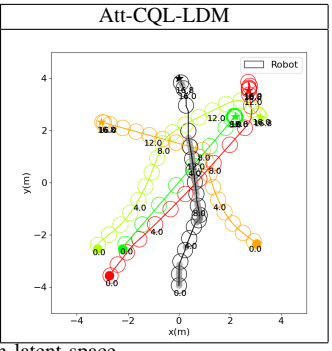
plotted in 2D (using t-SNE) were shown in Table III. Circles denoted trajectories (robot in black and human with other colors). Star denoted their goal. The intervention of the risk-avoiding policy was denoted by the gray shadow inside the robot's circles. The trajectories in latent space were shown in green, blue, and red for training data, and with and without LDM respectively. In this case, the trajectories following vanilla Att-CQL eventually went out of the data support and collided with the human at around 12s when it wanted to turn left to direct to the goal. In the contrast, the LDM detected the risk and made the intervention at the beginning and 6s to prevent the robot from going right. Especially around 6s, there was a clear separation of the trajectories in latent space. There are also long interventions around 8s to 12s. The trajectories in latent space suggested that this might be due to less data support in the middle of the scene, and thus the robot moved cautiously. A recovery policy [25] for navigation in the crowd that is capable to recover the path to the area with data support may reduce the frequency of interventions and improve the efficiency of the navigation.

VI. Conclusion

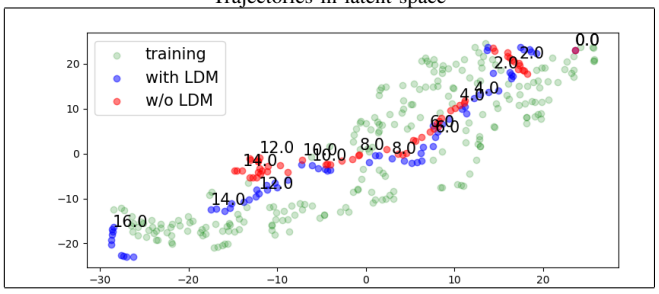
In this paper, a multi-policy control framework based on offline reinforcement learning for risk-sensitive robot navigation is proposed. The experiment results showed that the system was able to learn navigation from offline dataset and was robust against distributional shift between training data and input during deployment by detecting the risk of go-out-of-training data support using the Lyapunov density model, which substantially reduced the collision rate while maintaining the navigation efficiency outperformed the state-of-the-art work. As future work, a seamless co-working between efficiency-prior and safety-prior policies will be tackled to further improve navigation efficiency under safety constraints.

TABLE III

COMPARING TRAJECTORIES UNDER DIFFERENT METHODS.

Att-CQL	Att-CQL-LDM
	

Trajectories in latent space



REFERENCES

[1] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *Proceeding of the 2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6015–6022.

[2] C. Chen, S. Hu, P. Nikdel, G. Mori, and M. Savva, "Relational graph learning for crowd navigation," in *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 10 007–10 013.

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

[4] Y. Tamura, T. Fukuzawa, and H. Asama, "Smooth collision avoidance in human-robot coexisting environment," in *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 3887–3892.

[5] C. Mavrogiannis, P. Alves-Oliveira, W. Thomason, and R. A. Knepper, "Social momentum: Design and evaluation of a framework for socially competent robot navigation," *ACM Transactions on Human-Robot Interaction (THRI)*, vol. 11, no. 2, pp. 1–37, 2022.

[6] H. Bharadhwaj, A. Kumar, N. Rhinehart, S. Levine, F. Shkurti, and A. Garg, "Conservative safety critics for exploration," *arXiv preprint arXiv:2010.14497*, 2020.

[7] H. Karnan, A. Nair, X. Xiao, G. Warnell, S. Pirk, A. Toshev, J. Hart, J. Biswas, and P. Stone, "Socially compliant navigation dataset (scand): A large-scale dataset of demonstrations for social navigation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 807–11 814, 2022.

[8] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," vol. 33, 2020, pp. 1179–1191.

[9] K. Kang, P. Gradu, J. J. Choi, M. Janner, C. Tomlin, and S. Levine, "Lyapunov density models: Constraining distribution shift in learning-based control," in *Proceedings of the 39th International Conference on Machine Learning*, 2022, pp. 10 708–10 733.

[10] H. Nishimura, B. Ivanovic, A. Gaidon, M. Pavone, and M. Schwager, "Risk-sensitive sequential action control with multi-modal human trajectory forecasting for safe crowd-robot interaction," in *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 11 205–11 212.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[12] S. H. Semnani, H. Liu, M. Everett, A. De Ruiter, and J. P. How, "Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3221–3226, 2020.

[13] L. KU+000E4stner, J. Cox, T. Buiyan, and J. Lambrecht, "All-in-one: A drl-based control switch combining state-of-the-art navigation planners," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 2861–2867.

[14] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *Proceeding of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 1343–1350.

[15] T. v. d. Heiden, F. Mirus, and H. v. Hoof, "Social navigation with human empowerment driven deep reinforcement learning," in *Proceedings of the International Conference on Artificial Neural Networks*. Springer, 2020, pp. 395–407.

[16] Y. Liu, Q. Yan, and A. Alahi, "Social nce: Contrastive learning of socially-aware motion representations," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 118–15 129.

[17] S. Dhruv, B. Arjun, L. Hrishit, K. Ilya, R. Nicholas, and L. Sergey, "Offline reinforcement learning for visual navigation," in *Conference on Robot Learning*, 2022.

[18] A. Hakobyan and I. Yang, "Wasserstein distributionally robust motion planning and control with safety constraints using conditional value-at-risk," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 490–496.

[19] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 683–700.

[20] R. Hoque, A. Balakrishna, E. R. Novoseller, A. Wilcox, D. S. Brown, and K. Goldberg, "Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning," in *Conference on Robot Learning*, 2021.

[21] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International Conference on Machine Learning*, 2019, pp. 2052–2062.

[22] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn, "Combo: Conservative offline model-based policy optimization," vol. 34, 2021, pp. 28 954–28 967.

[23] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, "Neural spline flows," *Advances in neural information processing systems*, vol. 32, 2019.

[24] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart, "Optimal reciprocal collision avoidance for multiple non-holonomic robots," in *Distributed Autonomous Robotic Systems*. Springer, 2013, pp. 203–216.

[25] B. Thananjeyan, A. Balakrishna, S. Nair, M. Luo, K. Srinivasan, M. Hwang, J. E. Gonzalez, J. Ibarz, C. Finn, and K. Goldberg, "Recovery rl: Safe reinforcement learning with learned recovery zones," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4915–4922, 2021.