Deep Reinforcement Learning-Based Mapless Crowd Navigation with Perceived Risk of the Moving Crowd for Mobile Robots

Hafiq Anas, Wee Hong Ong, and Owais Ahmed Malik

***Abstract*— Current state-of-the-art crowd navigation approaches are mainly deep reinforcement learning (DRL)-based. However, DRL-based methods suffer from the issues of generalization and scalability. To overcome these challenges, we propose a method that includes a Collision Probability (CP) in the observation space to give the robot a sense of the level of danger of the moving crowd to help the robot navigate safely through crowds with unseen behaviors. By focusing on the most dangerous obstacle, the robot will not be confused when the crowd density is high, ensuring scalability of the model to increasing crowd density. Our approach was developed using deep reinforcement learning (DRL) and trained using the Gazebo simulator in a non-cooperative crowd environment with obstacles moving at randomized speeds and directions. We then evaluated our model on four different crowd-behavior scenarios with varying densities of crowds. The results showed that our method achieved a 100% success rate in all test settings. We compared our approach with a current state-of-the-art DRL-based approach, and our approach has performed significantly better. Importantly, our method is highly generalizable to different crowd behaviors and requires no fine-tuning after being trained once. We further demonstrated the crowd navigation capability of our model in real-world tests.**

I. INTRODUCTION

In crowd navigation, or social navigation, the classical navigation approaches of using global and local planners struggle in highly dense crowded environments and would often result in the robot being stuck in an endless replanning state. Recent research works have focused on deep reinforcement learning (DRL) methods [1]. Many recent DRL-based works are mapless and have empirical evidence that demonstrates the capability of DRL-based approaches with 2D laser scans for crowd navigation [2][3][4][5]. An early DRL-based solution to address the crowd navigation problem was the CrowdMove implementation [5]. CrowdMove was trained and tested in multiple dynamic environments using commonly used observation states, such as the robot's own velocity and relative target goal position. The authors concluded that their robot was able to avoid moving obstacles in real-world tests and that their trained model could be generalized to different environment settings unseen during training. We note that their approach relied on providing sufficient variation in the training data of multiple dynamic environments to improve generalization. In a recent work, Jin et al. [2] proposed that a robot moving in a crowded environment should have human-awareness competencies. Therefore, they implemented this through their reward setup by incorporating two conditions: ego-safety and social-safety violations. Using this reward setting, they trained their robot in one crowded environment and tested it in four different crowd behavior environments with varying number of moving obstacles. They achieved significant performance improvement over the then state-of-the-art DRL-based crowd navigation method, CADRL [6]. We consider their work a representative current state-of-the-art in crowd navigation using 2D laser scans.

Jin et al. [2] used ego and social scores in their reward function to model human-awareness, but this approach is limited by the lack of access to such information during deployment. In this sense, their model will require a large amount of training to infer the perceived risk from the typical observation states in different scenarios. Unlike Jin et al. [2], we incorporate perceived risk or human-awareness into the observation states, which allows the robot to perceive potential risk during testing or deployment. In addition, we have tested our robot at a significantly higher relative speed of the obstacles to the speed of the robot than in [2]. We evaluated our approach in different crowd behavior settings with varying numbers of obstacles at different speeds, and compared our results with the results of Jin et al. [2] under the same set of test conditions. The main contributions of this work are the idea of including risk perception in the observation space to allow the robot to perceive the danger level of moving crowds, and focusing on the most dangerous obstacle to ensure scalability of the model to high density crowd. We have also verified the ideas through successful implementations in both simulation and real-world settings.

II. APPROACH

A. Problem Formation

Our proposed method builds upon the techniques used in the existing DRL-based methods, with an addition of perceived risk in the observation states and prioritizing the most dangerous obstacle within a crowd. To determine which obstacle to prioritize, we compute the collision probability of all tracked moving obstacles within the robot's field of view (FOV) and focus on the obstacle with the highest probability of collision. Fig. 1 shows the overview of our deep reinforcement learning (DRL) system. The components in the system are described in the following subsections.

*1) **Observation space**:* The observation space contains input features to learn as well as perform crowd navigation behavior that solves both local and global navigation. To solve the global navigation problem, the information of relative distance to goal (DTG) and orientation (heading

All authors are with School of Digital Science, Universiti Brunei Darussalam, Jalan Tungku Link, Brunei. hafiq.anas@gmail.com, [weehong.ong, owais.malik]@ubd.edu.bn

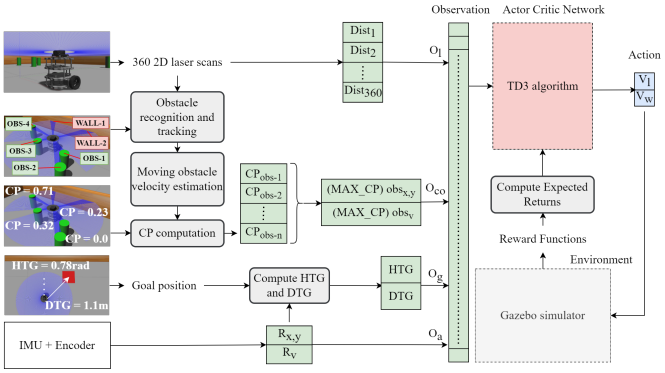

Fig. 1. Deep Reinforcement Learning system structure.

to goal, HTG) of the target goal location are used as the goal-related observations o_g. Meanwhile, o_l contains distance information from the 2D laser scan sensor that describes the static environment of the robot and is used to solve the local navigation problem. Given that a crowded environment is associated with moving obstacles, we added agent-related observations o_a and critical obstacle observation o_{co} to the observation space. o_a contains the robot's position ($R_{x,y}$) and velocity (R_v) estimated from its encoder and inertia sensor. o_{co} describes the position ($obs_{x,y}$) and velocity (obs_v) of the most dangerous moving obstacle (critical obstacle). We define an observation as o = [o_l, o_g, o_a, o_{co}] which describes the partial environment the robot can observe at a given time.

Obstacles tracking was implemented for obstacle velocity estimation and computation of Collision Probability (CP) from the 2D laser scans o_l. We define the Collision Probability (CP) as the sum of two component probabilities: the probability of collision based on the time to collision (P_{c-ttc}) and the probability of collision based on the distance to the obstacle (P_{c-dto}). We argue that the addition of distance to obstacle (*dto*) information allows the robot to better perceive the collision probability with a moving obstacle in the crowd. For example, an obstacle moving slowly near the robot can still pose substantial risk of collision while an obstacle moving fast toward the robot from a far distance is less dangerous. Therefore, a balance between the two CP components is made as given in (1).

$$CP = \alpha \cdot P_{c-ttc} + (1-\alpha) \cdot P_{c-dto} \tag{1}$$

where $\alpha \in [0, 1]$ is the parameter that decides the weight of collision probabilities P_{c-dto} and P_{c-ttc}. We have set $\alpha = 0.5$ in the experiments reported in this paper.

The calculation of collision probabilities uses the Collision Cone (CC) concept in [7][8]. P_{c-ttc} is computed based on time to collision as defined in (2). P_{c-dto} is computed based on relative distance to obstacle and defined in (3).

$$P_{c-ttc} = \begin{cases} \min(1, \frac{0.15}{t}), & \text{if } V_r' \in CC_{ro} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where t is the time-to-collision (TTC) when the relative velocity V_r' between the robot and the obstacle lies within the Collision Cone CC_{ro}. $V_r' = V_r - V_o$ is the resultant velocity between the robot velocity V_r and obstacle velocity V_o. $t = Dist_o/V_r'$ is the time to collision. CC_{ro} is the collision cone area between the robot and obstacle. Finally, 0.15 corresponds to the timestep value of the robot in seconds for executing its velocity commands.

$$P_{c-dto} = \begin{cases} \frac{l_{max} - Dist_o}{l_{max} - l_{min}}, & \text{if } Dist_o < l_{max} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where l_{max} and l_{min} are the maximum and minimum range of the laser scan respectively. $Dist_o$ is the distance from the robot to the obstacle of interest.

CP is computed for each obstacle in the list of tracked obstacles and the position ($obs_{x,y}$) and velocity ($obs_v = V_o$) of the obstacle with the highest CP is included in the observation space as o_{co}. This obstacle is seen as most probable to be in collision with the robot.

*2) **Action space**:* An action is defined as a = [V_l, V_w] which is sampled from a stochastic policy π given observation o: $a \sim \pi(a \mid o)$ where V_l is the linear velocity within the range [0, 0.22] ms^{-1} and V_w is the angular velocity within the range [-2.0, 2.0] $rad.s^{-1}$.

*3) **Reward functions**:* The reward function consists of the following terms:

$$R = R_{step} + R_{dtg} + R_{htg} + R_{goal} + R_{col} \tag{4}$$

$R_{step} = -2$ is the negative reward given to the robot for every step and serves to encourage the robot to avoid abusing the R_{dtg} and R_{htg} rewards by oscillating around the goal location without reaching it.

$$R_{dtg} = \begin{cases} +1, & \text{if } \mathrm{d}(r,g)_t < \mathrm{d}(r,g)_{t-1} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

R_{dtg} is the positive reward given to the robot whenever the distance from the robot to the target goal location $\mathrm{d}(r, g)$ has reduced between the current and previous timestep.

$$R_{htg} = \begin{cases} +1, & \text{if } \theta(r,g)_t < \theta(r,g)_{t-1} \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Similarly, R_{htg} is the positive reward given whenever the relative heading $\theta(r, g)$ has decreased.

$R_{goal} = +200$ is the large positive reward given to the robot when it reaches the target goal location. If a collision occurs, a penalty $R_{col} = -200$ is given instead.

B. Deep Reinforcement Learning

We use the Twin Delayed Deep Policy Gradient (TD3) [9] algorithm with the default parameters to learn the navigation policy.

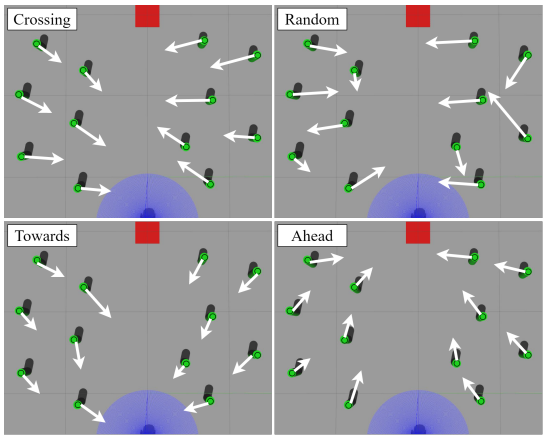

Fig. 2. The four crowd behavior settings with 12 obstacles in Gazebo. **Crossing**: The robot has to navigate through the crowd moving in the crossing directions. **Towards**: The crowd is moving toward the general direction of the robot. **Ahead**: The crowd is moving ahead of the robot. **Random**: The crowd is moving in random directions.

*1) **Model training**:* The robot was trained in the Gazebo simulator using Robotis TurtleBot3 Burger platform that is equipped with a LDS-01 360-degree 2D laser scanner and XL430-W250 encoder motors. The resolution of the laser scanner is 360 with a minimum and maximum range set to 0.105m and 0.6m respectively. The training process was done once in a 2m x 2m space with walls and 14 moving obstacles moving at a random speed of up to 0.2m/s in random directions. The moving obstacles were non-cooperative so they will ignore the robot's presence and can collide with the robot. The model was trained for 3000 episodes with the stopping criteria of collision with an obstacle or having reached the goal.

*2) **Model testing**:* The robot was trained in one simulation setting using TD3 and tested in different crowd behavior settings. The crowd was non-cooperating.

III. EVALUATION

We evaluated our robot in different crowd behavior environments similar to [2]: crossing, towards, ahead and random. For each crowd behavior, the model was tested with three different crowd densities of four, eight and twelve obstacles, except the random crowd behavior was only tested with twelve moving obstacles. This gave a total of ten test settings. For each crowd behavior setting, we computed the average of each metric over 10 separate runs. Fig. 2 shows the four crowd behavior settings with 12 moving obstacles.

To quantify performance, we used the same evaluation metrics from Jin et al.'s [2] work: success rate (%), arriving time (s), ego score (0-100) and social score (0-100). Let k be the number of ego-safety violation steps, and N be the total steps to reach the goal, then $Ego\_Score = (1 - k/N) * 100$. Let m be the number of social-safety violation steps, then $Social\_Score = (1 - m/N) * 100$.

An ego-safety violation is determined when an obstacle comes close to the robot within the ego radius of the robot. We have set the ego radius 0.787 times of the largest width of the Turtlebot3 base on the same ratio used in [2]. In [2], they have determined the social-safety violation when two rectangular spaces computed from the speed of the robot and the speed of an obstacle intersect. The rectangular spaces are similar to the concept of Collision Cone in our case. For the social-safety violation, we have used the CP to determine if our robot is in a collision trajectory course with an obstacle when the CP value is greater than 0.4.

For comparison purposes, we have determined by watching Jin et al.'s demonstration video [2] that their obstacles were moving about 5 times slower than the max speed of their robot (1.5m/s). In our case, we performed two separate tests in Gazebo with slow-moving and fast-moving obstacles. The slow-moving obstacles moved at a speed that is 2 times slower (0.1m/s) than our robot's max speed (0.22m/s). The fast-moving obstacles moved at a speed (0.2m/s) that is nearly the same as our robot's speed. We believe that in real-world situations, the crowd would be moving at a speed close to each other.

In real world test, the robot was tested in the four crowd behaviors with four obstacles. We have used mobile robots with similar size to the Turtlebot3 as moving obstacles. The moving obstacles were manually teleoperated by humans. It was difficult to teleoperate the obstacles when there are many of them, hence we have limited the real-world tests to four obstacles.

IV. RESULTS AND DISCUSSION

A. Crowd Navigation

Fig. 3 shows the evaluation results of our method in comparison to the results of Jin et al. [2]. Our robot achieved 100% Success Rates (SR) in all test environments. Jin et al.'s [2] success rates were between 60% to 100% with only 10 out of 20 tests achieved 100% success rate. We have observed that our approach did not exhibit freezing robot problem during training and testing as the robot could navigate smoothly to the goal positions without getting stuck in the crowd. In comparison, frequent freezing was observed when using the ROS Navigation stack in the same crowded environments.

Our robot took a longer time to reach the goal when the crowd was more dangerous. The arrival times for the test environments with fast-moving obstacles (more dangerous) were in general longer than with slow-moving obstacles. Likewise, higher crowd density (more dangerous) resulted in longer arrival time. Exceptions were seen in the ahead crowd behavior, where the arrival times with fast moving obstacles ahead were shorter than with slow moving obstacles. When the obstacles were moving fast ahead of the robot, there was very little chance of the robot being confronted by the obstacles. The ahead environments were quite safe. Consequently, there were very few safety violations as seen from the high ego and social scores in the results of ahead crowd behavior in Fig. 3. We note the model was not trained with ahead crowd behavior; however, it has learned to estimate the risk level given the observation space. While the arrival times of our robot were longer than the results of [2], we note that

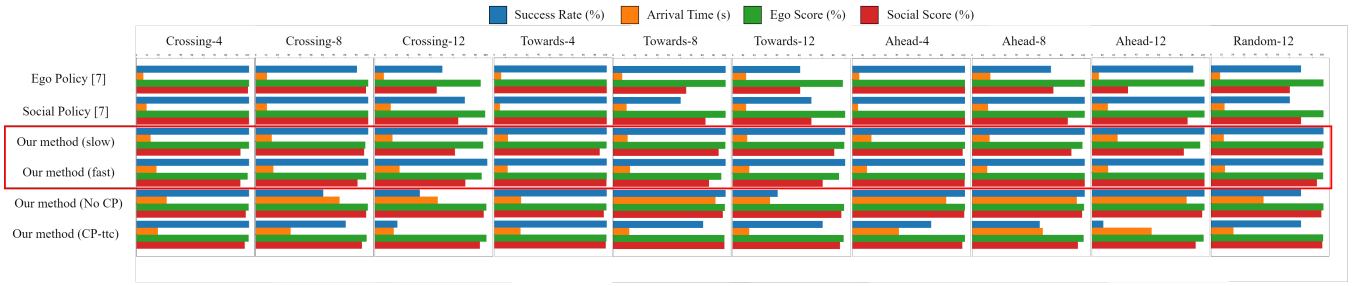

Fig. 3. Comparison between our approach and Jin et al.'s results [2]. Our method was tested with slow- and fast-moving obstacles. The label of the crowd behavior settings follows the pattern of "behavior-obstacles", e.g., crossing-4 is the test environment with crossing crowd behavior and 4 moving obstacles. The bottom two rows are results of ablation study.

their robot was traveling at a speed (1.5m/s) about 7 times faster than our robot (0.22m/s). Taking into account the speed difference, our approach has performed relatively faster and with higher success rate than the approach of [2].

In addition, we have observed that Ego-safety and social-safety violations do not necessarily result in collisions. They are measures of how risky the robot was navigating in the crowd. Given that in cases where the ego and social scores were low and the robot was able to successfully navigate through the crowd, it demonstrated that our robot could take a higher risk to reach the goal faster given its improved ability to perceive the risk.

Finally, in real-world tests, we observed similar crowd navigation capability as in the simulation. However, the physical robot could not move smoothly at the velocities setting used in the simulation.

B. Ablation Study

To investigate the effect of the Collision Probability (CP) (1) and its two components, we trained the model with two variations: one without P_{c-dto} (distance to obstacle CP) component (Model-CP-ttc), and one without CP completely (Model-no-CP). The results with fast obstacle speed are shown in bottom two rows in Fig. 3.

As anticipated the Model-no-CP achieved a lower success rate and was four times slower in arrival time on average than the model with complete risk perception (full CP, 4th row). During the tests, the robot was observed to avoid obstacles altogether by trying to detour. Without CP, the model was not able to estimate collision risk, so it learned that the best way to avoid collision was by avoiding the obstacles completely. Surprisingly, the Model-CP-ttc could only achieve 100% success rate in 2 out of 10 test conditions. The success rate was lower than the Model-no-CP. At first sight, it may seem that CP was not helpful. However, by observing the robot, we noticed that the robot was attempting to traverse through the crowd but collided with the obstacles when it was too close to an obstacle. This resulted in a lower success rate, however with a significantly faster arrival time than the Model-no-CP. The P_{c-ttc} (time to collision CP) alone underestimated the danger level of the moving obstacles and was insufficient to perceive the risk during crowd navigation. This caused the robot to be in a situation where it found itself unable to avoid a collision which caused the lower average success rate, especially in higher-density crowd tests (obstacle-12, ahead-12). The addition of P_{c-dto} (distance to obstacle CP) has improved the risk estimation as shown in the superior performance of the model with full CP (4th rows) with fast moving obstacles.

V. CONCLUSIONS

We have developed a navigation approach for mobile robots using 2D laser scans to improve their performance in crowded environments. Our experiments have shown that the inclusion of the Collision Probability of the most dangerous moving obstacle to the observation space has achieved outstanding performance in crowd navigation. Our model was trained in one crowd environment setting and tested on 10 different crowd environment settings. We achieved 100% success rate in all the 10 environment settings including the settings in which the obstacles were moving as fast as the robot. The perception of risk has enabled the robot to take calculated risk in navigating the crowd. Besides the superior performance in the simulated environment, we have also demonstrated the crowd navigation capability of our model in real-world tests. The robot has shown promising performance although not as dexterous as in the simulation. We plan to expand the real-world tests and improve the real-world performance in our future work. We will also investigate further ways to incorporate perceived risk or human awareness in our crowd navigation approach.

The source code and video demonstration of this work are made publicly available on GitHub [10].

REFERENCES

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

[2] 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 *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 6979–6985.

[3] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 31–36.

[4] O. Zhelo, J. Zhang, L. Tai, M. Liu, and W. Burgard, "Curiosity-driven exploration for mapless navigation with deep reinforcement learning," in *ICRA 2018 Workshop on Machine Learning in Planning and Control of Robot Motion*, May 2018.

[5] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6252–6259.

[6] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 285–292.

[7] L. Sun, J. Zhai, and W. Qin, "Crowd navigation in an unknown and dynamic environment based on deep reinforcement learning," *IEEE Access*, vol. 7, pp. 109 544–109 554, 2019.

[8] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The international journal of robotics research*, vol. 17, no. 7, pp. 760–772, 1998.

[9] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[10] H. Anas. Deep reinforcement learning based crowd navigation with perceived risk of the moving crowd for mobile robots. [Online]. Available: https://github.com/ailabspace/drl-based-mapless-crowd-navigation-with-perceived-risk