2023 IEEE 18th Conference on Industrial Electronics and Applications (ICIEA) | 979-8-3503-1220-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICIEA58696.2023.10241956

Cooperative Navigation of Differential-Drive Mobile Robots in Crowd Environments

1st Shijun Yan
Advanced Remanufacturing & Technology Centre (ARTC)
*Agency for Science, Technology and Research (A*STAR)*
Singapore
yan_shijun@artc.a-star.edu.sg

2nd Anthony Wen Hao Lee
School of Mechanical and Aerospace Engineering
Nanyang Technological University
Singapore
an0001ao@e.ntu.edu.sg

3rd Mingyang Guan
Advanced Remanufacturing & Technology Centre (ARTC)
*Agency for Science, Technology and Research (A*STAR)*
Singapore
guan_mingyang@artc.a-star.edu.sg

4th Ning Liu
Advanced Remanufacturing & Technology Centre (ARTC)
*Agency for Science, Technology and Research (A*STAR)*
Singapore
liu_ning@artc.a-star.edu.sg

***Abstract*—Mobile robotics has gained a lot of attention due to its wide range of applications in various domains such as warehouses, factories, and hospitals. In crowded environments, cooperative navigation is essential to ensure the safe and efficient movement of mobile robots. The cooperative navigation strategies must enable robots to coordinate their movements and adapt to the dynamic environment. In recent years, significant progress has been made in developing cooperative navigation algorithms for mobile robots. The algorithms can be grouped into centralised and decentralised methods, each with its advantages and disadvantages. However, most deep-learning-based methods have been designed for holonomic robots, while differential-drive mobile robots pose more challenges due to their limited degrees of freedom. The paper presents a method that uses deep reinforcement learning to enable differential-drive mobile robots to navigate cooperatively in crowded environments. The method employs a cooperation module to model the interactions among the robot and other agents, and a self-attention mechanism to prioritize the interaction features. The experimental outcomes show that the method outperforms several existing algorithms, which obtains better success rate and lower collision rate, indicating its effectiveness in guiding robots to their destinations while avoiding collisions.**

***Index Terms*—mobile robots, cooperative navigation, reinforcement learning, decentralised method**

I. Introduction

The emergence of mobile robotics has led to a growing interest in developing efficient and safe navigation algorithms for autonomous mobile robots. The mobile robots have been applied widely in warehouses [1], factories [2], [26], hospitals [3], [27], and etc.. In crowded environments, mobile robots must navigate among humans and other mobile robots without causing inconvenience or harm. It requires the development of cooperative navigation strategies that enable robots to coordinate their movements, and adapt to the dynamic environment. Without cooperation, robots may take suboptimal paths or get stuck in crowded areas, leading to congestion and delays. Cooperative navigation of mobile robots in crowded environments is a challenging task that involves addressing issues such as collision avoidance and path planning among robots.

In recent years, significant progress has been made in developing cooperative navigation algorithms for mobile robots. Generally, the navigation algorithms can be grouped into two categories, which are centralised and decentralised navigation methods. Centralised navigation methods [4]–[6] involve a central entity, such as a server or a central robot, that coordinates the movements of all robots in the environment. This method can be effective in environments with a small number of robots, as it enables the central entity to optimize the paths and avoid collisions. However, in larger environments with many robots, centralised methods can become computationally intensive, and delays in communication can lead to inefficient movements or collisions. Decentralised navigation methods [7]–[9], on the other hand, do not rely on a central entity and instead enable each robot to make its own decisions about its movements based on local information. Decentralised methods are more scalable than centralised methods and can handle large numbers of robots in a distributed way, while, they can be more challenging to design, as each robot must be able to make decisions autonomously and adapt to changing conditions in the environment.

With increasing development effort in cooperative navigation algorithms, Most deep-learning based methods for cooperative navigation of mobile robots have been designed for holonomic robots, mainly because these robots can move in any direction without changing their orientation. However, non-holonomic robots, such as differential drive robots, have more limited degrees of freedom and can only move in a limited set of directions. This makes their navigation more challenging. Differential-drive mobile robots are more common to be used than holonomic robots due to their much simpler actuation mechanisms. Although Long, et al. [10] present a decentralized navigation policy for differential-drive-robot systems, the performance is quite related to the testing scenarios.

The article provides an deep-learning based method for cooperative navigation of differential-drive mobile robots in crowded environments. With comparison with several other methods, the approach described in the article shows a great potential for achieving favorable navigation performance metrics such as success rate and collision rate.

II. Related Work

Cooperative navigation of mobile robots in crowd environments is an area of active research. Since decentralised navigation methods are more scalable than centralised methods, the article provides a brief literature review of decentralised methods. Based on the level at which the cooperative navigation operates, the decentralised methods can be categorised into sensor-level and agent-level co-operations.

A. Sensor-level co-operations

Long et al. [10] and Fan et al. [11] propose collision-free navigation policies for mobile robots based on unprocessed sensor data, such as 2D laser scans of environments and robot velocities. The method involves a direct mapping from the raw sensor data to the velocity of movement of an agent. The sensor-level methods allow robots to make independent navigation decisions without communicating with other robots. Sathyamoorthy et al. [12] extended this approach to consider visual sensors and trajectory prediction to anticipate pedestrian behaviors. The information is utilized along with a collision avoidance approach based on Deep Reinforcement Learning to generate trajectories during runtime. Sensor-level methods often rely on deep neural networks to train collision-free navigation policies using large datasets through supervised learning [13], [14]. However, these methods have several drawbacks, including the need for large amounts of training data, the difficulty in designing an accurate loss function, and the limited accuracy of using single-sensor data. Despite these efforts, the performance and success rate of sensor-level methods still need further improvements, especially when compared to centralized methods for navigating through dense and complex environments.

B. Agent-level co-operation

In crowd simulations and multi-agent systems with multiple robots, Reciprocal velocity obstacles (RVO) [15] and Optimal Reciprocal Collision Avoidance (ORCA) [16] have emerged as popular methods due to their ability to handle scenarios with many agents [17], [18]. These methods operate under the assumption that each agent has complete knowledge of its neighbors' velocities, positions, and shapes. With this information, the agents apply the policies' algorithms to compute velocities that guarantee their safety in the next time step. Additionally, the methods are developed specifically for holonomic robots. To make ORCA suitable for non-holonomic robots, Snape et al. [19] introduced ORCA differential-drive (ORCA-DD), which expands the radius of the robot used for navigation by a factor of two in simulations, allowing it to be considered holonomic under the original ORCA policy. However, the enlarged virtual size of the simulated robot makes it difficult for ORCA-DD to navigate through narrow passages or unstructured environments.

Deep Reinforcement Learning (DRL) has also attracted many researchers' attention to be applied in navigation of mobile robots. DRL-based methods involve training policies through multiple trials or episodes, where robots receive feedback in the form of rewards or penalties. In order to achieve effective training, it is necessary for robots to remain in the training environment for extended periods, allowing them to learn from the outcomes of their actions across a range of scenarios. Pathfinding via reinforcement and imitation multi-agent learning (PRIMAL), a learning-based method developed by Sartoretti et al. [20], combines both imitation and reinforcement learning to enable multi-agent navigation. In a subsequent study, PRIMAL2 [21] was introduced to address the issue of local inter-robot communication while maintaining full decentralization. Chen et al. [22] present an attention-based DRL framework designed for agent-level decentralized navigation. This approach leverages DRL to model interactions between robots and humans, as well as between humans themselves. By considering the collective importance of neighboring humans, the proposed method aims to achieve efficient navigation even in densely crowded environments. Nevertheless, similar to the majority of DRL-based navigation research, the robots employed for both training and experimentation in this study are holonomic in nature.

Holonomic robots are capable of unconstrained movement in any direction, but in practical applications, differential-drive robots are more frequently utilized. The article expands the application of DRL-based navigation to differential-drive robots and evaluates its performance against various other methods. The results demonstrate the potential of the approach for effective navigation in densely crowded environments.

III. Methodology

In this section, the article describes the DRL-based navigation approach utilized and a modeling method that enables a differential-drive robot to be treated as a holonomic robot.

A. Navigation policy

The method discussed in this article, which use reinforcement learning, are referred to as a decision-making problem [23]. In this framework, an agent, which can be a robot or a human, has an observable position, which can be denoted by $\mathbf{p} = [p_x, p_y]$, observable velocity, which can be denoted by $\mathbf{v} = [v_x, v_y]$, and radius of its footprint $\mathbf{r}$. The state of the i^{th} agent is denoted as $\mathbf{s}_t^i$ at time t, and the joint state, which include the i^{th} agent and its surrounding neighbor agents, are denoted as $\mathbf{s}_t^{i,n} = [\mathbf{s}_t^i, \mathbf{s}_t^{i,0}, \mathbf{s}_t^{i,1}, ..., \mathbf{s}_t^{i,n-2}]$, where n denotes the total number of the agents in the joint state. Each agent knows its preferred velocity, $\mathbf{v}_{pref}$, and its final goal position $\mathbf{p}_g$, which are not observable by other agents. One assumption is that every agent reaches its commanded velocity $\mathbf{v}_t$ instantaneously after receiving the action command

$\mathbf{a}_t$ at time t. An optimal policy can be created to maximize the return output, and the policy can be defined as shown in (1).

In (1), $R(\mathbf{s}_t^{i,n}, \mathbf{a}_t)$ is the reward the robot can obtain at time t with the state $\mathbf{s}_t^{i,n}$ by performing the action $\mathbf{a}_t$, and γ is a discount factor, which has a value within the interval $(0, 1)$. The discount factor is a term that helps to balance the influence of future rewards with that of immediate rewards. $P(\mathbf{s}_t^{i,n}, \mathbf{a}_t, \mathbf{s}_{t+\delta t}^{i,n})$ is the transition probability of the policy from time t to $t + \delta t$. V^* is the optimal state-value function and the integration represents the predicted cumulative reward that the robot can obtain by following the optimal policy.

The reward function is defined in (2) to reward the robot if it reaches its goal successfully but penalise the robot if its distance to other agents is less than the least safe distance. In (2), D is the least safe distance, and 0.2 is a buffer distance which is added onto the least safe distance to ensure safety.

$$R(\mathbf{s}_t^{i,n}, \mathbf{a}_t) = \begin{cases} -0.25 & \text{if } d_t < D \\ -0.1 + (d_t - D)/2 & \text{if } d_t < D + 0.2 \\ 1 & \text{if goal reached} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

B. Cooperation module

During navigation, the neighboring agents would affect navigation decisions. It is necessary to include neighboring agents into consideration during navigation. The module proposed in [22] is adopted in the article to model the interaction between a robot and other agents with a pairwise interaction module and the interactions between other agents with coarse local map. The local map is used to encode the presence and velocities of neighbors. The robot state and its jth neighbor's state, together with the local map, are fed into an embedding function with ReLU activation to obtain a fixed length vector v_j, and then fed into a fully-connected layer with ReLU activation to obtain the pairwise interaction features f_j.

The number of neighbors around the robot will vary at each time instance, as they are constantly moving, and the effect of each neighbor on the robot's navigation also varies. A self-attention mechanism proposed in [24] is used to identify and prioritise the interaction features. The average of all the fixed length vector v_j obtained from embedding function is calculated first, and then the average and the vector v_j are fed into a collection of interleaved fully connected layers with ReLU activation to obtain the attention score a_j. With the attention score, the interaction is modelled as (3).

$$c = \sum_{j=1}^{n-1} \text{softmax}(a_j) f_j \tag{3}$$

The interaction c and robot state $\mathbf{s}_t^i$ are fed into a collection of interleaved fully connected layers with ReLU activation to obtain next action of the robot as in (4).

$$v = \Phi(s, c; w_v) \tag{4}$$

where Φ is the collection of interleaved fully connected layers with ReLU activation and the weights in the layers are denoted by w_v.

The architecture is depicted in Figure 1.

C. Kinematic conversion

As mentioned in III-A, the mobile robot is supposed to be able to reach the commanded velocity instantaneously. The action obtained from the navigation policy is linear velocities in x and y directions, while it is not achievable for differential-drive robots due to the their motion constraints. The conversion method proposed in [19] is used to convert a differential-drive robot to a holonomic robot.

The conversion is plotted in Figure 2. The differential-drive robot has its centre at $q = (x, y)$ with a radius of r and a distance of L between its two driving wheels. Its orientation angle is denoted by θ. The velocities of left and right driving wheels are denoted by v_l and v_r. The corresponding holonomic robot has its centre at $p = (X, Y)$ with a radius of R. Its velocity is $v_p = (\dot{X}, \dot{Y})$. The distance between the centres p and q is D. The velocity v_p can be obtained with (5).

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \end{bmatrix} = M \begin{bmatrix} v_l \\ v_r \end{bmatrix} \tag{5}$$

where

$$M = \begin{bmatrix} \text{cos}\theta/2 + D\text{sin}\theta/L & \text{cos}\theta/2 - D\text{sin}\theta/L \\ \text{sin}\theta/2 - D\text{cos}\theta/L & \text{sin}\theta/2 + D\text{cos}\theta/L \end{bmatrix} \tag{6}$$

The matrix M is inversible due to $D > 0$ and $L > 0$. Therefore

$$\begin{bmatrix} v_l \\ v_r \end{bmatrix} = M^{-1} \begin{bmatrix} \dot{X} \\ \dot{Y} \end{bmatrix} \tag{7}$$

IV. Experiments

A. Setup

The PyTorch implementation of the policy is trained using Adam with a batch size of 100. The learning rate is set to 0.001, and the discount rate factor γ is set to 0.9. The exploration rate is reduced from 0.5 to 0.1 over the first 5000 episodes, and remains at 0.1 thereafter. For the conversion of differential-drive robot, $D = 0.7$, and $L = 0.4$. The robot and all its neighbors have 0 initial velocity. The method is compared with ORCA [7], CADRL [25] and LSTM-RL [23]. All the methods are test in the same simulated environment with the same number of neighbors. Since the ORCA, CADRL and LSTM-RL are designed for holonomic robots, they are tested with holonomic robots, while the method presented in the article is tested with a differential-drive robot. The holonomic and differential-drive robot have the same size. The evaluation is conducted using three different measure indices, which include: Success rate, which measures the rate of the robot reaching its goal safely in all the tests; Collision rate, which measures the rate of the collisions between the robot and its neighbors in all the tests; and Average navigation time, which captures the average duration it takes the robot to reach its goal in all the tests. All the methods are tested with 500 times.

$$\pi^*(\mathbf{s}_t^{i,n}) = argmaxR(\mathbf{s}_t^{i,n}, \mathbf{a}_t) + \gamma^{\delta t, \mathbf{v}_{pref}} \int_{\mathbf{s}_{t+\delta t}^{i,n}} P(\mathbf{s}_t^{i,n}, \mathbf{a}_t, \mathbf{s}_{t+\delta t}^{i,n}) V^*(\mathbf{s}_{t+\delta t}^{i,n}) d\mathbf{s}_{t+\delta t}^{i,n} \quad (1)$$

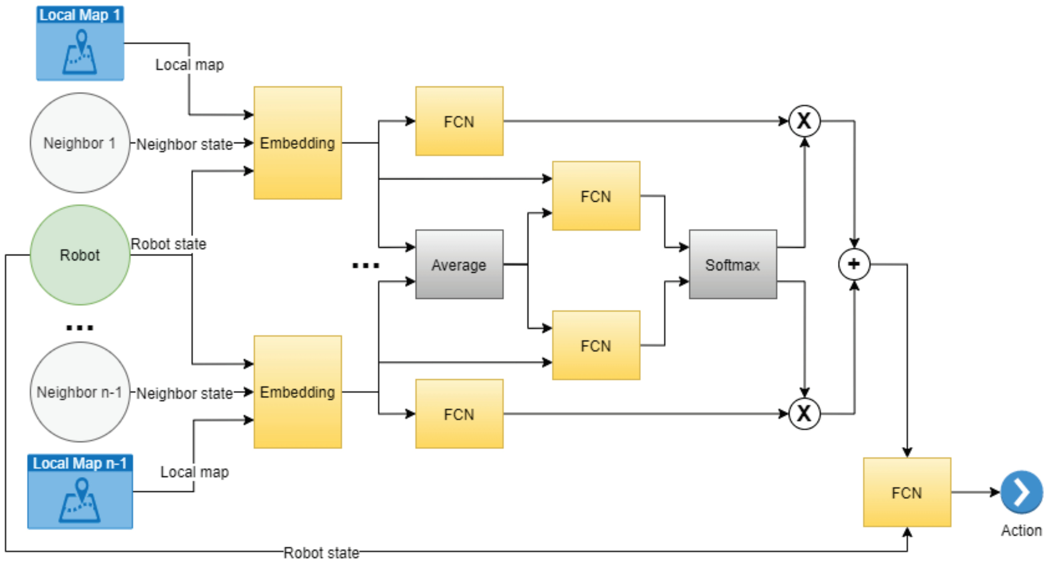

Fig. 1. Architecture of cooperative navigation. FCN means a collection of fully connected layers.

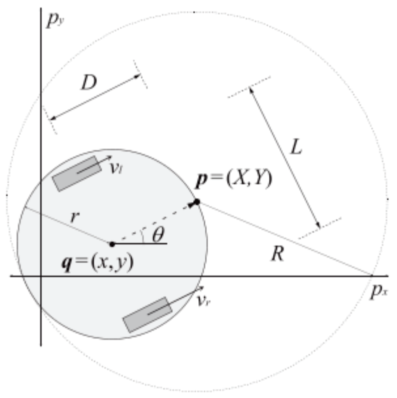

Fig. 2. Conversion of a differential-drive robot to a holonomic robot [19]. The small solid circle represents a differential-drive robot, and the large dotted circle represents the corresponding holonomic robot.

B. Result

Figure 3 depicts one example visulised in one test case at time 4.25 seconds. In all the five neighbors, the neighbor 0 has the highest attention score although it is not the closest neighbor to the robot, since the projection of its instantaneous velocity on the y axis is pointing towards to the robot. Table I shows the comparison result of the navigation policy presented in the article comparing to ORCA [7], CADRL [25] and LSTM-RL [23]. From the table, it can be seen that the method presented in the article can achieve the highest success rate and the lowest collision rate, which means that the robot has the highest chance to reach its goal but with the lowest chance to collide with its neighbors, while it is also noted that the success rate is not 100%, which means that the robot still has very low chance to fail to reach its target. Additionally, the CADRL method has the shortest average navigation time, other than the method presented in the article. It may be due to the method in the article is for differential-drive robots and differential-drive robots cannot move freely in the x and y directions.

TABLE I
QUANTITATIVE RESULTS OF POLICY PERFORMANCE COMPARING TO ORCA [7], CADRL [25] AND LSTM-RL [23]

Method	Success rate	Collision rate	Average navigation time (seconds)
ORCA	0.43	0.57	10.86
CADRL	0.78	0.22	10.80
LSTM-RL	0.95	0.03	11.82
Ours	**0.99**	**0.00**	11.12

V. CONCLUSION

Using deep reinforcement learning, the article presents an approach to address the decision-making problem and enable cooperative navigation of differential-drive mobile robots in crowded environments. To account for the impact of neighboring agents during navigation, the method employs a cooperation module that models robot-agent interactions, as well as the interactions among other agents. The number of neighboring agents around the robot varies over time due to their constant movement, and a self-attention mechanism is

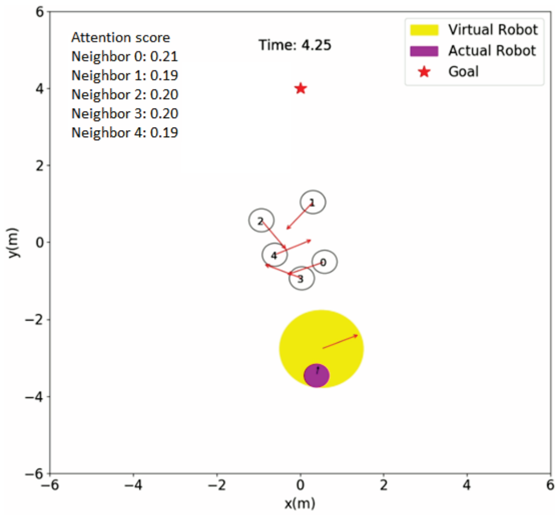

Fig. 3. An example of one test case using the method presented in the article. The differential-drive robot is colored purple, and its corresponding holonomic robot is colored yellow.

utilized to identify and prioritize the interaction features. The method is compared against the ORCA [7], CADRL [25], and LSTM-RL [23], and the findings demonstrate that the method achieves favorable outcomes in terms of high success rate, while also exhibiting low collision rate, suggesting that it is an effective method for guiding robots to their destinations while reducing the risk of collision with neighboring agents.

VI. Acknowledgments

This research is supported by A*STAR under its RIE2025 Manufacturing, Trade and Connectivity (MTC) Industry Alignment Fund- Pre-Positioning (IAF-PP) (Award M21J6a0080). Any opinions, findings, and conclusions, or recommendations expressed in this material are those of the author(s) and do not reflect the views of the A*STAR.

References

[1] Z. Li, A. V. Barenji, J. Jiang, R. Y. Zhong, and G. Xu, “A mechanism for scheduling multi robot intelligent warehouse system face with dynamic demand,” Journal of Intelligent Manufacturing, vol. 31, pp. 469–480, 2020.

[2] P. Fiorini and D. Botturi, “Introducing service robotics to the pharmaceutical industry,” Intelligent Service Robotics, vol. 1, pp. 267–280, 2008.

[3] A. G. Ozkil, Z. Fan, S. Dawids, H. Aanes, J. K. Kristensen and K. H. Christensen, ”Service robots for hospitals: A case study of transportation tasks in a hospital,” 2009 IEEE International Conference on Automation and Logistics, Shenyang, China, 2009, pp. 289-294.

[4] J. P. Van Den Berg and M. H. Overmars, “Prioritized motion planning for multiple robots,” in 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2005, pp. 430–435.

[5] R. Luna and K. E. Bekris, “Push and swap: Fast cooperative path-finding with completeness guarantees,” in IJCAI’11: Proceedings of the Twenty-Second international joint conference on Artificial Intelligence, 2011, pp. 294–300.

[6] J. Yu and D. Rus, “An effective algorithmic framework for near optimal multi-robot path planning,” Robotics Research, vol. 1, pp. 495–511, 2018.

[7] J. Van den Berg, M. Lin, and D. Manocha, “Reciprocal velocity obstacles for real-time multi-agent navigation,” in 2008 IEEE international conference on robotics and automation (ICRA), 2008, pp. 1928–1935.

[8] T. Fan, P. Long, W. Liu, and J. Pan, “Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios,” The International Journal of Robotics Research, vol. 39, no. 7, pp. 856–892, 2020.

[9] W. Wu, S. Bhattacharya, and A. Prorok, “Multi-robot path deconfliction through prioritization by path prospects,” in 2020 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2020, pp. 9809–9815.

[10] 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), IEEE, 2018, pp. 6252–6259.

[11] T. Fan, P. Long, W. Liu, and J. Pan, “Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios,” The International Journal of Robotics Research, vol. 39, no. 7, pp. 856–892, 2020.

[12] A. J. Sathyamoorthy, J. Liang, U. Patel, T. Guan, R. Chandra, and D. Manocha, “Densecavoid: Real-time navigation in dense crowds using anticipatory behaviors,” in 2020 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2020, pp. 11345–11352.

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

[14] P. Long, W. Liu, and J. Pan, “Deep-learned collision avoidance policy for distributed multiagent navigation,” IEEE Robotics and Automation Letters, vol. 2, no. 2, pp. 656–663, 2017.

[15] J. Van den Berg, M. Lin, and D. Manocha, “Reciprocal velocity obstacles for real-time multi-agent navigation,” in 2008 IEEE international conference on robotics and automation, Ieee, 2008, pp. 1928–1935.

[16] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, “Reciprocal n-body collision avoidance,” in Robotics Research: The 14th International Symposium ISRR, Springer, 2011, pp. 3–19.

[17] J. Snape, J. Van Den Berg, S. J. Guy, and D. Manocha, “The hybrid reciprocal velocity obstacle,” IEEE Transactions on Robotics, vol. 27, no. 4, pp. 696–706, 2011.

[18] D. Bareiss and J. Van den Berg, “Generalized reciprocal collision avoidance,” The International Journal of Robotics Research, vol. 34, no. 12, pp. 1501–1514, 2015.

[19] J. Snape, J. Van Den Berg, S. J. Guy, and D. Manocha, “Smooth and collision-free navigation for multiple robots under differential-drive constraints,” in 2010 IEEE/RSJ international conference on intelligent robots and systems, IEEE, 2010, pp. 4584–4589.

[20] G. Sartoretti et al., “Primal: Pathfinding via reinforcement and imitation multi-agent learning,” IEEE Robotics and Automation Letters, vol. 4, no. 3, pp. 2378–2385, 2019.

[21] M. Damani, Z. Luo, E. Wenzel, and G. Sartoretti, “PRIMAL2: Pathfinding via reinforcement and imitation multi-agent learning-lifelong,” IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 2666–2673, 2021.

[22] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, “Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning,” in 2019 international conference on robotics and automation (ICRA), IEEE, 2019, pp. 6015–6022.

[23] M. Everett, Y. F. Chen, and J. P. How, “Motion planning among dynamic, decision-making agents with deep reinforcement learning,” in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2018, pp. 3052–3059.

[24] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, “Self-attention with functional time representation learning,” Advances in neural information processing systems, vol. 32, 2019.

[25] 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), IEEE, 2017, pp. 285–292.

[26] G. Droge, ”Dual-mode dynamic window approach to robot navigation with convergence guarantees,” Journal of Control and Decision, vol. 8, no. 2, pp. 77-88, 2021.

[27] W. Jia, W. Zhao, Z. Song, and Z. Li, ”Object servoing of differential-drive service robots using switched control,” Journal of Control and Decision, pp. 1-12, 2022.