DRL-VO: Learning to Navigate Through Crowded Dynamic Scenes Using Velocity Obstacles

Zhanteng Xie and Philip Dames

Abstract— **This paper proposes a novel learning-based control policy with strong generalizability to new environments that enables a mobile robot to navigate autonomously through spaces filled with both static obstacles and dense crowds of pedestrians. The policy uses a unique combination of input data to generate the desired steering angle and forward velocity: a short history of lidar data, kinematic data about nearby pedestrians, and a sub-goal point. The policy is trained in a reinforcement learning setting using a reward function that contains a novel term based on velocity obstacles to guide the robot to actively avoid pedestrians and move towards the goal. A series of experiments in detailed simulated environments demonstrate the efficacy of this policy, which is able to achieve a higher success rate and a higher average speed than either standard model-based planners or state-of-the-art neural network control policies.**

I. Introduction

One common application of autonomous mobile robots is replacing manual labor to provide last-mile delivery services. For example, delivering sterile supplies and injection medicines to patients in hospitals, delivering materials to various packaging workstations in warehouses, and delivering delicious food or groceries to customers in restaurants and grocery stores [1]–[3]. All these tasks have time limits and require mobile robots to navigate autonomously and quickly to destinations through a partially known space filled with moving people and other static obstacles. The main challenges faced by these mobile robots are perceiving complex environments, especially unknown and dynamic pedestrians; extracting useful information; and generating a policy that yields autonomous navigation.

There is an abundance of studies that focus on robot navigation problems, with solutions ranging from traditional model-based approaches to learning-based approaches (*i.e.,* supervised learning-based approaches and reinforcement learning-based approaches). Typical model-based approaches compute efficient paths and the parameters are easily interpretable, but require manually adjusting model parameters for different scenarios, making them difficult to implement and adapt to new settings [4]–[15]. On the other hand, learning-based methods utilize machine learning techniques to automate these model generation and parameter tuning steps, either in a supervised setting using expert demonstrations or in a reinforcement learning setting using trial and error. Supervised learning-based approaches are purely data-driven and easy to use, but only work well in static or sparse dynamic environments and require laboriously collecting a representative set of expert demonstrations to train networks [16]–[21]. Reinforcement learning-based approaches are experience-driven, similar to human learning, and work in crowded dynamic environments, but typically rely solely on simulated data and require carefully designing a reward function [22]–[38].

Although each type of approach has its own advantages and disadvantages, we choose the deep reinforcement learning (DRL) -based framework to design a crowd-aware navigation control policy to address crowded dynamic navigation because it is difficult to manually design a general model or collect effective training data in uncontrolled and human-filled environments. The reward function design also gives us more freedom to integrate the strengths of model-based and learning-based approaches. In this paper, our primary contribution is designing a novel velocity obstacle (VO)-based reward function, which effectively guides the robot to learn a robust navigation policy with a good balance between collision avoidance and speed. Another key distinction is that we create a novel combination of preprocessed data representations for the input to our neural network-based control policy, using a short history of pooled lidar data, the current kinematics of nearby pedestrians, and a sub-goal point. We demonstrate the efficacy of our approach through a series of simulated experiments, showing that our approach achieves a better balance between collision avoidance and speed, and generalizes to unseen environments and crowd sizes better than state-of-art approaches, including a model-based controller [4], a supervised learning-based approach [21], and two DRL-based approaches [24]. Note that the full version of the paper and more details can be found in [39].

II. Navigation Policy

In this section, we formulate the navigation problem, with a focus on safety and speed through dynamic environments. We then detail our DRL-based approach, describing the preprocessed observation space, the DRL network architecture, and the novel VO-based reward function design.

A. Problem Formulation

In order to autonomous navigate to a goal through a dynamic environment with moving pedestrians, the robot must extract useful information from its sensors and process this information to get the partial observation of the environment, $\mathbf{o}^t$. The robot then uses this partial observation $\mathbf{o}^t$ to compute the suitable steering action $\mathbf{a}^t$ via a control policy π_θ, which

*This work was funded by the Amazon Research Awards Program, NSF grant IIS-1830419, and Temple University.

Zhanteng Xie and Philip Dames are with the Department of Mechanical Engineering, Temple University, Philadelphia, PA, USA {zhanteng.xie, pdames}@temple.edu

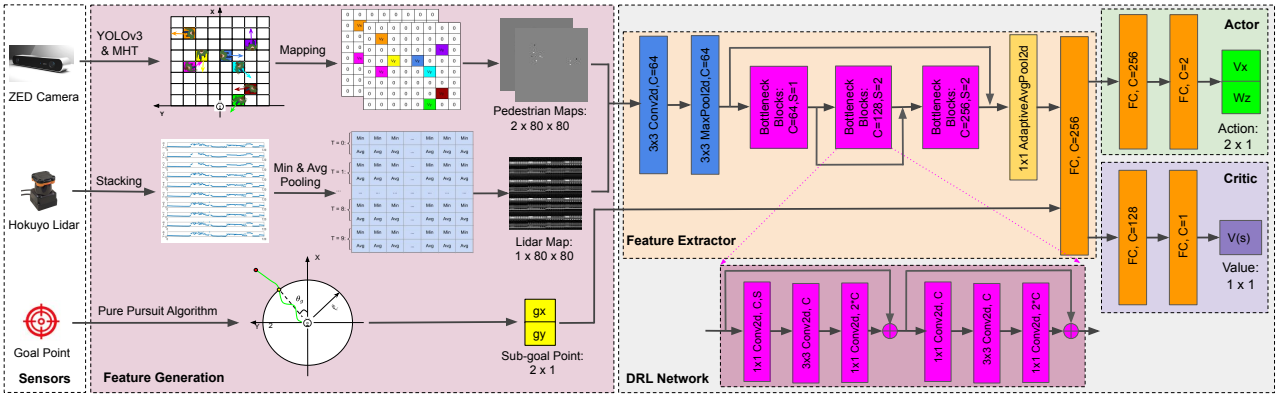


Fig. 1. Overall system architecture of our DRL-VO control policy.

takes the form of a parametric model

$$\mathbf{a}^t \sim \pi_\theta(\mathbf{a}^t|\mathbf{o}^t), \tag{1}$$

where θ are our model parameters. This complicated navigation decision-making process can be formulated as a large partially observable Markov decision process (POMDP). The deep reinforcement learning approach is a typical method to solve such a large POMDP [40].

B. Observation Space

The observation $\mathbf{o}^t = [\mathbf{p}^t, \mathbf{l}^t, \mathbf{g}^t]$ has three components in our control policy: pedestrian kinematics ($\mathbf{p}^t$), lidar history ($\mathbf{l}^t$), and the sub-goal position ($\mathbf{g}^t$). Note, all data in the observation $\mathbf{o}^t$ is expressed in the local robot frame.

1) Pedestrian Kinematics Observation: To extract information about pedestrians, we first feed the raw stereo camera data into the YOLOv3 [41] object detector to obtain bounding boxes and then extract the corresponding points from the 3-D point cloud to measure the pedestrian positions. These position measurements are fed into a multiple hypothesis tracker (MHT) [42], which performs data association to yield a collection of target tracks containing relative position and velocity information. Finally, we encode the pedestrian kinematics into occupancy grid-style maps specifically designed for our network architecture, as the ZED Camera track in the Feature Generation block of Fig. 1 shows.

2) Lidar History Observation: Most initial works using learning-based policies for robot navigation used the entire lidar scan message. However, our early fusion architecture requires that lidar feature map to be the same size as the pedestrian feature maps. Thus, we need to convert the lidar data into an 80×80 feature map. Motivated by the lidar data downsampling operation from [43], we use a combination of minimum pooling and average pooling to extract two separate distance measurements per scan region, as the Hokuyo Lidar track in the Feature Generation block of Fig. 1 shows, a process we found to yield the best performance.

3) Goal Position Observation: With the observation data from sensors, the robot also needs to know where the goal position is. Instead of feeding the final goal point to our DRL network, we choose the sub-goal point from a nominal path as our goal position observation. We use the pure pursuit algorithm [44] to extract this sub-goal point, as the Goal Point track in the Feature Generation block of Fig. 1 shows.

C. Action Space

The action $\mathbf{a}^t = [v_x^t, w_z^t]$ of our DRL control policy are the steering velocities, where v_x^t is the translational velocity and w_z^t is the rotational velocity in the robot's local coordinate frame. Note: The action space is a continuous space. The range of the translational velocity v_x^t is set to $[0, 0.5]$, and the range of the rotational velocity w_z^t is set to $[-2, 2]$.

D. Network Architecture

The Feature Extractor network is identical to the backbone CNN network of our previous work [21], which fuses the lidar historical observation $\mathbf{l}^t$ and pedestrian kinematics observation $\mathbf{p}^t$. We use the PPO algorithm to train our DRL network, and use Adam optimizer [45], a stochastic gradient descent method, to find the optimal model parameters θ^*.

E. Reward Function

Navigation has two competing objectives, we want the robot to move as quickly to reach the goal in minimum time but also safely to avoid colliding with any stationary objects or moving pedestrians. Thus, we design a multiobjective reward function:

$$r^t = r_g^t + r_c^t + r_w^t + r_d^t \tag{2}$$

where r_g^t making progress towards the goal, r_c^t penalizes passively approaching or colliding with an obstacle, r_w^t penalizes rapid changes in direction, and r_d^t rewards actively steering to avoid obstacles and point towards the sub-goal.

1) Reaching the Goal: The reward is given by

$$r_g^t = \begin{cases} r_{\text{goal}} & \text{if } \|p_g^t\|) < g_m \\ -r_{\text{goal}} & \text{else if } t \geq t_{\max} \\ r_{\text{path}}(\|p_g^{t-1}\| - \|p_g^t\|) & \text{otherwise,} \end{cases} \tag{3}$$

where p_g^t is the goal position (in the robot's frame) at time t. We use $r_{\text{goal}} = 20$, $r_{\text{path}} = 3.2$, $g_m = 0.3\,\text{m}$, and $t_{\max} = 25\,\text{s}$.

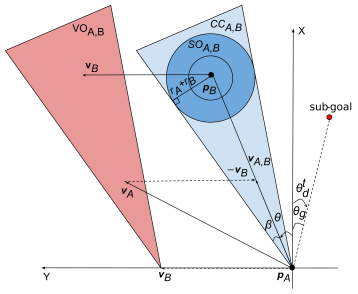


Fig. 2. The geometric pictogram for velocity obstacle $VO_{A,B}$, collision cone $CC_{A,B}$, and special occupancy $SO_{A,B}$.

2) Passive Collision Avoidance: The reward is given by

$$r_c^t = \begin{cases} r_{\text{collision}} & \text{if } \|p_o^t\| \leq d_r \\ r_{\text{obstacle}}(d_m - \|p_o^t\|) & \text{else if } \|p_o^t\| \leq d_m \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where p_o^t is obstacle position at time t. We use $r_{\text{collision}} = -20$, $r_{\text{obstacle}} = -0.2$, $d_r = 0.3\,\text{m}$, and $d_m = 1.2\,\text{m}$.

3) Path Smoothness: The reward is given by

$$r_w^t = \begin{cases} r_{\text{rotation}}|\omega_z^t| & \text{if } |\omega_z^t| > \omega_m \\ 0 & \text{otherwise,} \end{cases} \tag{5}$$

where $r_{\text{rotation}} = -0.1$ and $\omega_m = 1\,\text{rad/s}$.

4) Active Heading Direction: The reward is given by

$$r_d^t = r_{\text{angle}}(\theta_m - |\theta_d^t|), \tag{6}$$

where θ_d^t is the desired heading direction in the robot's local frame and θ_m is the maximum allowable deviation of the heading direction. We use $r_{\text{angle}} = 0.6$ and $\theta_m = \frac{\pi}{6}\,\text{rad}$.

The key to this reward term is to find the desired direction that moves towards the goal while also being collision free. To do this, we extend the concept of velocity obstacles [5], which create a collision cone $CC_{A,B}$ containing any relative velocities $\mathbf{v}_{A,B} \in CC_{A,B}$ that will cause a collision at future time, as Fig. 2 shows. Using the collision cone $CC_{A,B}$, the robot can know which heading direction angles will cause collisions with all of the moving pedestrians tracked by MHT. The robot then uses these collision cones and the direction of the sub-goal (θ_{sg}) to find the desired heading direction angle θ_d^t, as Algorithm 1 shows. This sampling-based search algorithm is motivated by [6].

III. Results

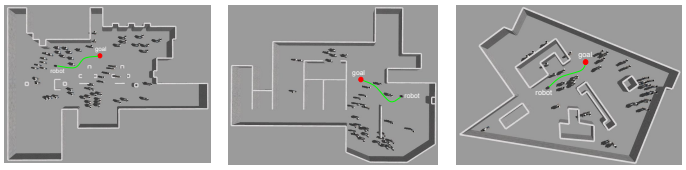
To demonstrate the efficacy and performance of our proposed control policy, we first design a 3D human-robot interaction simulator using the Gazebo simulator [46] and the PEDSIM library [47] and then conduct a set of 3D simulated experiments. This section describes this setup in greater detail, presents the procedure we used to train our network, and compares the results to other methods.

Algorithm 1: Search desired direction angle

Input: Sub-goal direction angle θ_g, pedestrians from MHT B_{peds}, robot linear velocity $\mathbf{v}_{A_x}$, number of samples N

Output: Optimal direction angle θ_d^t

```
initialize: θ_d^t ← π/2
if B_peds ≠ ∅ then
    θ_min ← ∞
    for i = 1, 2, ..., N do
        θ_u ← sample from [−π, π]
        free ← True
        for B in B_peds do
            θ_{v_{A,B}} ← atan2((v_{A_x} sin(θ_u) − v_{B_y}) / (v_{A_x} cos(θ_u) − v_{B_x}))
            if θ_{v_{A,B}} ∈ [θ − β, θ + β] then
                free ← False
                break
        if free then
            if ||θ_u − θ_g|| < θ_min then
                θ_min ← ||θ_u − θ_g||
                θ_d^t ← θ_u
else
    θ_d^t ← θ_g
return θ_d^t
```



(a) Lobby (b) Cumberland (c) Freiburg

Fig. 3. Gazebo simulation environments.

A. Simulation Configuration

The robot model is a Kobuki Turtlebot 2, which has a maximum velocity of 0.5 m/s, equipped with a Hokuyo UTM-30LX lidar and a ZED stereo camera. The Hokuyo lidar has a maximum range of 30 m, a FOV of 270°, and an angular resolution of 0.25° while the ZED camera has a minimum range of 0.3 m, a maximum range of 20 m, and a FOV of 90°.

B. Training Procedure

We use stable-baselines3 framework [48] to implement our DRL network. To train a control policy, we use the Lobby environment (Fig. 3(a)) with 34 pedestrians in it. The robot is then repeatedly assigned to reach a random goal from a random start position within the free space of the map. We used this procedure to train two different DRL-based control policies, one using the full reward (2) (DRL-VO) and one that does not use the heading direction reward, r_d^t (DRL).

TABLE I

NAVIGATION RESULTS AT DIFFERENT CROWD DENSITIES AND UNSEEN ENVIRONMENTS

Environment	Method	Success Rate	Average Time (s)	Average Length (m)	Average Speed (m/s)
Lobby world, 35 pedestrians	DWA [4]	0.82	14.18	**5.15**	0.36
	CNN [21]	0.81	14.30	5.40	0.38
	A1-RD [24]	0.86	17.82	6.06	0.34
	A1-RC [24]	0.77	16.81	6.89	0.41
	DRL	0.75	14.30	6.46	0.45
	DRL-VO	**0.88**	**11.42**	5.31	**0.46**
Lobby world, 45 pedestrians	DWA [4]	0.77	15.39	**5.16**	0.34
	CNN [21]	0.79	16.65	5.62	0.34
	A1-RD [24]	0.76	23.16	6.61	0.29
	A1-RC [24]	0.77	14.65	6.28	0.43
	DRL	0.69	13.96	6.41	0.46
	DRL-VO	**0.81**	**11.65**	5.37	**0.46**
Cumberland world, 35 pedestrians	DWA [4]	0.74	16.28	**5.57**	0.34
	CNN [21]	0.60	24.25	7.08	0.29
	A1-RD [24]	**0.88**	18.04	6.69	0.37
	A1-RC [24]	0.77	15.37	6.66	0.43
	DRL	0.56	15.79	6.97	0.44
	DRL-VO	0.78	**12.62**	5.84	**0.46**
Freiburg world, 35 pedestrians	DWA [4]	0.70	18.15	**5.78**	0.32
	CNN [21]	0.57	20.09	6.66	0.33
	A1-RD [24]	-	-	-	-
	A1-RC [24]	0.65	17.11	6.93	0.41
	DRL	0.39	18.21	7.61	0.42
	DRL-VO	**0.76**	**13.37**	6.16	**0.46**

C. Navigation Results

We test these two DRL-based policies (*i.e.,* DRL and DRL-VO), along with other's DRL-based policies [24] (*i.e.,* A1-RD and A1-RC), the CNN-based policy [21], and the DWA planner [4]. To test the ability of the control policy to generalize to new environments and crowd densities, we use the following environments (maps shown in Fig. 3):

- **Lobby with 35 and 45 pedestrians**: test generalization across different crowd densities.
- **Cumberland and Freiburg with 35 pedestrians**: test generalization to unseen environments.

We then compare the performance of the control policies using the following metrics, which are commonly used in the autonomous navigation literature [23], [31], [35]:

- **Success rate**: the fraction of collision-free trials.
- **Average time**: the average travel time of trials.
- **Average length**: the average trajectory length of trials.
- **Average speed**: the average speed during trials.

For each combination of environment and control policy, we run 4 tests from the same initial conditions, where each test consists of the robot navigating through the same sequence of 25 goal points. Although the initial conditions of each test are the same, each trial yields different navigation behavior due to sensor noise, social force interactions, etc.

Table I summarizes our results, where we observe three key phenomena. First, our DRL-VO policy has a much higher success rate than our DRL policy in each crowd size, while having a similar average speed. This shows that the proposed VO-based heading direction reward is beneficial and plays a key role in enabling the robot to maintain a good balance between collision avoidance and speed. Second, our DRL-VO policy has the highest success rate in every situation (except for the Cumberland with 35 pedestrians), with the largest advantage in the crowded environments. This indicates that our policy has strong generalizability to different crowd sizes and different unseen environments. Third, the average speed of the robot using our DRL-VO policy remains nearly constant across all situations, regardless of environment or crowd density, allowing it to reach the goal most quickly.

IV. CONCLUSION

In this paper, we proposed the DRL-VO control policy to enable autonomous robot navigation in crowded dynamic environments. Our approach differs from prior research in two key ways. First, we propose a new combination of preprocessed data representations, which can work well in crowded dynamic environments and bridge the appearance gap between an imperfect simulation and reality. Specifically, the robot fuses a short history of lidar data, current pedestrian kinematics, and a sub-goal point. All of this data is tracked in the robot's local frame, making it robust to errors in localization that are common in crowded, dynamic environments. Second, we design a novel reward function based on velocity obstacles, which we show to significantly reduce the collision rate and maintain a constant speed. We demonstrate that our DRL-VO policy generalizes better and maintains a better balance between collision avoidance and speed to different crowd sizes and different unseen environments than other state-of-the-art model-based and learning-based policies.

ACKNOWLEDGMENT

This research includes calculations carried out on HPC resources supported in part by the National Science Foundation through major research instrumentation grant number 1625061 and by the US Army Research Laboratory under contract number W911NF-16-2-0189.

REFERENCES

[1] J.-u. Kim, "Keimyung hospital demonstrates smart autonomous mobile robot," https://www.koreabiomed.com/news/articleView.html?idxno=10585, Mar 2021, (Accessed on 08/24/2021).

[2] J. Blyler, "One big 2020 robot trend that's hard to miss," https://www.designnews.com/automation/2020-robot-trend-could-explode-2021-and-beyond, Dec 2020, (Accessed on 08/24/2021).

[3] B. Marr, "Demand for these autonomous delivery robots is skyrocketing during this pandemic," https://www.forbes.com/sites/bernardmarr/2020/05/29/demand-for-these-autonomous-delivery-robots-is-skyrocketing-during-this-pandemic/, May 2020, (Accessed on 08/24/2021).

[4] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23–33, 1997.

[5] 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.

[6] D. Wilkie, J. Van Den Berg, and D. Manocha, "Generalized velocity obstacles," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 5573–5578.

[7] J. Van Den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics research*. Springer, 2011, pp. 3–19.

[8] S. H. Arul and D. Manocha, "V-rvo: Decentralized multi-agent collision avoidance using voronoi diagrams and reciprocal velocity obstacles," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 8097–8104.

[9] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, "Model predictive contouring control for collision avoidance in unstructured dynamic environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4459–4466, 2019.

[10] X. Shen, E. L. Zhu, Y. R. Stürz, and F. Borrelli, "Collision avoidance in tightly-constrained environments without coordination: a hierarchical control approach," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2674–2680.

[11] P. T. Singamaneni, A. Favier, and R. Alami, "Human-aware navigation planner for diverse human-robot interaction contexts," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5817–5824.

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

[13] G. Ferrer, A. Garrell, and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1688–1694.

[14] M. Sebastian, S. B. Banisetty, and D. Feil-Seifer, "Socially-aware navigation planner using models of human-human interaction," in *IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2017, pp. 405–410.

[15] M. Boldrer, M. Andreetto, S. Divan, L. Palopoli, and D. Fontanelli, "Socially-aware reactive obstacle avoidance strategy based on limit cycle," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3251–3258, 2020.

[16] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 1527–1533.

[17] L. Tai, S. Li, and M. Liu, "A deep-network solution towards model-less obstacle avoidance," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 2759–2764.

[18] A. Loquercio, A. I. Maqueda, C. R. Del-Blanco, and D. Scaramuzza, "DroNet: Learning to fly by driving," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1088–1095, 2018.

[19] G. Kahn, P. Abbeel, and S. Levine, "Badgr: An autonomous self-supervised learning-based navigation system," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1312–1319, 2021.

[20] A. Pokle, R. Martín-Martín, P. Goebel, V. Chow, H. M. Ewald, J. Yang, Z. Wang, A. Sadeghian, D. Sadigh, S. Savarese *et al.*, "Deep local trajectory replanning and control for robot navigation," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 5815–5822.

[21] Z. Xie, P. Xin, and P. Dames, "Towards safe navigation through crowded dynamic environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sep. 2021, accepted.

[22] 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.

[23] 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.

[24] R. Guldenring, M. Görner, N. Hendrich, N. J. Jacobsen, and J. Zhang, "Learning local planners for human-aware navigation in indoor environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 6053–6060.

[25] C. Pérez-D'Arpino, C. Liu, P. Goebel, R. Martín-Martín, and S. Savarese, "Robot navigation in constrained pedestrian environments using reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 1140–1146.

[26] X. Huang, H. Deng, W. Zhang, R. Song, and Y. Li, "Towards multi-modal perception-based navigation: A deep reinforcement learning method," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4986–4993, 2021.

[27] 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.

[28] 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.

[29] ——, "Collision avoidance in pedestrian-rich environments with deep reinforcement learning," *IEEE Access*, vol. 9, pp. 10 357–10 377, 2021.

[30] 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.

[31] 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 Robotics and Automation Letters*, vol. 5, no. 2, pp. 2754–2761, 2020.

[32] L. Liu, D. Dugas, G. Cesari, R. Siegwart, and R. Dubé, "Robot navigation in crowded environments using deep reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[33] S. Liu, P. Chang, W. Liang, N. Chakraborty, and K. Driggs-Campbell, "Decentralized structural-rnn for robot crowd navigation with deep reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3517–3524.

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

[35] 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. 11 345–11 352.

[36] D. Dugas, J. Nieto, R. Siegwart, and J. J. Chung, "Navrep: Unsupervised representations for reinforcement learning of robot navigation in dynamic human environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7829–7835.

[37] P. Xu and I. Karamouzas, "Human-inspired multi-agent navigation using knowledge distillation," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 8105–8112.

[38] 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 *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6057–6063.

[39] Z. Xie and P. Dames, "Drl-vo: Learning to navigate through crowded dynamic scenes using velocity obstacles," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2700–2719, 2023.

[40] A. Gosavi, "Reinforcement learning: A tutorial survey and recent

advances," *INFORMS Journal on Computing*, vol. 21, no. 2, pp. 178–192, 2009.

[41] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[42] K. Yoon, Y.-m. Song, and M. Jeon, "Multiple hypothesis tracking algorithm for multi-target multi-camera tracking with disjoint views," *IET Image Processing*, vol. 12, no. 7, pp. 1175–1184, 2018.

[43] M. Pfeiffer, S. Shukla, M. Turchetta, C. Cadena, A. Krause, R. Siegwart, and J. Nieto, "Reinforced imitation: Sample efficient deep reinforcement learning for mapless navigation by leveraging prior demonstrations," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4423–4430, 2018.

[44] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Carnegie-Mellon UNIV Pittsburgh PA Robotics INST, Tech. Rep., 1992.

[45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[46] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, 2004, pp. 2149–2154.

[47] C. Gloor, "PEDSIM: Pedestrian crowd simulation," *URL http://pedsim.silmaril.org*, vol. 5, no. 1, 2016.

[48] A. Raffin, A. Hill, M. Ernestus, A. Gleave, A. Kanervisto, and N. Dormann, "Stable baselines3," https://github.com/DLR-RM/stable-baselines3, 2019.