CrowdMove: Autonomous Mapless Navigation in Crowded Scenarios

Tingxiang Fan[1], Xinjing Cheng[1], Jia Pan[2], Dinesh Manocha[3] and Ruigang Yang[1]

Abstract— **Navigation is an essential capability for mobile robots. In this paper, we propose a generalized yet effective 3M (i.e., multi-robot, multi-scenario, and multi-stage) training framework. We optimize a mapless navigation policy with a robust policy gradient algorithm. Our method enables different types of mobile platforms to navigate safely in complex and highly dynamic environments, such as pedestrian crowds. To demonstrate the superiority of our method, we test our methods with four kinds of mobile platforms in four scenarios. Videos are available at https://sites.google.com/view/crowdmove**

I. INTRODUCTION

Safe and efficient navigation in highly dynamic unstructured environments remains an open problem in robotics [1], [2]. As a result, the mobility of robots nowadays is still limited in a crowded pedestrian scenarios, which greatly limits the mobile robot's application in many tasks, including the restaurant delivery and the surveillance.

Most existing robotic navigation approaches consist of two parts. First, a map is built online/offline using simultaneous localization and mapping (SLAM). Next, a collision-free trajectory is planned with respect to the map [3]. However, in an environment full of moving obstacles like pedestrians, SLAM may fail frequently due to occlusions or noises and may not be able to provide a reliable map. More importantly, a map about surrounding environment often is not necessary for a robust collision avoidance policy, since the collision avoidance is a reactive behavior that only requires a rough sensing about the position and velocity of nearby obstacles. In addition, the map construction is expensive and thus SLAM is not applicable in many mobile robotic platforms with limited computational resources.

Given the above limitations, mapless navigation approaches have been proposed recently, in which robots do not rely on the prior knowledge of surrounding environments [4], [5]. For the mapless navigation, the localization service is no longer provided by the SLAM, but by other low-cost solutions such as the Global Position System (GPS) for outdoor scenes and the Ultra Wide Band (UWB) technique for indoor scenes [6]. To develop the collision avoidance

capability, some manual rules such as artificial potential field (APF) [7] have been proposed. However, these hand-crafted rules is limited in generalization and the collision avoidance performance is sensitive to the rules' hyper-parameters.

Recently, deep learning based mapless navigation approaches have gained attention, which directly map sensor perception to steering commands [8], [9]. Unlike rule-based approaches, learning-based navigation methods are optimized over a large number of training data and thus no longer require the manual tuning of hyper-parameters and rules. Robot-learning schemes could be divided into two categories: the imitation learning and the reinforcement learning [10], [11]. In imitation learning, robots learn optimal policies by imitating demonstrations collected from human experts [12]. In reinforcement learning, the optimal policies are learned from the training data generated during the interaction between robots and the environment [13].

In this paper, we focus on the second category, i.e., the reinforcement learning, to address the problem of safe and efficient navigation in crowds. In our approach, the local planner is modeled by a deep neural network, which transfers raw sensor inputs to a collision-free steering command. Thanks to its excellent generalization, the local planner trained in a simulator can be deployed in the real world without tedious fine-tuning.

Main contributions: To learn a robust mapless local planner, we propose a novel 3M (i.e., multi-robot, multi-scenario, and multi-stage) training framework, which exploits a robust policy gradient based reinforcement learning algorithm that is trained in a large-scale robot system in a set of complex environments. We demonstrate that the collision avoidance policy learned from our approach can find collision-free paths for nonholonomic robots and can be generalized effectively to various scenarios and different mobile platforms, as shown in Figure 1.

II. RELATED WORKS

Collision avoidance is an essential capability for mobile robotic systems. Standard rule-based solutions to the collision-free navigation have two steps: first, the environment is modeled as an energy function, and then a collision-free path is computed as an optimization problem for reaching the destination [14]. These approaches benefit from the independent development of mapping-based localization and the motion planning [15], [16]. The major limitation of rule-based methods is that the energy functions are always hand-crafted, meaning that they need a lot of expertise and may

[1]Tingxiang Fan, Xinjing Cheng and Ruigang Yang are with Researcher of Robotics and Auto-driving Lab, Baidu Research, Baidu Inc., China v_fantingxiang@baidu.com, chengxinjing@baidu.com, yangruigang@baidu.com

[2]Jia Pan is with the Department of Mechanical and Biomedical Engineering, City University of Hong Kong, Hong Kong, China jiapan@cityu.edu.hk

[3]Dinesh Manocha is with Department of Computer Science, University of Maryland, College Park, USA dm@cs.umd.edu
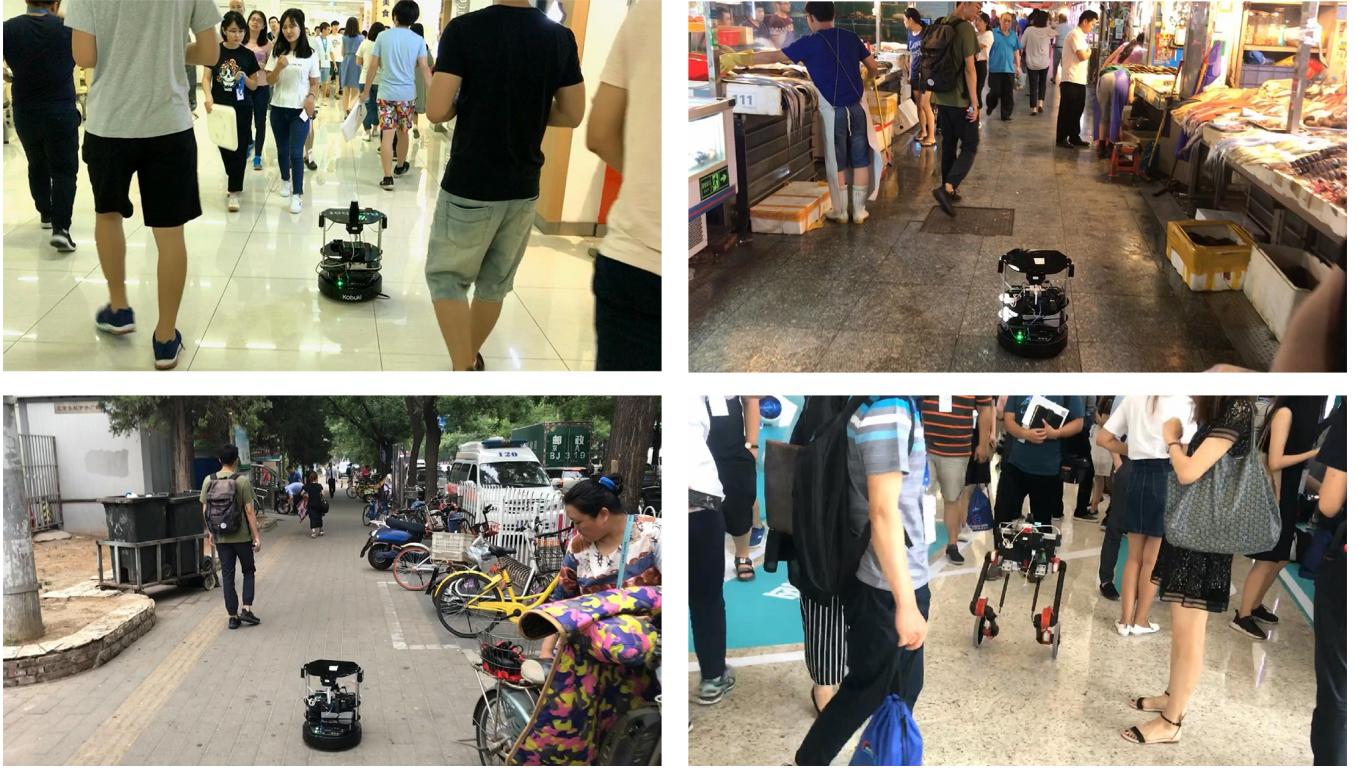
Fig. 1: Mapless navigation in complex and highly dynamic environments using different mobile platforms.

fail in complex and highly dynamic scenarios.

Learning-based collision avoidance techniques in which one robot avoids static obstacles have been studied extensively. Many approaches adopt the supervised learning paradigm to train a collision avoidance policy by imitating a dataset of sensor input and motion commands. Muller et al. [17] trained a vision-based static obstacle avoidance system in supervised mode for a mobile robot by training a 6-layer convolutional network which maps raw input images to steering angles. Sergeant et al. [18] proposed a mobile robot control system based on multimodal deep autoencoders. Ross et al. [19] trained a discrete controller for a small quadrotor helicopter with imitation learning techniques. The quadrotor was able to successfully avoid collisions with static obstacles in the environment using only a single cheap camera. Only discrete movements (left/right) must be learned and the robot need only be trained within static obstacles. Note that the aforementioned approaches only take into account static obstacles and require a human driver to collect training data in a wide variety of environments. Another data-driven end-to-end motion planner is presented by Pfeiffer et al. [20]. They trained a model to map laser range findings and target positions to motion commands using expert demonstrations generated by the ROS navigation package. This model can navigate the robot through a previously unseen environment and ensure that it successfully reacts to sudden changes. Nonetheless, like the other supervised learning methods, the performance of the learned policy is severely constrained by the quality of the labeled training sets. To overcome this limitation, Tai et al. [8] proposed a mapless motion planner trained through a deep reinforcement learning method. Kahn et al. [21] presented an uncertainty-aware model-based reinforcement learning algorithm to estimate the probability of collision in an priori unknown environment. However, the test environments are relatively simple and structured, and the learned planner is hard to generalize to scenarios with dynamic obstacles and other proactive agents. To address highly dynamic unstructured environments, some researches which applied decentralized multi-robot navigation approaches, are proposed recently. Godoy et al. [22] introduced Bayesian inference approach to predict surrounding dynamic obstacles and computed collision-free command through ORCA framework [23]. Chen et al. [24]–[26] proposed multi-robot collision avoidance policies by deep reinforcement learning, which also required deploying multiple sensors to estimate the states of nearby agents and moving obstacles. However, complex pipeline of these navigation approaches not only requires expensive online computation but makes the whole system less robust to the perception uncertainty.

III. APPROACH

The approach in this paper stems from our previous work [27]. Here we provide a brief description of this approach in terms of the key ingredients of our reinforcement learning framework, network structure and the training

procedure.

A. Reinforcement Learning Setup

In reinforcement learning, the environment is typically formulated as a Markov Decision Process(MDP) which can be described as a 4-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}$ is the state-transition model, $\mathcal{R}$ is the reward function. Below we describe the details of the state space, the action space, and the reward function.

1) State space: The state $\mathbf{s}^t$ consists of the readings of the 2D laser range finder $\mathbf{s}_z^t$, the relative goal position $\mathbf{s}_g^t$ and robot's current velocity $\mathbf{s}_v^t$. Specifically, $\mathbf{s}_z^t$ includes the measurements of the last three consecutive frames from a 180-degree laser scanner which has a maximum range of 4 meters and provides 512 distance values per scanning (i.e. $\mathbf{s}_z^t \in \mathbb{R}^{3 \times 512}$). The relative goal position $\mathbf{s}_g^t$ is a 2D vector representing the goal in polar coordinate (distance and angle) with respect to the robot's current position. The observed velocity $\mathbf{s}_v^t$ includes the current translational and rotational velocity of the differential-driven robot. The observations are normalized by subtracting the mean and dividing by the standard deviation using the statistics aggregated over the course of the entire training.

2) Action space: The action space is a set of permissible velocities in continuous space. The action of differential robot includes the translational and rotational velocity, i.e. $\mathbf{a}^t = [v^t, w^t]$. In this work, considering the real robot's kinematics and the real world applications, we set the range of the translational velocity $v \in (0.0, 1.0)$ and the rotational velocity in $w \in (-1.0, 1.0)$. Note that backward moving (i.e. $v < 0.0$) is not allowed since the laser range finder cannot cover the back area of the robot.

3) Reward design: Our objective is to avoid collisions during navigation and minimize the arrival time. A reward function is designed to guide robots to achieve this objective:

$$r^t = (^g r)^t + (^c r)^t + (^w r)^t. \tag{1}$$

The reward r received at timestep t is a sum of three terms, $^g r$, $^c r$, and $^w r$. In particular, the robot is awarded by $(^g r)^t$ for reaching its goal:

$$(^g r)^t = \begin{cases} r_{arrival} & \text{if } \|\mathbf{p}^t - \mathbf{g}\| < 0.1 \\ \omega_g(\|\mathbf{p}^{t-1} - \mathbf{g}\| - \|\mathbf{p}^t - \mathbf{g}\|) & \text{otherwise.} \end{cases} \tag{2}$$

When the robot collides with other robots or obstacles in the environment, it is penalized by $(^c r)^t$:

$$(^c r)^t = \begin{cases} r_{collision} & \text{if } collision \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

To encourage the robot to move smoothly, a small penalty $(^w r)^t$ is introduced to punish the large rotational velocities:

$$(^w r)^t = \omega_w |w^t| \qquad \text{if } |w^t| > 0.7. \tag{4}$$

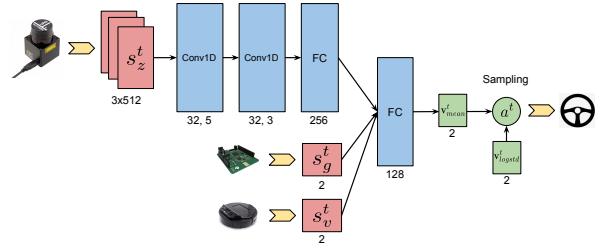We set $r_{arrival} = 15$, $\omega_g = 2.5$, $r_{collision} = -15$ and $\omega_w = -0.1$ in the training procedure.

Fig. 2: The architecture of the collision avoidance neural network. The network has the scan measurements $\mathbf{s}_z^t$, relative goal position $\mathbf{s}_g^t$ and current velocity $\mathbf{s}_v^t$ as inputs, and outputs the mean velocity $\mathbf{v}_{mean}^t$.

B. Network architecture

We design a 4-hidden-layer neural network as a non-linear function approximator to the policy π_θ. Its architecture is shown in Figure 2. The neural network maps the input state vector $\mathbf{o}^t$ to a vector $\mathbf{v}_{mean}^t$. The final action $\mathbf{a}^t$ is sampled from a Gaussian distribution $\mathcal{N}(\mathbf{v}_{mean}^t, \mathbf{v}_{logstd}^t)$, where $\mathbf{v}_{mean}^t$ serves as the mean and $\mathbf{v}_{logstd}^t$ refers to a log standard deviation which will be updated solely during training.

C. Multi-robot, multi-scenario, and multi-stage training

1) Training algorithm: We extend the state-of-the-art reinforcement learning algorithm, Proximal Policy Optimization (PPO) [28]–[30], to our parallel multi-robot training framework. The policy is trained with experiences collected by all robots in parallel. The parallel multi-robot training framework not only dramatically reduces the time cost of the sample collection but also makes the algorithm suitable for training many robots in various scenarios.

2) Training scenarios: To expose our robots to diverse environments, we create different scenarios with a variety of obstacles using the Stage mobile robot simulator[1] (as shown in Figure 3) and move all the robots concurrently. These rich, complex training scenarios enable robots to explore state space efficiently and are likely to improve the quality and robustness of the learned policy. Combined with the parallel *multi-robot* training algorithm, the collision avoidance policy is effectively optimized at each iteration over a variety of environments.

3) Training stages: Although training in multiple environments simultaneously enables robust performance in a variety of different test cases, it makes the training process harder. Inspired by the curriculum learning paradigm [31], we propose a two-stage training process, which accelerates the policy's convergence to a satisfying solution, and gets higher rewards than if the policy had been trained from scratch with the same number of epochs (as shown in Figure 4). In the first stage, we only train 20 robots on the random scenarios (scenario 7 in Figure 3) without any obstacles. Once the

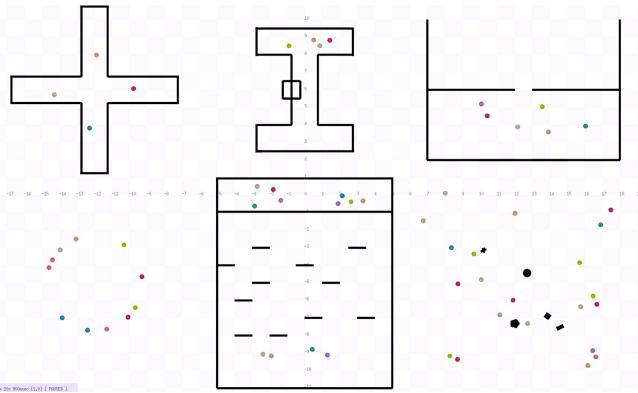[1]http://rtv.github.io/Stage/

Fig. 3: Scenarios used to train the collision avoidance policy. All robots are modeled as discs with the same radii. Obstacles are shown in black.
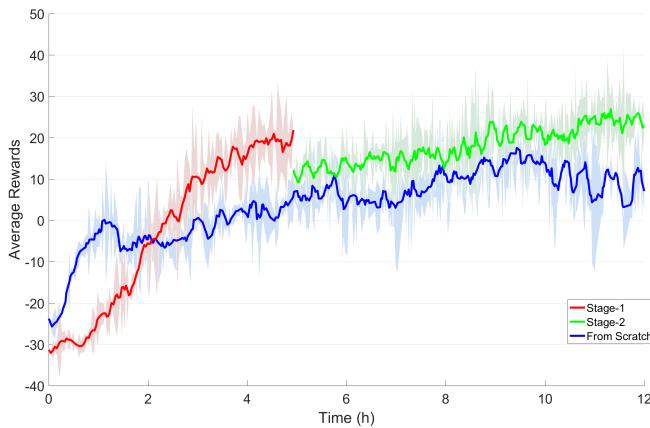

Fig. 4: Average rewards shown in wall time during the training process.

robots achieve reliable levels of performance, we stop Stage 1 and save the trained policy. The policy will continue to be updated in Stage 2, where the number of robots increases to 58, and they are trained on the richer and more complex scenarios shown in Figure 3.

IV. EXPERIMENTS AND RESULTS

In this section, we first describe the hardware setup of our robots. Then, we deploy our learning-based local planner to a Turtlebot platform and test the performance in various crowd scenarios. Finally, to validate the transferability of our model, we apply the collision avoidance policy on different mobile platforms.

A. Hardware setup

We now introduce the sensor kits and the mobile platforms used in our experiments. The sensor kits provide the input to the collision avoidance network, and the mobile platforms execute the steering command output from the collision avoidance network.

(a) Turtlebot

(b) Igor robot

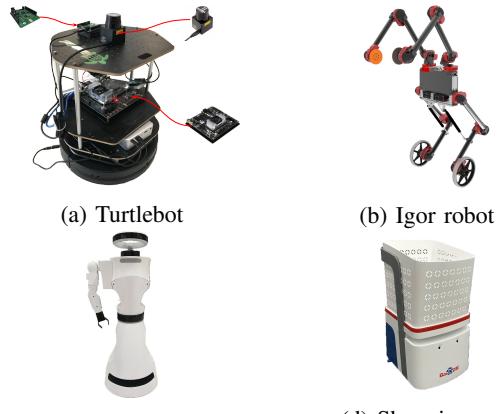(c) Human-like service robot

(d) Shopping cart

Fig. 5: Four different mobile robots used in our experiments. (a) shows the sensor kit installed in the Turtlebot. (b)(c)(d) show another three types of mobile platforms boarded with the same collision avoidance algorithm.

1) Sensor kits: We used the Hokuyo urg-04lx 2D LiDAR as our laser scanner, the Pozyx Ultra-WideBand(UWB) modules as the indoor localization system, and the Nvidia Jetson TX1 as our onboard computer. In this way, the cost of the sensor kit is economical.

2) Mobile platforms: We expect our policy to have a potential applicability to different mobile robots without being retrained. To analyze the transferability of our local planner from the simulation to the real-world, four different mobile platforms have been tested in the experiments (as shown in Figure 5).

In our experiments, to force the robot to encounter pedestrians, we let a person take the UWB localization tag, and then the robot can follow the target person according to the UWB signal. In this case, the target person can control the robot's moving directions and guide it to run through a dense pedestrian crowd.

B. Tests in different scenarios

We deployed our local planner on the Turtlebot and tested it in different environments to test the robustness of our collision avoidance policy in different scenarios. In particular, we let Turtlebot run in the canteen, the food market and the outdoor street because these environments provided numerous obstacles that never appear in simulation. In addition, the turtlebot (Figure 5a) is larger than the robot for which we trained the model in the simulation.

The experiments demonstrate that the Turtlebot can always avoid pedestrians and other static obstacles (Figure 6), although many people are curious about the robot and actively block the robot.

C. Tests on different robots

In testing different robots, we deployed our collision avoidance policy on different mobile platforms without retraining and fine-tuning. Noted that these robots have

Fig. 6: Turtlebot running in highly dynamic unstructured scenarios.

different shapes, sizes and dynamics, but our learning-based policy can adapt well to these differences.

The experiments demonstrate that, although there are huge differences between the physcial mobile platforms and the robots in simulation, the physical robots can still avoid the static obstacles and dynamic pedestrians reliably, as shown in Figure 7 and 8.

V. CONCLUSION

This work presents a multi-robot, multi-scenario, and multi-stage training framework to optimize a mapless navigation policy with a robust policy gradient algorithm in simulation. The experiments demonstrate that the mapless navigation policy can achieve autonomous navigation for different mobile platforms in a large variety of crowd scenes with moving pedestrians.

REFERENCES

[1] A. Vemula, K. Muelling, and J. Oh, "Modeling cooperative navigation in dense human crowds," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1685–1692.

[2] A. Bera, T. Randhavane, R. Prinja, and D. Manocha, "Sociosense: Robot navigation amongst pedestrians with social and psychological constraints," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 7018–7025.

[3] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft mav," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4974–4981.

[4] H. Oleynikova, D. Honegger, and M. Pollefeys, "Reactive avoidance using embedded stereo vision for mav flight," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 50–56.

[5] S. Choi, E. Kim, K. Lee, and S. Oh, "Real-time nonparametric reactive navigation of mobile robots in dynamic environments," *Robotics and Autonomous Systems*, vol. 91, pp. 11–24, 2017.

[6] R. Ye, S. Redfield, and H. Liu, "High-precision indoor uwb localization: Technical challenges and method," in *Ultra-Wideband (ICUWB), 2010 IEEE International Conference on*, vol. 2. IEEE, 2010, pp. 1–4.

[7] H. Haddad, M. Khatib, S. Lacroix, and R. Chatila, "Reactive navigation in outdoor environments using potential fields," in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 2. IEEE, 1998, pp. 1232–1237.

[8] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *International Conference on Intelligent Robots and Systems*, 2017.

[9] T. Ort, L. Paull, and D. Rus, "Autonomous vehicle navigation in rural environments without detailed prior maps," in *International Conference on Robotics and Automation*, 2018.

[10] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight," in *Advances in neural information processing systems*, 2007, pp. 1–8.

[11] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[12] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009.

[13] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[14] B. Siciliano and O. Khatib, *Springer handbook of robotics*. Springer, 2016.

[15] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Vision-based state estimation for autonomous rotorcraft mavs in complex environments," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1758–1764.

[16] K. Mohta, V. Kumar, and K. Daniilidis, "Vision-based control of a quadrotor for perching on lines," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3130–3136.

[17] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y. L. Cun, "Off-road obstacle avoidance through end-to-end learning," in *Advances in neural information processing systems*, 2006, pp. 739–746.

[18] J. Sergeant, N. Sünderhauf, M. Milford, and B. Upcroft, "Multimodal deep autoencoders for control of a mobile robot."

[19] S. Ross, N. Melik-Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, "Learning monocular reactive uav control in cluttered natural environments," in *International Conference on Robotics and Automation*, 2013, pp. 1765–1772.

[20] 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 *International Conference on Robotics and Automation*, 2017, pp. 1527–1533.

[21] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, and S. Levine, "Uncertainty-aware reinforcement learning for collision avoidance," *arXiv:1702.01182*, 2017.

[22] J. Godoy, I. Karamouzas, S. J. Guy, and M. L. Gini, "Moving in a crowd: Safe and efficient navigation among heterogeneous agents." in *IJCAI*, 2016, pp. 294–300.

[23] 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.

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

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

[26] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," *arXiv preprint arXiv:1805.01956*, 2018.

[27] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," *arXiv preprint arXiv:1709.10082*, 2017.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

Fig. 7: Igor the robot reacts quickly to the bottles, human leges, and bags that suddenly appear in its field of view.

Fig. 8: Our algorithm was implemented on a human-like service robot and an autonomous shopping cart.

[29] N. Heess, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, A. Eslami, M. Riedmiller *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv:1707.02286*, 2017.

[30] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International Conference on Machine Learning*, 2015, pp. 1889–1897.

[31] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *International conference on machine learning*, 2009, pp. 41–48.