Learning-Based Multi-Robot Formation Control With Obstacle Avoidance

Bai, Chengchao; Yan, Peng; Pan, Wei; Guo, Jifeng

DOI
10.1109/TITS.2021.3107336

Publication date
2022

Document Version
Final published version

Published in
IEEE Transactions on Intelligent Transportation Systems

Learning-Based Multi-Robot Formation Control With Obstacle Avoidance

Chengchao Bai, *Member, IEEE*, Peng Yan, Wei Pan, *Member, IEEE*, and Jifeng Guo, *Member, IEEE*

***Abstract*— Multi-robot formation control has been intensively studied in recent years. In practical applications, the multi-robot system's ability to independently change the formation to avoid collision among the robots or with obstacles is critical. In this study, a multi-robot adaptive formation control framework based on deep reinforcement learning is proposed. The framework consists of two layers, namely the execution layer and the decision-making layer. The execution layer enables the robot to approach its target position and avoid collision with other robots and obstacles through a deep network trained by a reinforcement learning method. The decision-making layer organizes all robots into a formation through a new leader–follower configuration and provides target positions to the leader and followers. The leader's target position is kept unchanged, while the follower's target position is changed according to the situation it encounters. In addition, to operate more effectively in environments with different levels of complexity, a hybrid switching control strategy is proposed. The simulation results demonstrate that our proposed formation control framework enables the robots to adjust formation independently to pass through obstacle areas and can be generalized to different scenarios with unknown obstacles and varying number of robots.**

***Index Terms*— Multi-robot systems, leader–follower formation control, deep reinforcement learning, collision avoidance, formation adjustment.**

I. Introduction

FORMATION control of multiple robots has received extensive research attention in the past decades. Compared with a single robot, a multi-robot system has better robustness and can perform more complex tasks. Multi-robot systems have broad applications in civilian and military areas, such as agriculture [1], [2], search and rescue [3], surveillance and reconnaissance [4], and environment exploration [5].

Several formation control methods have been developed, such as a virtual structure method [6], [7], a behavior-based approach [8], [9], and a leader–follower strategy [10], [11]. The leader–follower strategy has been widely applied because of its simplicity and stability. However, it suffers from the problem that obstacle collision occurs in practical applications, because multi-robot systems often work in unknown and unstructured environments. Therefore, obstacle avoidance is an essential ability for multi-robot systems during their operation.

Several methods have been studied to solve the obstacle avoidance problem. Wu *et al.* [12] proposed a method called IOAA to avoid obstacles by adjusting the angle between the leader and follower robots. The detection range of the robot is divided into five parts, and the angle is calculated by judging which part the obstacle belongs to. Wen *et al.* [13] used the artificial potential field (APF) method to solve the obstacle avoidance problem. This method considers the obstacle as high potential points, which produce repulsive forces to expel all agents away from them. Vilca *et al.* [14] used a limit-cycle approach as an obstacle avoidance strategy where the robots avoid obstacles by tracking limit-cycle trajectories. Xiao *et al.* [15] used the separation-distance scheme method for followers to avoid collision. When the distance between the follower and an obstacle is less than the safe distance, the follower will maintain a fixed position relationship with a virtual robot moving on the boundary of the obstacle to avoid this obstacle.

The common drawback of the above mentioned methods is that they assume the position of obstacles to be accurately known but do not obtain the position of obstacles in detail. Several methods have been proposed to address this gap. Dai and Lee [16] proposed a geometric obstacle avoidance control method to select waypoints for obstacle avoidance. The waypoints are calculated by the intersections between the measured range lines and the surface of the obstacle. Luo *et al.* [17] used the histogram obstacle avoidance method to avoid obstacles. The method searches for the navigation direction by using the polar density of obstacles, which is calculated on the basis of the weighted average of the inverse of range data at each orientation of the sensors. Fujimori *et al.* [18] proposed a reactive obstacle avoidance technique based on the robots' sonars. The obstacle avoidance commands are selected by the direction of the obstacle, judging by the outputs of the fired sonars. Although these methods have been successfully examined through experiments, they have some limitations. A significant limitation is that the above mentioned collision avoidance methods only have some fixed patterns, leading to their failure in complex and unknown environments beyond the pre-designed range.

The robot collision avoidance based on deep reinforcement learning (DRL) has been extensively studied because of DRL's human-level performance on specific tasks [19]. Chen *et al.* [20] presented a decentralized multi-agent collision avoidance algorithm called CADRL, in which a value

Manuscript received 23 September 2020; revised 1 May 2021; accepted 13 August 2021. Date of publication 1 September 2021; date of current version 9 August 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 61973101. The Associate Editor for this article was C. G. Claudel. *(Corresponding author: Chengchao Bai.)*

Chengchao Bai and Wei Pan are with the Department of Cognitive Robotics, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: baichengchao@hit.edu.cn; wei.pan@tudelft.nl).

Peng Yan and Jifeng Guo are with the School of Astronautics, Harbin Institute of Technology, Harbin 150001, China (e-mail: yanpeng@hit.edu.cn; guojifeng@hit.edu.cn).

Digital Object Identifier 10.1109/TITS.2021.3107336

network encoding the estimated time to the goal is developed to generate an obstacle avoidance path. The value network is initialized by supervised training and then refined by DRL. Based on [20], a time-efficient navigation policy, called SA-CADRL, adhering to common social norms was developed [21]. This method enables a robotic vehicle to move fully autonomously at human walking speed in an environment with many pedestrians. Everett *et al.* [22] extended their previous approach [21] by incorporating the following two aspects: (1) the agents' behavior rules are released and the agents can follow any particular behavior rules and (2) the long short-term memory (LSTM) [23] is used to enable the algorithm to process an arbitrary number of states of other agents. Besides, the distributed LSTM [24] can be used to deal with large number of robots. Long *et al.* [25] presented a decentralized sensor-level multi-robot collision avoidance policy, which maps raw laser scanner measurements to an agent's velocity commands. The policy is trained by using a policy gradient-based reinforcement learning algorithm. The convolutional neural network (CNN) [26] is used to process the raw sensor measurements. Fan *et al.* [27] further improved their previous work [25] by integrating the learning policy into a hybrid control framework, thereby enhancing the policy's robustness and effectiveness.

In this study, we focus on the leader–follower formation control of networked multi-robot systems in an unknown and unstructured environment with a critical focus on the ability of collision avoidance by independently changing the formation. Inspired by the research mentioned above, we combine the DRL method with the leader–follower formation control framework to improve the performance of networked multi-robot systems. The robots are equipped with a laser scanner to sense the obstacles and a communication module to communicate with their neighbor robots within the communication range. Motivated by [25] and [22], we use the LSTM to process the observations of an arbitrary number of other robots and the CNN to process the raw laser scanner measurements. There are two differences between our work and [22], [25]. First, there is a cooperative relationship between robots, and their target points have a definite position relationship in the training process, which is determined by the initial position relationship. Second, position information can be exchanged between robots through communication instead of using onboard sensors to estimate position information.

In comparison with the existing works, the main contributions of this study lie in the following four aspects: (1) A multi-robot adaptive formation control framework is proposed. The analysis, discussion, and verification are provided from the execution and decision-making levels. (2) DRL is used to avoid collisions where the LSTM is used to process observations of an arbitrary number of other robots and the CNN is used to process the laser scanner measurements. (3) A new leader–follower formation control strategy combined with DRL is proposed. This strategy enables the robots to adjust the formation shape independently when encountering obstacles. (4) Simulation results demonstrate that the proposed formation control strategy generalizes well in situations with different number of robots and various environments.

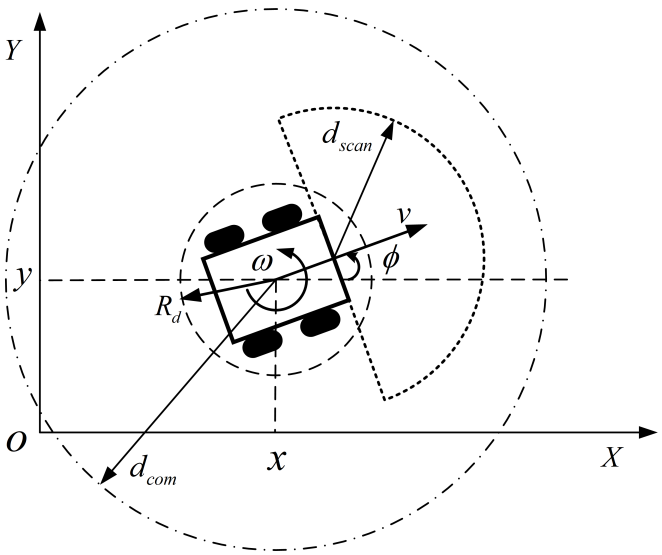


Fig. 1. Model of the mobile robot.

This paper is organized as follows. Section II provides the preliminaries of this work. Section III describes the multi-robot collision avoidance algorithm based on DRL. Section IV introduces the proposed leader–follower formation control strategy. Simulation results and corresponding discussions are presented in Section V. Finally, Section VI concludes this work.

II. Preliminaries

A. Mobile Robot Model

The mobile robot model is shown in Fig. 1. The position of the robot in a two-dimensional plane is represented as $\mathbf{p} = (x, y)$ and the orientation is represented as ϕ. The velocity vector is denoted as $\mathbf{v} = [\upsilon, \omega]$, where υ is the linear velocity and ω is the angular velocity.

The kinematics model can be represented as:

$$\begin{cases} \dot{x} = \upsilon \cos\phi \\ \dot{y} = \upsilon \sin\phi \\ \dot{\phi} = \omega \end{cases} \tag{1}$$

In this work, υ and ω are the command inputs of the mobile robot. The constraints of the command inputs are

$$\begin{cases} |\upsilon| \le \upsilon_{\max} \\ |\omega| \le \omega_{\max} \end{cases} \tag{2}$$

where $\upsilon_{\max}$ and $\omega_{\max}$ are the maximum absolute values of υ and ω, respectively.

The mobile robot is equipped with a laser scanner to perceive obstacles and a communication module to communicate with other robots. As shown in Fig. 1, d_{com} and d_{scan} are the maximum communication range and the maximum laser scanner detection range, respectively. The laser scanner is a 180-degree laser scanner that provides 180 distance values per scan. The mobile robot's collision area is modeled as a disc with radius R_d. All robots operate homogeneously.

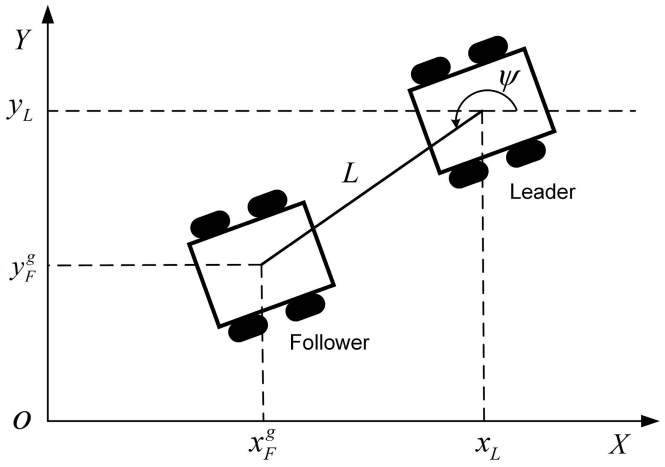


Fig. 2. Leader–follower formation configuration.

B. Leader–Follower Configuration

The formation control of multiple robots is implemented by the leader–follower approach. It is assumed that only one among all the robots is aware of the formation's target position and this robot is chosen as the leader. The remaining robots obtain the leader's information through their communication module and maintain a relative formation with the leader. The formation is constructed by a $L-\psi$ configuration as shown in Fig. 2. L is the desired distance between the leader and the follower, measured from the center of the leader to the center of the follower. ψ is the desired angle from the positive x-axis to the axis connecting the centers of the leader and the follower. Given the leader's position and the configuration $L-\psi$, the follower's target position is as follows:

$$\begin{cases} x_F^g = x_L + L\cos\psi \\ y_F^g = y_L + L\sin\psi \end{cases} \tag{3}$$

C. Problem Statement

This paper considers the problem of formation control of multiple mobile robots in an unknown environment, as shown in Fig. 3. The objective is to navigate the robots from start positions to goal positions while maintaining a formation and ensuring that no collision occurs among robots or between the robots and the obstacles in the environment. Each robot perceives the obstacles using its laser scanner and broadcasts its state information and receives the state information of other robots using its communication module. Note that when the robots encounter obstacles, the formation will change to avoid obstacles. At this time, formation maintenance means that the distances between the follower robots and the leader robot are maintained within the communication range so that the formation can still be controlled.

To solve this problem, we divide the formation control framework into two layers. The top layer is the decision-making layer, which provides a target point to each robot and different control strategies according to different situations for the leader and the followers. The bottom layer is the execution layer, which provides the robots the ability to follow target points and avoid obstacles and collision with other robots. The bottom layer is implemented by the DRL method as described in section III, and the top layer is described in section IV.

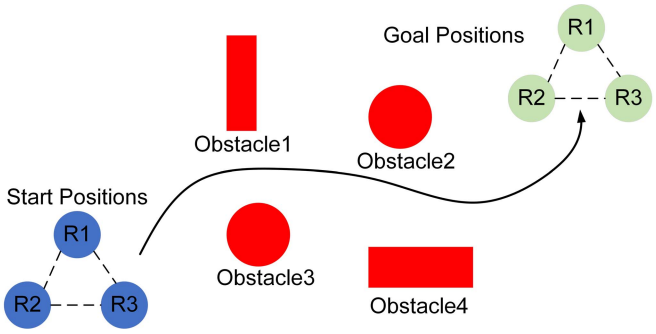


Fig. 3. Formation control in an unknown environment with obstacles.

III. DRL-Based Multi-Robot Collision Avoidance

In this section, we present the key elements of our reinforcement learning framework for multi-robot collision avoidance.

A. Problem Formulation

The multi-robot collision avoidance problem can be considered as a mobile robot moving on the 2D plane with obstacles and other decision-making robots that can be treated as dynamic obstacles with uncertainties. Therefore, the multi-robot collision avoidance problem can be formulated as a partially observable sequential decision-making problem for a single robot. At each time step t, the robot is at the position $\mathbf{p}_t$; it receives an observation $\mathbf{o}_t$ from the environment and computes an action $\mathbf{a}_t$ (actually a velocity vector) that drives it to approach the target position $\mathbf{p}_g$ without collision with the obstacles $\mathbf{B}_k (0 \le k \le M)$ and other decision-making robots, whose positions are denoted by $\tilde{\mathbf{p}}_t$. The action $\mathbf{a}_t$ is sampled from a stochastic policy:

$$\mathbf{a}_t \sim \pi_\theta\left(\mathbf{a}_t|\mathbf{o}_t\right) \tag{4}$$

where θ denotes the policy parameters.

The objective is to minimize the expected travel time $\mathrm{E}\left(t_g|\pi_\theta\right)$ of the mobile robot to reach its target position by determining an optimal policy π_θ:

$$\arg\min_{\pi_\theta} \mathrm{E}\left(t_g|\pi_\theta\right) \tag{5}$$

$$\text{s.t.} \quad \|\mathbf{p}_t - \tilde{\mathbf{p}}_t\| \ge 2\ R_d \tag{6}$$

$$\|\mathbf{p}_t - \mathbf{B}_k\| \ge R_d, \quad \forall k \in [1, M] \tag{7}$$

$$\mathbf{p}_t = \mathbf{p}_g \tag{8}$$

$$\mathbf{p}_t = \mathbf{p}_{t-1} + \Delta t \cdot \mathbf{a}_t \tag{9}$$

where (6) is the collision avoidance constraint among robots, (7) is the collision avoidance constraint between the mobile robot and the obstacles, (8) is the target position constraint, and (9) is the mobile robot kinematics constraint.

We solve this optimization problem through a reinforcement learning method, which is presented in the next subsection.

B. Reinforcement Learning Framework Design

The partially observable sequential decision-making problem introduced in section III-A can be described by a partially observable Markov decision process (POMDP). A POMDP has a six-tuple (S, A, P, R, Ω, O), where S is the state space, A is the action space, P is the state-transition model, R is the reward function, Ω is the observation space ($\mathbf{o} \in \Omega$), and O is the observation function mapping the system state to the observation ($\mathbf{o} \sim \mathrm{O}(\mathbf{s})$) [25]. Reinforcement learning (RL) [28] is a class of machine learning methods that can be used to solve the partially observable sequential decision-making problem. Let $\mathbf{s}_t$ ($\mathbf{s}_t \in S$) denote the environment state, $\mathbf{o}_t$ ($\mathbf{o}_t \in \Omega$) the agent's observed state, $\mathbf{a}_t$ ($\mathbf{a}_t \in A$) the agent's action, and r_t ($r_t \in R$) the agent's received reward at time t. The objective of the RL agent is to learn a policy, π_θ ($\mathbf{a}_t|\mathbf{o}_t, \mathbf{o}_t \sim O(\mathbf{s}_t)$), that selects an action $\mathbf{a}_t$ given the observed state $\mathbf{o}_t$ to maximize the accumulated discounted return:

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{10}$$

where γ ($0 < \gamma < 1$) is the discount factor.

The observation space, action space, and reward function are described in detail below.

1) Observation Space: At time t, the current robot observation $\mathbf{o}_t$ consists of the following four parts:

- As Fig. 4 shows, the relative target position $\mathbf{o}_t^g = \left[d_g, \alpha_g\right]$, which is a 2D vector representing the target position in polar coordinates (distance and angle) with respect to the robot's current position. $\mathbf{o}_t^g$ is normalized as follows:

$$d_g = \begin{cases} d_g/d_n & \text{if } d_g < d_n \\ 1.0 & \text{otherwise} \end{cases} \tag{11}$$

$$\alpha_g = \alpha_g/\pi \tag{12}$$

  where d_n is the value used to normalize the distance from the robot's current position to the target position, which is related to the scale of the environment.
- The robot's current velocity $\mathbf{o}_t^v = [v_t, \omega_t]$, which includes the current translational and rotational velocities of the robot. In this study, the norms of the robot's translational and rotational velocities are less than 1.0 (i.e., $v_{\max} = 1.0$m/s, $\omega_{\max} = 1.0$rad/s), so the robot's current velocity $\mathbf{o}_t^v$ is not normalized.
- Other robots' state $\mathbf{o}_t^r = \left[\mathbf{o}_t^{r_1}, \mathbf{o}_t^{r_2}, \cdots \mathbf{o}_t^{r_j}, \cdots, \mathbf{o}_t^{r_n}\right]$, ($\mathbf{o}_t^{r_j} = \left[d_j^r, \alpha_j^r\right], j = 1, 2, \cdots n, j \neq i$, where i is the current robot's index). $\mathbf{o}_t^r$ is a sequence of 2D vectors representing other robots' positions in polar coordinates (distance and angle) with respect to the current robot's position. It encompasses all other robots' position information within the current robot's communication range. The relative positions are listed in reverse order of distance to the robot to keep the most relevant information in the process of feature extraction by a deep neural network. $\mathbf{o}_t^r$ is normalized as follows:

$$\begin{cases} d_j^r = d_j^r/d_{com} \\ \alpha_j^r = \alpha_j^r/\pi \end{cases} \tag{13}$$

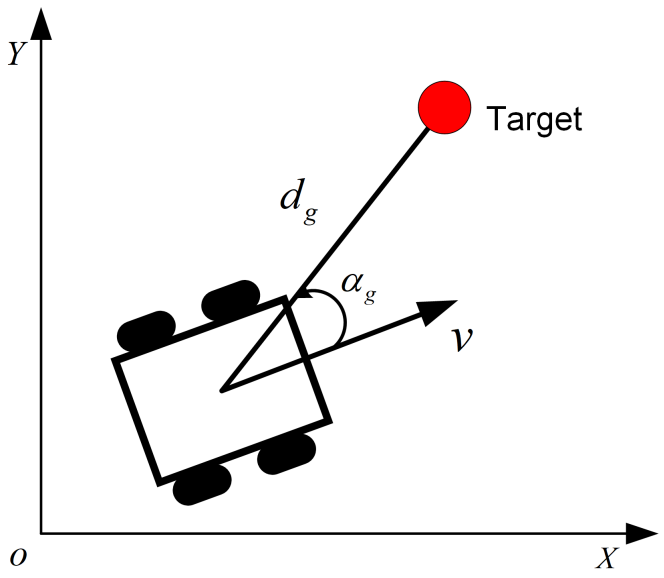


Fig. 4. Relative target position $\mathbf{o}_t^g = \left[d_g, \alpha_g\right]$ with respect to the robot's current position.

- The readings of the 2D laser range scanner $\mathbf{o}_t^z$, comprising the last three consecutive laser scanner measurements with 180 distance values per scanning and each measurement having a maximum range of 5.0 m (i.e., $d_{scan} = 5.0$m). $\mathbf{o}_t^z$ is normalized by dividing the maximum range of a laser range scanner.

Thus, the current robot observation is represented as $\mathbf{o}_t = \left[\mathbf{o}_t^g, \mathbf{o}_t^v, \mathbf{o}_t^r, \mathbf{o}_t^z\right]$.

2) Action Space: The action space $\mathbf{a}_t$ includes the translational velocity v_t^c and rotational velocity ω_t^c, that is, $\mathbf{a}_t = \left[v_t^c, \omega_t^c\right]$. The translational and rotational velocities are discretized as $v_t^c = \{0.0, 0.5, 1.0\}$m/s and $\omega_t^c = \{-1.0, -0.5, 0.0, 0.5, 1.0\}$rad/s, respectively. The action space is composed of the translational and rotational velocities and has 15 actions in total.

3) Reward Function: Our objective is to minimize the travel time of the mobile robot to reach its target while avoiding collisions with the other mobile robots and obstacles. Therefore, a reward function is designed as follows:

$$r_t = r_t^g + r_t^c + r_t^a \tag{14}$$

where r_t^g is the reward for reaching the target, r_t^c is the reward for avoiding collisions with other robots and obstacles, and r_t^a is the reward for action.

The robot receives reward r_t^g for reaching its target:

$$r_t^g = \begin{cases} 20.0 & \text{if } \left\|\mathbf{p}_t - \mathbf{p}_g\right\| \leq 0.5 \\ 0.5\left(\left\|\mathbf{p}_{t-1} - \mathbf{p}_g\right\| - \left\|\mathbf{p}_t - \mathbf{p}_g\right\|\right) & \text{otherwise} \end{cases} \tag{15}$$

The robot receives reward r_t^c for avoiding collisions:

$$r_t^c = \begin{cases} -1.0 & \text{if } \|\mathbf{p}_t - \tilde{\mathbf{p}}_t\| < 2\ R_d \text{or } \|\mathbf{p}_t - \mathbf{B}_k\| < R_d \\ -0.3 & \text{else if } \|\mathbf{p}_t - \tilde{\mathbf{p}}_t\| < 3\ R_d \text{ or } \|\mathbf{p}_t - \mathbf{B}_k\| < 2\ R_d \\ 0.0 & \text{otherwise} \end{cases} \tag{16}$$

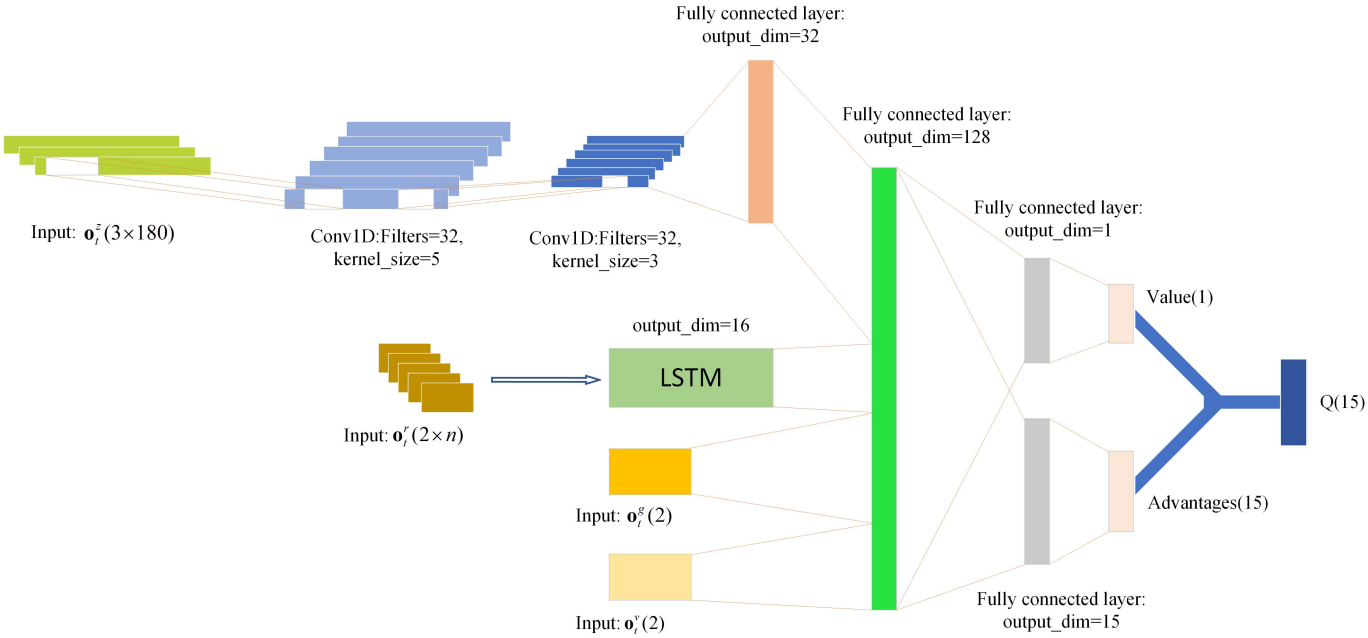


Fig. 5. Network structure.

The robot receives reward r_t^a for navigating smoothly, and it is calculated by the difference between the current action value and the last action value:

$$r_t^a = -0.05\left(\left|v_t^c - v_{t-1}^c\right| + \left|\omega_t^c - \omega_{t-1}^c\right|\right) \tag{17}$$

C. Network Structure

In this study, we use the dueling deep Q network (DQN) [29] structure, shown in Fig. 5, to extract feature information from the robot's observation and learn the action values for each action in the action space.

Inspired by [25], we use the first three hidden layers to process the laser scanner measurements $\mathbf{o}_t^z$. The first hidden layer uses the CNN to process the three input scans, comprising 32 one-dimensional filters with kernel size $=5$ and stride $=2$ and applying ReLU nonlinearities [30]. The second hidden layer is similar to the first hidden layer except with kernel size $=3$ and stride $=2$. The third hidden layer is a fully connected layer with 32 rectifier units. Similar to [22], we employ the LSTM to process other robots' state $\mathbf{o}_t^r$. The LSTM layer converts the other robots' states into a 16-dimensional vector. The above two outputs are concatenated with the other two inputs ($\mathbf{o}_t^g$ and $\mathbf{o}_t^v$) and then are fed into a fully connected layer with 128 rectifier units. The output of this layer is simultaneously processed by two fully connected linear layers to produce separate estimates of the value and corresponding advantages of the 15 actions. Finally, these outputs are combined to generate a single output with 15 action values.

Overall, the neural network maps the input observation $\mathbf{o}_t$ to an action value vector. The final action $\mathbf{a}_t$ is chosen according to the action values.

D. Training Process

1) Training Algorithm: We use double DQN [31] to train our network model. In double DQN, the target is defined as

$$y_t = r_{t+1} + \gamma\,\hat{Q}\left(S_{t+1}, \arg\max_a Q\left(S_{t+1}, a; \theta_t\right), \theta_t^-\right) \tag{18}$$

where r_{t+1} is the reward received at time $t+1$, S_{t+1} is the environment state at time $t+1$, θ_t is the parameter of the action-value function network Q to select an action, and θ_t^- is the parameter of the target action-value function network $\hat{Q}$ to evaluate an action. The parameter θ_t is updated at every time step:

$$\theta_{t+1} = \theta_t + \beta\left(y_t - Q\left(S_t, a_t; \theta_t\right)\right)\nabla_{\theta_t} Q\left(S_t, a_t; \theta_t\right) \tag{19}$$

where β is a scalar step size (or the learning rate) and a_t is the action selected at time t. The parameter θ_t^- is only updated with θ_t every C steps.

2) Training Stage: Inspired by the curriculum learning paradigm [32], we propose a two-stage training process. In the first stage, we only train one robot on the scenario with obstacles, allowing the robot to rapidly learn collision avoidance with obstacles and navigating to the target position. Once the robot completes learning, we stop Stage 1 and save the trained network parameters. The network parameters will continue to be updated in Stage 2, where the number of robots increases to four. In Stage 2, we randomly select one of the four robots as the trained robot at the beginning of each episode and update its network parameters at every time step. Its parameters are copied to the network parameters of the other robots every N steps, which guarantees that the environment dynamics are stationary within N steps and makes the learning more robust. The ε-greedy strategy is used to select actions and the Adam [33] optimizer is used to update the

Algorithm 1 Double DQN for Multi-Robot Collision Avoidance

Initialize replay memory D;
if *Stage 1* **then**
  Initialize action-value function network Q and target action-value function network $\hat{Q}$ with random weights θ and θ^- ($\theta = \theta^-$), respectively.
else
  Initialize action-value function network Q and target action-value function network $\hat{Q}$ with weights θ saved in Stage 1. (Note: The weights of the selected trained robot's action-value function network Q are denoted as θ and those of the other robots' action-value function network Q are denoted as $\tilde{\theta}$.)
end
for *episode = 1: max-episode* **do**
  Reset the training scenario settings;
  Randomly select a robot from all the robots in the training scenario as the trained robot and set its index to 1;
  for *step = 1: max-step* **do**
    for *robot* $i = 1, 2, \ldots, n$ **do**
      Get the observation $\mathbf{o}_i^t$;
      if $i == 1$ **then**
        With probability ε, select a random action;
        Otherwise, select $a_i^t = \arg\max_a Q\left(\mathbf{o}_i^t, a\ ; \theta\right)$;
      else
        Select $a_i^t = \arg\max_a Q\left(\mathbf{o}_i^t, a\ ; \tilde{\theta}\right)$;
      end
      Execute action a_i^t in the training scenario and observe reward r_i^t and next observation $\mathbf{o}_i^{t+1}$;
    end
    Store transition $\left(\mathbf{o}_1^t, a_1^t, r_1^t, \mathbf{o}_1^{t+1}\right)$ in D (Note: store only the trained robot's experience);
    Sample random minibatch of transitions $\left(\mathbf{o}_1^j, a_1^j, r_1^j, \mathbf{o}_1^{j+1}\right)$ from D;
    Set

$$y_1^j = \begin{cases} r_1^j & \text{if episode terminates at step } j+1 \\ r_1^j + \gamma\, \hat{Q}\left(\mathbf{o}_1^j, \arg\max_a Q\left(\mathbf{o}_1^{j+1}, a; \theta\right); \theta^-\right) & \text{otherwise} \end{cases}$$

    Perform a gradient descent step on $\left(y_1^j - Q\left(\mathbf{o}_1^{j+1}, a\ ; \theta\right)\right)^2$ with respect to the network parameters θ;
    Every C steps reset $\theta^- = \theta$;
    Every N steps reset $\tilde{\theta} = \theta$;
  end
end

parameters in both training stages. The workflow of the proposed multi-robot collision avoidance algorithm is shown in Algorithm 1.

IV. Leader–Follower Formation Control Strategy

In the previous section, we explained how the robot learns to navigate itself to the target, avoiding collisions with obstacles and other robots. In this section, we combine this ability and the leader–follower approach to implement the multi-robot formation control.

In a multi-robot formation control setting, we define some variables as follows:

- The minimum distance between robot i and the other robots is denoted as $d_i^{r\min}$ $(i = 1, 2, \ldots, n)$;
- The maximum distance between robot i and the other robots is denoted as $d_i^{r\max}$ $(i = 1, 2, \ldots, n)$;
- The minimum distance between robot i to obstacles is denoted as $d_i^{o\min}$ $(i = 1, 2, \ldots, n)$;
- The leader is denoted by subscript L and the follower is denoted by subscript F;
- The target of the formation is defined as the target of the leader, which is given as $\mathbf{p}_L^g = \left(x_L^g, x_L^g\right)$;
- The distance between the formation target and the leader is denoted as d_L^g.

A. Leader's Control Strategy

The control of the leader is considered in two situations:

- **Situation 1:**
  $\min\left(d_L^{r\min}, d_L^{o\min}\right) > d_{\text{safe}}$ or $d_L^g < \min\left(d_L^{r\min}, d_L^{o\min}\right)$. In this situation, the distances $d_L^{o\min}$ and $d_L^{r\min}$ are both greater than the safe distance d_{safe}, or the distance d_L^g is less than the minimum of $d_L^{r\min}$ and $d_L^{o\min}$. This situation is treated as a simple situation, and the leader is navigated

straight toward its target by using the PID control method.

- **Situation 2:** $\min\left(d_L^{r\min}, d_L^{o\min}\right) \le d_{\text{safe}}$ and $d_L^g \ge \min\left(d_L^{r\min}, d_L^{o\min}\right)$. This situation is treated as a complex situation and we use the trained network to navigate the leader to its target, where the observation $\mathbf{o}_t^g = \left[d_L^g, \alpha_L^g\right]$ denotes the target position in the leader's polar coordinate.

To ensure that the followers are within the communication range of the leader, the action speed of the leader is controlled as follows:

$$v_L^c = \begin{cases} v_L^c\left(5/d_L^{r\max}\right) & \text{if } d_L^{r\max} > 5\text{m} \\ v_L^c & \text{otherwise} \end{cases} \tag{20}$$

The above equation indicates that the action speed of the leader gradually decreases as the maximum distance between the leader and the followers exceeds 5 m and then increases, allowing the followers to move closer to the leader faster.

B. Follower's Control Strategy

The control of the follower is considered in three situations:

- **Situation 1:** $\min\left(d_F^{r\min}, d_F^{o\min}\right) > d_{\text{safe}}$. In this situation, the distances $d_F^{o\min}$ and $d_F^{r\min}$ are both greater than the safe distance d_{safe}. We use the PID control method to navigate the follower straight toward its target. The follower's target position is calculated by equation (3), where L and ψ define the shape of the formation relative to the leader.
- **Situation 2:** $d_F^{r\min} < d_{\text{safe}}$ and $d_F^{o\min} > d_{\text{safe}}$. In this situation, the distance $d_F^{r\min}$ is less than the safe distance d_{safe} and the distance $d_F^{o\min}$ is greater than the safe distance d_{safe}. We use the trained network to control the follower, where the observation $\mathbf{o}_t^g = \left[d_F^g, \alpha_F^g\right]$ denotes the follower's target position in the follower's polar coordinates. The follower's target position is calculated by equation (3).
- **Situation 3:** $\max\left(d_F^{r\min}, d_F^{o\min}\right) \le d_{\text{safe}}$. In this situation, the distances $d_F^{o\min}$ and $d_F^{r\min}$ are both less than the safe distance d_{safe}, which is the most complex situation. We use the trained network to control the follower, where the observation $\mathbf{o}_t^g = \left[d_F^g, \alpha_F^g\right]$ is different from the second situation. The observation $\mathbf{o}_t^g$ means the following:
  - d_F^g is the distance to the follower's target position calculated by equation (3).
  - α_F^g is the angle representing the center of the selected robots in the follower's polar coordinates with respect to its current position as shown in Fig. 6. The selected robots i meet the following conditions:

$$\cos\langle|\mathbf{p}_L - \mathbf{p}_F|, |\mathbf{p}_i - \mathbf{p}_F|\rangle > 0, \quad (i \ne F, i = 1, 2, \cdots, n) \tag{21}$$

The centers of the selected robots are calculated by

$$\begin{cases} x_c = \dfrac{1}{N}\sum_{i=1}^{n} x_i \\ y_c = \dfrac{1}{N}\sum_{i=1}^{n} y_i \end{cases} \quad (i \ne F) \tag{22}$$

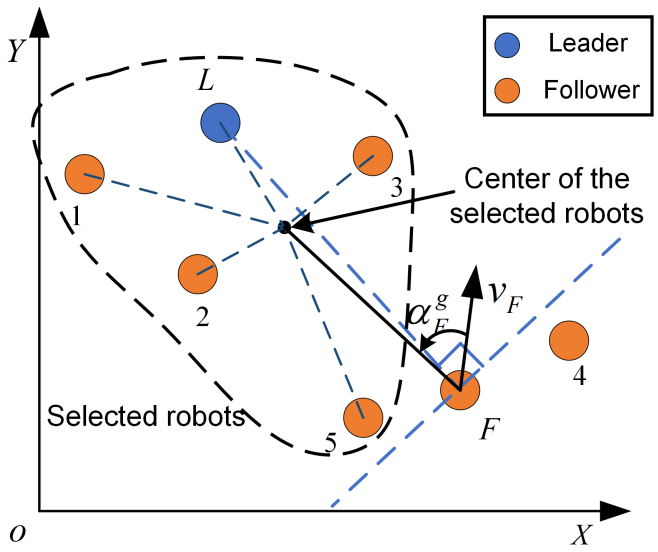


Fig. 6. Selected robots in situation 3.

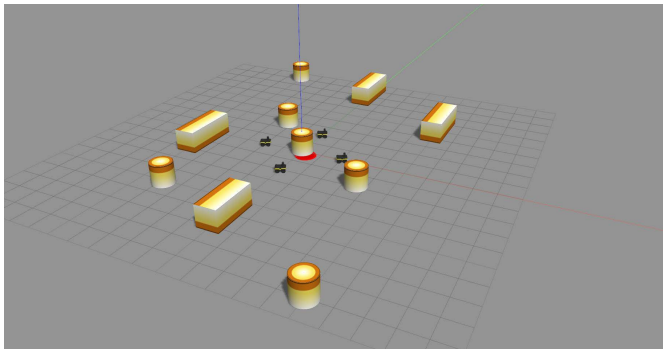


Fig. 7. Scenario used to train the collision avoidance policy.

where (x_c, y_c) are the coordinates of the center of the selected robots, N is the number of selected robots, and i satisfies equation (21).

In situation 3, the robot is likely to collide with obstacles; therefore, the formation shape must be changed to avoid the obstacles. To keep the robot in the communication range, the robot's command direction is designed toward the center of the selected robots, described by equation (21). This guarantees that the robot will approach the center of the formation to avoid obstacles.

The Leader–Follower formation control strategy is summarized in Algorithm 2.

V. Simulation Results

We demonstrated the performance of the proposed formation control strategy through a series of simulations and confirmed excellent generalization capability of the proposed method in these simulations. This section describes the simulation results.

A. Training of the Multi-Robot Collision Avoidance Policy

1) Training Scenario: We created training scenarios using gazebo[1] as shown in Fig. 7. In the training scenario,

[1]http://gazebosim.org/

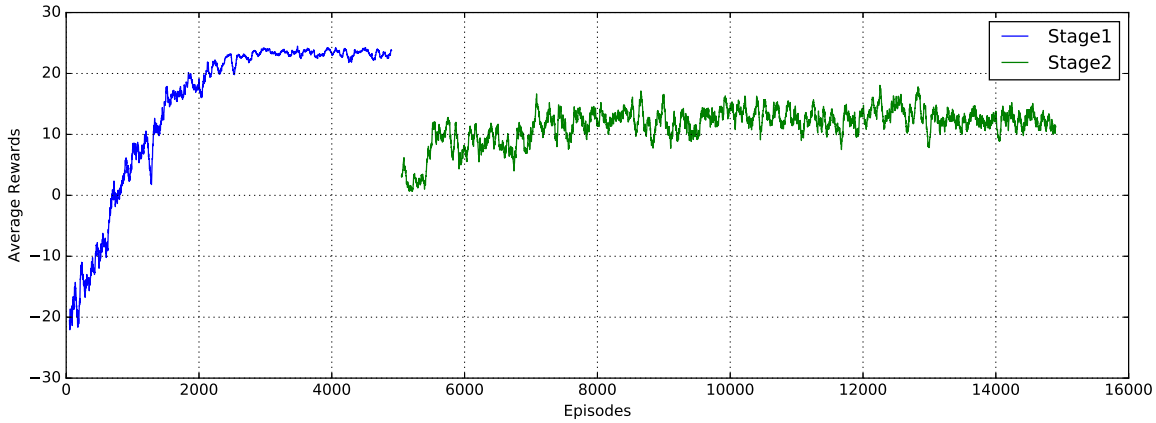


Fig. 8. Average reward curve in training Stage 1 and Stage 2. The reward is computed as the sum of the rewards accumulated in each episode.

Algorithm 2 Leader–Follower Formation Control Strategy

Initialize the environment settings (robots and obstacles);
Configure the formation shape (L and ψ);
Set the formation target position;
for *robot* $i = 1, 2, \ldots, n$ **do**
  if $i == L$ **then**
    if $\min\left(d_L^{r\min}, d_L^{o\min}\right) > d_{\text{safe}}$ *or* $d_L^g < \min\left(d_L^{r\min}, d_L^{o\min}\right)$ **then**
      Use the PID control method to navigate the leader straight toward its target;
    else
      Use the trained network to navigate the leader to its target;
    end
    if $d_L^{r\max} > 5\text{m}$ **then**
      $$v_L^c = v_L^c \left(5/d_L^{r\max}\right)$$
    end
  else
    if $\min\left(d_F^{r\min}, d_F^{o\min}\right) > d_{\text{safe}}$ **then**
      Use the PID control method to navigate the follower straight toward its target, as calculated by equation (3)
    else if $d_F^{r\min} < d_{\text{safe}}$ *and* $d_F^{o\min} > d_{\text{safe}}$ **then**
      Use the trained network to control the follower to approach its target, as calculated by equation (3)
    else
      Use the trained network to control the follower. d_F^g is the distance to the follower's target calculated by equation (3). α_F^g is the angle representing the center of the selected robot that satisfies equation (21) in the follower's polar coordinates with respect to its current position.
    end
  end
end

the cuboids and cylinders are the randomly scattered obstacles, and the mobile robot model is the Jackal[2] mobile robot. The positions of the robots and obstacles are generated randomly at the beginning of each episode (Note: to approximate the formation situation, the relative distance between the robots does not exceed 10 m). The target positions of the robots are obtained by translating the robots' initial positions for a random distance.

[2]http://wiki.ros.org/Robots/Jackal

TABLE I
TRAINING PARAMETERS IN ALGORITHM 1

Parameter	Value
batch size	32
replay memory size	2000
discount factor	0.99
learning rate	0.001
max-step	500
C	500
N	10000
d_n	3 m
step size	0.1 s

2) Training Results: The network models proposed in this study were implemented with TensorFlow [34] in Python and the robots are simulated in gazebo. We trained the multi-robot collision avoidance policy on a computer with an i9-9820X CPU and an Nvidia GTX 1080ti GPU. The model was trained within 8 h in Stage 1 and 3 h in Stage 2. The training parameters in Algorithm 1 are summarized in Table I and the parameters of the mobile robot are listed in Table II. The safe distance d_{safe} was set as 3.0 m. In particular, the max-episodes in Stage 1 and Stage 2 were 5000 and 10000, respectively. We set ε to decrease linearly from 1 to 0.1 in the first 3000 episodes in Stage 1 and 0.5 to 0.1 in the first 7000 episodes in Stage 2.

We record the average total reward of each 50 episodes. The results are shown in Fig. 8 and indicate that the average reward converges to approximately 24 after training 2500 episodes in Stage 1. This suggests that the robot's policy converges to a robust performance in a static environment and can navigate the robot to its target effectively without collision with obstacles. In Stage 2, the average reward at the beginning of training is higher than that in Stage 1 because of two reasons: (1) the policy used in Stage 2 has learned the ability to navigate the robot to its target without collision with obstacles; (2) ε is initialized to 0.5, resulting in the robot reaching

TABLE II
PARAMETERS OF THE MOBILE ROBOT

Parameter	Value
d_{com}	10 m
d_{scan}	5 m
$v_{\max}$	1 m/s
$\omega_{\max}$	1 rad/s
R_d	0.5 m

its target with a higher probability and thus receiving more rewards. The average reward in Stage 2 is less than that in Stage 1 because the environment in Stage 2 becomes more complex with the addition of robots.

B. Multi-Robot Formation Control

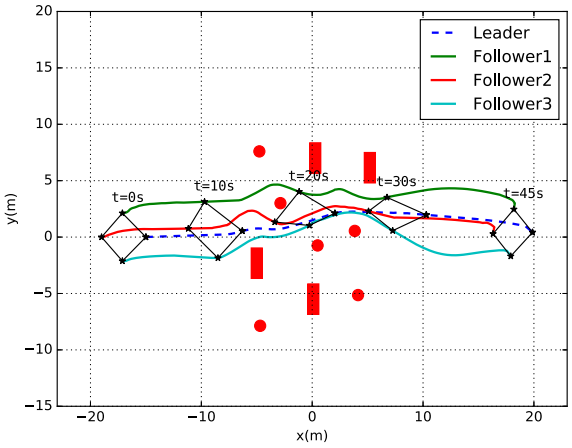
We used four mobile robots to verify the proposed leader–follower formation control strategy. The results are shown in Fig. 9.

Fig. 9(a) shows the trajectories of the entire formation. The stars represent the mobile robots and the red circles and rectangles represent obstacles. The dotted line is the trajectory of the leader and the solid lines are the trajectories of the followers. The configured formation at $t = 0$ s is as follows: $L_{F_1} = 3$ m, $\psi_{F_1} = 135°$, $L_{F_2} = 4$ m, $\psi_{F_2} = 180°$, $L_{F_3} = 3$ m, $\psi_{F_3} = -135°$. Initially, the four mobile robots form a diamond shape and their directions point to the target point. The formation shape independently change at t=10 s, t=20 s, and t=30 s when the robots encounter obstacles. Thus, the formation can easily pass through obstacle areas. The formation shape returns to the initially configuration at $t = 45$ s when it leaves the obstacle area.

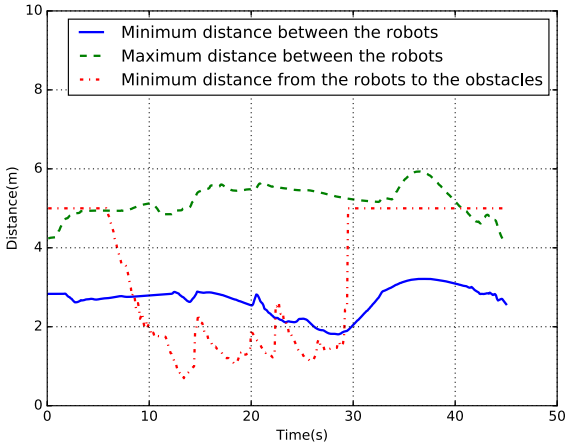
Fig. 9(b) shows the minimum and maximum distances between the robots and the minimum distance from the robots to the obstacles. The minimum distance between the robots is greater than 1.8 m during the entire simulation process, implying that there is no collision between robots. The maximum distance between the robots is less than 6 m during the entire simulation process, indicating that all robots can communicate with each other (the maximum communication range is 10 m). The minimum distance from the robots to the obstacles is greater than 0.5 m during the entire simulation process, implying that there is no collision between the robots and the obstacles.

Fig. 9(c) shows the distances from the robots to their target points. The distance from the leader to its target point decreases linearly with simulation time, indicating that the leader has been approaching its target point during the simulation process. The distances from the followers to their target points are always changing according to the environment the follower encounters. In particular, the distances from the followers to their target points change dramatically during the period from $t = 10$ s to $t = 30$ s. This is because the robots need to change the formation shape to pass through the obstacle area when they encounter obstacles.

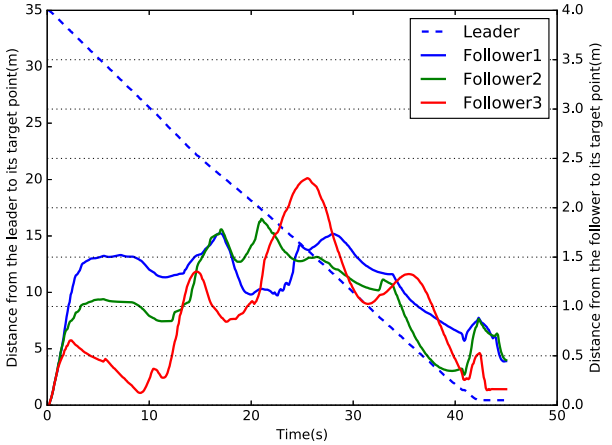
The above results demonstrate that the developed leader–follower formation control strategy enables the robots to adjust the formation shape independently to avoid collisions with obstacles and the other robots and maintain all robots within the communication range.



(a)



(b)

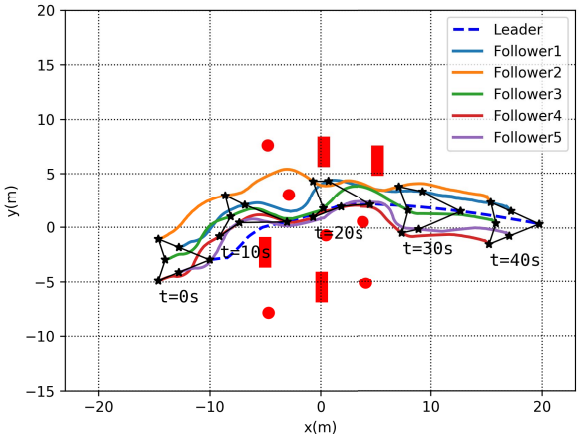


(c)

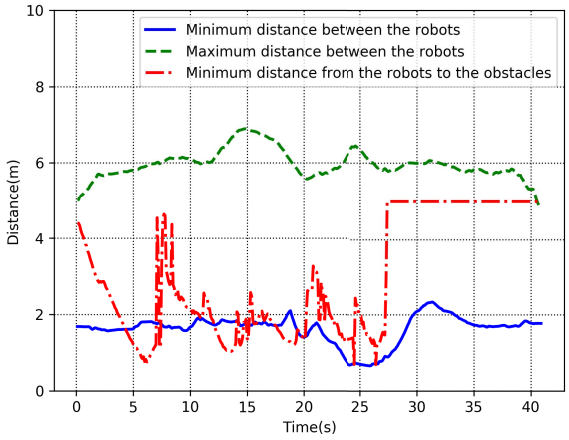
Fig. 9. Formation control results using four mobile robots.

C. Generalization Tests

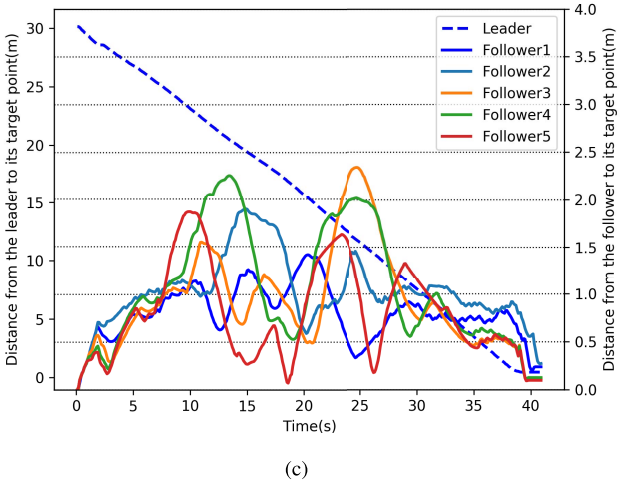
We performed two test cases to verify the generalization ability of the proposed leader–follower formation control strategy.



(a)

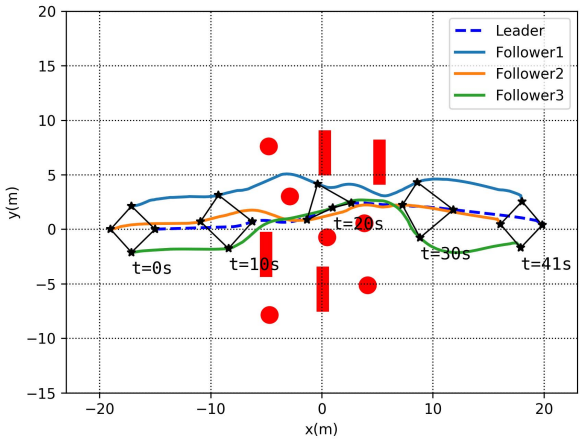


(b)



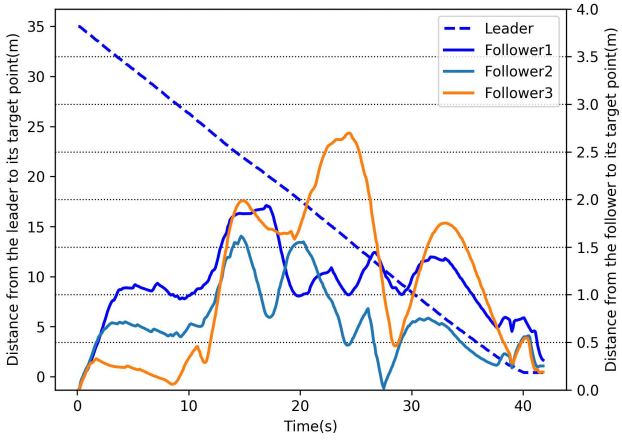
(c)

Fig. 10. Results of test case 1 (six mobile robots).



(a)

(b)



(c)

Fig. 11. Results of test case 2.

1) Case 1: In the first test case, we increased the number of robots but kept the environment consistent. The results of the first test case are shown in Fig. 10.

Fig. 10(a) shows the trajectories of the whole formation with six mobile robots. The configured formation at $t = 0$ s is as follows: $L_{F_1} = 3$ m, $\psi_{F_1} = 135°$, $L_{F_2} = 5$ m, $\psi_{F_2} = 157.5°$, $L_{F_3} = 4$ m, $\psi_{F_3} = 180°$, $L_{F_4} = 5$ m, $\psi_{F_4} = -157.5°$, $L_{F_5} = 3$ m, $\psi_{F_5} = -135°$. As Fig. 10(a) shows, the robots can adjust the formation shape independently at $t = 10$ s and $t = 20$ s to avoid collisions when they encounter obstacles. Fig. 10(b)

shows the minimum and maximum distances between the robots and the minimum distance from the robots to the obstacles. No collision occurs and the robots are always within the communication range during the entire simulation process. Fig. 10(c) shows the distances from the robots to their target points and indicate that the leader is always approaching its target point and the followers adjust their positions to pass through the obstacle area, which is consistent with Fig. 9(c).

2) Case 2: In the second test case, we changed the environment but kept the number of robots constant. The results of the second test case are shown in Fig. 11.

Fig. 11(a) shows the trajectories of the whole formation. The configured formation is the same as that in section V-B. When the robots encounter obstacles, the formation shape changes at $t = 10$ s to pass through the obstacle area; this indicates the robots can adjust different formation shapes according to varying environments. Fig. 11(b) shows the minimum and maximum distances between the robots and the minimum distance from the robots to the obstacles; no collision is observed and the robots are always within the communication range during the entire simulation process. Fig. 11(c) shows the distances from the robots to their target points. The leader is always approaching its target point and the followers adjust their positions to pass through the obstacle area, same as the results of case 1.

The above results show that our developed leader–follower formation control strategy can be generalized into different scenarios, including the scenario with different number of robots and various obstacles. The generalization ability of the formation control strategy owes largely to the generalization ability of the trained neural network, which has strong feature extraction capability and shows robust performance in different situations.

VI. Conclusion

This study developed a leader–follower formation control strategy for multi-robot systems. The formation control strategy is divided into two levels. The lower level controls each robot to approach its target and avoid collisions by using the deep network trained by the double DQN method. The dueling DQN structure effectively extracts features using the LSTM to process the observations of an arbitrary number of other robots and the CNN to process the raw laser scanner measurements. In addition, two stages of training are performed to enhance the robustness of learning. The higher level controls the leader and the followers separately according to their situation. The PID control method is employed in simple situations and the trained network is employed in complex situations. In particular, the follower's target position is selected carefully in complex situations. Finally, simulations were performed to confirm the effectiveness of the proposed method. The simulation results demonstrate that the proposed leader–follower formation control strategy enables the robots to adjust the formation shape independently when they encounter obstacles and can be applied to different scenarios with varying obstacles and number of robots. In the future work, we will investigate the problem of packet drops and communication delays among robots that is not considered in this study.

Acknowledgment

The authors would like to thank the reviewers for the comments and suggestions, and the editor who helped to improve the article significantly.

References

[1] D. Albani, J. IJsselmuiden, R. Haken, and V. Trianni, "Monitoring and mapping with robot swarms for agricultural applications," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2017, pp. 1–6.

[2] A. Guillet, R. Lenain, B. Thuilot, and V. Rousseau, "Formation control of agricultural mobile robots: A bidirectional weighted constraints approach," *J. Field Robot.*, vol. 34, no. 7, pp. 1260–1274, Oct. 2017.

[3] Y. Liu and G. Nejat, "Multirobot cooperative learning for semiautonomous control in urban search and rescue applications," *J. Field Robot.*, vol. 33, no. 4, pp. 512–536, 2016.

[4] K. Ovchinnikov, A. Semakova, and A. Matveev, "Cooperative surveillance of unknown environmental boundaries by multiple nonholonomic robots," *Robot. Auto. Syst.*, vol. 72, pp. 164–180, Oct. 2015.

[5] X. Dai, L. Jiang, and Y. Zhao, "Cooperative exploration based on supervisory control of multi-robot systems," *Int. J. Speech Technol.*, vol. 45, no. 1, pp. 18–29, Jul. 2016.

[6] D. Roy, A. Chowdhury, M. Maitra, and S. Bhattacharya, "Multi-robot virtual structure switching and formation changing strategy in an unknown occluded environment," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 4854–4861.

[7] Z. Sun and Y. Xia, "Receding horizon tracking control of unicycle-type robots based on virtual structure," *Int. J. Robust Nonlinear Control*, vol. 26, no. 17, pp. 3900–3918, Nov. 2016.

[8] G. Lee and D. Chwa, "Decentralized behavior-based formation control of multiple robots considering obstacle avoidance," *Intell. Service Robot.*, vol. 11, no. 1, pp. 127–138, 2018.

[9] D. Xu, X. Zhang, Z. Zhu, C. Chen, and P. Yang, "Behavior-based formation control of swarm robots," *Math. Problems Eng.*, vol. 2014, pp. 1–13, Jun. 2014.

[10] H. Xiao and C. L. P. Chen, "Leader-follower consensus multi-robot formation control using neurodynamic-optimization-based nonlinear model predictive control," *IEEE Access*, vol. 7, pp. 43581–43590, 2019.

[11] B. Tian, H. Lu, Z. Zuo, and W. Yang, "Fixed-time leader–follower output feedback consensus for second-order multiagent systems," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1545–1550, Apr. 2019.

[12] X. Wu, S. Wang, and M. Xing, "Observer-based leader-following formation control for multi-robot with obstacle avoidance," *IEEE Access*, vol. 7, pp. 14791–14798, 2019.

[13] G. Wen, C. L. P. Chen, and Y.-J. Liu, "Formation control with obstacle avoidance for a class of stochastic multiagent systems," *IEEE Trans. Ind. Electron.*, vol. 65, no. 7, pp. 5847–5855, Jul. 2018.

[14] J. Vilca, L. Adouane, and Y. Mezouar, "Adaptive leader-follower formation in cluttered environment using dynamic target reconfiguration," in *Distributed Autonomous Robotic Systems*. Tokyo, Japan: Springer, 2016, pp. 237–254.

[15] H. Xiao, Z. Li, and C. L. Philip Chen, "Formation control of leader–follower mobile robots' systems using model predictive control based on neural-dynamic optimization," *IEEE Trans. Ind. Electron.*, vol. 63, no. 9, pp. 5752–5762, Sep. 2016.

[16] Y. Dai and S. G. Lee, "Formation control of mobile robots with obstacle avoidance based on GOACM using onboard sensors," *Int. J. Control, Autom. Syst.*, vol. 12, no. 5, pp. 1077–1089, Oct. 2014.

[17] J. Luo, C.-L. Liu, and F. Liu, "A leader-following formation control of multiple mobile robots with obstacle," in *Proc. IEEE Int. Conf. Inf. Autom.*, Aug. 2015, pp. 2153–2158.

[18] A. Fujimori, H. Kubota, N. Shibata, and Y. Tezuka, "Leader–follower formation control with obstacle avoidance using sonar-equipped mobile robots," *Proc. Inst. Mech. Eng., I, J. Syst. Control Eng.*, vol. 228, no. 5, pp. 303–315, May 2014.

[19] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[20] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 285–292.

[21] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 1343–1350.

[22] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 3052–3059.
[23] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proc. 13th Annu. Conf. Int. Speech Commun. Assoc.*, 2012, pp. 1–4.
[24] G. Wen, J. Qin, X. Fu, and W. Yu, "DLSTM: Distributed long short-term memory neural networks for the Internet of Things," *IEEE Trans. Netw. Sci. Eng.*, early access, May 25, 2021, doi: 10.1109/TNSE.2021.3054244.
[25] P. Long, T. Fanl, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 6252–6259.
[26] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2014, pp. 806–813.
[27] T. Fan, P. Long, W. Liu, and J. Pan, "Fully distributed multi-robot collision avoidance via deep reinforcement learning for safe and efficient navigation in complex scenarios," 2018, *arXiv:1808.03841*. [Online]. Available: http://arxiv.org/abs/1808.03841
[28] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[29] Z. Wang, T. Schaul, M. Hessel, H. van Hasselt, M. Lanctot, and N. de Freitas, "Dueling network architectures for deep reinforcement learning," 2015, *arXiv:1511.06581*. [Online]. Available: http://arxiv.org/abs/1511.06581
[30] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.
[31] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 2094–2100.
[32] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 41–48.
[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[34] M. Abadi *et al.*, "Tensorflow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2016, pp. 265–283.

Chengchao Bai (Member, IEEE) received the B.S. and Ph.D. degrees in aerospace engineering from Harbin Institute of Technology, China, in 2013 and 2019, respectively. He is currently a Post-Doctoral Associate with TU Delft. His research interests include multi-robot learning, planetary exploration, intelligent sensing, and large scale resilience cooperation. He is also interested in robot intelligence and its application. He served as a member for the Youth Editorial Board of the Journal Unmanned Systems Technology. He is a Committee Member of the IEEE RAS Technical Committee on Multi-Robot Systems, CAAI (Chinese Association for Artificial Intelligence) Technical Committee on Intelligent Robot, CSIG (China Society of Image and Graphing) Technical Committee on Machine Vision, and CICC (Chinese Institute of Command and Control) Technical Committee on Unmanned Systems.

Peng Yan received the B.S. degree in aerospace engineering from Harbin Institute of Technology, China, in 2016, where he is currently pursuing the Ph.D. degree. His current research interests include motion planning, decision making, and behavior prediction.

Wei Pan (Member, IEEE) received the Ph.D. degree in bioengineering from Imperial College London. He is currently an Assistant Professor with the Department of Cognitive Robotics, Delft University of Technology. Until May 2018, he was a Project Leader with DJI, Shenzhen, China, responsible for machine learning research for DJI drones and AI accelerator. His research interests include machine learning, control theory, and robotics. He was a recipient of Dorothy Hodgkin's Postgraduate Awards, Microsoft Research Ph.D. Scholarship and Chinese Government Award for Outstanding Students Abroad, and Shenzhen Peacock Plan Award. He also serves as the Area Chair for CoRL and an Associate Editor for IROS.

Jifeng Guo (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in aerospace engineering from Harbin Institute of Technology, China, in 2001, 2004, and 2007, respectively. From 2007 to 2004, he served as a Lecturer and an Associate Professor with Harbin Institute of Technology. Since 2015, he has been a Professor with the School of Astronautics, Harbin Institute of Technology. He is the author of two books, more than 100 articles, more than 30 inventions, and holds ten patents. His research interests include intelligent sensing, autonomous planning, on-orbit service, and collaborative control. He is a member of the Editor Board of the *Journal of Unmanned Systems Technology*.