



## Research Article

# Communication-aided multi-UAV collision detection and avoidance based on two-stage curriculum reinforcement learning

Guanzheng Wang<sup>a,b</sup>, Xiangke Wang<sup>a,b</sup>, Zhiqiang Miao<sup>c</sup>, Zhihong Liu<sup>a,b,\*</sup>, Xinyu Hu<sup>a,b</sup>

<sup>a</sup> College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China

<sup>b</sup> National Key Laboratory of Equipment State and Smart Support, National University of Defense Technology, Changsha 410073, China

<sup>c</sup> College of Electrical and Information Engineering, Hunan University, Changsha 410012, China

## ARTICLE INFO

## Article history:

Received 3 April 2025

Revised 19 June 2025

Accepted 29 June 2025

Available online 25 July 2025

## Keywords:

Collision detection and avoidance

End-to-end

Multi-UAV

Two-stage

Curriculum reinforcement learning

## ABSTRACT

Currently, multi-UAV collision detection and avoidance is facing many challenges, such as navigating in cluttered environments with dynamic obstacles while equipped with low-cost perception devices having a limited field of view (FOV). To this end, we propose a communication-aided collision detection and avoidance method based on curriculum reinforcement learning (CRL). This method integrates perception and communication data to improve environmental understanding, allowing UAVs to handle potential collisions that may go unnoticed. Furthermore, given the challenges in policy learning caused by the substantial differences in scale between perception and communication data, we employ a two-stage training approach, which performs training with the network expanded from part to whole. In the first stage, we train a partial policy network in an obstacle-free environment for inter-UAV collision avoidance. In the second stage, the full network is trained in a complex environment with obstacles, enabling both inter-UAV collision avoidance and obstacle avoidance. Experiments with PX4 software-in-the-loop (SITL) simulations and real flights demonstrate that our method outperforms state-of-the-art baselines in terms of reliability of collision avoidance, including the DRL-based method and NH-ORCA (Non-Holonomic Optimal Reciprocal Collision Avoidance). Besides, the proposed method achieves zero-shot transfer from simulation to real-world environments that were never experienced during training.

© 2025 The Author(s). Published by Elsevier B.V. on behalf of Shandong University. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

## 1. Introduction

Collision avoidance in multi-UAV flights is challenging due to the complexity of dynamic and uncertain inter-UAV coordination, particularly when UAV objectives conflict. For small-rotor UAVs, this is compounded by limited perception, communication range, and computational resources [1–4]. Research on multi-UAV collision avoidance generally falls into two categories: traditional and learning-based methods. Traditional methods include motion planning, which coordinates UAVs and addresses global constraints [5–7]. However, these methods struggle with scalability and adaptability, especially in dynamic environments. Reactive approaches, such as velocity obstacle series methods [8–10], are more responsive but may underperform in complex scenarios and incur high computational costs.

The rapid advancement of deep learning (DL) has significantly contributed to the development of learning-based collision avoidance techniques, particularly those leveraging deep reinforcement learning (DRL). These methods capitalize on the strengths

of deep learning and reinforcement learning, excelling in feature extraction, nonlinear mapping, and self-learning capabilities. As a result, they can continuously enhance the performance of strategies through ongoing interaction with the environment [11–13]. In the context of small-rotor UAV collision avoidance, challenges such as limited perception range and low sample efficiency during training still persist.

Therefore, many studies have focused on improving training efficiency and enhancing convergence performance. One effective approach is curriculum learning, which involves training machine learning models in a structured sequence, beginning with simpler examples and gradually progressing to more complex ones. This strategy has been shown to outperform traditional training methods, which typically rely on random data shuffling, without introducing additional computational costs [14,15]. In this paper, we aim to integrate the strengths of both reinforcement learning and curriculum learning, which is referred to as curriculum reinforcement learning (CRL). This integration seeks to improve sampling efficiency while maintaining the robust self-learning and nonlinear fitting capabilities of reinforcement learning [16,17].

Curriculum learning primarily consists of two categories: training data ranging from simple to difficult, and models progressing

\* Corresponding author.

E-mail address: [zhliu@nudt.edu.cn](mailto:zhliu@nudt.edu.cn) (Z. Liu).

from simple to complex. Given the significant difference in dimensionality between perception data and communication data, we propose a learning method that effectively integrates both data and model curriculum learning to tackle these challenges. This method progressively increases task complexity while performing training. In the first stage, the focus is on collision avoidance between UAVs using partial neural networks, while the second stage expands the problem by incorporating additional obstacles, utilizing a full neural network for a more comprehensive solution. Extensive simulations using PX4 software-in-the-loop (SITL) [18] show significant performance improvements over the DRL-based method [19] and NH-ORCA [9], with strong generalization. A multi-UAV system with four UAVs was also tested in real-world flights, achieving zero-shot transfer from simulation to reality.

In summary, the contributions of this paper are as follows:

- We propose a communication-aided collision detection and avoidance method for multi-UAV systems based on two-stage curriculum reinforcement learning, integrating perception and communication data to ensure robust collision detection and avoidance.
- The proposed two-stage learning method first trains a partial policy network in an obstacle-free environment for inter-UAV collision avoidance. In the second stage, the full network is trained in a complex environment with obstacles, enabling both inter-collision avoidance and obstacle avoidance while improving training efficiency.
- Extensive simulation experiments using PX4 SITL, along with real-world flight tests, are conducted to validate the performance of the proposed method against two state-of-the-art baselines, demonstrating successful zero-shot transfer from simulation to reality.

The structure of the paper is organized as follows: Section 2 provides an overview of related work. Section 3 details the methods proposed in this study. Section 4 presents the experimental procedures and results, covering both simulations and real-world applications. Finally, Section 5 concludes the paper with a summary of key insights and final remarks.

## 2. Related work

### 2.1. Motion planning and VO-based approaches

To achieve fully autonomous and safe flight, researchers have extensively explored solutions to the aforementioned challenges. Traditional methods mainly include two approaches: motion planning and reactive collision avoidance. FastPlanner [5] and EgoPlanner [6] are representative works in motion planning, realizing real-time autonomous flight of an individual UAV. On this basis, Zhou et al. further proposed EgoSwarm [7] and a fully autonomous aerial swarm [20]. Tordesillas et al. proposed a decentralized and asynchronous trajectory planner MADER for UAVs to generate collision-free trajectories [21]. These methods can provide reliable path planning and collision avoidance but require mapping, which consumes significant computing resources. Besides, the potential error accumulation and delay in this cascaded architecture lead to a noticeable performance degradation in both FastPlanner and EgoPlanner as the flight speed increases, particularly with a significant decline in task success rate [22,23].

Additionally, most existing planning and control schemes place the responsibility of determining the speed constraint on the user, typically setting it conservatively as a constant during deployment [24]. Zhou et al. employed an online learning method based on Bayesian optimization to determine hyperparameters for trajectory planners [25]. However, the slow convergence rate

of Bayesian optimization limits the effectiveness of their approach, making it less capable of adapting to rapidly changing observations.

Compared to these planning methods, local reactive control methods, such as Velocity Obstacle (VO) and Reciprocal Velocity Obstacle (RVO) [8], are faster. RVO has solved the jitter problem in the original VO. Optimal reciprocal collision avoidance (ORCA) further advanced these approaches by reducing the problem to solving a low-dimensional linear program [9]. ORCA represents the state-of-the-art (SOTA) in the field of collision avoidance for multi-agent systems. However, these methods assume that each agent has perfect sensing of the surrounding environment, which limits their practical application. Multi-UAV collision avoidance using ORCA with limited field of view (FOV) has been studied in [10,26], but potential collisions with other UAVs outside the FOV are ignored. Besides, the VO-based policies are governed by several tunable parameters that are highly sensitive to the scenario conditions, requiring careful adjustment to achieve effective multi-robot motion. In some complex scenarios, these methods exhibit a noticeable decline in both time efficiency and success rate [13].

### 2.2. Learning-based approaches

With deep learning developments, end-to-end collision detection and avoidance methods are gradually emerging to overcome these shortcomings, such as [23,27]. End-to-end methods are mainly based on DL and DRL [28,29]. Among them, DL-based methods usually require large datasets for training, while building that is time-consuming and laborious. The DRL-based methods combine the advantages of DL and reinforcement learning (RL) and have strong capabilities of feature abstraction, nonlinear fitting, and decision-making.

Chen et al. proposed a decentralized non-communicating multi-agent collision avoidance method based on DRL [12], called CADRL. It has achieved better path quality (i.e., time to reach the goal) than ORCA by combining imitation learning [30] and DRL. Long et al. proposed a multi-scenario, multi-stage training framework for multi-robot [19]. The learned collision avoidance strategy can directly map the LiDAR data to the velocity control signal and realize less time and collision than NH-ORCA (non-holonomic ORCA). The studies mentioned above are primarily focused on ground robots and do not involve any form of inter-robot communication.

Compared to ground robots, UAVs have a higher dimensional action space, making multi-UAV motion planning challenging. The collision avoidance of multi-UAV systems in the presence of measurement noise has been studied in [31,32], whereas the airborne sensors used for perception are not considered. Wang et al. proposed a two-stage learning approach using local observations for multi-UAV collision avoidance under imperfect sensing [33]. Ourari et al. proposed a nearest-neighbor-based collision avoidance method for quadrotors via DRL [34]. These two works have not considered the perception sensors, and the UAVs can only avoid neighbors with known positions. Huang et al. proposed a vision-based distributed multi-UAV collision avoidance, taking depth images and inertial measurements as sensory inputs [35]. However, the FOV of the depth sensor is relatively small. This can easily lead to collisions with dynamic obstacles and neighboring UAVs.

Yu et al. proposed MAVRL (Memory-Augmented Varying-speed Reinforcement Learning), a novel obstacle avoidance method that utilizes a memory-augmented latent space and a varying-speed policy to enhance safety and efficiency in cluttered environments [36]. However, the current implementation focuses on static obstacles and does not address the prediction and avoidance of dynamic obstacles or other UAVs. Kulkarni et al. innovated

by proposing a modular deep learning approach for collision-free flight that utilizes a deep collision encoder for efficient representation of depth data and achieves robust navigation in complex environments [37], though it may require further investigation into handling dynamic obstacles and diverse terrain types for broader application.

### 3. Methodology

This section begins with a detailed description of the research problem, followed by a comprehensive explanation of the proposed collision avoidance methodologies. These methods integrate data from communication and sensory perception systems within the framework of CRL to improve navigational safety.

#### 3.1. Problem formulation

Considering a distributed multi-UAV system, each UAV only has local perception and communication capabilities. When approaching the target position, each UAV needs to avoid obstacles and other UAVs. Based on this, the task of each UAV is modeled as a partially observable Markov decision process (POMDP), which can be described as a tuple  $(\mathcal{S}, \Omega, \mathcal{A}, \mathcal{R}, \mathcal{P})$ . In this tuple,  $\mathcal{S}$  is the set of internal state  $\mathbf{s}$ ,  $\Omega$  is the observation space (observation  $\mathbf{o} \in \Omega$ ),  $\mathcal{A}$  is the action space (action  $\mathbf{a} \in \mathcal{A}$ ),  $\mathcal{R}(\mathbf{s}, \mathbf{a})$  is the reward function and  $\mathcal{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a})$  refers to the state transition function.  $\mathbf{s}'$  is the state at the next time step. The goal of DRL is to find an optimal policy  $\pi_{\theta}^* : \mathbf{o}^t \rightarrow \mathbf{a}^t$ , so as to maximize the discounted long-term numerical return  $G_t$  [11], defined as

$$G_t \doteq r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \dots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \quad (1)$$

In the definition of  $G_t$ ,  $r_t$  is the reward at time  $t$ , and  $\gamma$  is the discount factor. The  $i$ th UAV is desired to approach its destination  $\mathbf{g}_i$  from the starting point  $\mathbf{p}_i^0$ , avoiding other UAVs and obstacles  $\mathbf{B}_k$  in the environment. When  $\mathbf{p}_i^t = \mathbf{g}_i$ , the task of the  $i$ th UAV is finished. In addition, the velocity is constrained by  $v_{max}$ . Thus, the set of trajectories  $\mathbb{L}$  is defined as:

$$\begin{aligned} \mathbb{L} = & \{l_i, i = 1, \dots, N | \mathbf{v}_i^t \sim \pi_{\theta}(\mathbf{a}_i^t | \mathbf{o}_i^t), \\ & \mathbf{p}_i^t = \mathbf{p}_i^{t-1} + \Delta t \cdot \mathbf{v}_i^t, \\ & \forall j \in [1, N], j \neq i : \|\mathbf{p}_i^t - \mathbf{p}_j^t\| > 2R \\ & \wedge \forall k \in [1, M] : \|\mathbf{p}_i^t - \mathbf{B}_k\| > R \\ & \wedge \|\mathbf{v}_i^t\| \leq v_{max} \end{aligned} \quad (2)$$

In (2),  $l_i$  is the trajectory of the  $i$ th UAV,  $N$  is the number of UAVs in the system,  $\mathbf{v}_i^t$  is the velocity of the  $i$ th UAV at time  $t$ . The action  $\mathbf{a}_i^t$  at time  $t$  is the output of policy  $\pi_{\theta}$  based on the observation  $\mathbf{o}_i^t$ .  $\mathbf{p}_i^t$  is the position of the  $i$ th UAV at time  $t$ , which is updated according to the velocity  $\mathbf{v}_i^t$  and time duration  $\Delta t$ .  $\mathbf{p}_j^t$  is the position of  $j$ th UAV at time  $t$ ,  $\mathbf{B}_k$  is the position of the nearest point of the  $k$ th obstacle and  $R$  is the safe radius of the UAVs.  $M$  is the number of obstacles.  $v_{max}$  is the maximum velocity of the UAVs. To minimizing the expectation of the average arrival time of all UAVs, the optimal policy  $\pi_{\theta}$  shared by all UAVs is defined as

$$\arg \min_{\pi_{\theta}} \mathbb{E} \left[ \frac{1}{N} \sum_{i=1}^N t_i | \pi_{\theta} \right] \quad (3)$$

where  $t_i$  is the arrival time of the  $i$ th UAV. It should be pointed out that although minimizing the average time of arrival is the optimization goal, the precondition is that the velocity constraint is met and no collision occurs.

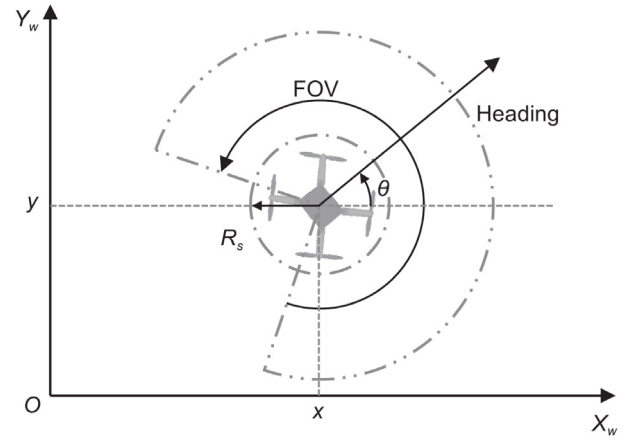


Fig. 1. The perception model of the UAV equipped with a 2D LiDAR.

#### 3.2. Kinematics model

We utilize the kinematic modeling of a UAV based on the Unicycle model, which is widely used in mobile robot dynamics. The model assumes the UAV moves in a 2D plane, with its motion defined by position and orientation, influenced by linear and angular velocities. Let the position of the drone be represented by the coordinates  $(x, y)$  in the world frame  $(X_w - Y_w)$ , and its orientation (heading) be denoted by  $\theta$ .

The UAV's velocity is defined by two components: the linear velocity  $v$  and the angular velocity  $\omega$ , which determine the rate of change of its position and orientation over time. The kinematic equations governing the UAV's motion are given by:

$$\begin{cases} \dot{x} = v \cos(\theta) \\ \dot{y} = v \sin(\theta) \\ \dot{\theta} = \omega \end{cases} \quad (4)$$

where  $\dot{x}$  and  $\dot{y}$  represent the velocity components in the  $x$ - and  $y$ -directions,  $\theta$  is the UAV's orientation relative to the global frame,  $v$  is the constant linear velocity, and  $\omega$  is the angular velocity, which controls the rate of change in orientation. This model is favored for its simplicity and ability to describe a wide range of 2D robot motions, where the control inputs directly influence the velocity and angular velocity.

#### 3.3. Reinforcement learning setup

##### 3.3.1. Observation

The observation at time  $t$  (denoted as  $\mathbf{o}^t$ ) consists of the last three consecutive LiDAR frames  $\mathbf{o}_i^t \in \mathbb{R}^{3 \times 512}$ , the relative positions of the destination  $\mathbf{o}_g^t \in \mathbb{R}^2$  and the nearest neighbor  $\mathbf{o}_c^t \in \mathbb{R}^2$ , as well as the current body velocity  $\mathbf{o}_v^t \in \mathbb{R}^2$ . Therefore, we have:

$$\mathbf{o}^t = [\mathbf{o}_i^t, \mathbf{o}_g^t, \mathbf{o}_c^t, \mathbf{o}_v^t] \quad (5)$$

LiDAR frames are planar distance measurements in the 270-degree FOV angle up to a maximum of 5 m, as shown in Fig. 1.  $R_s$  is the minimum safe radius. The size of a raw LiDAR frame is 1025 (Hokuyo UST-05LX), and we use down-sampling to reduce the size to 512. Each UAV can get its position via positioning systems such as Global Navigation Satellite System (GNSS), Ultra-Wideband (UWB) positioning system, motion capture system, or others. On the basis,  $\mathbf{o}_c^t$  is calculated by subtracting self-position from the nearest neighbors' position.

**Table 1**  
Parameters in RL setup.

Parameter	Value
$R_d$	15
$d_g$	0.5 m
$k_g$	2.5
$R_c$	-15
$R_s$	0.5 m
$k_\omega$	0.1
$\omega_m$	0.6 rad/s
$d_e$	5 m

### 3.3.2. Action

Considering the blind area of the LiDAR, the action of a UAV at time  $t$  is defined as

$$\mathbf{a}^t = [v_x, \omega_z] \quad (6)$$

where the linear velocity in  $x$  direction  $v_x \in [0, 1]$  m/s and the angular velocity in  $z$  direction  $\omega_z \in [-1, 1]$  rad/s.  $v_x$  is limited to non-negative because the LiDAR in this paper cannot perceive obstacles directly behind it. This paper focuses on collision avoidance in the two dimensional plane. The Proportional-Integral-Derivative (PID) algorithm [38] is used for height control.

### 3.3.3. Reward function

During the interaction between UAVs and the environment, reward is the key signal aligned with the optimization objective. The reward given to a UAV at time  $t$  is defined as

$$r^t = r_g^t + r_c^t + r_\omega^t \quad (7)$$

The three items are related to destination approaching, collisions, and angular velocity. Destination reward  $r_g^t$  is defined as (8).  $\mathbf{p}^t$  is the position of this UAV at time  $t$  and  $\mathbf{g}$  represents the position of the destination. When a UAV arrives at the destination (i.e., the distance to its destination is smaller than  $d_g$ ), it can obtain the reward  $R_d$ . Otherwise, the reward is determined based on the change in distance to the destination.  $k_g$  is a constant.

$$r_g^t = \begin{cases} R_d, & \text{if } \|\mathbf{p}^t - \mathbf{g}\| < d_g \\ k_g(\|\mathbf{p}^{t-1} - \mathbf{g}\| - \|\mathbf{p}^t - \mathbf{g}\|), & \text{otherwise} \end{cases} \quad (8)$$

Collision reward  $r_c^t$  is defined as (9). When a UAV collides with others or obstacles, it is punished with  $R_c$ . In (9),  $d_n$  is the distance between the UAV and its nearest neighbor, and  $\Delta d_n$  is the change in  $d_n$  within a time step.  $\mathbf{B}_n$  is the position of the closest point in obstacles to the UAV, and  $R_s$  is the minimum safe distance of the UAVs. When the distance between a UAV and another is less than  $d_e$ , a constant, it will be punished accordingly.

$$r_c^t = \begin{cases} R_c, & \text{if } d_n \leq 2R_s \text{ or } \|\mathbf{p}^t - \mathbf{B}_n\| \leq R_s \\ \ln(\frac{d_n}{d_e}) \Delta d_n, & \text{elif } 2R_s < d_n < d_e \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

The angular velocity related reward  $r_\omega^t$  is defined to suppress the excessive angular velocity for a more stable flight, as shown in (10).  $\omega_m$  is the expected upper bound of angular velocity and  $k_\omega$  is a constant. The values of the parameters mentioned before are shown in Table 1.

$$r_\omega^t = \begin{cases} -k_\omega(|\omega_z| - \omega_m), & \text{if } |\omega_z| > \omega_m \\ 0, & \text{otherwise} \end{cases} \quad (10)$$

## 3.4. Network architecture

In this paper, we primarily utilize convolutional layers to process LiDAR data for feature extraction. To this end, convolutional layers are combined with fully connected layers to construct the

## Algorithm 1: PPO for multi-UAV systems.

```

1 Network Initialization
2 for iteration = 1, 2, ..., do
3   for UAV = 1, 2, ..., NUAV do
4     Run Policy  $\pi_\theta$  for  $T_i$  time-steps, collecting
        $\{\mathbf{o}_i^t, \mathbf{r}_i^t, \mathbf{a}_i^t\}$ , when  $t \in [0, T_i]$ 
5     Estimate advantages using GAE [39],
        $\hat{A}_i^t = \sum_{l=0}^{T_i} (\gamma \lambda)^l \delta_i^{t+l}$ , where
        $\delta_i^t = r_i^t + \gamma V_\phi(s_i^{t+1}) - V_\phi(s_i^t)$ 
6     if  $\sum_{i=1}^N T_i > T_{max}$  then
7       break
8     end
9   end
10   $\pi_{old} \leftarrow \pi_\theta$ 
11  for  $j = 1, \dots, E_\pi$  do
12     $L^{PPO}(\theta) = \sum_{t=1}^{T_{max}} \frac{\pi_\theta(\mathbf{a}_i^t | \mathbf{o}_i^t)}{\pi_{old}(\mathbf{a}_i^t | \mathbf{o}_i^t)} \hat{A}_i^t - \beta KL[\pi_{old} | \pi_\theta] +$ 
        $\xi \max(0, KL[\pi_{old} | \pi_\theta] - 2KL_{target})^2$ 
13    if  $KL[\pi_{old} | \pi_\theta] > 4KL_{target}$  then
14      break and continue with next iteration i+1
15    end
16    Update  $\theta$  with  $lr_\theta$  by Adam [40] with respect to
        $L^{PPO}(\theta)$ 
17  end
18  for  $k = 1, \dots, E_v$  do
19     $L^V(\phi) = - \sum_{i=1}^N \sum_{t=1}^{T_i} (\sum_{t'=t}^{T_i} \gamma^{t'-t} r_i^{t'} - V_\phi(s_i^t))^2$ 
20  end
21  Update  $\phi$  with  $lr_\phi$  by Adam with respect to  $L^V(\phi)$ 
22  if  $KL[\pi_{old} | \pi_\theta] > \beta_{high} KL_{target}$  then
23     $\beta \leftarrow \alpha \beta$ 
24  end
25  else if  $KL[\pi_{old} | \pi_\theta] < \beta_{low} KL_{target}$  then
26     $\beta \leftarrow \beta / \alpha$ 
27  end
28 end

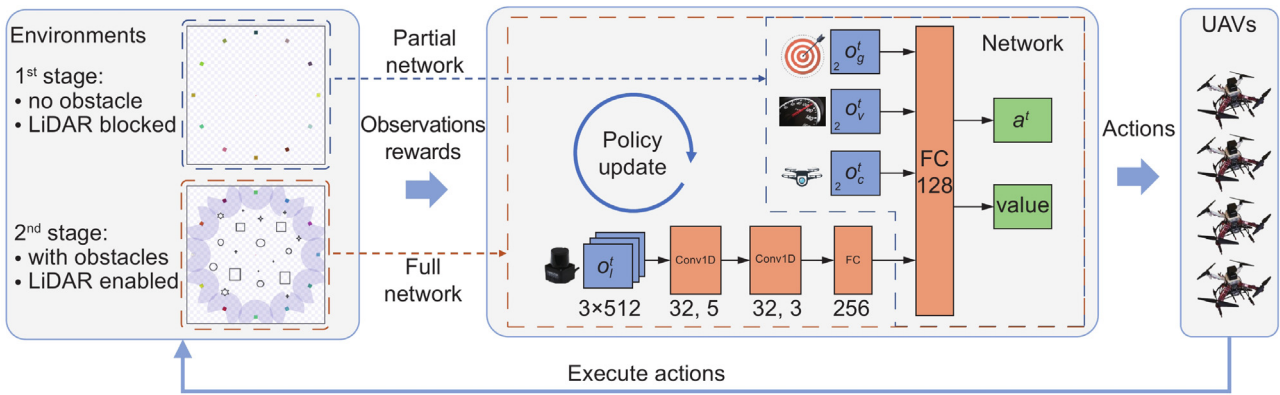
```

Actor-Critic network, as illustrated in Fig. 2. The actor network, a parameterized policy, maps from observation space  $\Omega$  to action space  $\mathcal{A}$ . The goal of training is to optimize the parameters of this network to maximize the long-term numerical reward  $G_t$ . Critic is used for value evaluation.

These two networks have the same architecture except for output. The first convolutional layer has 32 one-dimensional filters with kernel size = 5, stride = 2, and padding = 1. The second convolutional layer has 32 one-dimensional filters with kernel size = 3, stride = 2, and padding = 1. The third layer is a fully-connected layer with 256 rectifier units. The last hidden layer is a fully connected layer with 128 rectifier units. In the actor network, the output layer is a fully connected layer with two different activation functions: a sigmoid function for  $v_x$  and a hyperbolic tangent function for  $\omega_z$ . In the critic network, the output layer is a fully connected layer with a rectifier unit. To obtain more diverse experiences in training, the final output is sampled from the normal distribution  $\mathcal{N}_s(\mathbf{a}^t, \sigma_s)$ , where the mean (i.e.  $\mathbf{a}^t$ ) is the actual output of the policy network and the variation (i.e.  $\sigma_s$ ) is set manually.

## 3.5. Two-stage training approach based on CRL

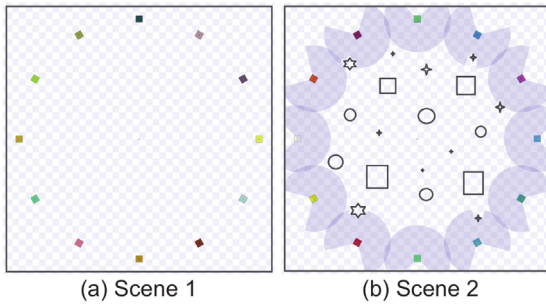
The observation includes LiDAR measurements and the relative position of the nearest neighbor, enabling the UAVs to avoid neighbor UAVs even if the LiDAR cannot detect them. However,



**Fig. 2.** The learning paradigm of the proposed communication-aided collision detection and avoidance method based on CRL. In the first stage, the UAVs learn to avoid other UAVs based on communication without LiDAR (partial network) and there are no obstacles in the environment. In the second stage, the UAVs can obtain the entire observation (full network) and there are various obstacles in the environment.

**Table 2**  
Hyper-parameters in Algorithm 1.

Parameter	Value	Parameter	Value
$\lambda$ in line 5	0.95	$\gamma$ in line 5 and 19	0.99
$T_{max}$ in line 6	8000	$E_{\pi}$ in line 11	20
$\beta$ in line 12	1.0	$KL_{target}$ in line 12	$15 \times 10^{-4}$
$\xi$ in line 12	50.0	$lr_{\theta}$ in line 16	$5 \times 10^{-5}$
$E_v$ in line 18	10	$lr_{\phi}$ in line 21	$1 \times 10^{-3}$
$\beta_{high}$ in line 22	2.0	$\alpha$ in line 23 and 26	1.5
$\beta_{low}$ in line 25	0.5		



**Fig. 3.** Training scenes in two different stages. The colored squares are UAVs, the black objects of various shapes are obstacles, and the light slate blue sectors represent lasers.

the dimension of the relative position  $\mathbf{o}_c^t \in \mathbb{R}^2$  is much lower than that of three consecutive LiDAR frames  $\mathbf{o}_l^t \in \mathbb{R}^{3 \times 512}$ . The difference in dimension and semantics makes it difficult to learn to avoid collisions based on data from communication.

In human education, the idea of curriculum learning is quite common. Learning from one course to another or simple courses to complex courses are prevalent but efficient methods, similar to machine learning [16]. Taking a step forward, we propose a two-stage training approach, which performs training with the neural network from part to whole, along with the increasing complexity of the environments, as shown in Fig. 2. In the first stage, all obstacles are removed, thus reducing the difficulty, as shown in Fig. 3(a). At the same time, LiDAR ranging is blocked, and the UAVs can focus on learning to avoid collisions using the nearest neighbor's relative position from communication (partial network). Then, the training scene is switched to scene 2 with complete observation for the second stage training with the entire network, as shown in Fig. 3(b).

The DRL algorithm for policy training is proximal policy optimization (PPO) [41], which has demonstrated good performance

in benchmarks for continuous control tasks [42,43]. The paradigm of centralized training with decentralized execution (CTDE) [44] is adopted to speed up the learning process and improve the generalization. In CTDE, the experience of all UAVs is collected together to update the network parameters in training, while in execution, each UAV makes decisions on its own. According to the computing performance of the desktop computer, twelve UAVs are set in the training environment to collect experiences ( $\{\mathbf{o}_i^t, \mathbf{r}_i^t, \mathbf{a}_i^t\}$ ) together and store them in the experience replay buffer. The complete algorithm flow is illustrated in Algorithm 1, with the parameters listed in Table 2.  $\theta$  and  $\phi$  denote the parameters of the Actor and Critic networks, respectively.  $\hat{A}_i^t$  represents the estimated advantage function, while  $V_{\phi}(s_i^{t+1})$  denotes the state-value function. In this context, the Kullback–Leibler (KL) divergence is employed to measure the discrepancy between the new and old policies. The loss functions  $L^{PPO}(\theta)$  and  $L^V(\phi)$  are utilized to update the Actor and Critic networks, respectively.

### 3.6. Training with stage

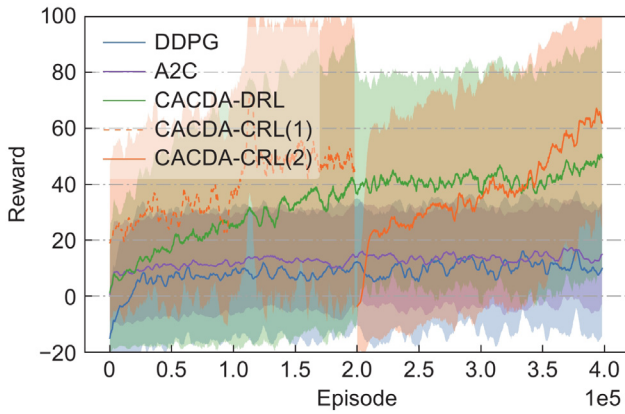
#### 3.6.1. Training details

The training is carried out on Stage<sup>1</sup> a multi-robot simulator based on Robot Operating System (ROS). The initial and target positions of the UAVs are presented in Table 3. The computer for training has an AMD 5950X CPU (Central Processing Unit) and an NVIDIA RTX3090 GPU (Graphics Processing Unit). The simulation speed of Stage is set to be five times faster than normal mode to save training time. For abbreviations, the proposed communication-aided collision detection and avoidance method based on DRL is written as CACDA-DRL, and the improved CRL-based method is written as CACDA-CRL (Communication-Aided Collision Detection and Avoidance with Curriculum Reinforcement Learning). We have also selected two additional reinforcement learning algorithms for comparison: Deep Deterministic Policy Gradient (DDPG) [45] and Advantage Actor-Critic (A2C) [46], while ensuring that the observation space remained consistent with that of CACDA-DRL. The total number of training episodes is  $4 \times 10^5$ . As for CACDA-CRL, there are  $2 \times 10^5$  episodes in each stage of the curriculum. The main parameters of the LiDAR are consistent with Hokuyo UST-05LX, with a horizontal FOV of  $270^\circ$  and a maximum range of 5 m.

<sup>1</sup> [http://wiki.ros.org/stage\\_ros](http://wiki.ros.org/stage_ros)

**Table 3**  
The initial and target positions of the UAVs in stage (m).

Axis	UAV 1	UAV 2	UAV 3	UAV 4	UAV 5	UAV 6
X	18.0 → -15.0	15.6 → -13.0	9.0 → -7.5	0.0 → 0.0	-9.0 → 7.5	-15.6 → 13.0
Y	0.0 → 0.0	9.0 → -7.5	15.6 → -13.0	18.0 → -15.0	15.6 → -13.0	9.0 → -7.5
	UAV 7	UAV 8	UAV 9	UAV 10	UAV 11	UAV 12
X	-18.0 → 15.0	-15.6 → 13.0	-9.0 → 7.5	0.0 → 0.0	9.0 → -7.5	15.6 → -13.0
Y	0.0 → 0.0	-9.0 → 7.5	-15.6 → 13.0	-18.0 → 15.0	-15.6 → 13.0	-9.0 → 7.5



**Fig. 4.** Training rewards with moving average. The orange dashed and solid lines represent two stages in CACDA-CRL.

### 3.6.2. Training results

The rewards in training are shown in Fig. 4. The curves in green and orange refer to CACDA-DRL and CACDA-CRL methods, as noted in the legend. The dashed-dotted line corresponds to the first stage (scene 1, Fig. 3(a)), and solid lines correspond to the second stage (scene 2, Fig. 3(b)). After  $2.0 \times 10^5$  episodes, CACDA-CRL has gained a higher reward due to the simpler environment. Besides, the ability to avoid collisions based on nearest-neighbor communication is formed, laying a good starting point for the training in the second stage. After switching to scene 2, the training speed of CACDA-CRL is even faster. Ultimately, CACDA-CRL receives an average reward of approximately 44.9% higher than CACDA-DRL. The effectiveness of curriculum learning is verified, which is consistent with the experience of human learning. The average rewards at the end of training for the DDPG and A2C algorithms, as benchmarks, are significantly lower than those achieved by the proposed CACDA-CRL method. Additionally, after a certain period of training, the rewards show little to no further improvement, highlighting the challenges in their learning and policy optimization. This further underscores the effectiveness of the proposed collision avoidance method based on CRL.

## 4. Verification experiments

In the experimental section, we validate the proposed method through extensive simulation experiments and real-world flight tests. The experiments were conducted using PX4 SITL and the Gazebo simulation environment, with comparisons made against two SOTA baseline methods. The scenarios involve complex environments with dynamic obstacles, assessing the collision avoidance performance and training efficiency of the multi-UAV system. The results demonstrate the successful zero-shot transfer from simulation to real-world applications, highlighting the effectiveness of the proposed approach.

### 4.1. Experimental details

The simulation experiments are carried out in Gazebo, a 3D, dynamic, multi-robot simulator that can provide high fidelity simulation.<sup>2</sup> PX4, an open-source autopilot for drones [18], is used for low-level control. PX4 software-in-the-loop (SITL) is adopted to make the simulation more realistic, as shown in Fig. 6. MAVROS is a middleware bridging ROS and the protocol MAVLink. More detailed instructions can be found in our open-source project XTDrone.<sup>3</sup> [47] The modular design makes it quite convenient to combine the collision avoidance module with others such as planning and tracking. We have constructed three scenes in Gazebo, as shown in Fig. 5. The white cylinder (in scenes 2 and 3) serves as a dynamic obstacle. The initial and target positions of the UAVs are shown in Table 4. Among them, scene 3 is designed to evaluate the generalization of DRL-based methods, and the relevant experiments will be detailed in Section 4.3.

The messages sent to MAVROS via topics mainly include velocity control output  $\{v_x, v_y, v_z, \omega_z\}$  in the body coordinate frame. In this paper,  $v_y$  is set to be 0, and  $v_z$  is obtained using the PID algorithm to maintain flight at a certain height. On this basis, the action ( $v_x$  and  $\omega_z$ ) generated by the proposed methods is used to guide collision avoidance. On real platforms, the messages to MAVROS are forwarded to the Pixhawk autopilot.

The SOTA baselines for comparison are NH-ORCA and the non-communicating DRL-based method [19], abbreviated as PPO-NC. The implementation of NH-ORCA is based on an open-source project NH-ORCA-python.<sup>4</sup> In NH-ORCA, the distance used to determine whether another UAV is a neighbor is 5 m. The maximum number of neighbors is set to five, and the time horizon is 3 s. Other parameters about the size and action space are consistent with the proposed DRL methods. The input of ORCA includes the UAVs' radius, position, and velocity, as well as the radius and position of obstacles. The policy of PPO-NC is trained in the scene shown in Fig. 3(b) for  $4 \times 10^5$  episodes.

To verify the robustness of the proposed methods, a noise  $\mathcal{N}_h(\mu_h, \sigma_h)$  obeying normal distribution is added to the height obtained from Gazebo. The mean value  $\mu_h$  and standard deviation  $\sigma_h$  are set to 0 and 0.1 m. The experiments in each scene are conducted 20 times repeatedly, and the results are shown in Table 5. The average number of collisions and time are the main indicators for performance evaluation, which correspond to reliability and efficiency. Besides, the average jerk is used to evaluate the smoothness of the trajectories.

### 4.2. In structured environments

The environments are shown in Fig. 5(a, b). The dynamic obstacle in scene 2 starts moving with the UAVs and maintains a constant velocity (1 m/s) in a straight line. Table 5 shows the average number of collisions, time and jerk. The corresponding standard deviation is in parentheses. Compared to PPO-NC, the

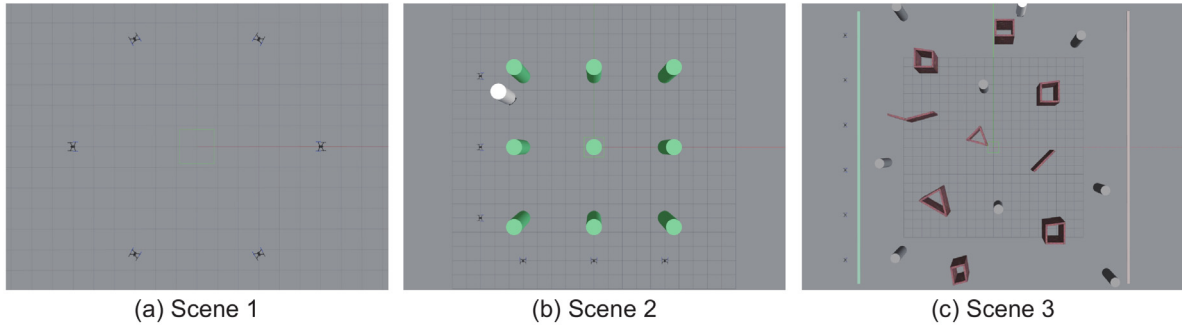
<sup>2</sup> <https://gazebo.org>

<sup>3</sup> <https://github.com/robin-shaun/XTDrone>

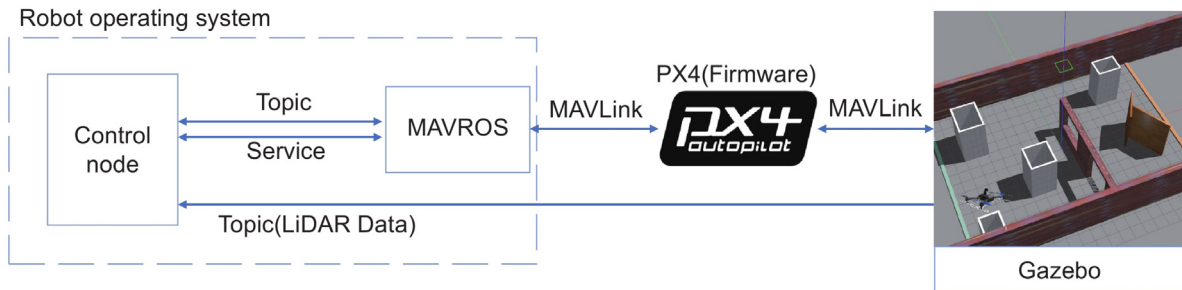
<sup>4</sup> <https://github.com/dongfangliu/NH-ORCA-python>

**Table 4**  
The initial and target positions of the UAVs in Gazebo (m).

Scene	Axis	UAV 1	UAV 2	UAV 3	UAV 4	UAV 5	UAV 6
Scene 1	X	6.0 → -6.0	3.0 → -3.0	-3.0 → 3.0	-6.0 → 6.0	-3.0 → 3.0	3.0 → -3.0
	Y	0.0 → 0.0	5.2 → -5.2	5.2 → -5.2	0.0 → 0.0	-5.2 → 5.2	-5.2 → 5.2
Scene 2	X	-8.0 → 8.0	-8.0 → 8.0	-8.0 → 8.0	-5.0 → -5.0	0.0 → 0.0	5.0 → 5.0
	Y	5.0 → 5.0	0.0 → 0.0	-5.0 → -5.0	-8.0 → 8.0	-8.0 → 8.0	-8.0 → 8.0
Scene 3	X	-16.5 → 16.5	-16.5 → 16.5	-16.5 → 16.5	-16.5 → 16.5	-16.5 → 16.5	-16.5 → 16.5
	Y	12.5 → -2.5	7.5 → -7.5	2.5 → -12.5	-2.5 → 12.5	-7.5 → 7.5	-12.5 → 2.5



**Fig. 5.** Scenes in Gazebo for evaluation experiments. The white cylinder (in scenes 2 and 3) serves as a dynamic obstacle.



**Fig. 6.** The simulation framework with PX4 software-in-the-loop.

**Table 5**

The average number of collisions, time (s) and jerk ( $m/s^3$ ). ANC stands for the average number of collisions, and the values in parentheses represent the standard deviation.

Methods	Scene 1			Scene 2			Scene 3		
	ANC	Time	Jerk	ANC	Time	Jerk	ANC	Time	Jerk
NH-ORCA	<b>0.00 (0)</b>	16.98 (0.55)	5.58 (1.44)	0.15 (0.48)	22.18 (0.66)	8.59 (1.96)	-	-	-
PPO-NC	0.40 (0.8)	<b>14.21 (0.52)</b>	<b>3.02 (0.93)</b>	1.30 (1.31)	<b>19.45 (0.32)</b>	2.37 (0.59)	4.45 (1.36)	<b>39.88 (0.99)</b>	1.81 (0.32)
CACDA-DRL	<b>0.00 (0)</b>	15.58 (0.54)	3.76 (0.59)	0.45 (0.80)	21.73 (1.19)	2.68 (0.61)	0.85 (0.48)	40.52 (1.23)	<b>1.42 (0.28)</b>
CACDA-CRL	<b>0.00 (0)</b>	15.02 (0.21)	3.11 (0.52)	<b>0.00 (0.00)</b>	21.22 (1.19)	<b>2.09 (0.77)</b>	<b>0.00 (0.00)</b>	41.94 (0.69)	1.67 (0.40)

collisions using CACDA-DRL and CACDA-CRL are significantly reduced. Besides, CACDA-CRL has achieved zero collision. PPO-NC is limited by the field of view, causing the most collisions.

The randomly selected trajectories are shown in Fig. 7(a-h). In Fig. 7(e), the collision of NH-ORCA occurs between the UAV and the dynamic obstacle, as marked by the black circle. The average time of CACDA-CRL has decreased by 11.5% in scene 1 and 4.3% in scene 2 compared to NH-ORCA. Besides, the average jerk of DRL-based methods is significantly smaller than that of NH-ORCA, showing better trajectory smoothness. NH-ORCA requires careful parameter tuning to balance efficiency and safety. Besides, it is difficult for ORCA to handle complex environments with various noises and non-cooperative moving obstacles.

#### 4.3. In a diverse environment

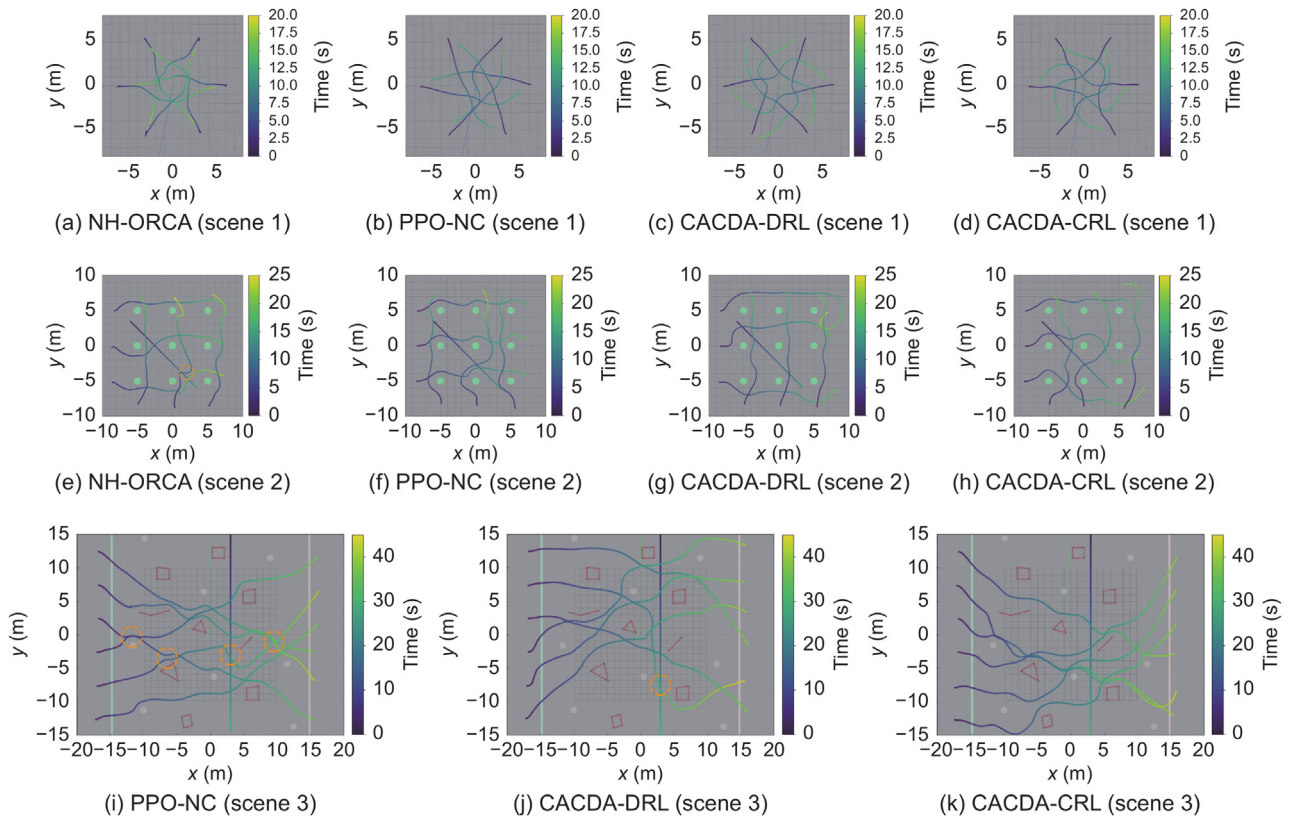
To verify the generalization of the three DRL-based methods, we design a larger unknown environment (scene 3) with diverse

irregular obstacles, as shown in Fig. 5(c). The average number of collision, time and jerk are shown in Table 5. The proposed CACDA-DRL and CACDA-CRL methods have significantly reduced the collision compared to PPO-NC, although the average time is 1.6% and 5.1% higher. In such a large and complex environment with dynamic obstacles, CACDA-CRL has still achieved non collision, showing good generalization ability.

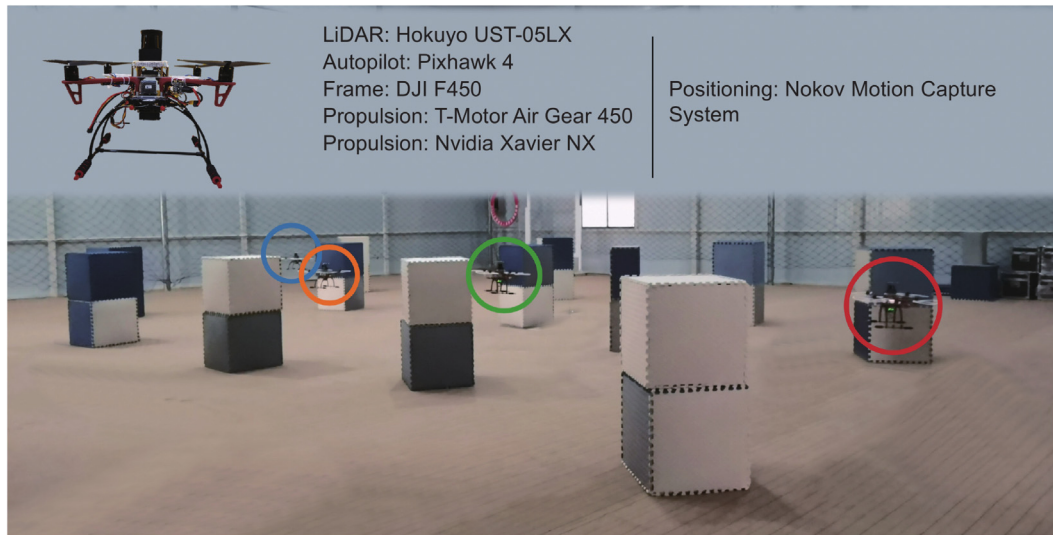
Fig. 7(i-k) shows the randomly selected trajectories. The black circles mark the positions where the collision happened. The collisions of PPO-NC in Fig. 7(i) mainly occur between UAVs. As for CACDA-DRL, the number of collisions has significantly reduced, and collisions between UAVs are rare. A UAV collides with the dynamic obstacle in Fig. 7(j).

#### 4.4. Evaluations in a real-world cluttered environment

A multi-UAV system, as shown in Fig. 8, is constructed to evaluate the performance of the proposed method on real-world



**Fig. 7.** The trajectories of UAVs and a dynamic obstacle (the straight line with color gradient). The orange circles indicate where the collision happened. The colors of the curves correspond to the flight time. Within the orange circles marking the collisions, the specific trajectories involved in the collisions can be identified by the degree of color matching between the trajectories.

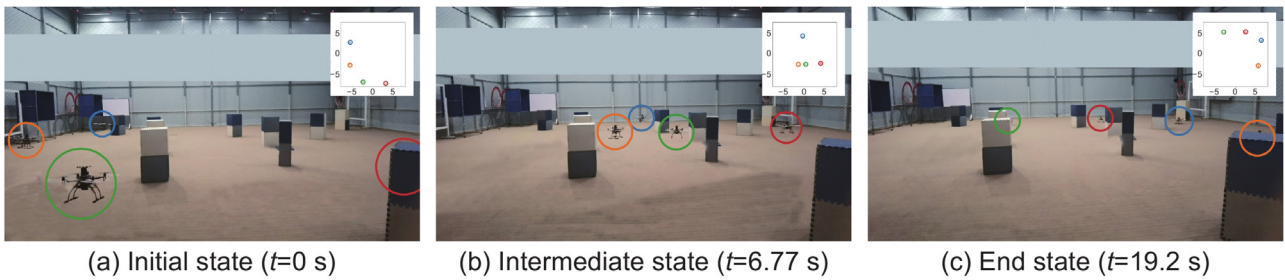


**Fig. 8.** The multi-UAV system and detailed configuration.

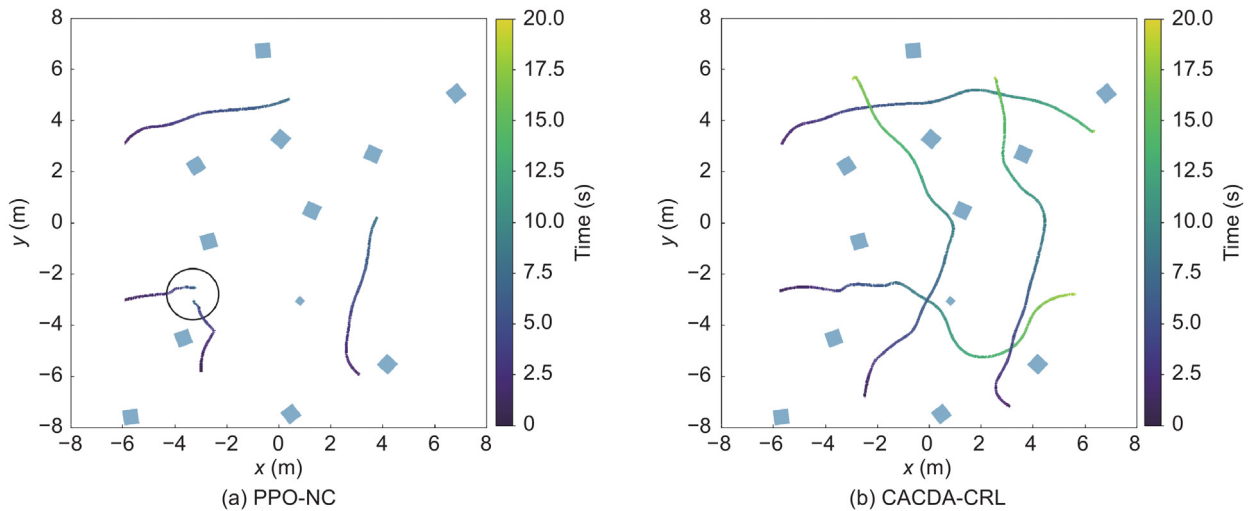
platforms. The UAV frame is the DJI F450, powered by T-Motor Air Gear 450 motors, with the Pixhawk 4 as the autopilot. The onboard computer is the Nvidia Xavier NX, and the LiDAR sensor is the Hokuyo UST-05LX, which offers precise distance measurement capabilities up to five meters. The flight area is about  $20 \times 20 \times 9 \text{ m}^3$  in size and is equipped with 72 Nokov cameras. The Nokov motion capture system<sup>5</sup> is used to track the positions of the UAVs, which are broadcast using UAV networks.

In the scene shown in Fig. 9, the baseline PPO-NC and the proposed CACDA-CRL have been verified. No fine-tuning was made to the network parameters while transferring from simulation to reality. The real-world scenarios differ from those in simulation to validate the generalization. The destination of each UAV is 12 meters in front of it, and the boxes (obstacles) are randomly placed. The flight trajectories are shown in Fig. 10. The UAVs with CACDA-CRL finished the task successfully without collision, while two UAVs collided with PPO-NC (marked with a black circle). This resulted in damage to both UAVs, and all UAVs were landed after

<sup>5</sup> <https://www.nokov.com>



**Fig. 9.** The snapshots of the real flight driven by CACDA-CRL. We use four colors to identify different UAVs. Boxes block some drones due to the shooting angle. The sub-figures in the upper right corner indicate their current positions.



**Fig. 10.** The UAV and trajectories in real flights. Light blue squares represent obstacles and the black circle in (b) indicates where the collision happens. The colors of the curves correspond to the flight time.

that. Fig. 9 shows a series of snapshots when UAVs avoid each other and obstacles, driven by CACDA-CRL.

As a whole, there are two main aspects verified in real flights. On the one hand, the feasibility of zero-shot transferring the proposed DRL methods from simulation to reality has been verified. On the other hand, the performance of the proposed CACDA-CRL method is confirmed. In the actual multi-UAV system, it is difficult to achieve perfect environmental sensing and accurate state estimation at all times. This is mainly due to possible visual blind areas and various disturbances. Therefore, combining sensor perception with neighbor communication for multi-UAV collision avoidance is particularly important.

#### 4.5. Real-time performance

The time spent on each processing component is recorded in real flights to verify the real-time performance of the proposed methods. The airborne platform used is NVIDIA Xavier NX. The forward inference is repeated for  $1 \times 10^5$  times, and the average latency of CACDA-CRL is 12.3 ms (ms). Since the network structure and processing flow of CACDA-DRL and CACDA-CRL are entirely consistent, testing one of them is sufficient. For comparison, the total latency of non-communicating baseline PPO-NC is 11.6 ms. Although the network architecture of CACDA-CRL is more complex compared with PPO-NC, the decision time can still meet the real-time requirement, and the frequency can reach

**Table 6**

Detailed latency including each processing component (ms).

Methods	Components	Total
FastPlanner	Pre-processing (14.6) Mapping (49.2) Planning (1.4)	65.2
PPO-NC	Pre-processing (1.6) Neural network inference (10.0)	11.6
CACDA-CRL	Pre-processing (2.1) Neural network inference (10.3)	12.3

81 Hz. As for the traditional planning method FastPlanner, the total processing latency is about 65.2 ms on a desktop computer with an Intel i7-8700 CPU and an NVIDIA RTX2080 GPU [23]. The detailed results of each processing component are shown in Table 6. From this table, mapping is the most time-consuming component of FastPlanner, which is not needed in the end-to-end methods.

## 5. Conclusion

Towards end-to-end collision detection and avoidance for multi-UAV systems, this paper introduced a CRL-based method combining data from LiDAR and nearest-neighbor communication. On this basis, a two-stage training method (CACDA-CRL) from simple to complex was proposed. CACDA-CRL performs training with the network expanded from part to whole, thereby handling the challenges of fusing data with different dimensions and improving training efficiency significantly. The experiments

with PX4 SITL demonstrated that CACDA-CRL outperforms the DRL-based baseline and NH-ORCA regarding reliability. In addition, CACDA-CRL achieved non-collision in the three scenes, exhibiting good generalization ability. Furthermore, we built a multi-UAV system with four UAVs and realized zero-shot transfer from simulation to real-world environments that were not experienced in training.

Looking ahead, there are still areas in our work that can be further refined or improved. First, the action space is two-dimensional, which makes UAVs unable to achieve collision avoidance through changing the altitudes. Extending the action space to three-dimensional is our future work. In the specific implementation, integrating a 3D LiDAR system can provide more comprehensive environmental observations for the UAVs, while extending the action space to include linear velocities along all three axes and yaw angular velocity. Furthermore, the decision-making network will require redesigning to accommodate these enhancements. Second, a motion capture system is required in the current implementation, which limits the application scope of the method. Extending the implementation to outdoor scenarios, where GNSS or UWB positioning is used, could further enhance its practicality. Third, we aim to conduct further research on multi-UAV collision avoidance in complex dynamic environments, focusing on real-world applications, to continuously explore the performance boundaries of the proposed method.

### CRedit authorship contribution statement

**Guanzheng Wang:** Writing – review & editing, Writing – original draft, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Xiangke Wang:** Writing – review & editing, Investigation, Funding acquisition, Conceptualization. **Zhiqiang Miao:** Writing – review & editing, Investigation, Conceptualization. **Zhihong Liu:** Writing – review & editing, Project administration, Methodology, Investigation, Conceptualization. **Xinyu Hu:** Validation, Software, Formal analysis, Data curation.

### Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

### Acknowledgments

This work was supported by National Natural Science Foundation of China (U23B2032 and U2241214), and Postgraduate Scientific Research Innovation Project of Hunan Province (CX20220055).

### Appendix A. Supplementary data

Supplementary material related to this article can be found online at <https://doi.org/10.1016/j.birob.2025.100253>.

### References

- [1] Jia Wu, Chunbo Luo, Yang Luo, Ke Li, Distributed UAV swarm formation and collision avoidance strategies over fixed and switching topologies, *IEEE Trans. Cybern.* 52 (10) (2021) 10969–10979.
- [2] Zhihong Liu, Xiangke Wang, Lincheng Shen, Shulong Zhao, Yirui Cong, Jie Li, Dong Yin, Shengde Jia, Xiaojia Xiang, Mission-oriented miniature fixed-wing UAV swarms: A multilayered and distributed architecture, *IEEE Trans. Syst. Man, Cybern.: Syst.* 52 (3) (2022) 1588–1602.
- [3] Chao Yan, Chang Wang, Xiaojia Xiang, Zhen Lan, Yuna Jiang, Deep reinforcement learning of collision-free flocking policies for multiple fixed-wing UAVs using local situation maps, *IEEE Trans. Ind. Informatics* 18 (2) (2021) 1260–1270.
- [4] Ruihua Han, Shengduo Chen, Shuaijun Wang, Zeqing Zhang, Rui Gao, Qi Hao, Jia Pan, Reinforcement learned distributed multi-robot navigation with reciprocal velocity obstacle shaped rewards, *IEEE Robot. Autom. Lett.* 7 (3) (2022) 5896–5903.
- [5] Boyu Zhou, Fei Gao, Luqi Wang, Chuhao Liu, Shaojie Shen, Robust and efficient quadrotor trajectory generation for fast autonomous flight, *IEEE Robot. Autom. Lett.* 4 (4) (2019) 3529–3536.
- [6] Xin Zhou, Zhepei Wang, Hongkai Ye, Chao Xu, Fei Gao, EGO-planner: An ESDF-free gradient-based local planner for quadrotors, *IEEE Robot. Autom. Lett.* 6 (2) (2021) 478–485.
- [7] Xin Zhou, Jiangchao Zhu, Hongyu Zhou, Chao Xu, Fei Gao, EGO-swarm: A fully autonomous and decentralized quadrotor swarm system in cluttered environments, in: *IEEE International Conference on Robotics and Automation, ICRA, 2021*, pp. 4101–4107.
- [8] Jur Van Den Berg, Ming Lin, Dinesh Manocha, Reciprocal velocity obstacles for real-time multi-agent navigation, in: *IEEE International Conference on Robotics and Automation, ICRA, 2008*, pp. 1928–1935.
- [9] Jur Van Den Berg, Stephen J Guy, Ming Lin, Dinesh Manocha, Reciprocal n-body collision avoidance, in: *Robotics Research: The 14th International Symposium ISRR, Springer, 2011*, pp. 3–19.
- [10] Parker Conroy, Daman Bareiss, Matt Beall, Jur Van Den Berg, 3-d reciprocal collision avoidance on physical quadrotor helicopters with on-board sensing for relative positioning, 2014, arXiv preprint [arXiv:1411.3794](https://arxiv.org/abs/1411.3794).
- [11] Richard S. Sutton, Andrew G. Barto, *Reinforcement learning: An introduction*, MIT Press, Cambridge, MA, USA, 2018.
- [12] Yu Fan Chen, Miao Liu, Michael Everett, Jonathan P. How, Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning, in: *IEEE International Conference on Robotics and Automation, ICRA, 2017*, pp. 285–292.
- [13] Tingxiang Fan, Pinxin Long, Wenxi Liu, Jia Pan, Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios, *Int. J. Robot. Res.* 39 (7) (2020) 856–892.
- [14] Petru Soviany, Radu Tudor Ionescu, Paolo Rota, Nicu Sebe, Curriculum learning: A survey, *Int. J. Comput. Vis.* 130 (6) (2022) 1526–1565.
- [15] Zhihong Liu, Xin Xu, Peng Qiao, Dongsheng Li, Acceleration for deep reinforcement learning using parallel and distributed computing: A survey, *ACM Comput. Surv.* 57 (4) (2024) 1–35.
- [16] Xin Wang, Yudong Chen, Wenwu Zhu, A survey on curriculum learning, *IEEE Trans. Pattern Anal. Mach. Intell.* 44 (9) (2021) 4555–4576.
- [17] Chao Yan, Chang Wang, Xiaojia Xiang, Kin Huat Low, Xiangke Wang, Xin Xu, Lincheng Shen, Collision-avoiding flocking with multiple fixed-wing UAVs in obstacle-cluttered environments: a task-specific curriculum-based MADRL approach, *IEEE Trans. Neural Networks Learn. Syst.* 35 (8) (2023) 10894–10908.
- [18] L. Meier, D. Honegger, M. Pollefeys, PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms, in: *IEEE International Conference on Robotics and Automation, ICRA, 2015*, pp. 6235–6240.
- [19] Pinxin Long, Tingxiang Fan, Xinyi Liao, Wenxi Liu, Hao Zhang, Jia Pan, Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning, in: *International Conference on Robotics and Automation, ICRA, 2018*, pp. 6252–6259.
- [20] Xin Zhou, Xiangyong Wen, Zhepei Wang, Yuman Gao, Haojia Li, Qianhao Wang, Tiankai Yang, Haojian Lu, Yanjun Cao, Chao Xu, Swarm of micro flying robots in the wild, *Sci. Robot.* 7 (66) (2022) eabm5954.
- [21] Jesus Tordesillas, Jonathan P. How, MADER: Trajectory planner in multi-agent and dynamic environments, *IEEE Trans. Robot.* 38 (1) (2021) 463–476.
- [22] Guangtong Xu, Tianyue Wu, Zihan Wang, Qianhao Wang, Fei Gao, Flying on point clouds with reinforcement learning, 2025, arXiv preprint [arXiv:2503.00496](https://arxiv.org/abs/2503.00496).
- [23] A. Loquercio, E. Kaufmann, R. Ranftl, M. Muller, V. Koltun, D. Scaramuzza, Learning high-speed flight in the wild, *Sci. Robot.* 6 (59) (2021) eabg5810.
- [24] Guangyu Zhao, Tianyue Wu, Yeke Chen, Fei Gao, Learning speed adaptation for flight in clutter, *IEEE Robot. Autom. Lett.* 9 (8) (2024) 7222–7229.
- [25] Xin Zhou, Chao Xu, Fei Gao, Automatic parameter adaptation for quadrotor trajectory planning, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2022*, pp. 3348–3355.
- [26] Steven Roelofsen, Alcherio Martinoli, Denis Gillet, 3D collision avoidance algorithm for unmanned aerial vehicles with limited field of view constraints, in: *IEEE Conference on Decision and Control, CDC, 2016*, pp. 2555–2560.
- [27] Pinxin Long, Wenxi Liu, Jia Pan, Deep-learned collision avoidance policy for distributed multiagent navigation, *IEEE Robot. Autom. Lett.* 2 (2) (2017) 656–663.
- [28] Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, Hongyang Li, End-to-end autonomous driving: Challenges and frontiers, 2023, arXiv preprint [arXiv:2306.16927](https://arxiv.org/abs/2306.16927).

- [29] B. Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A. Al Sallab, Senthil Yogamani, Patrick Perez, Deep reinforcement learning for autonomous driving: A survey, *IEEE Trans. Intell. Transp. Syst.* 23 (6) (2022) 4909–4926.
- [30] Luc Le Mero, Dewei Yi, Mehrdad Dianati, Alexandros Mouzakitis, A survey on imitation learning techniques for end-to-end autonomous vehicles, *IEEE Trans. Intell. Transp. Syst.* 23 (9) (2022) 14128–14147.
- [31] Andrew Clark, Control barrier functions for stochastic systems, *Automatica* 130 (2021) 109688.
- [32] Zhijun Wang, Tengfei Hu, Lijun Long, Multi-UAV safe collaborative transportation based on adaptive control barrier function, *IEEE Trans. Syst. Man, Cybern.: Syst.* 53 (11) (2023) 6975–6983.
- [33] Dawei Wang, Tingxiang Fan, Tao Han, Jia Pan, A two-stage reinforcement learning approach for multi-UAV collision avoidance under imperfect sensing, *IEEE Robot. Autom. Lett.* 5 (2) (2020) 3098–3105.
- [34] Ramzi Ourari, Kai Cui, Ahmed Elshamhory, Heinz Koepl, Nearest-neighbor-based collision avoidance for quadrotors via reinforcement learning, in: *International Conference on Robotics and Automation, ICRA, 2022*, pp. 293–300.
- [35] Huaxing Huang, Guijie Zhu, Zhun Fan, Hao Zhai, Yuwei Cai, Ze Shi, Zhaohui Dong, Zhifeng Hao, Vision-based distributed multi-UAV collision avoidance via deep reinforcement learning for navigation, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2022*, pp. 13745–13752.
- [36] Hang Yu, Christophe De Wagter, Guido C.H. E. de Croon, MAVRL: Learn to fly in cluttered environments with varying speed, *IEEE Robot. Autom. Lett.* 10 (2) (2025) 1441–1448.
- [37] Mihir Kulkarni, Kostas Alexis, Reinforcement learning for collision-free flight exploiting deep collision encoding, in: *IEEE International Conference on Robotics and Automation, ICRA, 2024*, pp. 15781–15788.
- [38] Ivan Lopez-Sanchez, Javier Moreno-Valenzuela, PID control of quadrotor UAVs: A survey, *Annu. Rev. Control.* 56 (2023) 100900.
- [39] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, Pieter Abbeel, High-dimensional continuous control using generalized advantage estimation, 2015, arXiv preprint arXiv:1506.02438.
- [40] Diederik P. Kingma, Jimmy Ba, Adam: A method for stochastic optimization, 2014, ArXiv Preprint ArXiv:1412.6980.
- [41] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov, Proximal policy optimization algorithms, 2017, arXiv preprint arXiv:1707.06347.
- [42] Robert Penicka, Yunlong Song, Elia Kaufmann, Davide Scaramuzza, Learning minimum-time flight in cluttered environments, *IEEE Robot. Autom. Lett.* 7 (3) (2022) 7209–7216.
- [43] Jiawei Fu, Yunlong Song, Yan Wu, Fisher Yu, Davide Scaramuzza, Learning deep sensorimotor policies for vision-based autonomous drone racing, in: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2023*, pp. 5243–5250.
- [44] Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, Shimon Whiteson, Counterfactual multi-agent policy gradients, in: *Proceedings of the AAAI Conference on Artificial Intelligence*, 32, (1) 2018.
- [45] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, Daan Wierstra, Continuous control with deep reinforcement learning, 2015, arXiv preprint arXiv:1509.02971.
- [46] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, Koray Kavukcuoglu,
- [47] Kun Xiao, Shaochang Tan, Guohui Wang, Xueyan An, Xiang Wang, Xiangke Wang, Xtdrone: A customizable multi-rotor UAVs simulation platform, in: *International Conference on Robotics and Automation Sciences, ICRAS, IEEE, 2020*, pp. 55–61.