Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 2399796, 12 pages
https://doi.org/10.1155/2022/2399796

Research Article

UAVs Maneuver Decision-Making Method Based on Transfer Reinforcement Learning

Jindong Zhu,[1,2] **Xiaowei Fu**,[1] **and Zhe Qiao**[1]

[1]*School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710072, China*
[2]*AVIC Shenyang Aircraft Design & Research Institute, Shenyang 110035, China*

Correspondence should be addressed to Xiaowei Fu; fxw@nwpu.edu.cn

Received 12 May 2022; Revised 25 September 2022; Accepted 22 October 2022; Published 14 November 2022

Academic Editor: Tapan Senapati

Aiming at the 1vs1 confrontation problem in a complex environment where obstacles are randomly distributed, the DDPG (deep deterministic policy gradient) algorithm is used to design the maneuver decision-making method of UAVs. Traditional methods generally assume that all obstacles are known globally. In this paper, a UAV airborne lidar detection model is designed, which can effectively solve the problem of obstacle avoidance when facing a large number of unknown obstacles. On the basis of the designed model, the idea of transfer learning is used to transfer the strategy trained by one UAV in a simple task to a new similar task, and the strategy will be used to train the strategy of the other UAV. This method can improve the intelligence of the UAVs in both sides alternately and progressively. The simulation results show that the transfer learning method can speed up the training process and improve the training effect.

1. Introduction

In the battlefield, UAVs can play a role in reconnaissance, detection, target tracking, attack interception, damage assessment, and others [1]. UAVs can also be used to intercept the enemy UAV [2]. How both sides maneuver to achieve the corresponding task objectives has aroused the attention and research interest of military experts and a large number of scholars.

At present, many experts have proposed different algorithms to solve the maneuver decision-making problems in different situations. In the traditional method, the main algorithms are the differential game method [3], expert system method [4], and guidance law [5]. These methods have shown good effect on simple tasks, but they cannot be applied to complex battlefields where the environment is unknown, and it is difficult to obtain analytical solutions. Therefore, scholars try to apply intelligent algorithms to UAV attack and defense confrontation problems, including bionic modeling [6], fuzzy cybernetics [7], and swarm intelligence algorithms [8].

Deep reinforcement learning, as an artificial intelligence technology that combines neural networks and reinforcement learning, is a new type of a decision-making method, which has good application prospects for the research of UAV countermeasures. For the scenario of UAV swarms chasing enemy targets, the DDPG algorithm is used to train UAVs to pursue targets [9]. Aiming at the confrontation problem with multiple UAVs, the cooperative decision-making method of multiple UAVs based on the multiagent reinforcement learning algorithm is proposed [10]. An MPPO algorithm is proposed to solve the confrontation problem of a large-scale UAV swarm [11]. A hierarchical framework based on reinforcement learning and two kinds of motion planning strategies for the problem of chasing and escaping games in the presence of obstacles is presented [12]. Liu and Wang proposed an adversarial decision generation method based on the generative adversarial network for the confrontation between UAVs in a barrier-free environment [13]. Wen and Shi proposed an intelligent decision making method for multicoupled tasks of cluster UAV confrontation in complex environments [14]. Wang and Guo

improved the reward function of the cluster UAV confrontation model and optimized the reward calculation method [15]. These works have verified the feasibility of applying deep reinforcement learning to the UAV confrontation problem. Most of the current research is carried out under the condition that the scene information is completely known, and the designed strategy is suitable for specific confrontation scenarios. If the scene becomes complicated, these studies may turn ineffective.

In this paper, to solve the problem of obstacle avoidance when facing a large number of unknown obstacles, a UAV airborne lidar detection model is designed, and a 1vs1 maneuver decision-making method based on the DDPG algorithm is proposed. To get a better training effect, three training methods are designed by the idea of transfer learning. The scenarios corresponding to these three training methods are interrelated, that is, gradually increasing the task difficulty and fixing the strategy of the other UAV when one UAV is trained so as to make the confrontation environment of the agent relatively stable. We can transfer the relevant experience gained during the interaction between the UAV and the environment into new training scenarios to improve the intelligence of the UAVs on both sides alternately and progressively. The experimental comparison between the transfer and nontransfer methods shows that the transfer reinforcement learning makes the two UAVs have their own intelligent strategies in a 1vs1 confrontation game. It also shows that the method can speed up the training process and improve the confrontation effect.

2. Problem Description and Modeling

2.1. 1vs1 Confrontation Problem. The scenario of 1vs1 confrontation can be described as that there are one blue UAV and one red UAV in a limited planar area, which are called the attack UAV and the defense UAV. The purpose of the attack UAV is to break through the interception of the defense UAV to reach the target area (light red area in the figure) from the initial position (blue flag). The purpose of the defense UAV is to intercept and destroy the attack UAV from the initial position (red flag). As shown in Figure 1, this paper assumes that circular obstacles (black areas) are distributed in the environment randomly. Only when the obstacles are within the detection range of the UAV's airborne radar, the UAV can obtain their positions.

In Figure 1, a and d represent the attack UAV and the defense UAV, respectively. $s_{ip} = (x_{ip}, y_{ip})(i = a, d)$ represents the position coordinates of the UAVs. $s_{id} = \psi_i (i = a, d)$ represents the heading angle of the UAVs. R_a and R_d represent the radar detection radius of the UAVs, respectively. (x_{tp}, y_{tp}) represents the position of the center point of the target area. R_t represents the effective radius of the target area. $s_{op}^k = (x_o^k, y_o^k)$ represents the position of the kth obstacle center point. For the convenience of research, there is a battlefield boundary in the limited confrontation environment, and neither UAV can move out of the boundary.

It is assumed that the defense UAV can obtain the position and heading of the attack UAV in real time through the ground surveillance radar, and both sides carry lidar to detect obstacles and boundary of the local environment. It is also assumed that the attack UAV knows the position of the ground target area in advance.

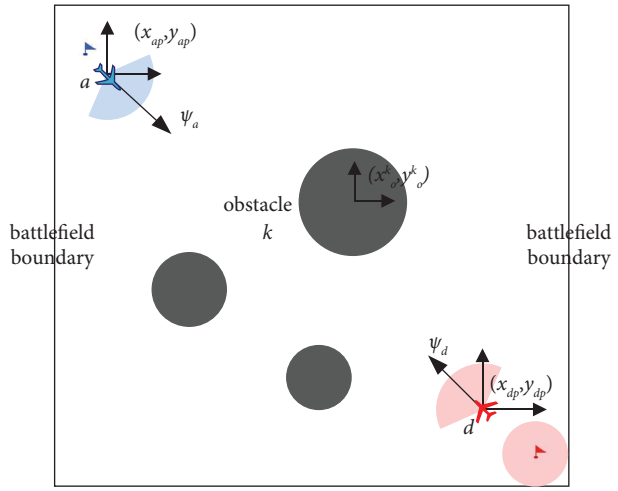


Figure 1: The scenario of 1vs1 attacking and defensive confrontation.

2.2. Kinematics Model of UAVs. It is assumed that the UAVs fly in a two-dimensional plane. The kinematics equations of the UAVs are shown in formula:

$$\begin{cases} \dot{x}_{ip} = v_i \cos \psi_i, \\ \dot{y}_{ip} = v_i \sin \psi_i, \\ \dot{v}_i = a_i, \\ \dot{\psi}_i = \omega_i, \end{cases} (i = a, d), \tag{1}$$

where v_i represents the speed of the UAVs. a_i and ω_i represent the acceleration and angular velocity of the UAVs, respectively.

$$\begin{cases} x_{\min} \le x_{ip} \le x_{\max}, \\ y_{\min} \le y_{ip} \le y_{\max}, \\ 0 \le v_i \le v_{i\max}, \\ 0 \le \psi_i < 2\pi, \end{cases} (i = a, d), \tag{2}$$
$$\begin{cases} -a_{i\max} \le a_i \le a_{i\max}, \\ -\omega_{i\max} \le \omega_i \le \omega_{i\max}, \end{cases} (i = a, d),$$

where $x_{\min}, x_{\max}$ and $y_{\min}, y_{\max}$ represent the boundary of the area. $v_{i\max}$ represents the upper limit of the UAV speed. $a_{i\max}$ represents the maximum value of the UAV acceleration. $\omega_{i\max}$ represents the maximum value of the UAV angular velocity.

The current state of the UAV i is $[x_{ip}^t, y_{ip}^t, v_i^t, \psi_i^t]$, and the state will change under the action of the acceleration a_i and angular velocity ω_i. The state $[x_{ip}^{t+1}, y_{ip}^{t+1}, v_i^{t+1}, \psi_i^{t+1}]$ at the next moment will be determined by the state transition equation as

$$\left.\begin{aligned} x_{ip}^{t+1} &= x_{ip}^{t} + v_i \cdot \Delta T \cdot \cos\left(\psi_i^t + \omega_i \cdot \Delta T\right), \\ y_{ip}^{t+1} &= y_{ip}^{t} + v_i \cdot \Delta T \cdot \sin\left(\psi_i^t + \omega_i \cdot \Delta T\right), \\ v_i^{t+1} &= v_i^t + a_i \cdot \Delta T, \\ \psi_i^{t+1} &= \psi_i^t + \omega_i \cdot \Delta T, \end{aligned}\right\} (i = a, d). \quad (3)$$

2.3. Radar Detection Model. It is assumed that both UAVs are equipped with lidar to detect the circular obstacles and enemy in the environment. As shown in Figures 2 and 3, the detection area of the UAVs is discretized into m state variables. In the figures, $R_i\,(i = a, d)$ represents the UAV radar detection radius. $\theta_i\,(i = a, d)$ represents the detection angle range. $R_o^k\,(k = 1, \cdots, N_k)$ represents the radius of the circular obstacle, where N_k represents the number of obstacles with different radius sizes. $(x_o^k, y_o^k)\,(k = 1, \cdots, N_o)$ represents the position of the obstacles, where N_o represents the total number of obstacles.

As shown in Figures 2 and 3, in order to better represent the detection state of the radar, the detection angle range of the UAV radar is discretized into $l\,(l = 7)$ directions at equal intervals. In the figure, it is represented by 7 rays, and the length of each ray is $D_n\,(n = 1, \ldots, l)$. The length of the blue ray is the maximum detection radius of the UAV radar, and the length of the red ray is the relative distance between the UAV and the obstacle or boundary detected in the corresponding direction. $x_{io}^n\,(i = a, d)\,(n = 1, \cdots, m)$ represents the ratio of D_n to the UAV radar maximum detection radius. If the ratio is closer to 1, it indicates that the UAV is farther from the obstacle or boundary in this direction. Otherwise, it indicates that the UAV is closer to the obstacle or boundary in this direction.

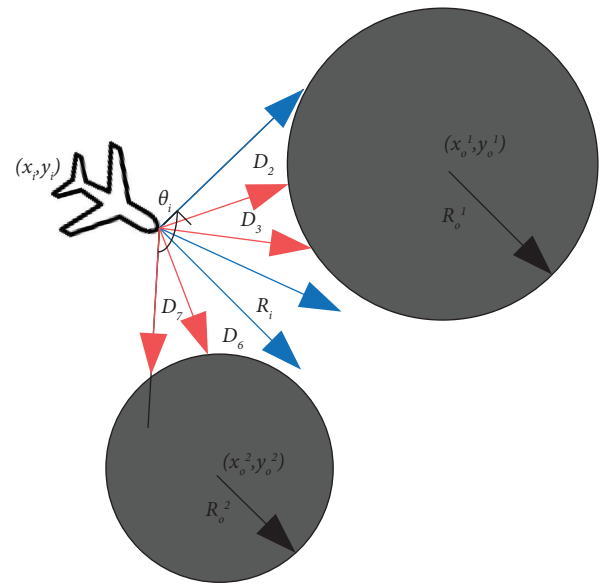


Figure 2: Obstacles detected by the UAV radar.

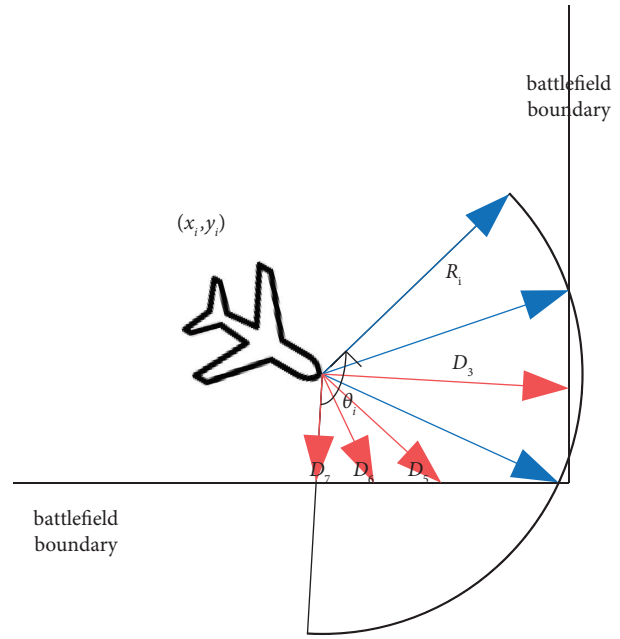


Figure 3: UAV radar detects the boundary of the battlefield.

3. 1vs1 Confrontation Maneuver Decision-Making Method Based on Reinforcement Learning

In this paper, the reinforcement learning algorithm of DDPG is used to study the 1vs1 confrontation scenarios. Before using this algorithm, it is necessary to define the state space, action space, and reward function.

3.1. State Space. The position, speed, and heading of the attack UAV a can be characterized as $[x_{ap}, y_{ap}, v_a, \psi_a]$. The discretization number l of the radar detection range is set to 7, so the detection state can be characterized as $s_{ao} = [x_{ao}^1, x_{ao}^2, x_{ao}^3, x_{ao}^4, x_{ao}^5, x_{ao}^6, x_{ao}^7]$. The attack UAV usually knows the position of the target area in advance. To simplify the input state dimension of the UAV, the position of the target is combined with the radar detection state. As shown in Figure 4, the direction corresponding to the maximum value of state quantity x_{ao}^i, $(i = 1, \ldots, l)$ in s_{ao} (there may be multiple such directions, such as the four blue ray directions in Figure 4) will be determined, and then the direction with the smallest angle with the UAV target line of sight direction will be selected as the optimal heading (such as the green ray direction in the figure) of the attack UAV. The number of this direction is marked as $c\,(1 \le c \le 7)$, and let x_{ao}^c equals to 2, which means that the attack UAV moves in this direction as much as possible.

In summary, the state of the attack UAV includes the UAV's own position, speed, heading angle, the radar's detection state, and the target's direction. Therefore, the state contains 10 dimensional data in total, which is defined as formula:

$$\begin{aligned} s_a &= \left[s_{ap}, s_{ad}, s_{av}, s_{ao}\right] \\ &= \left[x_{ap}, y_{ap}, \psi_a, v_a, x_{ao}^1, x_{ao}^2, x_{ao}^3, x_{ao}^4, x_{ao}^5, x_{ao}^6, x_{ao}^7\right]. \end{aligned} \quad (4)$$

For the defense UAV d, the status is similar to the attack UAV, which is defined as formula.

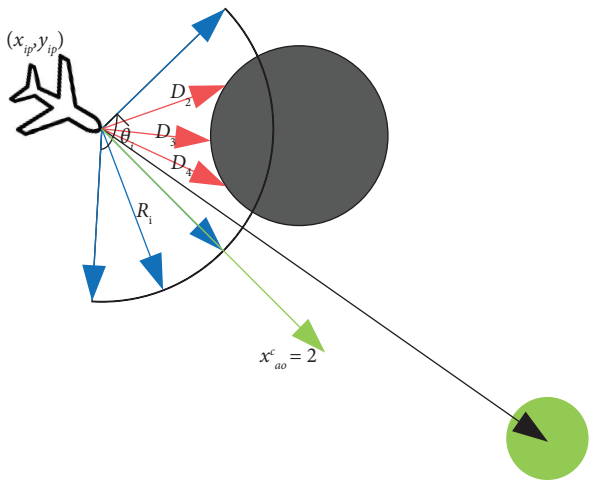


FIGURE 4: Schematic diagram of UAVs selecting the optimal direction.

$$s_d = \left[s_{dp}, s_{dd}, a_{dv}, s_{do}\right] = \left[x_{dp}, y_{dp}, \psi_d, v_d, x^1_{do}, x^2_{do}, x^3_{do}, x^4_{do}, x^5_{do}, x^6_{do}, x^7_{do}\right]. \tag{5}$$

3.2. Action Space. It is assumed that the attack UAVs have stronger maneuverability. The control inputs of both UAVs are acceleration and angular velocity, and the action space is shown as formula:

$$A = \left[a_i, \omega_i\right] (i = a, d). \tag{6}$$

3.3. The Reward Function. Reinforcement learning mostly uses sparse rewards in the field of AI games and has achieved good results [16]. However, the sparse reward cannot make the UAVs to learn efficiently at the beginning of the confrontation task.

Therefore, the reward function of this experiment is set by the combination of guided reward and sparse reward. The design of guided reward R_g is shown in the following formula:

$$\begin{cases} R_d = d_{t-1} - d_t, \\ R_h = \sum_{n=1}^{7} \left(x^n_{io} - 1\right), \\ R_v = \dfrac{v_i}{v_{i\max}}, \\ R_c = \left|\psi_{opti} - \psi_i\right|, (i = a, d), \end{cases} \tag{7}$$

where d_{t-1} and d_t represent the relative distance between the UAV and the target at time $t-1$ and t, respectively. R_d represents the variation of relative distance. R_h represents the cumulative value of the UAV radar detection state variable x^n_{io} relative to 1. R_v represents the reward of the current speed of the UAV. R_c represents the deviation of the current heading ψ_i of the UAV from the optimal heading ψ_{opti}.

$$R_g = \alpha_1 R_d + \alpha_2 R_h + \alpha_3 R_v + \alpha_4 R_c. \tag{8}$$

The design of sparse reward R_s is expressed in the following formula:

$$R_s = \begin{cases} R_1, & \text{if}\left(x_{ip} \le x_{\min}\right) or \left(x_{ip} \ge x_{\max}\right) or \left(y_{ip} \le y_{\min}\right) or \left(y_{ip} \ge y_{\max}\right), \\ R_2, & if\ dis\left(s_{ip}, s_{io}\right) \le R^k_o \left(k = 1, L, N_k\right), \\ R_3, & if\ dis\left(s_{ip}, s_{it}\right) \le R_t or R_f, \end{cases} \quad (i = a, d). \tag{9}$$

where R_1 represents the penalty for the UAV colliding with the boundary. R^k_o represents the radius of the $k-$th obstacle. $dis(\cdot)$ represents the Euclidean distance in two-dimensional space. R_2 represents the penalty for UAV colliding with the obstacle. R_t represents the radius of the target area. R_f represents the attack distance of the defense UAV. R_3 is the reward of the attack UAV to reach the target or the punishment for it being destroyed. The success signal of the defense UAV is that the attack UAV is destroyed.

3.4. The DDPG Algorithm. The DDPG algorithm is a classic reinforcement learning algorithm based on the actor-critic framework [17]. It is a deterministic policy gradient algorithm referring to the experience playback mechanism and the dual network structure in the DQN algorithm, and it realizes the direct mapping from the continuous state space to the specific high-dimensional action space through the actor network. The network architecture of DDPG is shown in Figure 5.

As shown in Figure 5, the algorithm mainly includes the interactive environment, the experience pool, and the network module of the algorithm. Before the UAV interacts with the environment, it is necessary to determine the number of layers and nodes of the network. We need to initialize the current network parameters randomly and copy the evaluated network parameters to the corresponding target network for the first time. In each step of interaction, the initial state s_t of environmental feedback is taken as the state input of the actor evaluated network, and the action

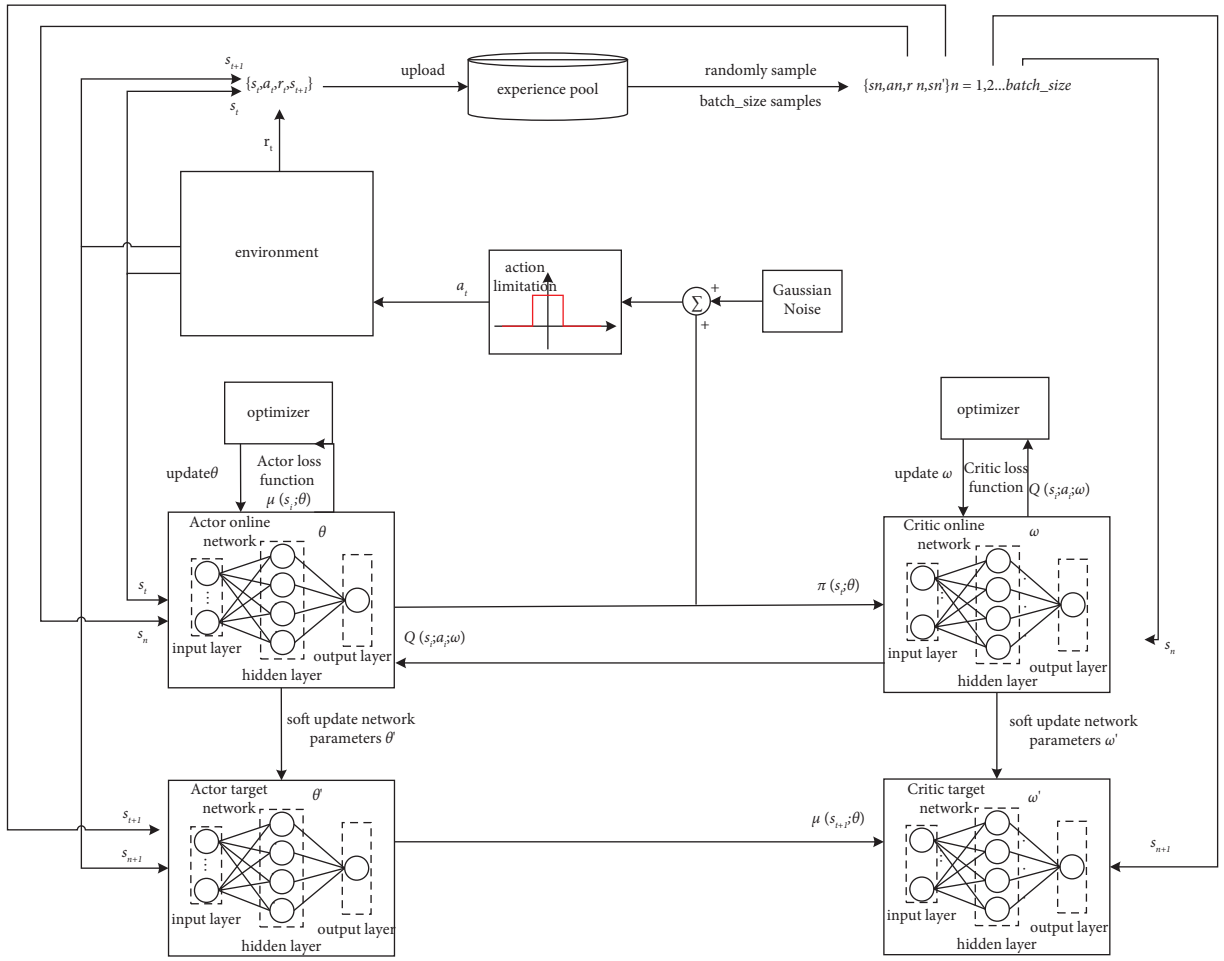


Figure 5: The DDPG algorithm network structure.

value $\mu(s_t;\theta)$ of UAV is obtained by the actor network. We need to add Gaussian noise to increase the exploration of the action space on this basis. Due to the limitation of the UAV's angular velocity, the action of the UAV is the combination of Gaussian noise and motion constraints, which is expressed in the following formula:

$$a_t = f_{\text{clip}}\left(\mu(s_t;\theta) + \mathcal{N}\right), \tag{10}$$

where f_{clip} represents the limitation function of the UAV action, $\mathcal{N}$ is Gaussian noise, which should obey the formula :

$$\mathcal{N} \sim N\left(0, \sigma^2\right), \tag{11}$$

where σ represents the variance of action noise. The state of the UAVs is determined by the state transition formula (3), and the corresponding reward is obtained according to the reward function. Then, the network training sample $[s_t, a_t, r_t, s_{t+1}]$ is obtained, and we stored it in the experience pool. If the number of samples reaches the requirements for starting training, the parameters of the network are trained according to the method of random sampling. The specific method is to randomly take m sets of sample data from the experience pool. $\{s_n, a_n, r_n, s_n'\}$ represents the $n-$ th sample. The back propagation algorithm can be used to update the evaluated network parameters.

The loss function $J(\omega)$ of the critic evaluated network is calculated as formula:

$$J(\omega) = \frac{1}{m}\sum_{n=1}^{m}\left(\left(y_n - Q(s_n;a_n;\omega)\right)^2\right), \tag{12}$$

where ω represents the parameters of the critic evaluated network, $Q(s_n, a_n;\omega)$ represents the evaluation value of the critic evaluated network of the current state and the actions performed, and y_n is defined as formula:

$$y_n = r_n + \gamma Q'\left(s_n';\mu'\left(s_n';\theta'\right);\omega'\right), \tag{13}$$

where r_n represents the reward after the UAV performs action a_n, γ represents the attenuation coefficient of the reward, and $Q'\left(s_n';\mu'\left(s_n';\theta'\right);\omega'\right)$ represents the evaluation value of the critic target network.

The parameter of the critic evaluated network is updated as formula:

$$\omega = \omega - \alpha_C \nabla_\omega J(\omega), \tag{14}$$

where α_C is the learning rate of the critic evaluated network, and $\nabla_\omega J(\omega)$ is calculated as formula:

$$\nabla_{\omega} J(\omega) = \frac{1}{m}, \sum_{n=1}^{m} \left(y_n - Q\left(s_n; a_n; \omega\right)\right) \nabla_{\omega} Q(s; a; \omega)|_{s=s_n, a=a_n}. \tag{15}$$

The parameter updating method of the actor evaluated network:

$$\theta = \theta + \alpha_A \nabla_{\theta} J(\theta), \tag{16}$$

where α_A is the learning rate of the actor evaluated network. $\nabla_{\theta} J(\theta)$ is calculated as formula:

$$\nabla_{\theta} J(\theta) = \frac{1}{m} \sum_{n=1}^{m} \left(\nabla_a Q(s, a; \omega) \,\middle|\, s = s_n, a = \mu\left(s_n; \theta\right) * \nabla_{\theta} \mu(s; \theta) \,\middle|\, s = s_n\right). \tag{17}$$

The parameters of the actor target network and the critic target network are updated through a soft update method. Such a slow updating process makes the training process more stable. The process of updating as formula:

$$\begin{cases} \omega' \leftarrow \tau\omega + (1 - \tau)\omega', \\ \theta' \leftarrow \tau\theta + (1 - \tau)\theta', \end{cases} \tag{18}$$

where τ represents the soft update coefficient.

4. Confrontation Maneuver Decision-Making Method Based on Transfer Reinforcement Learning

4.1. Transfer Learning. It is common that the trained strategies of deep reinforcement learning can only be applied to specific environments. As the complexity of the task increases, it is more difficult for the strategies to apply to new scenarios. Transfer learning is an algorithm that can make full use of the knowledge and experience that could be gained in previous related tasks and applied to new tasks [18]. Transfer learning has a strong ability of model generalization. This idea can also be reflected in daily learning. For example, people use their mother tongue to learn foreign languages. People who are familiar with C++ can quickly learn other programming languages. A solid mathematical foundation is helpful for learning professional courses. All those mentioned previously are based on the previous knowledge to continue learning to solve new problems. Different scenarios or tasks in transfer learning are generally called domains. The domains that have learned experience and knowledge are called source domains, and the domains to be learned are called target domains. The definition of transfer learning is as follows.

Based on the given source domain D_s and source domain task T_s, the knowledge K_s learned in the source domain is used to learn K_t in the target domain D_t to complete the task T_t of the target domain.

The idea of transfer learning can also be applied to reinforcement learning. In this paper, the parameter transfer method of transfer learning is used to deal with the scenario of 1vs1 confrontation. The core idea of this method is that the agent learns in a simple task firstly, and if the learned strategy is getting better, the difficulty of the agent's task can be gradually increased. The agent strategies which are suitable for simple tasks will be transferred to more complex tasks to continue learning. This process can reduce the difficulty of exploring complex tasks effectively and avoid the problems caused by sparse rewards successfully.

4.2. Confrontation Maneuver Decision-Making Method Based on Transfer Learning. Aiming at the 1vs1 confrontation model established in Section 2, this paper lets the UAV learn in a simple environment firstly and gradually transfer the learned experience to more difficult mission scenarios. In the learning process, when one side's strategy is to be trained, the other side's strategy trained in the previous scenario will be used initially. After the training is completed, the strategy of this training will be used to train the other side. We can use alternate training methods to improve the strategy of the UAVs from the two sides progressively. The specific training process is shown in Table 1.

The pseudocode of the strategy training algorithm for DDPG-based 1vs1 confrontation is shown in Table 2.

5. Simulation Experiment

5.1. Experimental Environment and Parameter Settings. The experimental software package is PyCharm 2020.1 and Anaconda3. The experimental program is based on the Python language. The settings of the confrontation scenario are shown in Figure 1. This paper uses the standard GUI writing library named Tkinter of Python to build a two-dimensional environment. The neural network is constructed by the PyTorch module, and the version of it is 1.8.1.

The specific parameters of the experimental environment are introduced as shown in Table 3. The obstacles are distributed in each episode randomly, and they are limited in the specific area.

The simulation step ΔT is 1s. The PyTorch module is used to build the neural networks of this paper, which all are 3-layer fully connected feedforward neural networks. The number of neurons in each layer of the actor network is [10, 128, 64, 2], and the number of neurons in each layer of the critic network is [12, 128, 64, 1]. The activation function is the ReLU function. To ensure that the action output by the actor network is reasonable, the value output by the final output layer is multiplied by the maximum action limit value by the tanh function. The network parameter optimizer uses the AdamOptimizer module. To reduce the burden of the neural network and speed up the training of the network, the state input of both UAVs will be processed in advance. In this paper, the position coordinates are divided by the maximum boundary length, and the angle is limited to $[0, 2\pi$ and divided by 2π.

The algorithm training parameter settings are shown in Table 4.

In addition, there are two specific conditions of episode termination in this experiment. One is the number of time steps that the UAV interacts with the environment reaching

Table 1: The training method of 1vs1 confrontation maneuver decision-making based on transfer learning.

Step	Different scenario requirements from simple to difficult
1	Set that there is only one attack UAV in the battlefield environment and train the UAV to avoid collision with obstacles and boundaries until it can reach the target area
2	Use the strategy of the attack UAV in step 1 and add a defense UAV in the environment. The maneuverability of the defense UAV is not as good as that of the attack UAV. The defense UAV is trained to avoid collision with obstacles and boundaries, and we perform the task of intercepting and attacking the attack UAV
3	Use the strategy of the defense UAV trained in step 2. It is set that the attack UAV can detect the defense UAV in advance. Use the transfer strategy and the nontransfer strategy for training, respectively

Table 2: Pseudocode of the 1vs1 countermeasure algorithm based on DDPG transfer learning.

Pseudocode of the 1vs1 countermeasure algorithm based on DDPG
(1) Randomly initialize the parameters θ and ω of the evaluated network of actor and critic. Initialize experience pool D with a capacity of M. The number of initialization batch samples is batch_size. The initial attenuation factor is γ. The initial soft update coefficient is τ. The initial Gaussian noise variance is noise. The maximum number of initialization rounds is Max_Episode. The maximum number of initialization steps per round is Max_Step
(2) For episode = 1 to Max_Episode do
(3) Obtain the respective state s_t of both sides according to the initial settings of the simulation environment
(4) For $t = 1$ to Max_Step do
(5) Enter s_t as the input of the actor evaluated network to get the UAV's action $a_t = f_{clip}(\mu(s_t;\theta) + \mathcal{N})$, where f_{clip} represents the function of the upper and lower limits of the UAV's restricted action
(6) If there is an enemy UAV, the enemy UAV takes the corresponding confrontation maneuver decision-making according to the description in Table 2and we need to execute action a_{ct} and update its own state s_{ct} to $s_{c(t+1)}$
(7) Select the action according to the ε − gree dy strategy, that is, training the UAV to randomly select the action within the action range with a certain probability or the action a_t of step 5, then obtain the corresponding reward value r_t, and change the environment state to s_{t+1} at the next moment
(8) Store the sample data $[s_t, a_t, r_t, s_{t+1}]$ of the interaction between the UAV and the environment in the experience pool D
(9) Randomly select batch_size of training sample data $[s_n, a_n, r_n, s_n']$ from experience pool D
(10) Calculate the loss function of the critic evaluated network and update the parameter ω of the critic evaluated network through backpropagation to minimize the loss function
(11) Calculate the loss function of the actor evaluated network and update the parameter θ of the actor evaluated network through backpropagation loss function
(12) Update the parameters θ' and ω' of the actor and critic target network for every step C
(13) end for
(14) end for

Table 3: Design of the training method for 1vs1 confrontation strategies.

Experiment environment parameters	Parameter values
Task area boundary $x_{\min}, x_{\max} y_{\min}, y_{\max}$	$[0, 100] \times [0, 80]$ (km)
Number of obstacles, N_o	9
The radius R_o^k and the number of different kinds of obstacles	4 km (3), 5 km (3), 6 km (3)
Area where obstacles appear randomly	$[15, 85] \times [15, 65]$ (km)
Radar detection range of attack UAV, $R_a * \theta_a$	$12\,\text{km} \times 120^\circ$
Radar detection range of defense UAV, $R_d * \theta_d$	$8\,\text{km} \times 120^\circ$
Discrete number of detection range, m	7
Maximum speed of attack UAV, $v_{a\max}$	340 m/s
Maximum speed of defense UAV, $v_{d\max}$	300 m/s
The upper limit of the acceleration of the attack UAV, $a_{a\max}$	$20\,\text{m/s}^2$
The upper limit of the acceleration of the defense UAV, $a_{d\max}$	$20\,\text{m/s}^2$
The maximum angular velocity of the attack UAV, $\omega_{a\max}$	$\pi/15.7$
The maximum angular velocity of the defense UAV, $\omega_{d\max}$	$\pi/22.6$
The initial position coordinates and heading angle of the attack UAV	[2.5, 2.5] (km), $\pi/4$
The initial position coordinates and heading angle of the defense UAV	[97.5, 77.5] (km), $5\pi/4$
The center point coordinate (x_{tp}, y_{tp}) and radius R_t of the target area	[95, 75] (km), 5 (km)
The attacking radius of the defense UAV, R_f	1 km

the maximum number of time steps per episode. The other is that the UAVs collide with obstacles and boundaries or successfully achieve their required targets. For sparse rewards, if the UAV collides with an obstacle or boundary, the rewards R_1 and R_2 are set to −10. If the UAV completes the required task, the reward R_3 is set to 10. For the guided

Table 4: Algorithm training parameter settings.

Parameters	Description	Values
Discount factor, γ	Decay factor of cumulative reward	0.95
Inertial update rate, τ	Calculate the parameters of the target network by the soft update method	0.01
Experience pool size, M	The sample size of the experience pool, which is the source of training samples	1e5
Number of samples per batch, batch_size	The number of samples used for learning in each batch, randomly chosen from the experience pool	64
Actor network learning rate, α_A	Update the parameters of the actor network	1e − 5
Critic network learning rate, α_C	Update the parameters of critic network	1e − 4
Exploration rate, ε	The exploration rate of the agent's random actions	$0.1 \longrightarrow 0$
Action noise, σ (variance of the normal distribution)	Action noise variance	$3 \longrightarrow 0$
Maximum number of rounds, Max_Episode	Total number of rounds in training	1500
Maximum number of steps per episode, Max_Step	Maximum number of steps in each episode	500
Number of steps between parameter soft update, c	Number of steps between parameter soft update	10

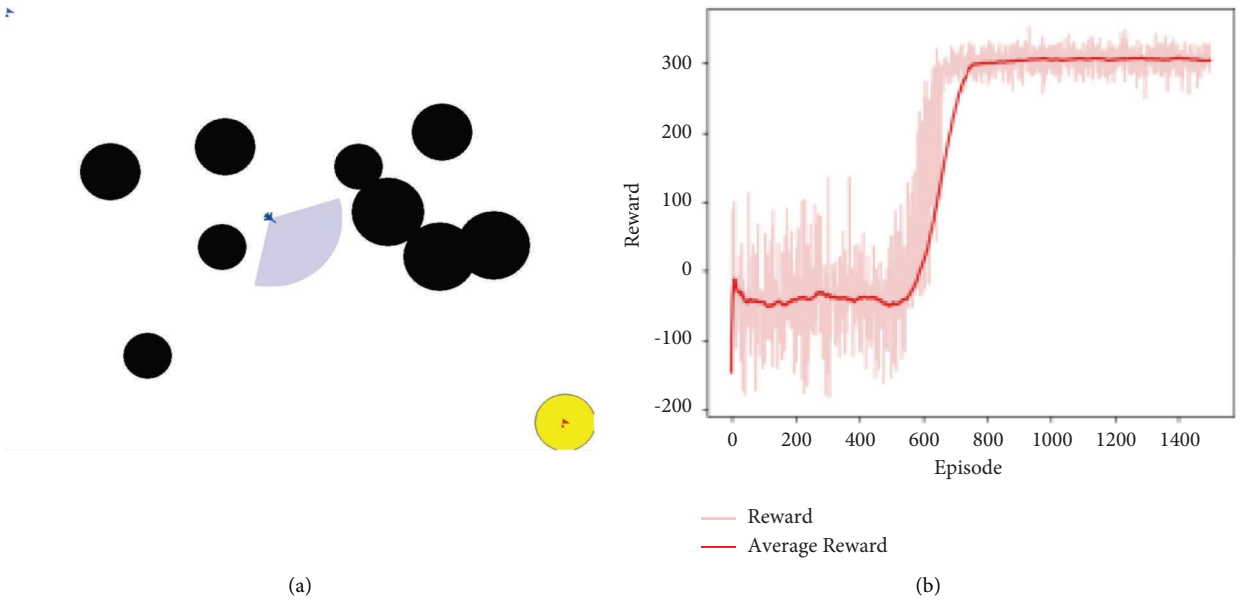


Figure 6: (a) The training environment of step 1 and (b) the average training reward curve.

rewards, different reward coefficients α_1, α_2, α_3, and α_4 are set to 0.3, 0.2, 0.2, and 0.3 in formula (8).

5.2. Training Result Analysis. The purpose of the reinforcement learning algorithm is to train the agent's strategy to maximize its cumulative reward expectation. The evaluation index of training results can generally be the average reward value of the episode. It is a graph which shows the change of the reward value obtained by the agent training with the number of the episodes. The faster the reward value rises and the more stable and higher the reward value converges, the better the training effect is. This paper uses the average reward of the last 100 episodes as the final average reward value. If there are less than 100 episodes from the beginning of training, only the average reward value of the existing rounds will be used.

According to the training steps in Table 2, we can use the strategies trained by the UAVs in the simple task scenarios in step 1 and step 2 to in the scenario of step 3. In step 3, the task difficulty increases gradually, and the transfer and nontransfer methods are used for comparative analysis, respectively. The migration methods are based on the network parameters of 1500 episodes previously trained. The details are as follows:

As shown in Figure 6(a), the offensive UAV has prior information of its starting position and goal position in the environment of step 1, and it is trained to avoid obstacles and boundaries. After 1500 episodes of training, the reward function curve of the attacking UAV is shown in Figure 6(b).

The abscissa of Figure 6(b) represents the number of training episodes, and the ordinate represents the average rewards of the most recent 100 episodes. It can be seen from the figure that the UAV is not clear about what it is

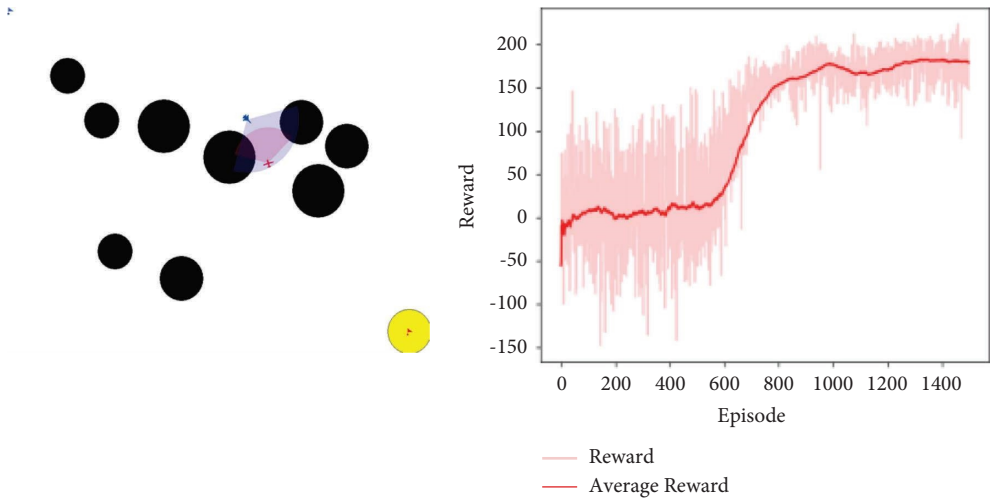


Figure 7: (a) Step 2 training environment and (b) training average reward curve.

going to do at the beginning. It is just exploratory interaction with the environment, and the data of these interactions are extremely useful. After the experience pool is filled (about 520 rounds), as the algorithm begins to train, the reward curve begins to rise gradually, and it starts to show a trend of convergence after 720 episodes with good stability.

As shown in Figure 7(a), in step 2, the defense UAV uses the trained strategy of the attack UAV in step 1 to avoid obstacles and boundaries, and on this basis, the defense UAV is trained to intercept the attack UAV. If the distance from the attack UAV to the target location (yellow) is less than the distance from the defense UAV to the target location, the defense UAV cannot complete the interception and strike mission, it is due to the fact that the maneuverability of the attack UAV is better than the defense UAV. Therefore, the episode will be terminated early, and it means that the attack UAV completes its task successfully and the defense UAV fails to defend.

The abscissa of Figure 7(b) represents the number of training episodes, and the ordinate represents the average rewards of the most recent 100 episodes. It can be seen from the figure that after the experience pool is filled (approximately 580 episodes), the training curve begins to gradually rise and begins to converge around 850 episodes with good stability.

In step 3, the defense UAV used the defensive strategy trained in step 2. It is assumed that the attack UAV can detect the defense UAV by its airborne lidar and take the defense UAV as obstacles to avoid. Then, the attack UAV is trained by the strategy of the attack UAV trained in step 1 and the nontransfer method, respectively. The training results are shown in Figure 8. Similarly, if the distance between the attack UAV and the target position (yellow) is less than the distance between the defense UAV and the target position, the episode will be terminated in advance, and it will be judged that the attack of the attack UAV is successful and the defense of the defense UAV is failed.

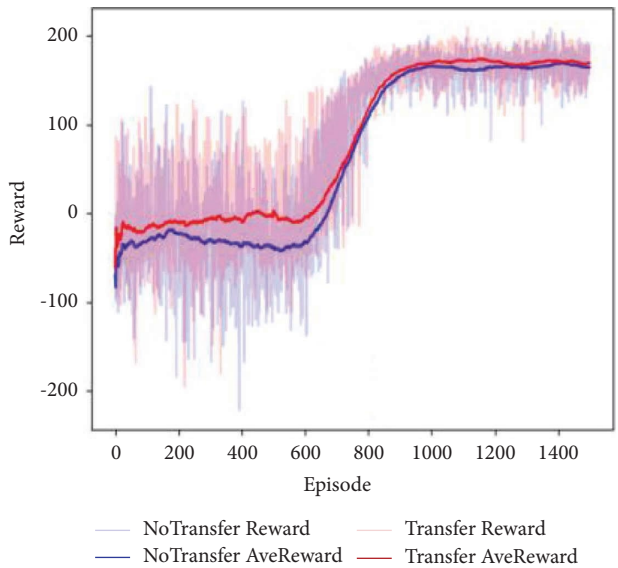


Figure 8: The average reward curve of step 3.

The abscissa of Figure 8 represents the number of training episodes, and the ordinate represents the average rewards of the most recent 100 episodes. It can be seen from the figure that both transfer and nontransfer methods can converge within a certain period of time. In contrast, the transfer method has a better round reward value before training and a higher reward value after convergence.

5.3. Experiment Result Analysis. In this paper, the training results after 1500 episodes are tested by Monte Carlo for 10000 times. The parameters of the trained actor evaluated network are set in the UAVs.

Three different scenarios are tested. The effects of this test are shown in Figures 9–11 (each small circle represents the current position of the UAV at every time step, which is 5 s.). The test result data are shown in Figure 12.

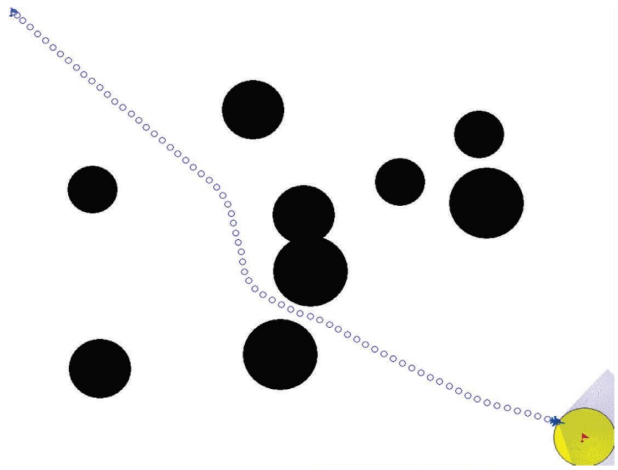
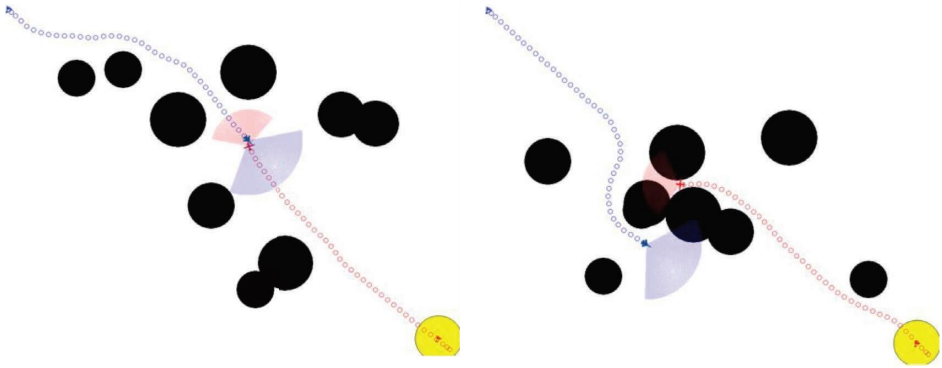
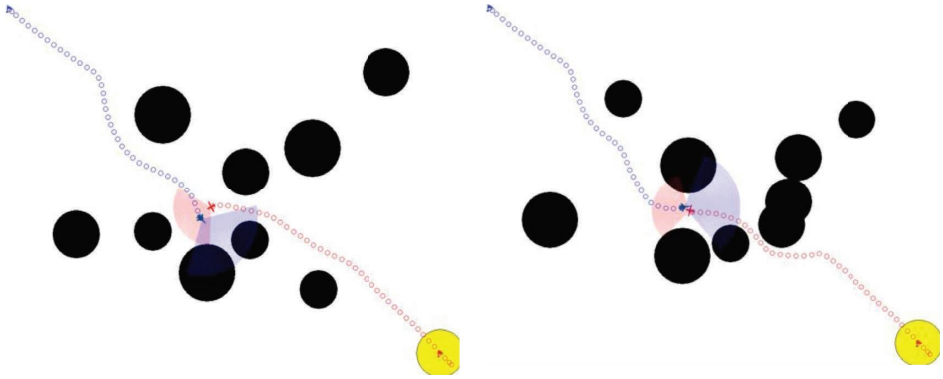


Figure 9: Training results of test step 1 (offensive success).



(a) (b)

Figure 10: Training results of test step 2 (a) (defense success) and (b) (defense failure).



(a) (b)

Figure 11: Training results of test step 3 (a) (offense success) and (b) (offense failure).

The test results of step 1 show that the attack UAV trained by the presented method can avoid obstacles successfully. The final strategy can achieve stable convergence, and the success rate of avoiding obstacles and reaching the designated area is 99.29%.

The test results of step 2 show the success and failure of the defense UAV, respectively. As shown in Figure 12, both UAVs can avoid obstacles successfully. The defense success rate of the defense UAV is 55.54%. Most of the cases of defense failure are that the two UAVs evade from different

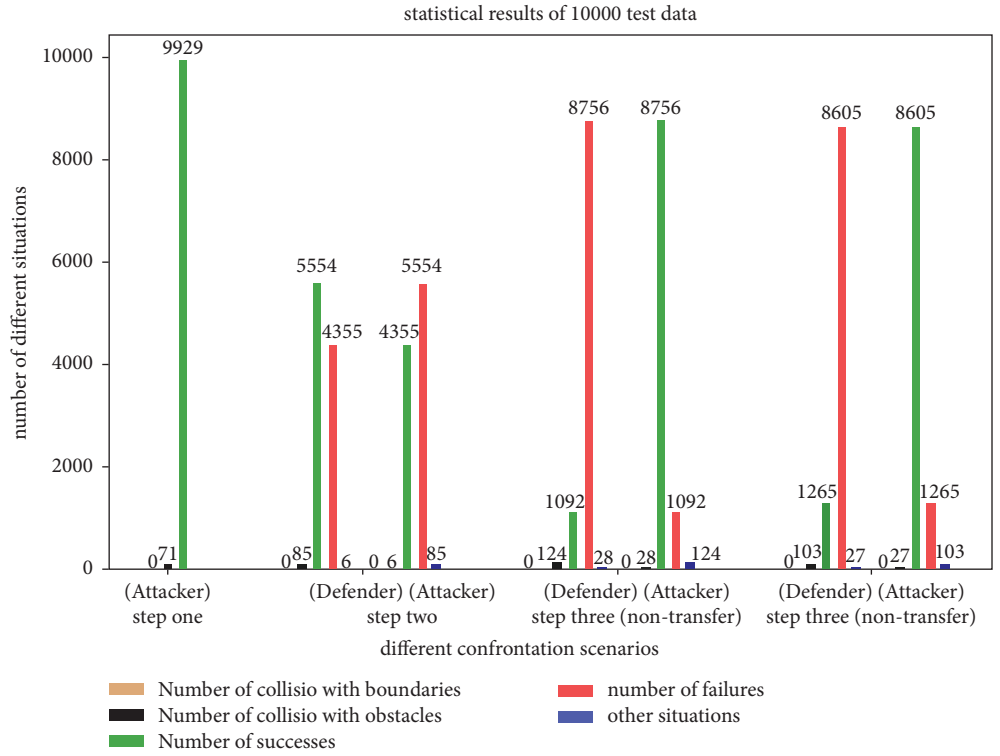


Figure 12: Statistical graph of test data results.

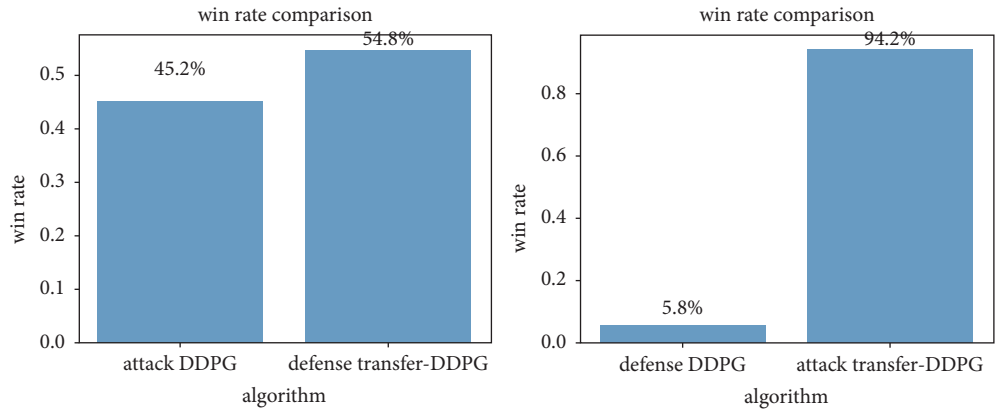


Figure 13: Win rate comparison between the transfer learning algorithm and the DDPG algorithm.

sides of the obstacle, so the defense UAV cannot intercept effectively.

The test results of step 3 show the success and failure of the attack UAV, respectively. Compared with the results of the nontransfer method (86.05%), the transfer reinforcement learning method proposed in this paper can increase the offensive success rate (87.56%). Moreover, the results of both sides are greatly improved compared to step 2 (43.55%).

1000 Monte Carlo experiments are conducted between the attackers and defenders trained by the traditional MADDPG algorithm and the attackers and defenders trained by the DDPG algorithm based on transfer learning. The experiment results are shown in Figure 13.

As shown in Figure 13, on the attack side, the winning rate of the transfer learning algorithm is 94.2%, which is significantly higher than MADDPG's 45.2% winning rate, while on the defense side, the winning rate of the transfer learning algorithm is 54.8% which is also significantly higher than MADDPG's 6.8% winning rate. These results demonstrate the effectiveness and superiority of the algorithm proposed in this paper.

6. Conclusion

In this paper, reinforcement learning is applied to the UAV confrontation problem, and a 1vs1 confrontation method is designed based on the DDPG algorithm. Based on the model, transfer learning is introduced to train the UAVs.

The results show that the proposed method can make training converge faster and can increase the offensive success rate.

Due to its limited mobility, the task success rate of a single defense UAV is not high. Therefore, the next step will continue to study the maneuver decision-making of multiple defense UAVs against a single offensive UAV on the basis of the method proposed in this paper. In the far future, optimizing the framework structure of the algorithm or complicating the environment and adding more UAVs to the scenario will be the development direction.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This research was funded in part by the Aeronautical Science Foundation of China under Grant number 2020Z023053001.

References

[1] Y. W. Zhu, *Research on the Key Technology of Unmanned Arial Vehicle*, Beijing Institute of Technology, Beijing, China, (in Chinese), 2016.

[2] D. L. Luo, Y. Xu, and J. P. Zhang, "New progresses on UAV swarm confront," *Science and Technology Review*, vol. 35, no. 7, pp. 26–31, 2017, (in Chinese).

[3] R. Isaacs, "Differential games a mathematical theory with applications to warfare and pursuit, control and optimization," *Physics Bulletin*, vol. 17, no. 2, pp. 1-2, 1966.

[4] X. Wang, W. J. Wang, K. P. Song, and M. W. Wang, "UAV air combat decision based on evolutionary expert syste," *Ordnance Industry Automation*, vol. 38, no. 1, pp. 42–47, 2019, (in Chinese).

[5] K. D. Huang, D. Y. Zhou, C. H. Zhou, and J. Yu, "UCAV close range combat occupying strategy and pro-gram," *Command Control and Simulation*, vol. 35, no. 4, pp. 44–48, 2013, (in Chinese).

[6] V. Isler, S. Kannan, and S. Khanna, "Randomized pursuit-evasion in a polygonal environment," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 875–884, 2005.

[7] X. Chen and Y. F. Wang, "Study on multi-UAV air com-bat game based on fuzzy strategy," *Applied Mechanics and Materials*, vol. 495, pp. 1102–1105, 2014.

[8] W. Z. Ran, *Design and Implementation of Coopera-Tive Adversarial System Based on Swarm Intelligence for Unmanned Aerial Vehicles*, University of Elec-tronic Science and Technology, Chengdu, China, (in Chinese), 2020.

[9] Y. Z. Zhang, J. L. Xu, K. J. Yao, and J. L. Liu, "Pursuit mis-sions for UAV swarms based on DDPG algorithm," *ACTA AERONAUTICAET ASTRONAUTICA*, vol. 41, no. 10, p. 324000, 2020 (in Chinese).

[10] C. Chen, L. Mo, D. Zheng, Z. H. Cheng, and D. F. Lin, "Cooperative attack-defense game of multiple UAVs with asymmetric maneuverability," *ACTA AERONAUTICAET ASTRONAUTICA SINICA*, vol. 41, no. 12, p. 324152, 2020 (in Chinese).

[11] S. Z. Xuan and L. J. Ke, "Stduy on attack-defense countermeasure of UAV swarms based on multi-agent Reinforcement learning," *Radio Engineering*, vol. 51, no. 5, pp. 360–366, 2021, (in Chinese).

[12] Z. Huo, S. Dai, and M. Yuan, "A reinforcement learning based multiple strategy framework for tracking a moving target," in *Proceedings of the 2020 IEEE/ASME In-ternational Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 1292–1297, MA, USA, July 2020.

[13] X. Liu and H. Wang, "UAV game and adversarial techniques based on generative adversarial networks," *Command Information Systems and Technology*, vol. 12, no. 5, pp. 1–5, 2021.

[14] Y. Wen and X. Shi, "An intelligent decision method for UAV-clustered against multi-coupled tasks," *Journal of Astronautics*, vol. 42, no. 4, pp. 504–512, 2021.

[15] E. Wang and J. Guo, "To proved the objective gain function," *Journal of Nanjing University of Aeronautics & Astronautics*, vol. 53, no. 6, pp. 888–897, 2021.

[16] Z. J. Hu, K. F. Wan, X. G. Gao, Y. W. Zhai, and Q. L. Wang, "Deep reinforcement learning approach with multiple experience pools for UAV's autonomous motion planning in complex unknown environments," *Sensors*, vol. 20, no. 7, pp. 1890–1920, 2020.

[17] R. Yang, J. P. Yan, and X. Li, "A survey on sparse reward algorithms in reinforcement learning theory and experiment," *CAAI Transactions on Intelligent Systems*, vol. 47, pp. 1–15, 2021, http://kns.cnki.net/kcms/detail/23.1538.TP.20200921.1556.004.html.

[18] T. P. Lillicrap, J. J. Hunt, P. Alexander et al., *Continuous Control with Deep Rein-Forcement Learning*, 2020, https://arxiv.org/abs/1509.02971.