Journal of Systems Engineering and Electronics
Vol. 34, No. 5, October 2023, pp.1211 – 1224

A UAV collaborative defense scheme driven by DDPG algorithm

ZHANG Yaozhong[1,*], WU Zhuoran[1], XIONG Zhenkai[2], and CHEN Long[3]

1. School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710072, China;
2. College of New Energy and Intelligent Connected Vehicle, Anhui University of Science & Technology, Hefei 231131, China;
3. China Research and Development Academy of Machinery Equipment, Beijing 100089, China

Abstract: The deep deterministic policy gradient (DDPG) algorithm is an off-policy method that combines two mainstream reinforcement learning methods based on value iteration and policy iteration. Using the DDPG algorithm, agents can explore and summarize the environment to achieve autonomous decisions in the continuous state space and action space. In this paper, a cooperative defense with DDPG via swarms of unmanned aerial vehicle (UAV) is developed and validated, which has shown promising practical value in the effect of defending. We solve the sparse rewards problem of reinforcement learning pair in a long-term task by building the reward function of UAV swarms and optimizing the learning process of artificial neural network based on the DDPG algorithm to reduce the vibration in the learning process. The experimental results show that the DDPG algorithm can guide the UAVs swarm to perform the defense task efficiently, meeting the requirements of a UAV swarm for non-centralization, autonomy, and promoting the intelligent development of UAVs swarm as well as the decision-making process.

Keywords: deep deterministic policy gradient (DDPG) algorithm, unmanned aerial vehicles (UAVs) swarm, task decision making, deep reinforcement learning, sparse reward problem.

DOI: 10.23919/JSEE.2023.000128

1. Introduction

Since its emergence, unmanned aerial vehicle (UAV) has been widely used in People's daily life and scientific research because its good performance and great application potential. Besides, UAV has natural advantages in carrying out military missions because of its concealment and security [1]. However, with the continuous advancement of the application of UAVs in the military field, it is difficult to guarantee the flexibility and mission completion rate of a single UAV in the execution of tasks. Therefore, the use of multiple UAVs to form a swarm and utilizing UAVs swarm to perform related military tasks has become one of the important development trends in the militarization applications of UAVs [2]. Recently, some researchers have worked on the intelligence algorithms of the UAVs swarm, such as the ant colony optimization (ACO) algorithm and the wolf group algorithm (WGA) to realize the command-and-control process of the UAV swarm [3,4]. However, these methods have to take a long time to compute, lack flexibility, and have the weakness of low degree of intelligence. In recent years, the emerging field of artificial intelligence, with its powerful non-linear processing ability and the ability of perception and understanding of high-dimensional information, is expected to enable UAV swarm to have enough intelligence to complete tasks in a complex environment on the battlefield.

At present, some scholars have used artificial intelligence methods to carry out exploratory studies on relevant issues of the UAVs swarm. For example, Chandarana et al. applied classification methods in artificial intelligence to study the human-machine interaction mode of UAV and compared the interactive control methods of UAV based on mouse, voice, and gesture [5,6]. Xu et al. developed deep reinforcement learning to research the autonomous target area coverage problem with UAVs swarms, which are used to solve the task planning problem and the challenge of high-dimensional state space under the joint action of multiple UAVs [7,8]. Wang et al. proposed a utility maximization approach for tuning the reward generation of a multi-arm bandits (MAB)-based reinforcement learning approach for selecting energy and processing optimized data offload paths between a source

Manuscript received November 12, 2021.
*Corresponding author.
This work was supported by the Key Research and Development Program of Shaanxi (2022GY-089) and the Natural Science Basic Research Program of Shaanxi (2022JQ-593).

and target UAV in a decentralized edge UAV swarm [9]. Li et al. [10,11] used deep reinforcement learning methods to investigate the relevant factors that affect the autonomous air combat on UAV, which provides a theoretical basis for future intelligent air combat. The reinforcement learning approach is also applied to the resource allocation problem [12,13], and the global deep reinforcement learning model can be directly downloaded to the newly activated agent, which avoids the time-consuming training process. Wang et al. used deep reinforcement learning for pursuing an omni-directional target with multiple, homogeneous agents that are subject to unicycle kinematic constraints. With shared experience to train a policy for a given number of pursuers, executed independently by each agent at run-time, they achieved good training results [14]. Liu et al. employed the deep neural network as the function approximator, and combined it with Q-learning to achieve the accurate fitting of action-value function for the intelligent tactical decision problem [15]. Price et al. leveraged the deterministic policy gradient algorithms in an actor-critic construct using artificial neural networks as nonlinear function approximators to define agent behaviors in a continuous state and action space [16]. They used the method in target defense differential game and gained optimal results in both the single and multi-agent cases [16].

Moreover, some experiments on the application of artificial intelligence in UAV swarm have been undertaken in many countries, and the United States has also experimented with many UAV swarm projects. Especially in 2016, the UAV swarm experiment conducted by the U.S Army in California successfully applied artificial intelligence to the action decision-making of UAV swarm, and this experiment realized UAV swarm autonomous collaboration in the air, formed UAV cluster formation, and completed the scheduled tasks, which have been fully embodied the drone of the cluster's decentralized, self-independence and autonomous. These research results show that the ad-hoc network and task in UAV swarm decision-making is ready for commercial use [17]. Hu et al. [18] presented an actor-critic, model-free algorithm based on the deterministic policy gradient (DPG) that can operate over continuous action spaces. Based on the DPG algorithm [19], they also combined the actor-critic approach with insights from the recent success of deep Q network (DQN) [20]. The model-free approach, deep DPG (DDPG), proposed by Timothy et al. requires only a straightforward actor-critic architecture and learning algorithm with very few "moving parts", and it is easy to implement and scale to more difficult problems and larger networks.

This paper investigates the UAV swarm defense mission using the DDPG algorithm. Due to the use of neural networks in DDPG, its powerful simulation ability makes the state space and action space of agents be extended from limited discrete type to infinite continuous type, which is closer to the real scene of UAVs defense missions. We successfully overcome the "sparse reward" problem by setting the reward function to obtain the appropriate reward value before an episode ends, thus making the algorithm easier to converge. At the same time, we also add random noise elements to the act selection to improve the random exploration ability of the UAV in the early training, which can obtain more abundant experience.

2. Task description

Among the numerous UAVs swarm tasks, the defense task is always the typical one of them. The defense task is to deploy the UAVs cluster around the defense target to prevent the possible attack from the incoming enemy and achieve the defense mission of the target.

2.1 Environment description

In this paper, a continuous two-dimensional battlefield map is constructed to serve as the exploration environment for UAV swarm defense task. The position of UAVs, incoming enemy, and the defended target are represented by continuous coordinate position, and the original state of the defensive task is shown in Fig. 1.

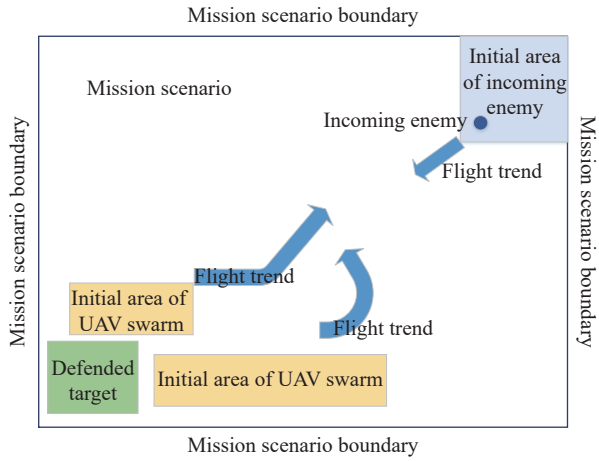


Fig. 1 Schematic diagram of UAV swarm defensive mission

As shown in Fig. 1, there are insurmountable battlefield boundaries in the environment. It is stipulated that the defended target is generated in the lower left part of the environment and moves horizontally to the right and

assumed that the movement of the defended target is not affected by the movement state of the incoming enemy and the UAV swarm. The initial region of the UAV swarm is on both sides of the defended target. The UAV swarm is randomly generated from the two initial regions of the UAV and it is accompanied by a certain random initial speed. In the upper right, there is the initialization region of the incoming enemy. The incoming enemy is randomly generated in this area.

It is assumed that there is only a single incoming enemy in the mission, and the incoming enemy aims to destroy the defended target. The incoming enemy knows the position and speed of the defended target, so it can maneuver according to the two guidance methods, and the movement of the incoming enemy will not be affected by the movement state of the UAV swarm.

The objective of the UAV swarm is to eliminate the incoming enemy and achieve defensive missions. To realize the destruction of UAVs to an incoming enemy, it is assumed that the UAV carries a weapon unit. To simplify the model, there is no limit on the number of weapons. It is only possible for an incoming enemy to be destroyed by UAVs when it enters the attack range of UAVs.

Therefore, there are three end states for UAV swarm to perform defense tasks: the incoming enemy is destroyed and the defense mission is completed; the defended target is destroyed and the defense mission fails; UAVs hit battlefield boundary and the defense mission fails.

2.2 Incoming enemy model

As the purpose of the incoming enemy is to destroy the defended target, the movement of the incoming enemy should be modeled. It is assumed that the position, speed, and other information of the defended target can be obtained by the incoming enemy, and the movement of the incoming enemy is guided through the information. It is known that the movement speed of the incoming enemy is 1.5 times the maximum movement speed of the UAV's. In other words, $v_{\text{enemy}} = 1.5\max(v_{\text{agent}})$. The movement model of the incoming enemy is constructed based on the fire-control attack seeker which has two kinds of motion rules.

(i) Pure tracking attack

Pure tracking attack is an easy and common attack method, which is often used as the guidance method of infrared-guided missiles and other controllable weapons. The core of the pure tracking guidance method requires that the velocity vector of the incoming enemy always points to the center of the defended target at every moment, as shown in Fig. 2.

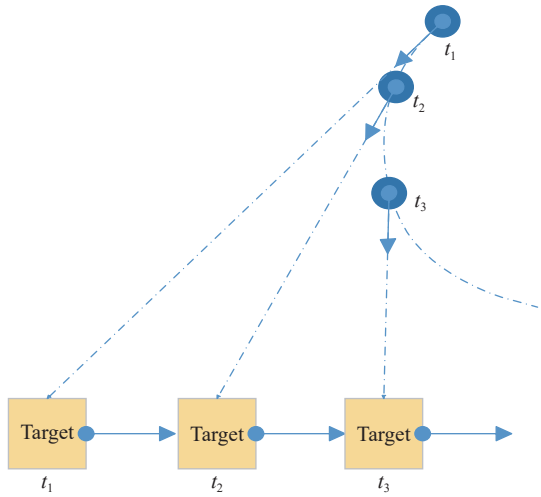


Fig. 2 Schematic diagram of pure tracking attack

The speed direction of the incoming enemy always points to the defended target, that is,

$$\frac{\boldsymbol{v}}{|\boldsymbol{v}|} = \frac{\boldsymbol{d}}{|\boldsymbol{d}|} \tag{1}$$

where $\boldsymbol{v}$ represents the velocity vector; $\boldsymbol{d}$ represents the distance vector from the incoming enemy to the defended target. Given that the velocity of the incoming enemy is a constant as v_{enemy}, the movement strategy of the incoming enemy in the pure tracking attack mode can be obtained.

(ii) Pure collision attack

Pure collision attack is another common attack guidance method, which is often used for weapons that cannot be controlled autonomously after launching and the main guidance method of tactical contact with the enemy. Its core is that the velocity vector of the incoming enemy always points to the predicted impact point between the incoming enemy and the defended target, as shown in Fig. 3.

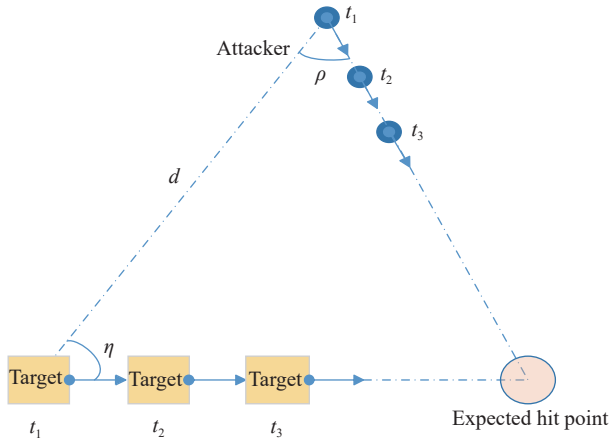


Fig. 3 Schematic diagram of a pure collision attack

According to the characteristics of pure collision guidance, in the process of navigation for an incoming enemy, it is necessary to calculate the expected impact point through the initial position, speed of the target, initial position, and speed of the defended target in the process of navigation. Then the velocity vector of the incoming enemy always points to the expected impact point with a constant velocity. And according to the characteristics of a pure collision attack, the speed of the incoming enemy always points to the defensive target relative to the defensive target's velocity.

According to the sine theorem, it can be obtained that

$$\frac{\sin\eta}{\sin\rho}=\frac{v_{\text{attacker}}t}{v_{\text{target}}t}=\frac{v_{\text{attacker}}}{v_{\text{target}}} \tag{2}$$

where ρ is the angle between the line connecting the positions of the attacker and the target and the velocity vector of the attacker, η is the angle between the line connecting the positions of the target and the attacker and the velocity vector of the target, v_{attacker} is the speed of the attacker, and v_{target} is the speed of the target.

It is assumed that the speed ratio between the incoming enemy and the defended target is $\frac{v_{\text{attaker}}}{v_{\text{target}}}=k$, and $\sin\eta=\frac{y}{d}$, y is the ordinate distance between the incoming enemy and the defended target, d is the distance between the incoming enemy and defended target. Therefore, it can be known that

$$\rho=\arcsin\left(\frac{y}{kd}\right). \tag{3}$$

Therefore, the direction of the incoming enemy can be obtained. According to the guidance mode of pure collision attack, the incoming enemy will keep this speed direction until it hits the target.

The incoming enemy uses the above attack guidance method to attack the defended target. Each turn randomly selects one of the above attack guidance methods to attack the defended target.

2.3 Model of the UAV swarm

This paper uses a deep reinforcement learning algorithm to study the UAV swarm defense missions. Different from the traditional grid-based dimensionality reduction method used in the interaction between UAV and environment under the deep reinforcement learning algorithm, we construct a continuous UAV exploration environment, that is, the UAV can reach any point on the map, and the UAV has an infinite state space. The continuous environment greatly improves the authenticity of the environment model, which makes deep reinforcement learning an important step to practical application. Based on the continuous environment space, the UAV is modeled as follows.

(i) UAV kinematics model

For the continuous environment model, acceleration is used as the action control unit of the UAV, and the specific acceleration action space is shown in Fig. 4.

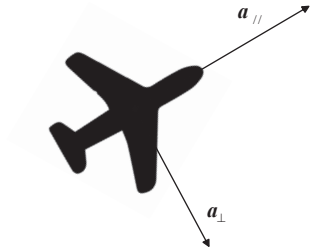


Fig. 4 UAV acceleration model diagram

The UAV is modeled because the UAV swarm defends the mission environment and uses acceleration to control its action. Moreover, the DDPG algorithm can control the continuous action of the UAV. As shown in Fig. 4, the action space of UAV is controlled by its tangential acceleration $\boldsymbol{a}_{//}$ and normal acceleration $\boldsymbol{a}_{\perp}$, and the value range of acceleration is limited.

$$\begin{cases}\boldsymbol{a}_{//}\in(-2,2)\\ \boldsymbol{a}_{\perp}\in(-1,1)\end{cases}$$

At the same time, the speed of UAV is limited, and the speed of UAV is not affected by the acceleration unlimited, $v\in[3,7]$. In the initial state, the state of UAV i is given by $s^i=\left[x_i,y_i,v_{xi}^t,v_{yi}^t\right]$, the initial speed v_x^t, v_y^t and position coordinates x_i, y_i which are initialized randomly within the specified range.

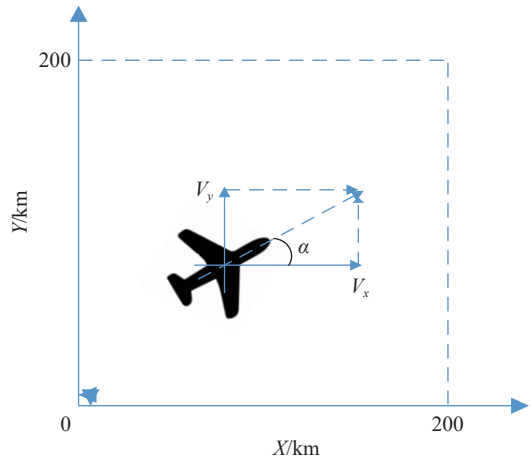


Fig. 5 UAV kinematics model diagram

Then the next state of a UAV is calculated according to the basic kinematic formula for each state. As shown in

Fig. 5, the kinematic model is given by

$$\begin{cases}\sin\alpha = \boldsymbol{v}_y^t / \left(\boldsymbol{v}_x^{t\,2} + \boldsymbol{v}_y^{t\,2}\right) \\ \cos\alpha = \boldsymbol{v}_x^t / \left(\boldsymbol{v}_x^{t\,2} + \boldsymbol{v}_y^{t\,2}\right)\end{cases}, \tag{4}$$

$$\begin{cases}\boldsymbol{v}_x^{t+1} = \boldsymbol{v}_x^t + \boldsymbol{a}_{//} \cdot \cos\alpha \cdot t \pm \boldsymbol{a}_{\perp} \cdot \sin\alpha \cdot t \\ \boldsymbol{v}_y^{t+1} = \boldsymbol{v}_y^t + \boldsymbol{a}_{//} \cdot \sin\alpha \cdot t \pm \boldsymbol{a}_{\perp} \cdot \cos\alpha \cdot t\end{cases}, \tag{5}$$

$$\begin{cases}x_{t+1}^i = x_t^i + v_x \cdot t \\ y_{t+1}^i = y_t^i + v_y \cdot t\end{cases}. \tag{6}$$

In (4)–(6), variables $\boldsymbol{a}_{//}$, $\boldsymbol{a}_{\perp}$, $\boldsymbol{v}_x^t$, $\boldsymbol{v}_y^t$, $\boldsymbol{v}_x^{t+1}$, $\boldsymbol{v}_y^{t+1}$ are vectors: $\boldsymbol{v}_x^t$ and $\boldsymbol{v}_y^t$ are the speeds of the UAV at time t; $\boldsymbol{v}_x^{t+1}$ and $\boldsymbol{v}_y^{t+1}$ are the speeds of the UAV at time $t+1$; $\boldsymbol{a}_{//}$ and $\boldsymbol{a}_{\perp}$ are the accelerations of the UAV output by the neural network in current state; x_t^i and y_t^i are position coordinates of UAV at time t; x_{t+1}^i and y_{t+1}^i are position coordinates of UAV at time $t+1$; t is movement time of the UAV in one state; α is angle between velocity direction of UAV and X-axis direction.

(ii) UAV detection model

It is assumed that the UAV can sense its environment globally. To make the model closer to the real environment, a normal distribution error is added to the UAV sensor information. The relevant parameters of the error are determined as follows:

$$\begin{cases}\mu = 0 \\ \sigma = \dfrac{1}{60} d_{\text{agent\_target}}\end{cases}. \tag{7}$$

Therefore, the UAV detection results are as follows:

$$\begin{cases}x_{\text{get}} = x + \varepsilon \\ y_{\text{get}} = y + \varepsilon\end{cases} \tag{8}$$

where x_{get} is position in the x-direction of the incoming enemy detected by the UAV; y_{get} is position in the y-direction of the incoming enemy detected by the UAV; x is true x-direction position of the incoming enemy; y is true y-direction position of the incoming enemy; ε is a random value that follows a normal distribution.

The detection results of the UAV on the incoming enemy's speed are as follows:

$$\begin{cases}v_{x\_\text{get}} = \dfrac{x_{\text{now}} - x_{\text{old}}}{t} \\ v_{y\_\text{get}} = \dfrac{y_{\text{now}} - y_{\text{old}}}{t}\end{cases} \tag{9}$$

where $x_{\text{now}}, x_{\text{old}}, y_{\text{now}}, y_{\text{old}}$ respectively represent the incoming enemy's position detected under the current conditions and the previous conditions and t represents the detection cycle of the UAV swarm. Because of the detection error of the UAV swarm to the incoming enemy's position, the detection error of the UAV swarm to detected velocity state always exists.

(iii) UAV attack model

It is assumed that the UAV has an attack mode, which is set as a circular attack area to simplify the model. Assuming that the attack radius of the UAV is r when the incoming enemy enters the attack area of the UAV, the UAV begins to attack, and the UAV can launch an attack on the incoming enemy every three simulation cycles. In each launch, the UAV has a certain probability of destroying the incoming enemy, and the probability of destroying is related to the distance between the incoming enemy and the UAV. The probability of destroying an incoming enemy in each attack is assumed to be as follows:

$$p = -3.777 d^2_{\text{ageng\_target}} + 0.85 \tag{10}$$

where $d_{\text{ageng\_target}}$ represents the distance between the drone and the incoming target. When $d_{\text{ageng\_target}} < r$, the incoming enemy is within the attack area of the UAV.

(iv) UAV communication model

To make UAVs have better action decisions, local communication between UAVs is required. Different from centralized communication which needs the central node, we adopt the flat distributed communication structure. Assume that each UAV can communicate with its adjacent drones, and each drone can obtain its three nearest neighbor's position, velocity, and other related information, as shown in Fig. 6.

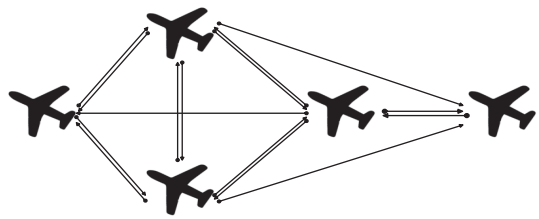


Fig. 6 Schematic diagram of UAVs communication

The arrow direction in the figure shows the process of the UAV transmitting its position, speed, and other state information and corresponding detected information to the adjacent UAV. Besides, the state of the UAV itself and the state of the adjacent UAV transmitted by communication is taken as the perception state of the UAV to the current environment.

In this paper, the UAV i can sense the following properties about other UAVs within its neighborhood:

$d^{i,j}$ is the distance to its neighboring UAVs;

$\phi^{i,j} = \arctan\left(\dfrac{y^j - y^i}{x^j - x^i}\right) - \phi^i$ is the bearing angle to its neighboring UAVs;

$\theta^{i,j} = \arctan\left(\dfrac{y^i - y^j}{x^i - x^j}\right) - \phi^j$ is relative orientation;

$\Delta v^{i,j} = v^i[\cos\phi^i, \sin\phi^i] - v^j[\cos\phi^j, \sin\phi^j]$ is relative velocity.

Furthermore, each UAV has access to the following local properties:

$d^i_{\text{boundary}} = \min(x^i - x_{\min}, y^i - y_{\min}, x_{\max} - x^i, y_{\max} - y^i)$ is the distance to closest battlefield boundary;

$\phi^i_{\text{boundary}} = \varphi^i_{\text{boundary}} - \phi^i$ is orientation to closest battlefield boundary;

v^i is the velocity of the UAV itself.

$\varphi^i_{\text{boundary}}$ denotes the absolute bearing angle of UAV i to the closet battlefield boundary segment.

3. DDPG algorithm

The DDPG algorithm is a deep reinforcement learning algorithm combining value iteration and policy iteration [21,22]. The DDPG algorithm uses neural networks to learn and preserve the UAV action strategy, which can explore the environment and learn optimal strategy in infinite size state space and action space.

3.1 Algorithm structure

DDPG algorithm combines advantages of the "actor & critic" algorithm and the DQN algorithm, which is a new deep reinforcement learning algorithm [23,24]. The algorithm structure is shown in Fig. 7.

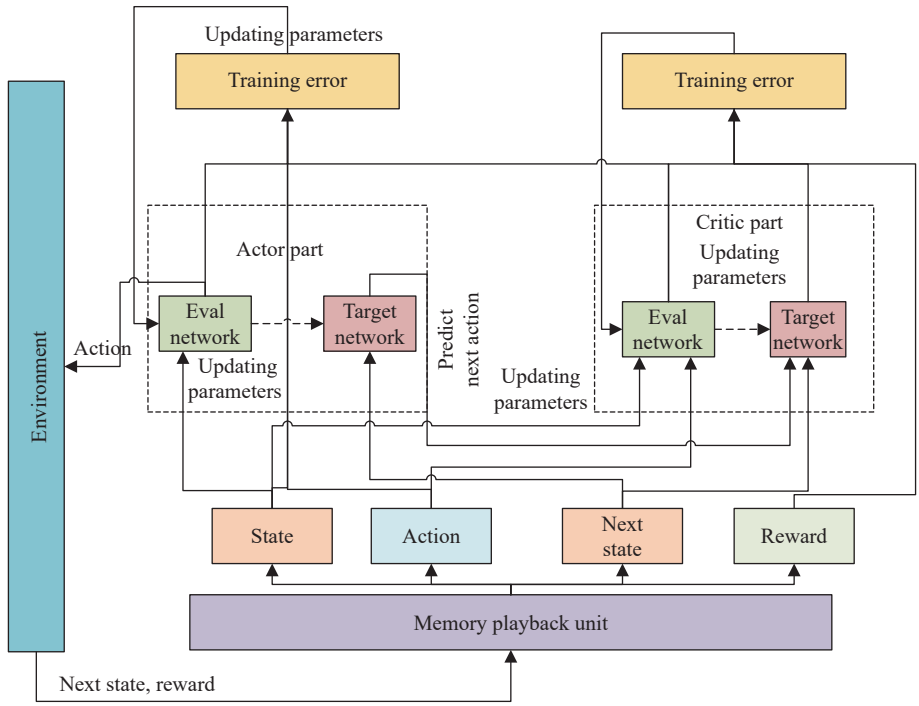


Fig. 7 Structure of the DDPG algorithm

The actor part and the critic part each have a pair of artificial neural networks with exactly the same structure, which is called Eval neural network and Target neural network respectively. Among them, Eval neural network is used for training and Target for updating the parameters of Eval network, which are followed by periodic soft update strategy and assist in the training process of the neural network.

The actor neural network determines the probability of UAV's action selection. When the UAV makes behavioral decisions, it will interact with the environment according to the probability provided by the actor neural network. The critic neural network receives the state and action and generates the value evaluation of the "state-action", in which the Eval neural network is used to judge the value of the current state and action, and the Target neural network receives the state at the next moment and the action at the next moment from the actor Target neural network and makes the value judgment.

Actors and critics have different training processes using different loss functions. For the critic network, TD-error is used to update the parameters of Eval neural network to minimize loss during training, as shown in (11) and (12).

$$\text{TD}-\text{error}=\text{reward}(s_t,a_t)+\gamma v'\left(s_{t+1},a_{t+1};\theta'_{\text{critic}}\right)-v(s_t,a_t;\theta_{\text{critic}}), \tag{11}$$

$$\text{Loss}=(\text{TD}-\text{error})^2. \tag{12}$$

In (11) and (12), $\text{reward}(s_t,a_t)$ is the environmental reward of the current state and action, given by the training sample; $v'(s_{t+1},a_{t+1};\theta'_{\text{critic}})$ is the evaluation of the value of state and action at the next moment, given by the critic's neural network; $v(s_t,a_t;\theta_{\text{critic}})$ is status assessment of current temporal states and actions, given by critic's neural network; a_{t+1} is the action of the next moment given by the neural network of the actor's part; s_{t+1} is the status at the next moment given by the training sample; a_t is the action at the current moment given by the training sample; s_t is the current status given by the training sample; γ is the discount factor.

For the training process of actors neural network, it is necessary to maximize the value function of state and action, so the mean value of evaluation of state and action is used as the loss function, as shown in (13):

$$\text{Loss}=-\text{mean}(v(s,a;\theta_{\text{critic}})). \tag{13}$$

3.2 Balance of DDPG algorithm in exploration and experience

Although the DDPG algorithm outputs the probability distribution of UAV's action selection in the process of UAV's action selection, in the final process of UAV interaction with the environment, the only action is still selected according to the given probability distribution [18]. Therefore, in the process of UAV interaction with the environment, it always uses a deterministic strategy, which will lead to insufficient exploration of the environment by UAV. Therefore, it is necessary to add some exploration to the strategy selection of DDPG algorithm [25].

The exploratory method to increase the exploration ability for the DDPG algorithm needs to add some random noise to the action selection process of UAV. As shown in (14):

$$\text{action}=\text{action}'+\text{Noise} \tag{14}$$

where action is the interaction between the UAV and the environment; action′ is action selected according to the probability distribution of actions given by the actor-network; Noise is the random noise.

The action decision made by the DDPG algorithm can output the continuous action of UAV. Therefore, it is feasible to enhance the exploratory performance of the DDPG algorithm by using the above method, and the random noise is normally distributed:

$$\text{Noise}\sim \text{N}(0,\sigma^2). \tag{15}$$

The variance of the random noise is the quantity related to the iteration round. As the training progress goes on, the variance decreases gradually. According to the characteristics of the normal distribution, we give the initial value of the variance as follows:

$$\sigma_0=(\text{action}_{\max}-\text{action}_{\min})/4=1.5, \tag{16}$$

$$\sigma=\kappa^{\text{episode}}\sigma_0. \tag{17}$$

The variance change of random noise used to increase the exploratory of DDPG algorithm is shown in (16) and (17), and here based on the training experience $\kappa=0.999\,5$, episode represents the training cycle, which can ensure that the action selection of UAV has a large exploration rate. At the same time, when the action selection of UAV exceeds the maximum range, it is restricted to the corresponding action boundary.

3.3 Network model of DDPG algorithm

The network structure diagram of the DDPG algorithm is constructed by using an artificial neural network, where the "actor" and "critic" parts of the DDPG algorithm have a pair of neural networks with the same structure, and the two neural networks will be introduced below.

(i) "Actor" part of artificial neural network

It is well known that the neural network of the "actor" part is used to select the action of UAVs, that is, to realize the mapping of the state space of UAV swarm to the action space of UAVs.

In UAV swarm defense mission scenario, the state space for the UAV swarm is defended target position $x_{\text{center}},y_{\text{center}}$, UAVs' position $x_{\text{agent}},y_{\text{agent}}$, UAVs' speed $v_{x\_\text{agent}},v_{y\_\text{agent}}$, and the incoming enemy's position $x_{\text{target}},y_{\text{target}}$, the incoming enemy's speed $x_{\text{target}},y_{\text{target}}$, and other information that gained by the interaction with other UAVs in the swarm $x_{\text{agent\_comm\_i}}$, $y_{\text{agent\_comm\_i}}$ $v_{x\_\text{agent\_comm\_i}},v_{y\_\text{agent\_comm\_i}}$, as well as the other UAVs to detect the incoming enemy's information $x_{\text{agent\_comm\_i\_get}}$, $y_{\text{agent\_comm\_i\_get}}$, $v_{x\_\text{agent\_comm\_i\_get}}$, $v_{y\_\text{agent\_comm\_i\_get}}$, a total of 34 dimensions for the UAV swarm state space. These state data are sorted and coded in a fixed order to form the state space structure of the UAV swarm, as shown in Fig. 8.

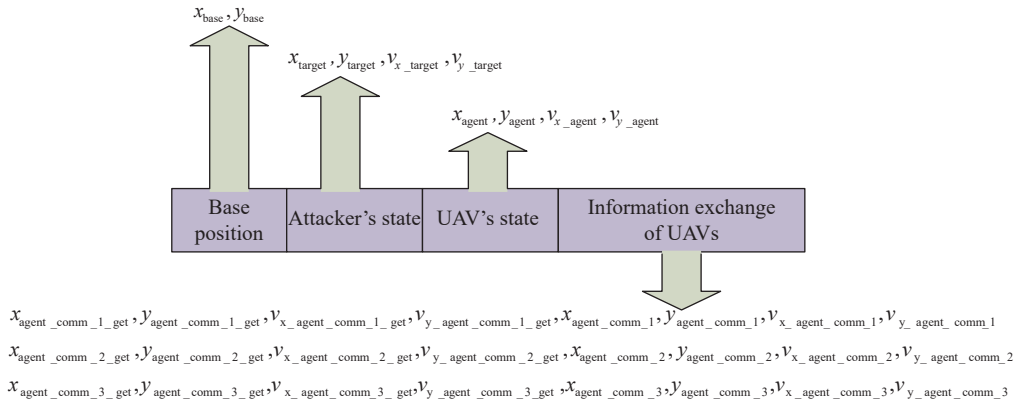


Fig. 8 State space of UAV swarm

Therefore, for the "actor" part: the Target and Eval neural network, two fully-connected artificial neural networks (excluding the input layer) with the same structure are constructed, and we have a six layers neural network, the number of neurons in each layer is respectively $[100, 100, 300, 100, 10, 2]$. Neurons in the last layer of the neural network use $\tanh(x)$ as nonlinear activation functions to realize the mapping between the network output and UAV's action, while other neurons use $\text{relu}(x) = \max(0, x)$ as nonlinear activation functions. And the root mean square prop (RMSP) algorithm is used as the optimization algorithm of training. The neural network structure is shown in Fig. 9.

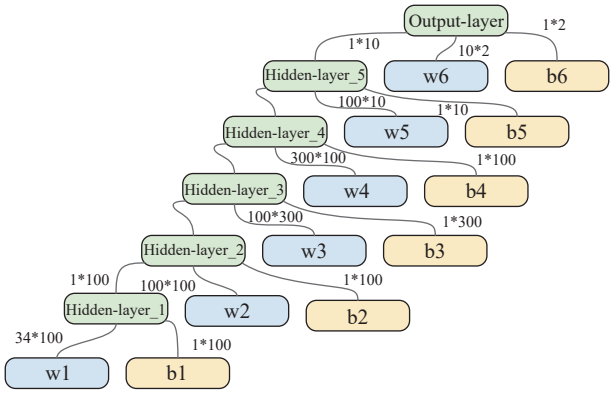


Fig. 9 Artificial neural network structure of the "actor's part"

(ii) "Critic" partial artificial neural network

Through the determined value of "state-action", the critic's partial neural network guides the training process of the "actor" neural network [26]. Therefore, the input state of the critic neural network is the state information and action information of the UAV swarm, and the state space structure of the critic network is shown in Fig. 10.

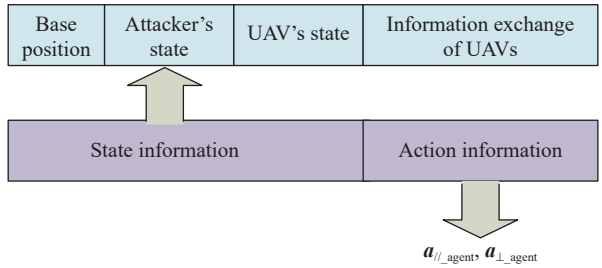


Fig. 10 Input data structure of "critic section" neural network

It is also built via two completely identical 5-layer all connected artificial neural networks (excluding the input layer) for the "critic" part of the Target and Eval artificial neural network, the number of neurons in each layer is respectively $[100, 300, 100, 10, 1]$. The neurons of the last output layer use $\tanh(x)$ as a nonlinear activation function, while the other neurons use $\text{relu}(x)$ as a nonlinear activation function, and the RMSP algorithm is used as the optimization algorithm for training. The structure of the neural network is shown in Fig. 11.

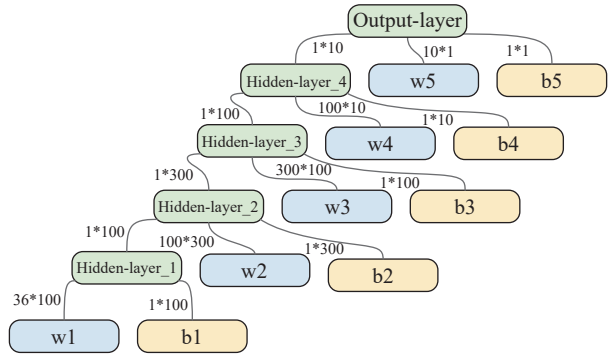


Fig. 11 Artificial neural network structure of the "critic" part

For the "actor" part and the "critic" part, there exist simultaneously the Target and Eval neural networks, Eval is used for the training process, and the parameters of the Target neural network change periodically with the corresponding parameters of the training network. For the parameter update of Target neural network, the soft

update strategy based on the sliding average value is adopted, namely,

$$\theta_{\text{Target}} = k\theta_{\text{Target}} + (1-k)\theta_{\text{Eval}} \tag{18}$$

where θ_{Target} refers to Target neural network parameters; θ_{Eval} refers to Eval neural network parameters; k is the sliding factor, with a value of 0.2.

3.4 Sparse reward problem of DDPG algorithm

Since the continuous state space and action space are used in this research, the UAV will undergo a long interaction with the environment to reach the final state after initialization [24]. At this point, the method of rewarding UAV only after it reaches the final state has the defect of too long return period, which leads to the failure of effective learning in reinforcement learning, that is, the problem of sparse reward.

For the sparse reward problem, there are two common solutions: one is to divide the main task and set related small goals for the main task so that the UAV starts learning from the small goal first, to speed up learning [27]. Another one is to modify the goal to increase the effective reward, to speed up learning. We adopt the second method to accelerate the training and construct the reward function of UAVs to guide the learning direction of deep reinforcement learning.

$$\begin{cases} r = 100 \\ r = -10 \\ r = -100 \\ r = d_{\text{agent\_target}} - d'_{\text{agent\_target}} + v_{\text{agent}} \cos\beta - \dfrac{20}{d_{\text{center\_target}}} \end{cases} \tag{19}$$

where $d_{\text{agent\_target}}$ is the distance between the UAV and the incoming enemy at the current state; $d'_{\text{agent\_target}}$ is the distance between the UAV and the incoming enemy in the next state; β is angle between speed direction of the UAV and connection line between the UAV and the incoming enemy at the current state; v_{agent} is the velocity of the UAV at the current state; $d_{\text{center\_target}}$ is the distance between the defended target and the incoming enemy.

According to the above different termination states, there are different reward function values. When the UAV swarm completes the defense task, the UAV swarm that destroys the incoming enemy will receive a positive reward $r = 100$. When the collision boundary of UAV swarm leads to the early end of this turn, the UAV colliding with the battlefield boundary is given a negative reward $r = -10$. If the UAV swarm fails to complete the task and the defended target is destroyed, the UAV will be given a negative reward $r = -100$. For the exploration phase of UAV in the environment, the reward function with enlightened characteristic is adopted:

$$r = d_{\text{agent\_target}} - d'_{\text{agent\_target}} + v_{\text{agent}} \cos\beta - \frac{20}{d_{\text{center\_target}}}. \tag{20}$$

The reward function of the enlightened property is composed of the distance between the UAV and the incoming enemy, the speed and direction of the UAV, and the distance between the defended target and the incoming enemy. Obviously, in defense missions, the reward function corresponding to the smaller distance between UAV and the incoming enemy is positive. When the UAV's velocity vector points to the incoming enemy, the higher the UAV's velocity is, the higher the reward function will be. When the UAV's velocity vector deviates from the incoming enemy, the higher the UAV's velocity is, the higher the negative reward function will be. At the same time, it increases the distance between the incoming enemy and the defended target as the reference of time and the threat of the incoming enemy to the defended target. The closer the distance between the incoming enemy and the defended target is, the greater the negative reward will be. To sum up, (19) reward function can accurately reflect the state value of UAV swarm.

3.5 DDPG algorithm flow

A detailed description of DDPG algorithm is provided in Algorithm 1.

Algorithm 1 DDPG algorithm

1: **Initialize** maximum iterations T, state feature dimension n_s, action space dimension n_a, and batch size;
2: **Initialize** parameters γ, α, n and artificial neural network parameters;
3: **Initialize** memory playback unit and temporary storage unit;
4: **for** training rounds episode = $1, T$
5: Initialize the state space and action space
6: **do** time $t = 1$
7: Select UAV's action using the equation: $a_t = \arg\max(Q(s_t, a_t : \theta_{\text{actor}}))$
8: Generate random noise error using: $a_t = a_t + \text{Noise}$
9: UAV performs an action a_t, get new state s_{t+1} and reward γ
10: Store $\{s_t, a_t, \gamma, s_{t+1}\}$ on the temporary storage unit
11: **If** the temporary storage unit is full
12: Calculate the value of the sample using the value function
$\text{TD} - \text{error} = \text{reward}(s_t, a_t) + \gamma^* v'(s_{t+1}, a_{t+1}) - v(s_t, a_t)$
13: Remove low-value samples from the temporary

storage unit by probability formula

$$P(i)=\frac{(p+\varepsilon)^{-1}}{\sum_{i=1}^{k}(p_i+\varepsilon)^{-1}}$$

14: Calculate the sample value under the new state using the value function

$$\text{TD}-\text{error}=\text{reward}(s_t,a_t)+\gamma^{*}v'(s_{t+1},a_{t+1})-v(s_t,a_t)$$

15: Select high-value samples to store in the temporary storage unit according to probability

$$P(i)=\frac{P_i^X}{\sum_{i=1}^{k}P_i^X}$$

16: Empty temporary storage unit

17: **If** in training phase:

18: In the memory playback unit through a value function

$$\text{TD}-\text{error}=\text{reward}(s_t,a_t)+\gamma^{*}v'(s_{t+1},a_{t+1})-v(s_t,a_t)$$

19: Select batch_size training samples by priority sampling according to probability

$$P(i)=\frac{\mu^{*}p_i^x+\beta}{\sum_{i=1}^{k}(\mu^{*}p_i^x+\beta)}$$

20: Calculate the loss function $\text{loss}=(\text{TD}-\text{error})^2$ and get the gradient

21: Update neural network with RMSP optimizer

22: **If** reach "target-network" update cycle

23: Soft update parameters

$$\theta'_{\text{critic}}=a*\theta_{\text{critic}}+(1-a)*\theta'_{\text{critic}}$$

and

$$\theta'_{\text{actor}}=a*\theta_{\text{actor}}+(1-a)*\theta'_{\text{actor}}$$

24: **while** the UAV reaches the end state, ending the cycle

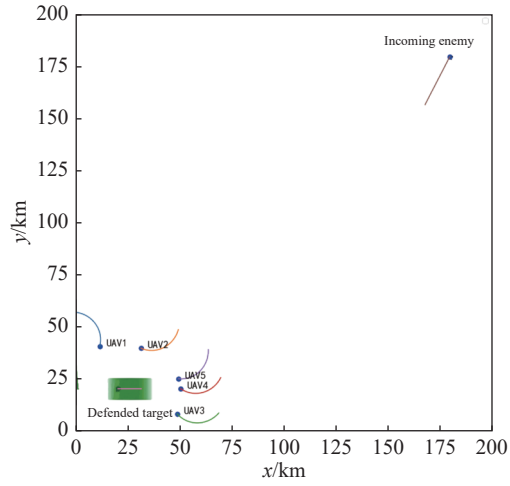
25: **end for**

4. Simulation results

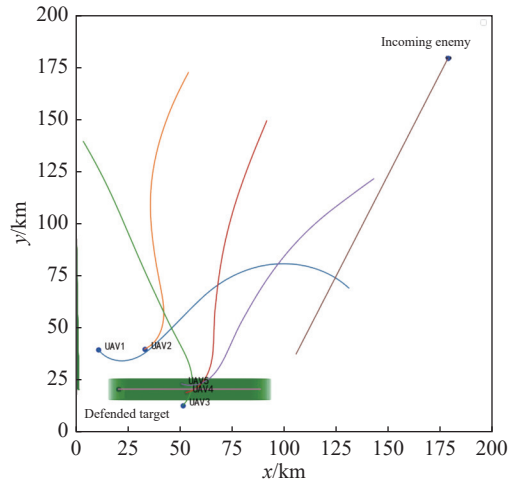
The DDPG algorithm is used to study the defense task of UAV swarm against the incoming enemy which is randomly generated using either pure pursuit or pure collision attack mode against the defended target. The initial position of the UAV swarm is generated randomly in the designated area, and its initial state is random. Therefore, the DDPG algorithm is used to command the action decision of UAV swarm, to observe the performance of UAV swarm in defense tasks.

4.1 Training process

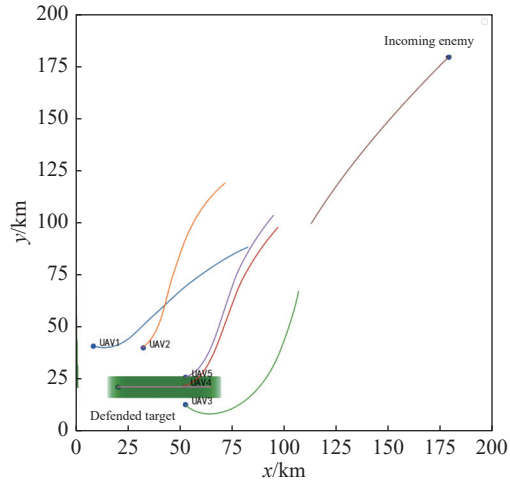
The artificial neural network constructed above is used for training, during which five isomorphic drones are used for defense missions. It is assumed that each UAV receives relevant information from three adjacent UAVs to assist in decision making and the maximum training epoch is 1 250. The snapshots of the training peocess are shown in Fig. 12.



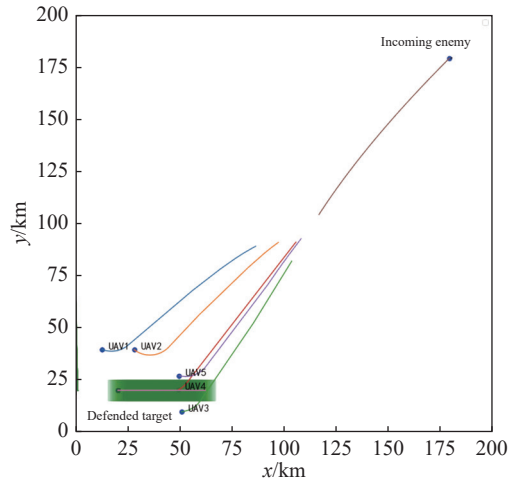
(a) 200 rounds of training renderings



(b) 400 rounds of training renderings



(c) 600 rounds of training renderings



(d) 1 250 rounds of training renderings

Fig. 12 Snapshots of different training epochs

Firstly, the convergence of the artificial neural network is analyzed, and the parameters of the neural network layers of "critic" and "actor" are selected for relevant statistics. Since there are massive parameters in the neural network, the mean and variance of the parameters are statistically observed. The statistical results are shown in Figs. 13−16.

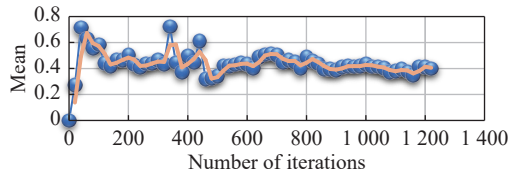


Fig. 13 Mean changes of network parameters in "critic" section from Eval network

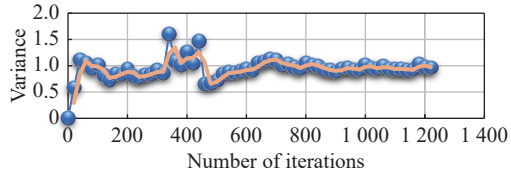


Fig. 14 Variance changes of network parameters in "critic" section from Eval network

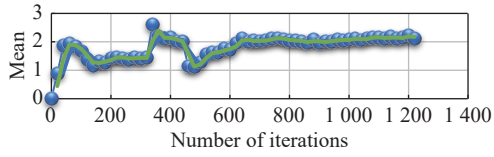


Fig. 15 Mean changes of network parameters in "actor" section from Eval network

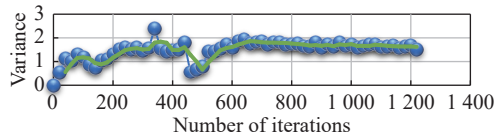


Fig. 16 Variance changes of network parameters in "actor" section from Eval network

The parameter statistics of the above figure are selected from the parameter statistics of all neurons in the neural network of the "critic" part and "actor" part. The solid line is the statistical truth value of the parameter, while the dotted line is the result of sliding average processing of the statistical truth value with a period of 3. In Fig. 17 and Fig. 18, it can also be seen from the distribution trend of target network parameters of actor and critic that the parameters of neural network are constantly changing during training and eventually tend to be stable. It is proved that the neural network tends to converge during training. It can be seen from the above statistics that the artificial neural network converges during the training process.

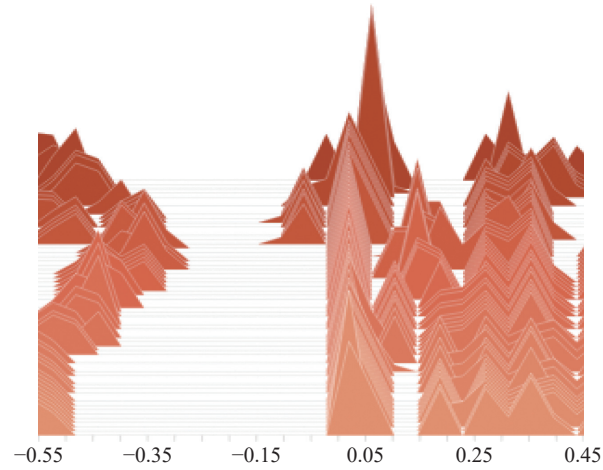


Fig. 17 Changes of Target network parameters distribution in the "actor" section

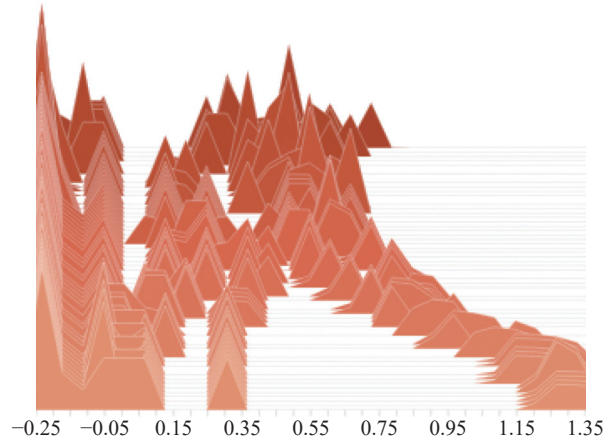


Fig. 18 Changes of Target network parameters distribution in the "critic" section

This paper processes the data in the UAV training process and uses memory playback unit to store the samples, and the reward value in each sample is the return of the interaction between the UAV and the environment, as well as the performance reference index when the UAV interacts in the environment. Fig. 19 shows the reward values of the samples selected for training against the increase of training rounds.

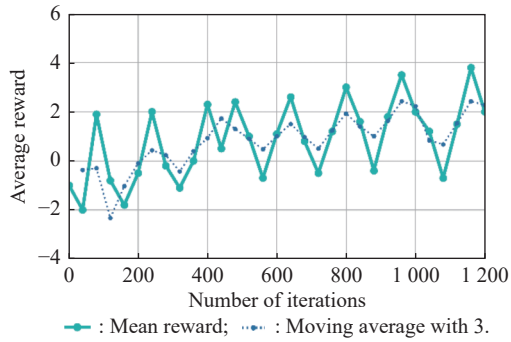


Fig. 19 Mean reward of samples in different epochs

As shown in Fig. 19, with the increase of training rounds, the average reward of training samples rises slowly, that is to say, the UAV behavior represented by the samples in the memory playback unit is trying to be beneficial to the direction adjustment of the task target, and the UAV training will have better and better performance, while the dotted line is the result of sliding average processing of the statistical truth value with a period of 3.

Fig. 20 shows the evolution of the average episode rewards in different rounds during the training process.

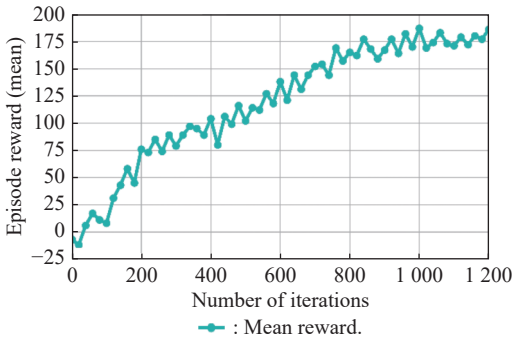


Fig. 20 Mean episode reward under different training rounds

Fig. 20 shows that DDPG achieves very good performance on our task. As expected, we see that with the increase of the total training rounds, the total rewards of the round increases steadily. When it reaches 800 rounds, it is close to convergence.

Using the above method to train five UAVs, and the mission success rate of each round is shown in Fig. 21.

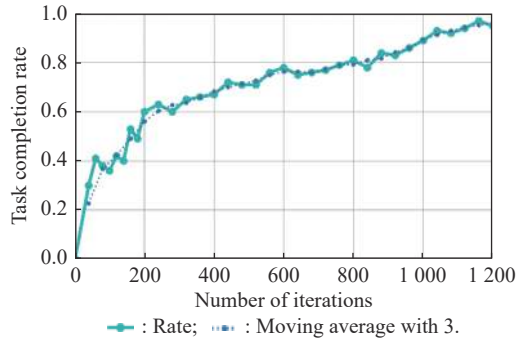


Fig. 21 Task completion rate for UAV swarm

As can be seen from Fig. 21, the UAV clusters after training can well complete the defense task, with an average success rate of 95%.

4.2 Execution process

After the completion of the defense task training with five UAVs, the artificial neural network verifies the completed training model. The artificial neural network model completed by training is used as the behavioral decision-making unit of the UAV swarm. The UAV has a certain random initial state, and the initial position of the incoming enemy is also randomly generated in the designated area. Firstly, the behavioral decision of using five UAVs for defense missions is verified, and the movement trajectory of the UAV swarm is obtained, as shown in Fig. 22.

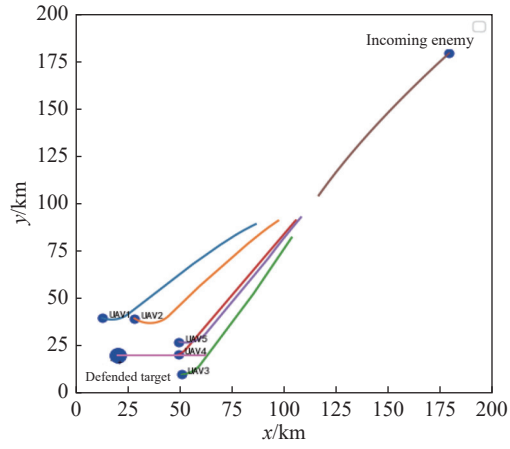


Fig. 22 Trajectory of five UAVs in a defense mission

As shown in Fig. 22, the trained model can well accomplish the defense tasks of the cluster of five UAVs. Then, increasing the number of UAVs to 30 without changing the state space structure of the UAVs cluster. It is to verify the UAV swarm defense mission by using the trained network (in the cluster defense mission environment composed of five UAVs, the artificial neural net-

work has undergone 1 250 epochs of training). Also, the initial position and initial velocity of the UAV swarm are random to some extent. The performance of the swarm composed of more UAVs when the defense task is verified, and the mission trajectory of the UAV swarm is shown in Fig. 23.

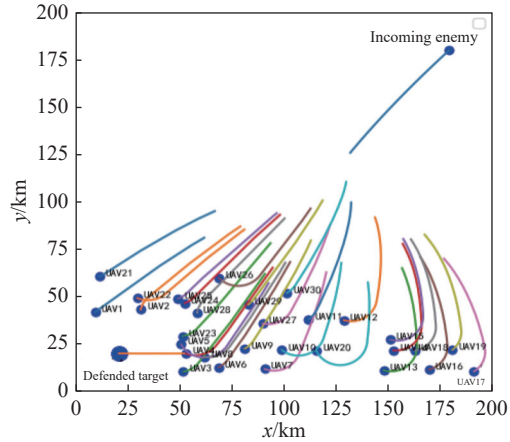


Fig. 23 Trajectory of 30 UAVs in defense mission

As shown in Fig. 23, the decision-making model of UAV swarm defense mission obtained by training with five UAVs can be applied to the swarm defense mission composed of more UAVs without modifying the state space structure. At the same time, it also shows that the behavior decision of UAV swarm using DDPG algorithm can well meet the challenge of UAV swarm to a dynamic number of individuals, reflecting the characteristics of UAV cluster autonomy.

5. Conclusions

In this paper, the DDPG algorithm is used to study defense task of the UAV swarm, which have a continuous state space and action space. To balance the contradiction of "experience-exploration" in the DDPG algorithm, we add random noise elements to the act selection to improve the exploration ability of the algorithm. At the same time, the random noise will decrease with the increase of iteration epochs to ensure the convergence of the algorithm. To improve the performance of the algorithm, the parameter updating process of a neural network is optimized. To solve the problem of "sparse reward" in deep reinforcement learning, we build a guiding reward function to give the UAV swarm appropriate reward when an episode is not over, which effectively improves the convergence of the algorithm.

According to simulation results, we use the DDPG algorithm to guide the UAV swarm for decision making which successfully defend the target from the incoming enemy. Meanwhile, under the same state space, the model trained from a small number of UAVs can be directly applied to more UAVs to form swarm to perform the corresponding tasks, which fully embodies the characteristics of UAV swarm, such as decentralization and autonomy. This also reflects the use of artificial intelligence method to study the strong generalization ability of the UAV swarm decision-making problem.

References

[1] ZHANG Q X, JIANG M L, FENG Z Y. IoT enabled UAV: network architecture and routing algorithm. IEEE Internet of Things Journal, 2019, 6(2): 3727–3742.

[2] LAN T, QIN D, SUN G. Joint optimization on trajectory, cache placement, and transmission power for minimum mission time in UAV-aided wireless networks. International Journal of Geo-Information, 2021, 10(7): 426.

[3] ARAFAT M Y, MOH S. Localization and clustering based on swarm intelligence in UAV networks for emergency communications. IEEE Internet of Things Journal, 2019, 6(5): 8958–8976.

[4] MUKHERJEE A, MISRA S, CHANDRA V S P, et al. Resource-optimized multiarmed bandit-based offload path selection in edge UAV swarms. IEEE Internet of Things Journal, 2019, 6(3): 4889–4896.

[5] CHANDARANA M, MESZAROS E L, TRUJILLO A, et al. Natural language based multimodal interface for UAV mission planning. Proceedings of the Human Factors and Ergonomics Society Annual Meeting, 2017, 61(1): 68–72.

[6] CHANDARANA M, MESZAROS E L, TRUJILLO A, et al. "Fly like this": natural language interface for UAV mission planning. Proc. of the International Conference on Advances in Computer-Human Interactions, 2017: 1−7.

[7] XU G Q, JIANG W L, WANG Z L, et al. Autonomous obstacle avoidance and target tracking of UAV based on deep reinforcement learning. Journal of Intelligent & Robotic Systems, 2022, 104(4): 1–13.

[8] HU Y, CHEN M Z, SAAD W, et al. Distributed multi-agent meta learning for trajectory design in wireless drone networks. IEEE Journal on Selected Areas in Communications, 2021, 39(10): 3177–3192.

[9] WANG H, LI Y M, QIAN J B. Self-adaptive resource allocation in underwater acoustic interference channel: a reinforcement learning approach. IEEE Internet of Things Journal, 2020, 7(4): 2816–2827.

[10] LI G L, MA Y F. Feature extraction algorithm of air combat situation based on deep neural networks. Journal of System Simulation, 2017, 29(S1): 98–105, 112. (in Chinese)

[11] HANG W. Research of UCAV air combat based on reinforcement learning. Harbin: Harbin Institute of Technology, 2015. (in Chinese)

[12] XU Y H, XIE J W, ZHANG Y G, et al. Reinforcement learning (RL)-based energy efficient resource allocation for energy harvesting-powered wireless body area network. Sensors, 2020, 20(44): 1–22.

[13] LI Q Y, YAO H P, MAI T L, et al. Reinforcement and belief learning-based double auction mechanism for edge computing resource allocation. IEEE Internet of Things Journal. 2020, 7(7): 5976−5985.

[14] WANG S Y, DUAN J J, SHI D, et al. A data-driven multi-

agent autonomous voltage control framework using deep reinforcement learning. IEEE Trans. on Power Systems, 2020, 35(6): 4644–4654.
[15] LIU P, MA Y F. A deep reinforcement learning based intelligent decision method for UCAV air combat. Proc. of the Asian Simulation Conference, 2017: 274−286.
[16] PRICE J K, PINON-FISCHER O J, MAVRIS D N. Definition of optimal agent behaviors using reinforcement learning. Proc. of the AIAA Scitech Forum, 2019. DOI: 10.2514/6.2019-2200.
[17] LUO D L, YANG X, ZHANG J P, et al. New progress on UAV swarm confrontation. Science & Technology Review, 2017, 35(7): 26–31.
[18] HU J W, WANG L H, HU T, et al. Autonomous maneuver decision making of dual-UAV cooperative air combat based on deep reinforcement learning. Electronics, 2022, 11(3): 467.
[19] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human level control through deep reinforcement learning. Nature, 2015, 518(7540): 529–533.
[20] SILVER D, LEVER G, HEESS N. Deterministic policy gradient algorithms. Proc. of the 31st International Conference on International Conference on Machine Learning, 2014: 387−395.
[21] SHI H B, SUN Y R, LI J. Dynamical motor control learned with deep deterministic policy gradient. Computational Intelligence and Neuroscience, 2018, 2018: 8535429
[22] ARULKUMARAN K, DEISENROTH M P, BRUNDAGE M, et al. A brief survey of deep reinforcement learning. IEEE Signal Processing Magazine, 2017, 34(6): 26–38.
[23] WANG G H, SHI J L. Actor-critic for multi-agent system with variable quantity of agents. Proc. of the International Conference on Internet of Things as a Service, 2018: 48−56.
[24] HUANG W R, WANG Y Z, YI X D. A deep reinforcement learning approach to preserve connectivity for multi-robot systems. Proc. of the 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, 2017: 1−7.
[25] YI H. Deep deterministic policy gradient for autonomous vehicle driving. Proc. of the International Conference on Artificial Intelligence, 2018: 191−194.
[26] ANDERSEN P A, GOODWIN M, GRANMO O C. Deep RTS: a game environment for deep reinforcement learning in real-time strategy games. Proc. of the IEEE Conference on Computational Intelligence and Games, 2018: 1−8.
[27] NIE H H, CHEN Y, SONG Y K, et al. A general real-time OPF algorithm using DDPG with multiple simulation platforms. Proc. of the IEEE Innovative Smart Grid Technologies, 2019: 3713–3718.

Biographies

ZHANG Yaozhong was born in1974. He is an associate professor at the School of Electronics and Information in Northwestern Polytechnical University. His research interests include modeling, simulation and effectiveness evaluation of complex systems, reinforcement learning and its application in the complex system.
E-mail: zhang_y_z@nwpu.edu.cn

WU Zhuoran was born in 1999. He received his bachelor's degree in detection guidance and control technology from the School of Electronics and Information, Northwestern Polytechnical University, and master's degree in control science and engineering in the School of Electronics and Information, Northwestern Polytechnical University. His research interest is reinforcement learning.
E-mail: 542391943@qq.com

XIONG Zhenkai was born in 1979. He received his Ph.D. degree from the College of Mechanical and Electrical Engineering, Harbin Engineering University in 2012. His main research interests include intelligent control, high-precision control, embedded system, optimal estimation theory and application, and filter algorithm.
E-mail: 1223959392@qq.com

CHEN Long was born in 1967. He received his bachelor's degree from Changchun Institute of Optics and Precision Mechanics, and master's degree from Beijing Institute of Technology. He is a researcher at the China Research and Development Academy of Machinery Equipment, specializing in intelligent systems. He has been extensively involved in equipment system research and has served as the chief designer for the development of multiple types of equipment. He has received numerous provincial and ministerial-level scientific and technological advancement awards.
E-mail: dragon-cl@sohu.com