UAV Confrontation Decision-Making Based on Fictitious Self-play Multi-agent Proximal Policy Optimization

WANG Mingming[1], *ZHANG Baoyong*[1], *WU Chong*[2], *PING Yuan*[2], *QI Juntong*[3*]

1. School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, P.R. China;
2. The R&D Center, EFY (Hainan) Technology Co. Ltd., Sanya 572025, P.R.China;
3. School of Future Technology, Shanghai University, Shanghai 200444, P.R.China

(Received 10 November 2023; revised 22 November 2023; accepted 5 December 2023)

Abstract: This paper addresses the confrontation decision-making problem of unmanned aerial vehicles (UAVs) based on fictitious self-play multi-agent proximal policy optimization. UAV confrontation relies on autonomous decision-making, enabling the UAV to generate action instructions based on environmental information. An innovative autonomous decision-making methodology for UAV confrontations is proposed within the context of red-blue air combat tasks. Initially, the current situation is evaluated by employing the relative angle between the missile attack area and the UAV. Following this, guided by the evaluated scenario, the design of state space, action space, and real-time reward feedback is implemented to streamline the training process. Subsequently, an advanced method is introduced for optimizing strategy through a virtual autonomous agent's proximity, aiming to derive the advantage function and average strategy from the experience buffer of training data. Ultimately, the efficacy and superiority of the proposed method are validated through simulations of UAVs engaging in red-blue countermeasure tasks.

Key words: unmanned aerial vehicle (UAV); air combat; multi-agent proximal policy optimization (MAPPO); decision-making

CLC number: TP181 **Document code:** A **Article ID:** 1005-1120(2023)06-0627-14

0 Introduction

Owing to their high cost-effectiveness, robust maneuverability, and formidable concealment capabilities, unmanned aerial vehicles(UAVs) are widely applied in military area encompassing tasks such as reconnaissance, surveillance, and targeted attacks. Facing the complex battlefield environment, the confrontation and decision-making capabilities of UAVs have become a key development direction. UAV confrontation decision-making is a coordinated air battle formed by a group of UAVs intercepting another group of UAVs. UAVs, endowed with self-organizing adaptability and anthropomorphic cognition, assess their surroundings through environmental sensing. They employ strategies such as attack, avoidance, dispersion, concentration, cooperation, and assistance, guided by specific rules. In aggregate, research in UAVs has revealed the characteristics of its cluster confrontation.

Generally speaking, UAV adversarial decision- making approaches can be categorized into two distinct groups: Conventional methods and intelligent methods. Conventional methods pertain to leveraging expert experiences, formulaic derivations, and similar techniques to attain optimal decisions. These methodologies primarily rely on preexisting knowledge or mathematical computations, lacking the innate capacity for self-optimization associated typically. Gacovski et al.[1] proposed a combat decision modeling method based on expert knowledge and the fuzzy system. Although the complexity of the model was reduced, it was difficult to deal with

*Corresponding author, E-mail address: qijt@tju.edu.cn.

How to cite this article: WANG Mingming, ZHANG Baoyong, WU Chong, et al. UAV confrontation decision-making based on fictitious self-play multi-agent proximal policy optimization[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2023, 40(6): 627-640.

http://dx.doi.org/10.16356/j.1005-1120.2023.06.001

states outside the empirical samples. Zhou et al.[2] referred human knowledge structure to design a brain-like air combat system that could learn maneuvering tactics of human pilots without human guidance. Huang et al.[3] used the Bayesian theory to calculate the air combat situation, and adaptively adjusted the weight of decision factors according to the situation evaluation results, which improved the robustness and effectiveness of maneuver decision. Scukins et al.[4] used the Monte Carlo tree to evaluate the most optimal course of action from the vantage point of the opposing aircraft, while using the convex optimization to search for available flight paths. Shou et al.[5] proposed a game of constrained strategy solving algorithm based on linear programming to generate intelligent decisions of multiple UAVs in complex air combat environments. Although the above methods can realize the decision-making task in UAV confrontation, some problems still exist. For example, global optimization cannot be guaranteed, calculation time is long, and the ability to deal with unpredictable and non-deterministic tasks is inferior.

Intelligent methods refer to the use of intelligent methods to achieve decision-making and task allocation in UAV confrontation, such as population optimization algorithm, genetic algorithm, reinforcement learning, etc. These methods focus on the establishment and training of models, and can achieve optimization of strategies independently based on decision objectives. Li et al.[6] used the improved discrete particle swarm optimization algorithm to solve the target advantage function and encircle advantage function, and converted the occupation process into formation switching task. Gao et al.[7] proposed an air combat maneuver decision method based on the improved symbiotic search (SOS) algorithm, expanded and improved the traditional basic maneuver action library, constructed the fighter maneuver decision advantage function, and improved the convergence speed, convergence accuracy and the ability to escape the local optimal of the traditional algorithm. The matrix game method was proposed by Deng et al.[8] to obtain the approximate range of optimal selection strategy of UAVs, and then genetic algorithm was used to find the optimal strategy within this range. Dantas et al.[9] used artificial neural networks to generate responses related to tactical states according to sensor parameters, improving situational awareness and thus speeding up decision-making in air combats. However, the optimization speed of the swarm intelligence method is slow, the parameters of the genetic algorithm are not easy to design, and the maneuver decision based neural networks require a large number of sample data.

Reinforcement learning is an intelligent technique that employs a "trial and error" strategy to engage with the environment, acquires knowledge from it, and progressively enhances performance. This approach effectively overcomes the limitations of other methods, such as intricate modeling, challenging sample labeling, and tedious problem-solving. Without manual intervention, it can generate reasonable decision sequences through self-interactive training, which can well solve the problem of intelligent decision-making in UAV air combats. Yang et al.[10] constructed the action space of the state observation space based on the basic maneuver library and the proximal policy optimization, and designed the reward function, which had a win rate of 62% and a loss rate of only 11% in the simulation of air combats. Li et al.[11] introduced an autonomous operational decision model that employs the expert-based soft actor-critic algorithm. This innovative approach reconstructed the empirical replay buffer by leveraging expert experiences, significantly enhancing the exploration and utilization efficiency of deep reinforcement learning. Crumpacker et al.[12] constructed a Markov decision model representing aircraft action and energy, and used an approximate strategy iteration algorithm based on neural networks to generate high-quality maneuvering strategies.

In summary, regarding the UAV system as a typical agent system, many scholars have engaged in the application of multi-agent reinforcement learning algorithms for UAV combats.

However, there are still some difficulties in the research of intelligent maneuvering decision-making

of UAV based on reinforcement learning: The simulation environment is simple; most of the simulation environments are two-dimensional space; the impact of weapons on air combats is ignored; dimensional explosion exists; reward is sparse or delayed; situation assessment is incomplete; and generalizability is low.

An approach is presented to address the UAV combat decision-making problem, focusing on the design of a UAV countermeasure decision-making that utilizes a combination of the reinforcement learning algorithm and the game theory. The proposed method is evaluated through simulation experiments, wherein it is compared against alternative reinforcement learning algorithms to assess its effectiveness.

The contributions of this paper are as follows:

(1) A comprehensive situation model for UAVs is established, specifically targeting action space, state space, and reward function for effective decision-making. Furthermore, an algorithm, named FSP-MAPPO, is designed for decision training. This algorithm incorporates multi-agent proximal policy optimization (MAPPO) and fictitious self-play (FSP) to enhance the decision-making and generalizability capabilities of the UAV.

(2) To facilitate the learning process of the UAV's decision-making abilities regarding missile launch and evasion, a three-dimensional battlefield environment is constructed using JSBSim. Additionally, a missile system is integrated into the UAV to replicate real-world scenarios for effective decision-making learning. Moreover, the performance of the proposed approach is compared against alternative reinforcement learning methods to assess its effectiveness.

The paper is structured as follows:

Section 1 provides a comprehensive overview of the decision-making problem. Section 2 presents the design of the air combat countermeasure algorithm based on deep reinforcement learning. Section 3 conducts a thorough simulation analysis. Finally, Section 4 offers a concise summary of the study.

1 Problem Description

The core component of UAV confrontation involves the aerial firefight between UAVs, wherein they employ missiles and other weapons to engage enemy UAVs. In this paper, the scenario of air combat is designed as that UAVs equipped with identical missiles are divided into the red and the blue teams to fight. The objective is to precisely strike enemy targets while evading the threat posed by enemy missiles. To address this problem, the process is illustrated in Fig.1. Consequently, in order to establish the state space, action space, and reward function, various models need to be developed. These include the UAV dynamics model, the missile attack area model, and the confrontation angle situation model. Moreover, to facilitate decision-making training using reinforcement learning algorithms, it becomes essential to construct a Markov-based reinforcement learning model.

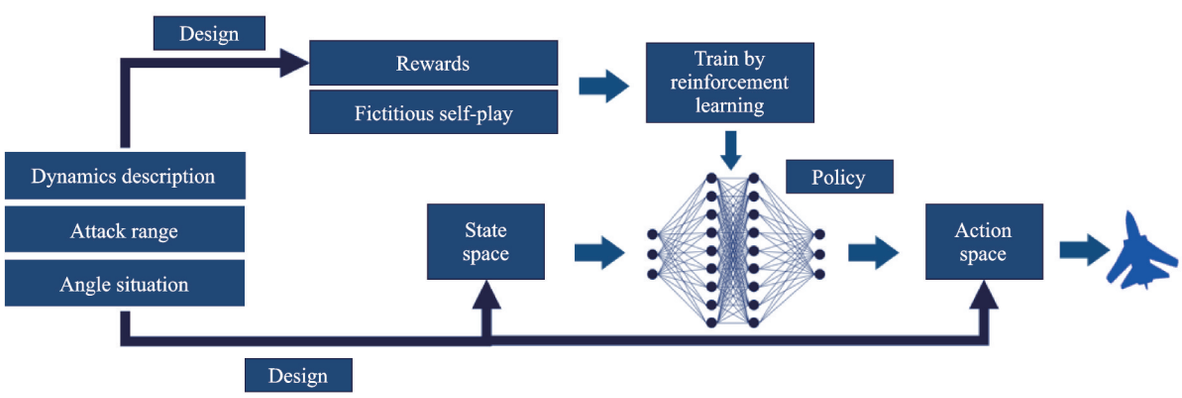

Fig.1 Framework for decision-making

1.1 Dynamics description of UAV

As depicted in Fig.2, in the ground coordinate system, the axes Ox, Oy, and Oz correspond to the eastward direction, the northward direction, and the vertical upward direction, respectively[13]. The motion model of the UAV in this coordinate system is defined as

$$\begin{cases} \dot{x} = v\cos\alpha\sin\psi \\ \dot{y} = v\cos\alpha\cos\psi \\ \dot{z} = v\sin\alpha \end{cases} \tag{1}$$

where the position of the UAV is denoted by x, y, and z; v represents the current direction of UAV velocity, while $\dot{x}$, $\dot{y}$, and $\dot{z}$ represent the rate of change of v along the three axis directions; v' corresponds to the projection of v onto the xOy plane; α denotes the pitch angle as the angle between v' and v; β represents the yaw angle referred to as the angle between v' and the y-axis. Accordingly, the dynamic model of the UAV in this coordinate system can be expressed as

$$\begin{cases} \dot{v} = g(n_x - \sin\alpha) \\ \dot{\alpha} = \dfrac{g}{v}(n_z\cos\theta - \cos\alpha) \\ \dot{\beta} = \dfrac{gn_z\sin\theta}{v\cos\alpha} \end{cases} \tag{2}$$

where g represents the acceleration due to gravity, $n_x \in \mathbf{R}$ and $n_z \in \mathbf{R}$ denote the tangential overload and the normal overload, respectively; $\theta \in [-\pi, \pi]$ represents the roll angle: $[n_x, n_z, \theta] \in \mathbf{R}^3$ serves as a practical fundamental control parameter in the UAV maneuver control mode, effectively governing the direction and magnitude of UAV speed. As a result, these three parameters are commonly employed as command inputs for air combat decision making in UAV operations. Since only the influence of the states of UAV and missile on decision-making is considered in this paper, UAV flight interference factors are not considered in the UAV dynamics modelling process.

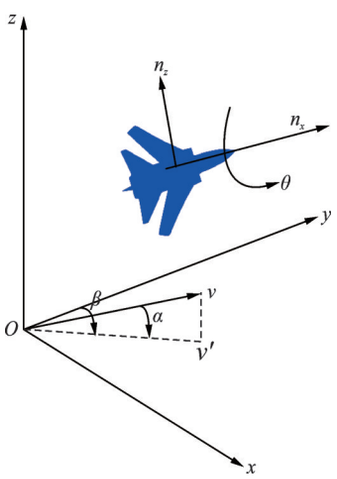

Fig.2 Motion model of UAV

1.2 Air combat model of angle and attack range

As shown in Fig.3, the effective launch area of an air-to-air missile is called the missile attack zone[14]. By launching missiles within this designated area, an UAV can effectively strike enemy aircraft while minimizing the risk of self-inflicted damage from missile explosions. The attack zone range of an air-to-air missile is determined by several factors, including the maximum attack distance $D_{\max}$, the minimum attack distance $D_{\min}$, and the maximum off-axis launch angle $\mu_{\max}$.

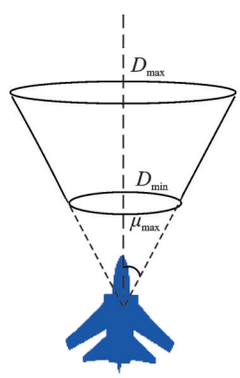

Fig.3 Effective attack area

The air-to-air missile needs time to calculate the missile launch data according to the target position, speed and other data, which is called the target locking time $t_{\max}$. Therefore, the definition is as follows: When the enemy UAV is located in the effective launch area of the UAV, the UAV is superior to the enemy UAV, and if its residence time is longer than or equal to the target locking time, the successful launching of the UAV's missile led to the destruction of the enemy UAV by the missile. It is defined as

$$\begin{cases} D_{\min} < D < D_{\max} \\ \mu < \mu_{\max} \\ t_{in} \geqslant t_{\max} \end{cases} \tag{3}$$

where t_{in} is the stay time of UAV in effective attack area; $D_{\min} < D < D_{\max}$ means the UAV is in the range of the enemy UAV attack, and this attack range is a circular truncated cone. And $D = D_{AB}\cos\varphi_A$, D_{AB} and φ_A are defined below.

According to the needs of the attack model, the angle of attack and escape of UAV are also designed. They needs to be considered during missile launching. As shown in Fig.4, $\boldsymbol{R}_A$ and $\boldsymbol{R}_B$ represent the position of the UAV and the target, respectively. φ_A represents the angle between vectors $(\boldsymbol{R}_A - \boldsymbol{R}_B)$ and $\boldsymbol{v}_A$, which is called the angle of attack. Similarly, φ_B represents the angle between vectors $(\boldsymbol{R}_B - \boldsymbol{R}_A)$ and $\boldsymbol{v}_B$, called the escape angle. They are defined as

$$\varphi_A = \arccos\left(\frac{\boldsymbol{v}_A(\boldsymbol{R}_B - \boldsymbol{R}_B)}{\|\boldsymbol{v}_A\| D_{AB}}\right) \tag{4}$$

$$\varphi_B = \arccos\left(\frac{\boldsymbol{v}_B(\boldsymbol{R}_A - \boldsymbol{R}_B)}{\|\boldsymbol{v}_B\| D_{AB}}\right) \tag{5}$$

where $D_{AB} = \|\boldsymbol{R}_A - \boldsymbol{R}_B\|$ is the distance between the drone and the target. And $0 \leqslant \varphi_A, \varphi_B \leqslant \pi$.

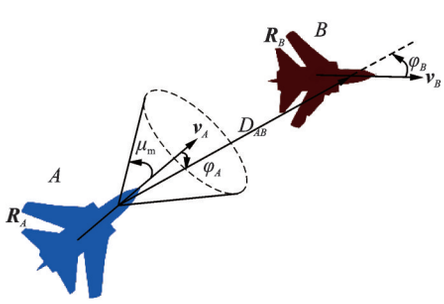

Fig.4 UAV attack and escape model

1.3 Markov decision process

The Markov decision process (MDP)[15] is often used to simulate the random strategies and rewards that agents can achieve in system states with Markov properties. If the state information of the system contains all the historical information, and the future state can be predicted from the current state without historical information, then the system has Markov property. Through the interaction between the environment and the agent, the state of the environment only depends on the state and action of the current moment, and the MDP-based reinforcement learning structure is shown in Fig.5.

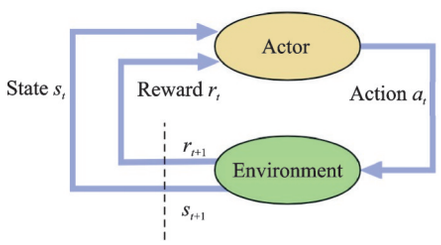

Fig.5 MDP-RL structure

In Fig.5, the agent takes the current state and reward as input, generates and executes the action after being processed by the algorithm, which will change the environment and drive the environment to the next new state. At the same time, the environment feeds back a reward value of the agent to judge the quality of this action and guides the agent to optimize the next strategy. After repeated trial-and-error learning with the environment, the best decision-making strategy for the tasks is generated and a higher long-term cumulative benefit can be obtained finally.

A quintuple $\langle S, A, P, R, \gamma \rangle$ can be used to describe a complete Markov decision process: S is the state set, $S = \{s_1, s_2, \cdots, s_t\}$, which represents all the sets of states that may exist in the environment, and can be discrete or continuous. A is the action set, $A = \{a_1, a_2, \cdots, a_t\}$, which represents all the actions that the agent may perform, and can be discrete or continuous. P is the probability of the state transition, which means that state s_t will transit to a new state s_{t+1} after action a_t is performed. The transition to a certain state cannot be completely guaranteed after a definite action is taken in a certain state, because it depends on the state transition probability. R is the reward feedback, which represents the reward value r obtained when the action a_t is taken in a certain state s_t and transferred to the next state s_{t+1}. If the agent takes a more favorable action, the environment will give a higher reward. Otherwise, it will give a certain punishment. γ is the discount factor, and $\gamma \in (0, 1)$, which is used to measure the importance of short-term and long-term incentives.

By calculating the reward value obtained by strategy π under state s_t, the expected value function $V^{\pi}(s_i)$ is obtained, and its mathematical expression is

$$V^{\pi}(s_i)=R(s_i)+\gamma\sum_{s_{i+1}\in S}P(s_{i+1}\mid s_i,\pi)V^{\pi}(s_{i+1}) \quad (6)$$

where $R(s_i)$ is an immediate reward, and $P(s_{i+1}\mid s_i,\pi)$ represents the probability that the state is transferred from s_i to s_{i+1} after the corresponding action is performed by using strategy π in a certain state s_i.

2 Algorithm Design

According to the task and decision-making of UAVs in confrontation, as well as the characteristics of cooperative operation of multi-UAVs, and according to the needs of agents in the construction of the above model, the state space, action space, reward function and multi-agent proximal policy optimization algorithm based on self-play are designed in this part. The overall framework is shown in Fig.6.

2.1 State space and action space

According to the situation data, the UAV makes decisions through the trained policy network, and executes the action after making the decision, which makes the enemy enter the missile attack area of its own side and completes the combat mission. The air combat model part has described and calculated the variables needed for UAV air combat.

Therefore, according to the requirements of air combat, combined with Eqs.(3—5), the information of the UAV itself and the enemy information that can be obtained in the state space can be described as a tuple of the following eight elements, and expressed as

$$\left\langle \boldsymbol{R}_{RB},\boldsymbol{V}_{RB},\varphi_{RB},\boldsymbol{R}_{RM},\boldsymbol{V}_{RM},\varphi_{RM},\varphi_A,\varphi_B\right\rangle \quad (7)$$

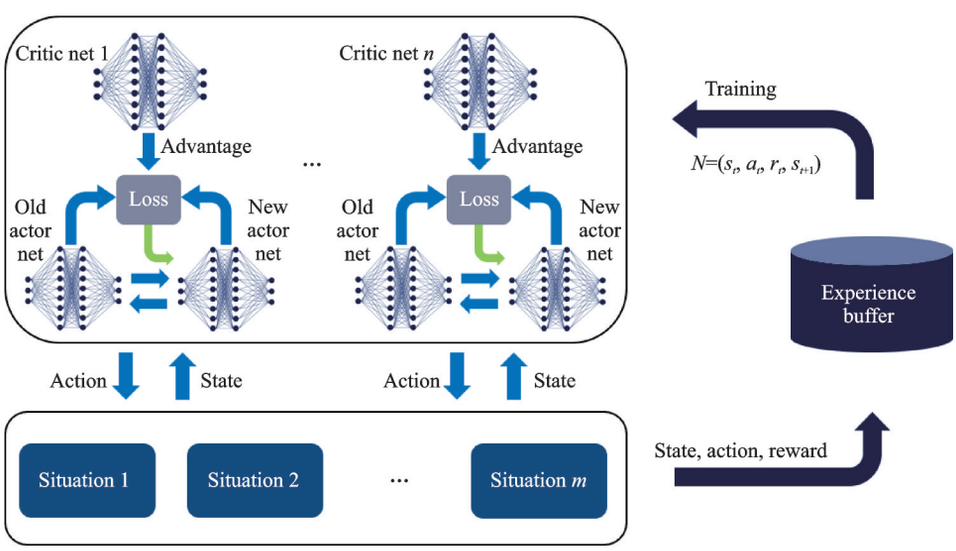

Fig.6 Training structure of UAV air combat decision

where $\boldsymbol{R}_{RB}$, $\boldsymbol{V}_{RB}$, φ_{RB} represent the relative position, the relative velocity and the velocity angle of the UAV on both sides of the confrontation, respectively; $\boldsymbol{R}_{RM}$, $\boldsymbol{V}_{RM}$, φ_{RM} the relative position, the relative velocity and the velocity angle between the UAV and the attack missile, respectively; φ_A is the attack angle and φ_B the escape angle.

It can be seen from the dynamic model of UAVs that the motion of the UAV is controlled by tangential overload n_x, normal overload n_z and roll angle θ along the speed direction. Therefore, the action space of the UAV can be designed as a tuple of three elements and represented as

$$\langle n_x,n_z,\theta\rangle \quad (8)$$

2.2 Reward function

The function of the reward algorithm is to let the UAV learn how to make the best choice in the confrontation environment in order to get the maxi-

mum reward value. In the UAV 2 v.s. 2 confrontation scenario, if the global reward is shared by all UAVs, it is easy to cause "lazy agent". When some UAVs in the group perform well in the confrontation, they will be well paid, but some UAVs will give up exploration and remain in the current state as a result, because continuing to explore may bring punishment. Therefore, it is necessary to destroy the enemy UAVs as soon as possible without being attacked by the enemies. In this paper, a reward function is designed to destroy enemy aircraft, assist friendly aircraft and avoid collision.

(1) Destroy enemy aircraft

The angle and the distance are the key factors that affect the confrontation. The following two reward functions are designed for destroying enemy aircraft.

Attack angle advantage reward function

$$R_{\varphi}=1-\frac{\left|\varphi_{A}\right|+\left|\varphi_{B}\right|}{2\pi} \tag{9}$$

where the angle of attack and the angle of escape of the two sides are φ_A and φ_B, respectively.

Distance advantage reward function

$$R_{D}=\exp\left(-\frac{d-D_{\text{opt}}}{k}\right) \tag{10}$$

where $d=|\boldsymbol{R}_{RB}|$ is the distance between the opposing sides; D_{opt} the best air combat distance; and k the adjustment factor, which adjusts the gradient of the reward function of distance superiority.

(2) Assist friendly aircraft

A reward function based on cluster collaborative goal is designed to encourage each UAV to maximize the benefits of the cluster. The reward function for coordinated attack and mutual support UAVs is defined as

$$R_{\text{H}}=-\max D \tag{11}$$

where $D=\{D_i^1,\cdots,D_i^M,\cdots,D_N^1,\cdots,D_N^M\}$ represents the collection of distances between our own drones and all enemy drones; N the number of friendly UAVs; M the number of enemy UAVs; D_i^j the distance between the ith friendly UAVs and the jth enemy UAVs; and $\max D$ the maximum distance. When multiple UAVs are close to the enemy UAVs, $\max D$ is small, and the reward value R_{H} is larger. If the other UAV in the cluster is farther away from the enemy aircraft, and the $\max D$ value is larger, the reward value R_{H} is smaller.

(3) Avoid collision

In order to avoid collisions between UAVs, collision penalties should be set. When there is no collision, the agent is given a small but positive reward value. When the minimum distance D_{c} between UAVs is less than the safe distance D_{safe}, it will be punished accordingly. The reward function is set as

$$R_{\text{C}}=\min\left(R_{\text{c1}},D_{\text{c}}-D_{\text{safe}}\right) \tag{12}$$

The final reward function is as

$$R=\lambda_{\varphi}R_{\varphi}+\lambda_{D}R_{D}+\lambda_{\text{H}}R_{\text{H}}+\lambda_{\text{C}}R_{\text{C}} \tag{13}$$

where R_{φ}, R_D, R_{H} and R_{C} are the attack angle reward, the distance reward, the assist friendly UAVs reward and the collision avoidance reward, respectively; λ_{φ}, λ_D, λ_{H} and λ_{C} the weights of the attack angle reward, the distance reward, the assist friendly UAVs reward and the collision avoidance reward, respectively.

2.3 Multi-agent proximal policy optimization

MAPPO[16] represents a pioneering multi-agent reinforcement learning algorithm (MARL) designed to address intricate multi-agent cooperative decision-making problems. Unlike conventional single agent reinforcement learning algorithms, MAPPO embraces a centralized training-distributed execution paradigm, wherein multiple agents are collectively considered as an integrated entity. Consequently, during the training phase, the strategies of all agents are simultaneously optimized, establishing a synchronized approach towards enhancing overall performance.

Multi-agent reinforcement learning is divided into three architectures: Centralized, distributed and centralized training and decentralized execution (CTDE)[17]. The system of distributed architecture is dynamic and non-Markov. The centralized structure has the problem that the input and output space is huge, so it is difficult to adapt to multi-agent systems.

To address the aforementioned challenges, a prevalent approach in multi-agent reinforcement

learning adopts the CTDE framework. During training, agents maintain a centralized critic that takes joint state-action as the input, providing an estimate of the expected reward. This enables agents to leverage information from other agents and adapt to non-Markovian and non-stationary environments. In the training, agents employ a decentralized policy based solely on their local implementation, alleviating the need for inter-agent communication and facilitating scalability in large-scale scenarios. The CTDE framework has been widely used in practical applications, making it an effective and utilized approach in MARL.

The fundamental premise behind the MAPPO algorithm lies in employing proximal policy optimization, which enables efficient coordination amid the training process of multiple agents. Primarily leveraging the proximal policy optimization (PPO) method, the PPO algorithm accurately computes the optimization gradient directly from the current policy distribution by optimizing the action strategy, consequently facilitating updates to the policy parameters.

Building upon the PPO framework, the MAPPO algorithm fosters harmonious training interactions amidst diverse agents through the incorporation of a shared value network and a traceable experience playback buffer. These additions serve to streamline the training process, ensuring cooperative learning and enabling efficient knowledge transfer among the agents.

Utilizing the acclaimed actor-critic framework[18], as illustrated in Fig.7, the MAPPO algorithm incorporates an actor network and a critic network, employing a vector parameterized neural network to represent strategies. The actor network plays a pivotal role in processing pertinent battlefield information, acquired by the agents, through neural network computations, ultimately generating the agent's action output. Such battlefield information, in this study, encompasses crucial details such as the position and speed information pertaining to both adversaries.

The input parameters of the actor network include the UAV relative position the $\boldsymbol{R}_{RB}$, the UAV relative velocity $\boldsymbol{V}_{RB}$, the UAV velocity angle φ_{RB}; the UAV missile relative position $\boldsymbol{R}_{RM}$, the missile relative velocity $\boldsymbol{V}_{RM}$, the missile velocity angle φ_{RM}; the attack angle φ_A and escape angle φ_B. The attack angle and the escape angle are described in detail in the air combat model of this paper.

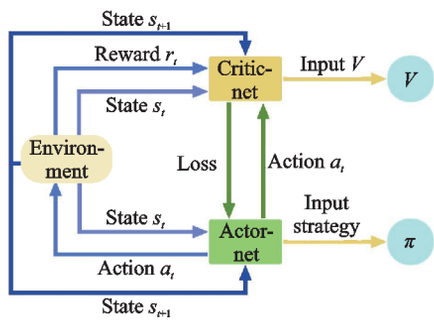

Fig.7 Actor-critic framework

The output parameters of the actor network are the action information of the UAV, including the tangential overload n_x, the normal overload n_z and the roll angle θ.

The objectives of the actor network optimization are

$$L_A=\frac{1}{Bn}\sum_{i=1}^{B}\sum_{k=1}^{n}\left[\min\left(r_i^k A_{fi}^k,\operatorname{clip}\left(r_i^k,1-\varepsilon,1+\varepsilon\right)A_{fi}^k\right)\right]+\sigma\frac{1}{Bn}\sum_{i=1}^{B}\sum_{k=1}^{n}S_e(\pi_i^k) \tag{14}$$

$$\operatorname{clip}(r_i^k,1-\varepsilon,1+\varepsilon)=\begin{cases}r_i^k & 1-\varepsilon\leqslant r_i^k\leqslant 1+\varepsilon\\ 1-\varepsilon & r_i^k<1-\varepsilon\\ 1+\varepsilon & 1+\varepsilon<r_i^k\end{cases} \tag{15}$$

where B is the size of the batch_size; n the number of agents; A_f the advantage function; π the policy function; S_e the entropy of the strategy; σ a super parameter of the control entropy coefficient; and r the ratio of the new and the old strategy functions; and ε a super parameter of clip function.

Critic network guides the training of the actor network by evaluating the value of the state behavior. Critic network receives battlefield situation information and agent action, and outputs the action state value.

Battlefield situation information includes the $\boldsymbol{R}_{RB}$, $\boldsymbol{V}_{RB}$, φ_{RB}, $\boldsymbol{R}_{RM}$, $\boldsymbol{V}_{RM}$, φ_{RM}, φ_A and φ_B. Agent action parameters are n_x, n_z and θ. The output pa-

rameter of the critic network is the state value $V(s, a)$ of the agent.

The objectives of the critic network optimization are

$$L_C = \frac{1}{Bn}\sum_{i=1}^{B}\sum_{k=1}^{n}[\max((V(s_i^k)-R_i)^2, (\mathrm{clip}(V(s_i^k), V_{\mathrm{old}}(s_i^k)-\varepsilon, V_{\mathrm{old}}(s_i^k)+\varepsilon)-R_i)^2)] \tag{16}$$

$$\mathrm{clip}(V(s_i^k), V_{\mathrm{old}}(s_i^k)-\varepsilon, V_{\mathrm{old}}(s_i^k)+\varepsilon) = \begin{cases} V(s_i^k) & V_{\mathrm{old}}(s_i^k)-\varepsilon \leqslant V(s_i^k) \leqslant 1+\varepsilon \\ V_{\mathrm{old}}(s_i^k)-\varepsilon & V(s_i^k) < V_{\mathrm{old}}(s_i^k)-\varepsilon \\ V_{\mathrm{old}}(s_i^k)+\varepsilon & V_{\mathrm{old}}(s_i^k)+\varepsilon < V(s_i^k) \end{cases} \tag{17}$$

where V is the value function and R the discount reward.

Remark 1 In this paper, the traceable buffer with the characteristic of remembering the history information is used to weight each advantage function from 1 to n iteratively, and to take the result as the final advantage function. It can balance the span of agent exploration and reduce variance. The advantage function is designed as

$$A_{\mathrm{fi}}^k = \sum_{j=0} \xi^j \omega_i^k \tag{18}$$

where $\omega_i^k = R_i^k + \gamma V^\pi(s_{i+1}, a_{i+1}) - V^\pi(s_i, a_i)$; ξ is the weight that balances the steps from 1 to n, γ a discount factor.

2.4 FSP-MAPPO

Based on the game theory, it can be argued that UAV confrontation is characterized as an imperfect information game. In other words, the UAV serves as the decision-maker, with each participant holding the ability to observe their own state and make decisions. However, when participants act simultaneously, they are unable to obtain action information from others. Applying the reinforcement learning autonomous training method to facilitate the UAV game process, even though both UAVs operate in the same combat environment, they possess independent decision networks and replay buffers. This enables them to train strategy networks independently during confrontations.

The strategy model acquired through autonomous training tends to excessively adapt to the training environment and specific situations, resulting in a limited capacity for generalization. Consequently, when it is implemented in a novel environment, the policy model may struggle to make accurately informed decisions. In such scenarios, UAVs are compelled to begin the learning process from scratch, demanding considerable training time. To address the challenge of over-adaptation in UAV strategy models, the FSP-MAPPO algorithm drawing from the principles of FSP theory is introduced[19]. This algorithm serves the purpose of training UAVs in confrontational settings.

The algorithm flow is shown in Algorithm 1. In the process of self-training, agents generate experiences or data such as states and actions in different simulation scenes and put them into the same experience buffer, and then each agent calculates the optimal strategy by strengthening the behavior in the experience pool. Finally, each agent updates its own strategy through the action-critic network.

Algorithm 1 FSP-MAPPO

(1) Initialize the relevant network parameters of the MAPPO algorithm: The reward discount rate, the learning rate of the actor and the critic network, batch_size
(2) Initialize the experience buffer
(3) Initialize m combat situations
(4) For episode $= 1, 2, \cdots, E$:
(5) Obtain the initial state s of UAVs in m situations
(6) For $t = 1, 2, \cdots, T$:
(7) If the confrontation is not over:
(8) Input the normalized combat s_t to the actor network to get action a
(9) and carry out the action in various situations to obtain various states s_{t+1};
(10) Calculate the UAVs attack angle reward, distance reward, reward to assist friendly aircraft
(11) and collision avoidance reward in different situations, and get the final reward r_t by weighting;
(12) Store m groups of experience in the experience pool.
(13) If the number of experience entries in the experience pool reaches the preset batch_size:

(14) Input the status, actions and rewards of the agent into the value network
(15) For UAV $i = 1, 2, \cdots, N$:
(16) Input the status, actions and rewards of the agent into the value network L;
(17) Update the value network according to Eq.(14);
(18) Input advantage estimate L into the actor network, calculate action an according to state S,
(19) compare new and old networks, and update Eq.(15);
(20) end for
(21) else: break
(22) else: break
(23) end for
(24) episode = episode + 1
(25) end for

Remark 2 Specifically, in Algorithm 1, each FSP agent plays the game with other FSP agents in different situations. The result of each FSP agent's play, denoted as (s_t, a_t, r_t, s_{t+1}), is saved in the experience buffer, and the best response (s_B, a_B) is calculated. When the buffer reaches its maximum size, the action critic network is used for iteration. The loss function of the actor network and critic network are formulated as Eq.(14) and Eq.(15), respectively.

The specific update method for the action policy is as follows. In this paper, the historical actions and best responses stored in the traceable buffer are used to train the average policy and the best policy. The average policy is updated based on the weight function.

$$\pi(s,a) \propto \sum_{k=1}^{n} \mu_k x_k(s) \Pi_k(s,a) \tag{19}$$

$$\pi_i = (1-\lambda_i)\pi_{i-1} + \lambda_i \zeta_i \tag{20}$$

where i represents the iteration count; k the network identifier; π the average policy; Π the policies generated by each network; ζ the best response to the current average policy obtained through policy network training; $\mu x(s)$ a normalization constant related to the state s; and $\lambda \in (0,1)$ a weight coefficient.

3 Simulation and Analysis

In this paper, the environment model of attack and flight air combat of UAV is constructed by using python language and JSBSim dynamics model[20]. The application of FSP-MAPPO algorithm in UAV countermeasure missions is realized, and the effectiveness of the algorithm is compared. The simulation experiment takes the red-blue air combat confrontation, as shown in Fig.8, as the background, and UAVs compete for the airspace of the key places. The mission goal of both sides is to shoot down all the other UAVs, or to limit the local UAV to be in an inferior position in the airspace of the key area. In the experiment, the blue side UAV is an agent trained by the deep reinforcement learning algorithm, and the red side UAV is a non-agent. The algorithm parameters are shown in Table 1.

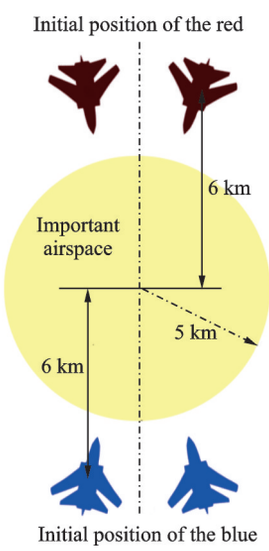

Fig.8 Diagram of simulation confrontation plane

Table 1 Parameter settings of MAPPO algorithm

Parameter	Value
Training episode	1 000
Time step	0.2
Buffer size	3 000
Discount factor	0.99
Clip params	0.2
Learning rate	0.000 3
Entropy coef	0.000 1
Recurrent hidden size	128
Number of parallel training	4

In order to verify the effectiveness of the decision-making and show the process of UAV confrontation, the simulation experiments are carried out and the results are shown in the Figs.9—11.

Fig.9 shows the process of 2B v.s. 2R confrontation. Fig.9(a) represents the initial positions of the UAVs, and the confrontation positions of the UAVs are generated when they meet the requirements of the confrontation environment.

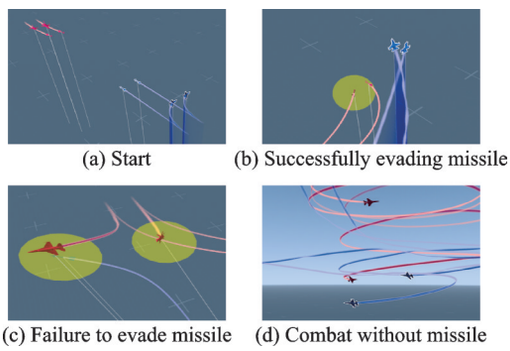

Fig.9 Process of 2B v.s. 2R confrontation

Figs.9(a—c) show the process from the beginning of the operation of the UAV to the end of the confrontation. At the beginning, the drones of both sides fired missiles at each other, but the blue UAV equipped with the training algorithm turned to both sides in a more timely manner, thus avoiding the explosion range of the missile. However, without the training algorithm, the red UAV did not escape in time, and did not flee the explosion range of the missile in the first time, so it failed in the confrontation.

Fig.9(d) shows the confrontation process of a UAV when the missile is exhausted or when the missile is not carried by the UAV. When the UAV does not carry missiles or runs out of missiles, the UAV will change the previous strategy of launching missiles-evading missiles, to cooperate with friendly aircraft to limit each other's drones to an inferior posture.

To validate the generalization ability of decision-making in different situations, the scenario based on a 2 v.s. 2 adversarial setup is designed, as shown in Figs.10, 11.

In Fig.10(a), the number of blue team drones is reduced to one, and the initial position is set in such a way that the side of the blue team faces the front of the red team. At the beginning, the blue UAV immediately accelerates to fly within the minimum missile attack range of the red UAV to avoid being hit by missiles launched by the red team in a disadvantaged positions as shown in Fig.10(b). Then, it continues to evade the red team encirclement and ultimately avoids their attacks as shown in Figs.10(c,d).

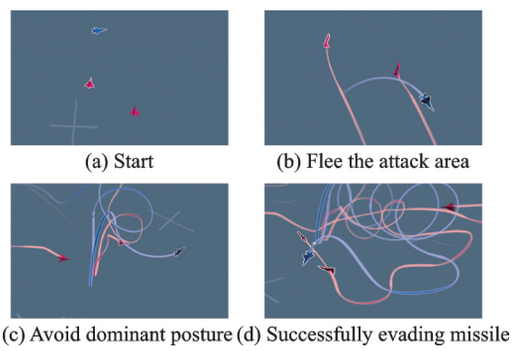

Fig.10 Process of 1B v.s. 2R confrontation

In Fig.11(a), the number of UAVs for the red and the blue teams are set as 4 and 3, respectively. The red team are arranged in a neat formation, while the blue team are in a scattered formation. At the beginning, one blue UAV engages in a distraction tactic, similar to that in Fig.10 to divert the attention of two red UAVs. The other two blue UAVs cooperate to destroy one red UAV. That is in Figs.11(b, c). Although the distracting UAV is destroyed, it creates an advantageous position for its friendly UAVs. Ultimately, the blue team manages to destroy three red team UAVs and emerges victorious in Fig.11(d).

As depicted in Fig.12, the incorporation of a collision avoidance reward function helps to mitigate the collisions occurring due to conflicts during coop-

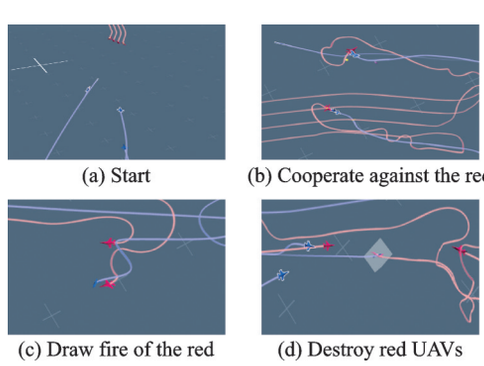

Fig.11 The process of 3B v.s. 4R confrontation

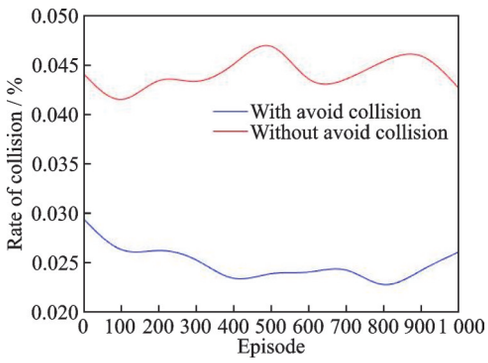

Fig.12 Collision rate with or without collision avoidance

erative adversarial scenarios, consequently minimizing the unnecessary damage incurred by the drones during the adversarial process.

In order to better evaluate the convergence speed of the algorithm, the total reward and the loss value of the strategy function in each round are recorded in the experiment, and the convergence of the training network can be judged. The total rewards curve and strategy function loss curve of the FSP-MAPPO algorithm, the MAPPO algorithm and the Q-learning algorithm under the same initial conditions are shown in Figs.13—14.

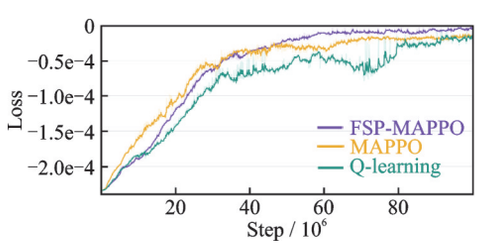

Fig.13 Loss of policy

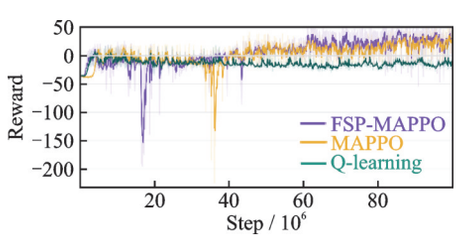

Fig.14 Reward curves

It can be seen that in 20×10^6—40×10^6 steps, because the FSP-MAPPO algorithm needs self-game training, the early convergence and reward acquisition of the FSP-MAPPO algorithm are worse than those of the MAPPO algorithm without self-play, but the three algorithms achieve local convergence in 40×10^6—60×10^6 steps. From the point of view of convergence and reward acquisition, there is little difference between the FSP-MAPPO algorithm and the MAPPO algorithm, but both are better than the Q-learning algorithm, and the FSP-MAPPO algorithm quickly jumps out of the local convergence.

The MAPPO algorithm does not jump out of the local convergence until 60×10^6 steps, and the Q-learning algorithm does not jump out of the local convergence until 80×10^6 steps. Moreover, the convergence speed, the ability to jump out of the local convergence, the final convergence and the reward of FSP-MAPPO are superior to those of other comparison algorithms.

4 Conclusions

Aiming at the countermeasure problem of UAVs with missiles, an autonomous decision-making algorithm of intelligent air combat based on the deep reinforcement learning is proposed in this paper. The UAV situation assessment model including missile attack area is established. The attack angle advantage, distance advantage, assist friendly aircraft and avoid collision reward function are comprehensively considered to guide the agent to converge to the optimal solution and avoid the sparse reward problem in reinforcement learning. Based on the MAPPO algorithm and the FSP algorithm, FSP-MAPPO is proposed to optimize the advantage function and policy generation, improving the performance and generalization ability of decision-making.

On this basis, the air combat simulation of this method is realized and compared with other methods. The results show that the FSP-MAPPO algorithm has better convergence speed and reward acquisition ability, and is more suitable for solving UAV confrontation decision-making problems.

References

[1] GACOVSKI Z, DESKOVSKI S. Modelling of combat actions via fuzzy expert system[C]//Proceedings of the RTO NMSG Conference on Future Modelling and Simulation Challenges. Breda, Netherlands: [s. n.], 2001: RTO-MP-073.

[2] ZHOU K, WEI R, XU Z, et al. A brain like air combat learning system inspired by human learning mechanism[C]//Proceedings of 2018 IEEE CSAA Guidance, Navigation and Control Conference (CGNCC). Xiamen, China: IEEE, 2018: 1-6.

[3] HUANG C, DONG K, HUANG H, et al. Autonomous air combat maneuver decision using Bayesian inference and moving horizon optimization[J]. Journal of Systems Engineering and Electronics, 2018, 29(1): 86-97.

[4] SCUKINS E, KLEIN M, KROON L, et al. Monte carlo tree search and convex optimization for decision support in beyond-visual-range air combat[C]//Proceedings of 2023 International Conference on Unmanned Aircraft Systems (ICUAS). Warsaw, Poland : IEEE, 2023: 48-55.

[5] LI S, CHEN M, WANG Y, et al. Air combat decision-making of multiple UCAVs based on constraint strategy games[J]. Defence Technology, 2022, 18(3): 368-383.

[6] LI W, SHI J, WU Y, et al. A multi-UCAV cooperative occupation method based on weapon engagement zones for beyond-visual-range air combat[J]. Defence Technology, 2022, 18(6): 1006-1022.

[7] GAO Yangyang, YU Minjian, HAN Qisong, et al. Air combat maneuver decision-making based on improved symbiotic organisms search algorithm[J]. Journal of Beijing University of Aeronautics and Astronautics, 2019, 45(3): 429-436. (in Chinese)

[8] DENG Ke, PENG Xuanqi, ZHOU Deyun. Study on air combat decision method of UAV based on matrix game and genetic algorithm[J]. Fire Control & Command Control, 2019, 44(12): 61-66. (in Chinese)

[9] DANTAS J P A, MAXIMO M R O A, COSTA A N, et al. Machine learning to improve situational awareness in beyond visual range air combat[J]. IEEE Latin America Transactions, 2022, 20(8): 2039-2045.

[10] YANG K, DONG W, CAI M, et al. UCAV air combat maneuver decisions based on a proximal policy optimization algorithm with situation reward shaping[J]. Electronics, 2022, 11(16): 2602.

[11] LI B, BAI S, LIANG S, et al. Maneuver decision-making of unmanned aerial vehicles in air combat based on an expert actor-based soft actor critic algorithm[J]. CAAI Transactions on Intelligence Technology, 2023, 8(4): 1608-1619.

[12] CRUMPACKER J B, ROBBINS M J, JENKINS P R. An approximate dynamic programming approach for solving an air combat maneuvering problem[J]. Expert Systems with Applications, 2022, 203: 117448.

[13] HU J, WANG L, HU T, et al. Autonomous maneuver decision making of dual-UAV cooperative air combat based on deep reinforcement learning[J]. Electronics, 2022, 11(3): 11030467.

[14] JIAO Ke, PI Yiming, LU Jizhen. A fast calculation method of launch envelope of air-to-air missile[J]. Fire Control Command Control, 2010, 35(6): 100-102.(in Chinese)

[15] PUTERMAN M L. Markov decision processes: Discrete stochastic dynamic programming[M]. [s.n.]: John Wiley & Sons, 2014.

[16] GUO D, TANG L, ZHANG X, et al. Joint optimization of handover control and power allocation based on multi-agent deep reinforcement learning[J]. IEEE Transactions on Vehicular Technology, 2020, 69(11): 13124-13138.

[17] FOERSTER J, FARQUHAR G, AFOURAS T, et al. Counter-factual multi-agent policy gradients[C]//Proceedings of the AIAA Conference on Artificial Intelligence. [S.l.]:AIAA, 2018.

[18] FU H, LIU W, WU S, et al. Actor-critic policy optimization in a large-scale imperfect information game[C]//Proceedings of International Conference on Learning Representations. [S.l.]: [s.n.], 2021: 1-28.

[19] HEINRICH J, LANCTOT M, SILVER D. Fictitious self-play in extensive-form games[C]//Proceedings of International Conference on Machine Learning. [S.l.]: ACM, 2015: 805-813.

[20] DE MARCO A, D'ONZA P M, MANFREDI S. A deep reinforcement learning control approach for high-performance aircraft[J]. Nonlinear Dynamics, 2023, 111(18): 17037-17077.

Acknowledgement This work was supported by the National Natural Science Foundation of China (No.62173242).

Authors Dr. **WANG Mingming** received her B.S. degree in computer science and technology from Henan University, Kaifeng, China, in 2008, and her Ph.D. degree in pattern recognition and intelligent systems from Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China, in 2016. She is currently a lecturer with School of Electrical and Information Engineering, Tianjin University, Tianjin, China. Her research interests include unmanned aerial vehicle systems environmental perception and decision-making, etc.

Prof. **QI Juntong** received his B.S. degree in automation

from Tianjin University, Tianjin, China, in 2004, and his Ph. D. degree in pattern recognition and intelligent systems from Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China, in 2009. He is currently a professor, and doctoral supervisor with School of Future Technology, Shanghai University, Shanghai, China. His current research interests include environmental perception and understanding, behavior and decision-making optimization, environmental interaction, and multirobot cooperation, etc.

Author contributions Dr. **WANG Mingming** and Mr. **ZHANG Baoyong** designed studies, modeled experiments, conducted analyses, interpreted results, and wrote the manuscript. Dr. **WU Chong** provided data and modeled components for the UAV model and for the analysis of the comparative trials. Dr. **PING Yuan** and Prof. **QI Juntong** contributed to the discussion and background of the study. All authors commented on the draft manuscript and approved submission.

Competing interests The authors declare no competing interests.

(Production Editor: ZHANG Bei)

基于虚拟自博弈多智能体近端优化策略的无人机对抗决策

王明明[1], 张宝勇[1], 吴 冲[2], 平 原[2], 齐俊桐[3]

(1.天津大学电气与信息工程学院,天津 300072,中国; 2.一飞(海南)科技有限公司研发中心,三亚 572025,中国; 3.上海大学未来学院,上海 200444,中国)

摘要:研究了基于虚拟自博弈多智能体近端策略优化的无人机对抗决策问题。无人机对抗依赖自主决策,使无人机能够根据环境信息生成行动指令。提出了一种基于红蓝空战任务的无人机对抗自主决策方法。首先,采用导弹攻击区域与无人机之间的相对角度来评估当前情况。然后,以场景评估为指导,进行状态空间、动作空间和实时奖励反馈的设计,简化训练过程。在此基础上,提出了一种利用虚拟自博弈多智能体近端策略的方法,旨在从训练数据的经验缓冲区中推导出优势函数和平均策略。最后,通过对无人机执行红蓝对抗任务的仿真,验证了该方法的有效性和优势所在。

关键词:无人机;空战;多智能体近端优化策略;决策