SCIENCE CHINA
Information Sciences

August 2024, Vol. 67, Iss. 8, 180204:1–180204:18
https://doi.org/10.1007/s11432-023-4088-2

• **RESEARCH PAPER** •
Special Topic: UAV Swarm Autonomous Control

UAV swarm air combat maneuver decision-making method based on multi-agent reinforcement learning and transferring

Zhiqiang ZHENG, Chen WEI & Haibin DUAN*

State Key Laboratory of Virtual Reality Technology and Systems,
School of Automation Science and Electrical Engineering, Beihang University, Beijing 100083*, China*

Received 28 September 2023/Revised 19 March 2024/Accepted 17 April 2024/Published online 24 July 2024

Abstract During short-range air combat involving unmanned aircraft vehicle (UAV) swarms, UAVs must make accurate maneuver decisions based on information from both enemy and friendly UAVs. This dual requirement of competition and cooperation presents a significant challenge in the field of unmanned air combat. In this paper, a method based on multi-agent reinforcement learning (MARL) is proposed to address this issue. An actor network containing three subnetworks that can handle different types of situational information is designed. Hence, the results from simpler one-on-one scenarios are leveraged to enhance the complex swarm air combat training process. Separate state spaces for local and global information are designed for the actor and critic networks. A detailed reward function is proposed to encourage participation. To prevent lazy participants in air combat, a reward assignment operation is applied to distribute these dense rewards. Simulation testing and ablation experiments demonstrate that both the transfer operation and reward assignment operation can effectively deal with the swarm air combat scenario, and reflect the effectiveness of the proposed method.

Keywords UAV swarm, short-range air combat, multi-agent reinforcement learning, reward assignment, transfer

1 Introduction

Owing to advancements in related technologies and their various excellent features, like structural simplicity and minimized human risk, unmanned aerial vehicles (UAVs) have been successfully utilized in inspection, search, rescue tasks, and air-to-air combat [1, 2]. The demand for transitioning from single UAV missions to swarm deployments and the inherent challenges of model uncertainty, strong coupling, and complex disturbances have led to the continuous proposal of highly efficient control strategies [3, 4]. The short-range air combat is a significant application domain for UAVs. However, the capabilities of a single UAV are often insufficient to handle the increasing complexity of air combat missions [5]. Consequently, multi-UAV cooperative air combat has emerged as a crucial contemporary combat paradigm, presenting both significant research opportunities and formidable challenges [6]. Recent advancements of artificial intelligence (AI) technology have expanded the combat capabilities of single UAVs and brought fresh vigor into UAV swarm short-range air combat [7].

During unmanned air combat, both sides constantly execute maneuvers according to their respective policies, resulting in rapidly changing dynamic situations. Numerous methods have been used to solve the problem of unmanned air combat problem, mainly classified into optimization methods, game theory methods, and AI-based methods [8, 9].

The optimization methods aim to transform the air combat maneuver decision-making problem into an optimization problem through modeling [10]. Typically, this involves solving a multi-objective optimization problem using a suitable optimization method [11]. Duan et al. [12] framed the dynamic task allocation problem in swarm air combat as a real-time decision-making problem, considering engagement,

* Corresponding author (email: hbduan@buaa.edu.cn)

attrition, and other factors, then an improved particle swarm optimization (PSO) is devised to solve this intricate problem. Bayesian inference is also employed to estimate the combat situations, with the entire air combat maneuver decision-making process solved by moving horizon optimization [13]. Although optimization methods can search for better solutions within a vast solution space, they require significant computational resources, making real-time decision-making challenging [5]. The game theory methods establish maneuver decision-making policies based on several types of game theories, such as differential games and influence diagrams. For highly dynamic swarm pursuit scenarios, the innovative differential game theory provides a reference for the overall index design [14]. Liu et al. [15] have designed an air combat embedded training system based on the extended influence diagram framework, improving the system's authenticity and practicality. However, the strong dynamism of air combat scenarios leads to increased computational complexity, making it difficult for game theory methods to consider all influencing factors and deliver real-time performance.

Increasingly, researchers are turning on AI-based methods, especially reinforcement learning (RL), which enables the agent to evolve through "trial and error" interactions with the environment, facilitating real-time maneuver decision-making [5, 16, 17]. Yang et al. [9] have designed a maneuver decision-making model based on deep Q network (DQN), combined with a one-on-one air combat evaluation model, to achieve autonomous decision-making in high-dimensional state and action spaces. Motivational curriculum learning has been integrated with a type of RL algorithms to provide special rewards to the agent when it displays with unsatisfactory behaviors in [18]. To address UAV swarm air combat, multi-agent reinforcement learning (MARL) algorithms are introduced, which are widely used in multi-individual cooperation and competition problems [19–21]. An improved communication network, serving as the UAVs' communication channel, is introduced to design the actor network, thereby enhancing cooperation abilities [22]. To eliminate the need for expert knowledge, a multi-agent hierarchical policy gradient algorithm learns maneuver policies through self-play, achieving excellent performance in both defense and offense scenarios [7]. Parallel and decoupling strategies are introduced into the unmanned swarm confrontation game based on MARL algorithms with the simplified UAV motion model [23].

One-on-one air combat, as a special case of swarm air combat, involves the agent interacting with an opposing player, resulting in a more stable system. However, in swarm air combat, agents must engage with both teammates and opponents, collaboratively making decisions, which significantly increases system complexity and instability. Moreover, the expanded decision space in swarm combat environments further complicates decision-making increasing the likelihood of encountering "lazy agent", thus challenging cooperative operations.

Simplified scenarios, such as one-on-one air combat, can expedite the training process for complex scenarios, providing foundational insights into essential aspects of swarm air combat, including maneuvering tactics and evasion techniques. These insights offer valuable assistance for swarm air combat and constitute a viable method to tackle swarm control issues. However, current methods mainly focus on one-on-one or swarm air combat, rarely considering the interplay between the two.

Informed by the above discussion, this paper proposes an MARL-based method for short-range air combat maneuver decision-making. The method includes designing a detailed swarm air combat environment and accelerating the training process by transferring and reward assignment. Compared to previous studies [5, 17, 22], the main contributions of this paper are as follows.

(1) Tailored UAV swarm air combat environment. A comprehensive representation of the air combat process, including a strategy to depict the relative relationships between opposing parties. Departing from previous studies, this paper introduces a blood mechanism that escalates complexity and interaction intensity between both sides. Additionally, it devises a local state space for distributed execution by RL agents and a global state space designed to encapsulate an overall situation evaluation.

(2) Network transfer for UAV swarm air combat. The actor network design is modularized, allowing the transfer of an actor network from one-on-one scenario to handle secondary enemy UAV information. This establishes connections within more complex UAV swarm air combat scenarios. Simultaneously, a phased multistep training process is adopted to ensure both feasibility and expediency in training RL agents.

(3) Reward function and reward assignment for swarm air combat. A reward function encompassing situation, events, and end-game specifics is designed to enhance RL agents' learning efficiency. Inspired by the credit assignment problem in multi-agent systems, a reward assignment tip redistributes individual rewards based on their contributions to the swarm. This effectively mitigates the emergence of "lazy agents" and enhances collaborative capabilities.

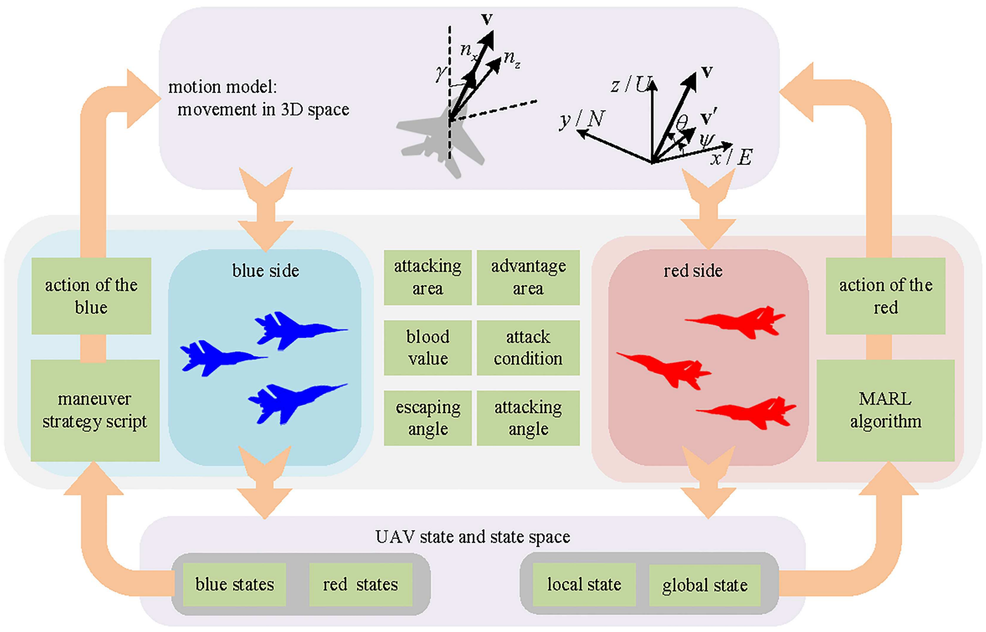

Figure 1 (Color online) Environment of UAV swarm short-range combat.

The rest paper is organized as follows. Section 2 introduces the UAV swarm air combat environment, mainly including the motion model, designed action and state spaces, and maneuver strategy script for the enemy. Section 3 presents the maneuver decision method using RL, including the designed networks, reward function, and training process. Next, the simulation results are discussed in Section 4. Ultimately, Section 5 concludes the findings of the study.

2 UAV swarm air combat environment

The UAV swarm short-range air combat maneuver decision-making problem is complex, requiring consideration of environmental physical constraints and the states of UAVs from both sides, which are dynamically changing. This section combines the characteristics of air combat to introduce the UAV model, the designed action and state space, and the opponent's maneuver strategy script to complete the modeling of the air combat environment, as shown in Figure 1.

2.1 UAV motion model

During short-range air combat, the UAV is primarily concerned with the positional and velocity relationships with other UAVs. Consequently, the UAV motion model is simplified to a three-degree-of-freedom motion model, placing greater emphasis on the decision-making process rather than the underlying flight control. In this model, the body coordinate system Σ_b is fixed with the UAV, and the UAV's velocity direction is always aligned with the ox axis of Σ_b. The simplified motion model is established in the ground coordinate system Σ_g, which is regarded as an inertial coordinate system. In Σ_g, the directions are defined as follows: east (E) along the ox axis, north (N) along the oy axis, and upward direction (U) along the oz axis. The UAV motion model can be expressed as [9]

$$\begin{cases} \dot{x} = v\cos\theta\cos\psi, \\ \dot{y} = v\cos\theta\sin\psi, \\ \dot{z} = v\sin\theta, \\ \dot{v} = g\left(n_x - \sin\theta\right), \\ \dot{\theta} = \dfrac{g}{v}\left(n_z\cos\gamma - \cos\theta\right), \\ \dot{\psi} = \dfrac{gn_z\sin\gamma}{v\cos\theta}, \end{cases} \tag{1}$$

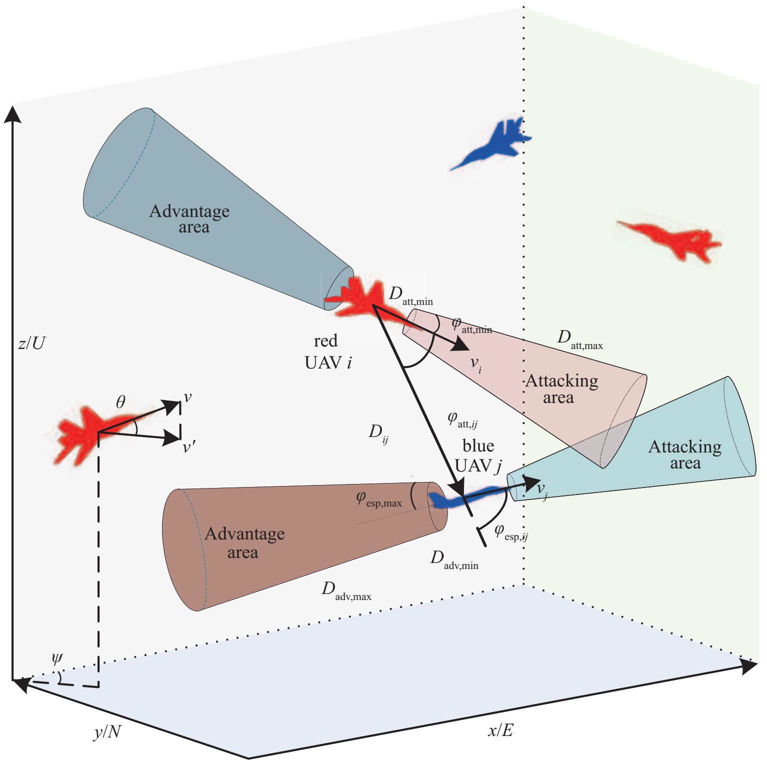

Figure 2 (Color online) Scenario of UAV swarm short-range combat in the ground coordinate system.

where $\boldsymbol{p} = [x, y, z]^{\mathrm{T}}$ and $\boldsymbol{v}$ are the UAV's position and velocity vectors in Σ_g, and $\dot{x}$, $\dot{y}$ and $\dot{z}$ are the component values of $\boldsymbol{v}$ in three directions, which means $\boldsymbol{v} = [\dot{x}, \dot{y}, \dot{z}]^{\mathrm{T}}$. γ is the flight-path bank angle around $\boldsymbol{v}$. θ represents the angle between $\boldsymbol{v}'$ and $\boldsymbol{v}$, and ψ is the angle between $\boldsymbol{v}'$ and ox axis, where $\boldsymbol{v}'$ is the projection of $\boldsymbol{v}$ on the xoy plane. g signifies the acceleration of gravity, n_x denotes the overload in velocity direction, and n_z is the normal overload. The control input vector $\boldsymbol{u} = [n_x, n_z, \gamma]^{\mathrm{T}} \in \mathbb{R}^3$ of the motion model is utilized to control the UAV state $\boldsymbol{s} = [\boldsymbol{p}^{\mathrm{T}}, \boldsymbol{v}^{\mathrm{T}}]^{\mathrm{T}}$ while the model is working. In addition, considering constraints, such as the flight performance of the UAV, the UAV model has to fulfill some conditions, which are represented as

$$\begin{cases} v_{\min} \leqslant v \leqslant v_{\max}, \\ \theta_{\min} \leqslant \theta \leqslant \theta_{\max}, \\ -\pi < \gamma \leqslant \pi, \\ 0 \leqslant \psi < 2\pi, \\ n_{x\min} \leqslant n_x \leqslant n_{x\max}, \\ n_{z\min} \leqslant n_z \leqslant n_{z\max}, \end{cases} \tag{2}$$

where subscripts min and max denote the minimum and maximum values. Therefore, with $\boldsymbol{s}$ and $\boldsymbol{u}$ at time step t, $\boldsymbol{s}$ at next time step can be found by (1) and (2) using the Runge-Kutta method.

2.2 Air combat scenario

In an episode of UAV swarm air combat, the mission for UAVs on both sides is to cooperate with their teammates to destroy all opposing UAVs. The red UAVs, denoted as Ω_r with a membership count of n_r, adopt a maneuver decision-making controller based on an RL algorithm. Meanwhile, the UAVs of the blue side, represented as Ω_b with a membership count of n_b, follow a maneuver strategy script designed to direct the blue UAV to tail and destroy the red UAVs. The scenario is illustrated in Figure 2. Ω_b' and Ω_r' denote the surviving UAVs on two sides and taking UAV $i \in \Omega_r'$ and UAV $j \in \Omega_b'$ in the following text.

The UAV i can launch an attack against UAV j when the attack conditions are as follows:

$$\begin{cases} D_{\text{att,min}} \leqslant D_{ij} \leqslant D_{\text{att,max}}, \\ \varphi_{\text{att},ij} \leqslant \varphi_{\text{att,max}}, \\ \varphi_{\text{esp},ij} \leqslant \varphi_{\text{esp,max}}, \end{cases} \tag{3}$$

where $D_{ij} = \|\boldsymbol{p}_i - \boldsymbol{p}_j\|$ is the distance between UAV i and j, and $D_{\text{att,max}}$ and $D_{\text{att,min}}$ are the maximum and minimum attackable distance, respectively. $\varphi_{\text{att},ij}$ and $\varphi_{\text{esp},ij}$ are the attacking angle and escaping angle between UAV i and j, respectively, and $\varphi_{\text{att,max}}$ and $\varphi_{\text{esp,max}}$ are the thresholds. $\varphi_{\text{att},ij}$ and $\varphi_{\text{esp},ij}$ are defined as [24]

$$\begin{aligned} \varphi_{\text{att},ij} &= \arccos \frac{\boldsymbol{v}_i \cdot (\boldsymbol{p}_j - \boldsymbol{p}_i)}{\|\boldsymbol{v}_i\| \cdot \|\boldsymbol{p}_j - \boldsymbol{p}_i\|}, \\ \varphi_{\text{esp},ij} &= \arccos \frac{\boldsymbol{v}_j \cdot (\boldsymbol{p}_j - \boldsymbol{p}_i)}{\|\boldsymbol{v}_j\| \cdot \|\boldsymbol{p}_j - \boldsymbol{p}_i\|}. \end{aligned} \tag{4}$$

Furthermore, in the designed short-range air combat environment, the damage inflicted on an enemy UAV by a single attack is variable and finite. It is assumed that each UAV has a blood value B, and each attack will reduce the specified blood value ΔB with a probability of p_{att}. A UAV will be destroyed when its blood value B is less than or equal to 0. The relationship of ΔB and p_{att} is defined as

$$\Delta B = \begin{cases} -B_1, & 0 \leqslant p_{\text{att}} < p_{\text{att1}}, \\ -B_2, & p_{\text{att1}} \leqslant p_{\text{att}} < p_{\text{att2}}, \\ -B_3, & p_{\text{att2}} \leqslant p_{\text{att}} < p_{\text{att3}}, \\ 0, & p_{\text{att}} \geqslant p_{\text{att3}}, \end{cases} \tag{5}$$

where p_{att1}, p_{att2} and p_{att3} are the thresholds of p_{att} and satisfy $0 < p_{\text{att1}} < p_{\text{att2}} < p_{\text{att3}} < 1$; B_1, B_2 and B_3 are the reduced blood values after an attack. If $B_i < 0$ or colliding with another UAV or the ground, the damaged flag of UAV i, dam_i, will be set as True.

However, the attack conditions are too harsh and difficult to meet in the early stages of air combat, which is not conducive to training. Therefore, an attack area located in front of its nose and an advantage area located behind the enemy's tail is proposed. The judgment conditions for those areas are represented as

$$\text{attack area} \quad \begin{cases} D_{\text{att,min}} \leqslant D_{ij} \leqslant D_{\text{att,max}}, \\ \varphi_{\text{att},ij} \leqslant \varphi_{\text{att,area}}, \end{cases} \tag{6}$$

$$\text{advantage area} \quad \begin{cases} D_{\text{adv,min}} \leqslant D_{ij} \leqslant D_{\text{adv,max}}, \\ \varphi_{\text{esp},ij} \leqslant \varphi_{\text{esp,area}}, \end{cases} \tag{7}$$

where $D_{\text{adv,max}}$ and $D_{\text{adv,min}}$ are the maximum and minimum lengths of the advantage area. $\varphi_{\text{att,area}}$ is the maximum attacking angle of the attack area, which satisfies $\varphi_{\text{att,area}} > \varphi_{\text{att,max}}$, and $\varphi_{\text{esp,area}}$ is the maximum escaping angle of the advantage area, which satisfies $\varphi_{\text{esp,area}} > \varphi_{\text{esp,max}}$.

In summary, the process of UAV swarm air combat in an episode is as follows. Firstly, the positions and velocities of the UAVs on both sides are initialized. Then, the UAVs engage in air combat based on their respective maneuver decision-making strategies, provided the maximum allowable air combat time $T_{\max}$ has not been reached. The UAV will execute an attack if the attack conditions are realized. It is worth noting that if the UAV's height is less than 0 or larger than the maximum allowable height, it will be directly deemed damaged. Finally, the episode ends when one side has destroyed all the opponent's UAVs or when $T_{\max}$ is reached. The side with more surviving UAVs wins, while the result is a tied if both have the same number of surviving UAVs.

2.3 Action space

The UAV maneuver decision-making controller generates $\boldsymbol{u}$ according to the current situation to control the UAV to participate in the swarm air combat. Tactical maneuvers in air combat, such as Immelmann

Table 1 Basic action library

No.	Action	Values for $[n_x, n_z, \gamma]^{\rm T}$	No.	Action	Values for $[n_x, n_z, \gamma]^{\rm T}$
1	Forward, maintain	0; 1; 0	2	Forward, accelerate	2; 1; 0
3	Forward, decelerate	-1; 1; 0	4	Upward, maintain	0; 3.5; 0
5	Upward, accelerate	2; 3.5; 0	6	Upward, decelerate	-1; 3.5; 0
7	Downward, maintain	0; -3.5; 0	8	Downward, accelerate	2; -3.5; 0
9	Downward, decelerate	-1; -3.5; 0	10	Left turn, maintain	0; 3.5; arccos(2/7)
11	Left turn, accelerate	2; 3.5; arccos(2/7)	12	Left turn, decelerate	-1; 3.5; arccos(2/7)
13	Right turn, maintain	0; 3.5; $-$arccos(2/7)	14	Right turn, accelerate	2; 3.5; $-$arccos(2/7)
15	Right turn, decelerate	-1; 3.5; $-$arccos(2/7)			

and Cobra maneuvers, are complex, ambiguous, and flexible, lacking specific control command values. This complexity presents significant challenges in modeling. To facilitate analysis and simulation testing, the complex maneuvers are often broken down into a serial of basic maneuvers. For example, NASA scholars have devised seven basic maneuvers [22, 25].

The series of basic maneuvers is usually referred to as the action library, or action space $\mathcal{A}$ in this paper. A more complex action space allows for greater flexibility in UAV maneuvers during air combat, but it also increases the complexity of air combat problem. Therefore, the action space utilized here, shown in Table 1, consists of fifteen basic maneuvers. These can be disassembled into five movements, namely forward, upward, downward, left turn and right turn, and three speed changes, namely maintain, accelerate and decelerate. Through executing maneuver sequences composed of these basic actions, various tactical maneuvering actions are fitted. This encourages the UAVs to explore various tactical maneuvers driven by the RL algorithm.

2.4 State space

For the UAV swarm air combat problem, each red UAV must consider its relationships with other UAVs on both the red and blue sides. To describe the positional relationship between UAV h and any other UAV k, the relative pitch angle $\theta_{D,hk}$ and relative yaw angle $\psi_{D,hk}$ are defined as

$$\begin{aligned} &\theta_{D,hk} = \arcsin \frac{z_k - z_h}{\|\boldsymbol{p}_k - \boldsymbol{p}_h\|}, \\ &\psi_{D,hk} = \arctan \frac{y_k - y_h}{x_k - x_h}, \end{aligned} \quad \forall h, k \in \Omega_r \cup \Omega_b, h \neq k. \tag{8}$$

The designed state space $\mathcal{S}$ for any red UAV i is divided into two parts, namely $\mathcal{S}_{\rm red}$ and $\mathcal{S}_{\rm blue}$, which are composed by

$$\begin{aligned} &\mathcal{S}_{\rm red} = \bigcup_{k\in\Omega_r, i\neq k} \{D_{ik}, \theta_{D,ik}, \psi_{D,ik}, \dot{x}_i - \dot{x}_k, \dot{y}_i - \dot{y}_k, \dot{z}_i - \dot{z}_k\}, \\ &\mathcal{S}_{\rm blue} = \bigcup_{k\in\Omega_b} \mathcal{S}_{{\rm blue},k}, \qquad \forall i \in \Omega_r. \\ &\mathcal{S}_{{\rm blue},k} = \{x_i - x_k, y_i - y_k, z_i, D_{ik}, \theta_{D,ik}, \psi_{D,ik}, \dot{x}_i - \dot{x}_k, \dot{y}_i - \dot{y}_k, \dot{z}_i - \dot{z}_k, \varphi_{{\rm att},ik}, \varphi_{{\rm esp},ik}\}, \end{aligned} \tag{9}$$

Furthermore, $\mathcal{S}$ is normalized in order to avoid the impact of these components' values. For each component $\lambda \in \mathcal{S}$, it is normalized by $a \cdot \frac{\lambda}{\lambda_0} - b$, where a and b are adjustment factors, and λ_0 is the reference value.

Obviously, $\mathcal{S}$ cannot completely and objectively display the global state of the UAV swarm air combat; for example, the action and damage flag of each UAV are not included. Therefore, the global state $\mathcal{S}_g$ is defined as

$$\begin{aligned} &\mathcal{S}_g = \{{\rm dam}_i | i \in \Omega_r\} \cup \mathcal{S}_{g,{\rm state}} \cup \{a_i | i \in \Omega_r\}, \\ &\mathcal{S}_{g,{\rm state}} = \bigcup_{i\in\Omega_r, j\in\Omega_b} \left\{D_{ij}, \theta_{D,ij}, \psi_{D,ij}, \dot{x}_i - \dot{x}_j, \dot{y}_i - \dot{y}_j, \dot{z}_i - \dot{z}_j, \arccos \frac{\boldsymbol{v}_i \cdot \boldsymbol{v}_j}{\|\boldsymbol{v}_i\| \cdot \|\boldsymbol{v}_j\|}, \varphi_{{\rm att},ij}, \varphi_{{\rm esp},ij}\right\}, \end{aligned} \tag{10}$$

where a_i is the index of the action of UAV i in $\mathcal{A}$, and it is not normalized. For the boolean variable dam, it is normalized as $\varepsilon \cdot (-1)^{\rm dam}$, where ε is a constant. The left components of $\mathcal{S}_g$ are normalized in the same way as those in $\mathcal{S}$.

Algorithm 1 Maneuver strategy script

```
Require: Decision period $f_m$, attack target index $i$ and its state $\boldsymbol{s}_i$, own state $\boldsymbol{s}$, the maximum allowable air combat time $T_{\max}$, time step $t_{\text{step}}$, the set of alive enemy UAVs $I_{\text{alive}}$, the set for alive enemy UAVs that are not being pursued $I_{\text{free}}$;
Ensure: Action $a$;
 1: Set decision counter $f = f_m$, time $t = 0$, $I_{\text{alive}} = I_{\text{free}} = \emptyset$, $a = 0$;
 2: for $t < T_{\max}$ do
 3:     if $f < f_m$ then
 4:         $f = f + 1$;
 5:     else
 6:         $f = 0$;
 7:         if $i$ is damage then
 8:             Choose the target: select the indexes of alive enemy UAVs and store them in $I_{\text{alive}}$. Select the indexes of enemy UAVs that are not being pursued and store them in $I_{\text{free}}$. If $I_{\text{alive}} = \emptyset$, set $i$ = None; If $I_{\text{free}} = \emptyset$, select index of the nearest UAV from $I_{\text{alive}}$ as $i$. If $I_{\text{free}} \neq \emptyset$, select index of the nearest UAV from $I_{\text{free}}$ as $i$;
 9:             Set $I_{\text{alive}} = I_{\text{free}} = \emptyset$;
10:         end if
11:         Predict target action: for each action in Table 1, predict $i$'s state $\boldsymbol{s}_{p,i}$ by $\boldsymbol{s}_i$ and the action. Calculate $i$'s threat using $\boldsymbol{s}$, $\boldsymbol{s}_{p,i}$ and (11), then select the action that poses the greatest threat as the predicted action $a_{p,i}$;
12:         Predict target state: predict $i$'s state $\boldsymbol{s}_{p,i}$ by $a_{p,i}$ and (1);
13:         Make decision: for each action in Table 1, predict its state $\boldsymbol{s}_p$ after performing the action. Calculate the threat using $\boldsymbol{s}_p$, $\boldsymbol{s}_{p,i}$ and (11), and then select the action that poses the least threat as the decision action $a$;
14:     end if
15:     Output the action $a$;
16:     Update $\boldsymbol{s}$ and $\boldsymbol{s}_i$;
17:     $t = t + t_{\text{step}}$;
18: end for
```

2.5 Maneuver strategy script

The smarter and more flexible the opponent's maneuver strategy is, the more challenging the training becomes, and, consequently, the more significant the training results. The maneuver strategy script employed, shown in Algorithm 1, is divided into three main steps: target selection, prediction, and decision-making. The thread value T is defined as

$$T = 0.41 \cdot T_{\varphi} + 0.26 \cdot T_v + 0.19 \cdot T_d + 0.14 \cdot T_h, \tag{11}$$

where T_{φ}, T_v, T_h, and T_d represent the angle thread value, speed thread value, height thread value and distance thread value, respectively, while the definitions of the four thread values can be found at [26].

3 Maneuver decision method by RL

In this section, the proposed network framework is described in detail for the MARL algorithm. The reward function and training process are also presented. It is worth noting that the red UAVs are the agents guided by the MARL algorithm.

3.1 Multi-agent proximal policy optimization algorithm

The multi-agent proximal policy optimization (MAPPO) algorithm, which extends the capabilities of the proximal policy optimization (PPO) algorithm into the realm of multi-agent systems, is employed to train the red UAVs. This algorithm plays a significant role in the swarm cooperation problem [27, 28]. The PPO algorithm enhances its stability and convergence speed through several strategies, notably including important sampling, generalized advantage estimation (GAE), and clipping [29]. These strategies are integrated into the MAPPO, ensuring reliable and efficient policy optimization in a multi-agent environment.

The adopted MAPPO is trained based on centralized training and decentralized execution (CTDE) learning mechanism. Every UAV on the red side is an RL agent for MAPPO. Each agent makes decisions based on its own observation by $\mathcal{S}$ and its actor network, while the critic network evaluates based on the global state by $\mathcal{S}_g$. The agents are isomorphic, meaning they have the same physical properties and play the same role in the swarm. Therefore, the sharing strategy is adopted, where the agents share the actor and critic networks, but the input values they provide to the networks are obtained according to their respective perspectives.

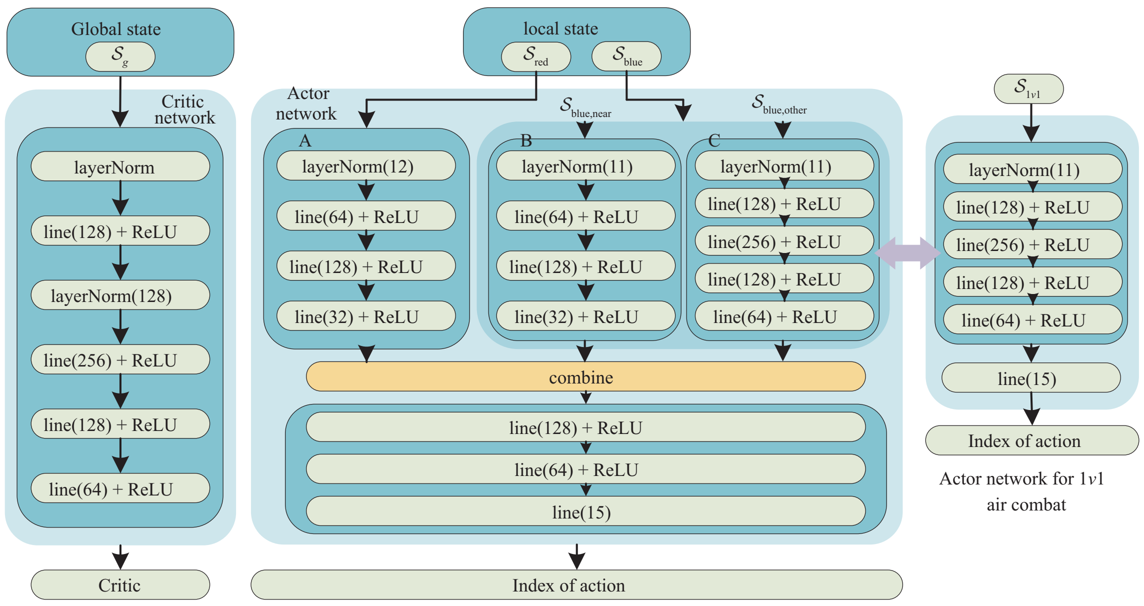

Figure 3 (Color online) Networks for the MARL algorithm.

3.2 Designed networks

UAV swarm air combat is not a simple summation of one-on-one air combat encounters, and it requires consideration of the effects of other UAVs on both friendly and enemy sides. However, insights from one-on-one air combat can, to a certain extent, guide the maneuver decisions of UAVs in swarm air combat. Thus, A framework for UAV swarm air combat based on transfer RL is devised, which transfers the results from one-on-one training to swarm air combat scenarios. This paper focuses on the three-on-two UAV swarm air combat.

The details of the proposed networks are shown in Figure 3. The actor network is the core component of the decision-making architecture and is devised to produce real-time decisions. The actor network, expressed as π_θ with parameter θ, comprises three dedicated subnetworks to accommodate and process the information from different UAVs. The input of π_θ is $\mathcal{S}$ which consists of $\mathcal{S}_{\text{red}}$ and $\mathcal{S}_{\text{blue}}$. The subnetwork A handles the information related to friendly UAVs, denoted as $\mathcal{S}_{\text{red}}$. At the same time, $\mathcal{S}_{\text{blue}}$ is divided into two parts: the state space with nearer blue UAV $\mathcal{S}_{\text{blue,near}}$ and the state space with other blue UAVs $\mathcal{S}_{\text{blue,other}}$. Note that $\mathcal{S}_{\text{blue,near}}$ and $\mathcal{S}_{\text{blue,other}}$ have the same composition as $\mathcal{S}_{\text{blue},k}$. It is obvious that more attention should be paid to the nearer UAV, and the subnetwork B should be trained carefully. Then, the subnetwork C is designed to deal with other blue UAV, which are not as important as the nearer one. To simplify π_θ and accelerate training, the parameters of subnetwork C are loaded from a trained actor network used in one-on-one air combat. Note that $\mathcal{S}_{1v1}$ has the same composition as $\mathcal{S}_{\text{blue},k}$. Finally, the output of π_θ is the action's index a in Table 1. As for the critic network V_ϕ with parameter ϕ, it is utilized to evaluate the current situation based on $\mathcal{S}_g$. The critic network is trained to evaluate the global state, and its value function is used to calculate the advantage function and the loss function during the training process.

3.3 Reward function design

When training the neural networks, the RL algorithm updates the networks based on the rewards obtained [30]. Therefore, to guide the RL algorithm in training the networks in the desired direction, a suitable reward function is necessary. In this paper, the reward function R_i for $i \in \Omega_r$ consists of three components as follows.

3.3.1 *Situation reward*

In each decision step, after all the UAVs have performed their actions, UAV i will receive its own situation reward $r_{s,i}$ from the air combat environment. $r_{s,i}$ provides real-time feedback to UAV i on the value of the performed action in the air combat process, thus improving the UAV's search efficiency and accelerating the training process.

For UAV j, the devised situation reward $r_{s,ij}$ consists of the angle reward r_φ, distance reward r_d, speed reward r_v and height reward r_h, and $r_{s,ij}$ is defined as

$$r_{s,ij} = w_\varphi \cdot r_\varphi + w_d \cdot r_d + w_v \cdot r_v + w_h \cdot r_h, \tag{12}$$

where w_φ, w_d, w_v and w_h are the weights falling in $[0,1]$, and their summation is 1. Notice that some subscripts ij in (12) have been omitted for the sake of brevity of expression and this will also be denoted as such later on. r_φ represents the azimuth relationship between the two UAVs, which can be specified as

$$r_\varphi = \frac{\pi - \varphi_{\text{att},ij}}{\pi} \cdot \frac{\pi - \varphi_{\text{esp},ij}}{\pi}. \tag{13}$$

The distance reward r_d indicates the environment's evaluation of current distance, which can be divided into two parts, r_{d1} and r_{d2}, and specified as

$$r_d = r_{d1} + r_{d2}, \tag{14}$$

$$r_{d1} = \begin{cases} 0.25, & \Delta D_{ij} < 0 \text{ and } D_{ij} > D_{\text{mid}}, \\ 0, & \text{other}, \end{cases} \tag{15}$$

$$r_{d2} = \begin{cases} 0.25 \cdot \left(a_1 \left(D - D_{\text{adv,max}}\right)^2 + 1\right), & D_{\text{adv,max}} < D_{ij} \leqslant D_s, \\ 0.25 + 0.25 \cdot \left(a_2 \left(D - D_{\text{att,max}}\right)^2 + 1\right), & D_{\text{att,max}} < D_{ij} \leqslant D_{\text{adv,max}}, \\ 0.50 + 0.25 \cdot a_3 \left(D - D_{\text{att,min}}\right)\left(D - D_{\text{att,max}}\right), & D_{\text{att,min}} < D_{ij} \leqslant D_{\text{att,max}}, \\ 0, & \text{other}, \end{cases} \tag{16}$$

where $D_{\text{mid}} = \left(D_{\text{att,min}} + D_{\text{att,max}}\right)/2$, $a_1 = -\left(D_s - D_{\text{adv,max}}\right)^{-2}$, $a_2 = -\left(D_s - D_{\text{adv,max}}\right)^{-2}$ and $a_3 = \left(D_{\text{mid}} - D_{\text{att,min}}\right)^{-1}\left(D_{\text{mid}} - D_{\text{att,max}}\right)^{-1}$ are the coefficients. D_s is the desired maximum distance. ΔD_{ij} denotes the distance difference from the previous moment. It is evident that r_{d1} guides UAV i to approach UAV j, while r_{d2} adopts a segmented function form to encourage UAV i to keep its distance from UAV j in the attack range.

The height reward r_h is defined as

$$r_h = \begin{cases} 0.1, & H_{\max} < z_i - z_j \leqslant D_{\text{att,max}}, \\ h_1\left(z_i - z_j - H_{\text{adv}}\right)^2 + 1, & H_{\text{adv}} < z_i - z_j \leqslant H_{\max}, \\ 1, & H_{\text{att}} < z_i - z_j \leqslant H_{\text{adv}}, \\ h_2\left(z_i - z_j - H_{\text{att}}\right)^2 + 1, & H_{\min} < z_i - z_j \leqslant H_{\text{att}}, \\ 0, & \text{other}, \end{cases} \tag{17}$$

where $H_{\max}$, H_{adv}, H_{att} and $H_{\min}$ are four height thresholds during the combat. The coefficients are defined as $h_1 = -0.9\left(H_{\max} - H_{\text{adv}}\right)^{-2}$ and $h_2 = -\left(H_{\min} - H_{\text{att}}\right)^{-2}$. r_h leads the UAV i to occupy a favorable height advantage against the enemy.

The speed reward r_v is used to evaluate the speed relationship between UAVs i and j. Upon defining $k_v = v_i/v_j$, r_v can be expressed as

$$r_v = \begin{cases} 0.1, & k_v > 1.5, \\ 1, & 1.0 \leqslant k_v \leqslant 1.5, \\ 5k_v - 4, & 0.8 \leqslant k_v < 1.0, \\ 0, & \text{other}. \end{cases} \tag{18}$$

It is clear that r_φ, r_d, r_v, r_h and $r_{s,ij}$ fall within the range of $[0,1]$. Finally, the situational reward $r_{s,i}$ achieved by UAV i at decision step is defined as

$$r_{s,i} = \max_{j \in \Omega'_b} r_{s,ij} \quad i \in \Omega'_r. \tag{19}$$

Table 2 Involved event rewards in the view of red UAV i

No.	Name	Description	Condition	Reward value	Score
1	Occupy advantage area	UAV i is in advantage area behind blue UAV j	See (7)	r_{adv} by (20)	1
2	Be occupied advantage area	Blue UAV j is in advantage area behind UAV i	Similar to (7)	-1	0
3	Strike the ground	UAV i strikes the ground and is damaged	$z_i < 0$	-0.5	0
4	Exceed the ceiling	UAV i exceeds the maximum allowable height	$z_i > z_{\max}$	-0.5	0
5	Collide with others	UAV i too close to one of other UAVs	$D_{ik} < D_{\min}$ a)	-0.5	0
6	Occupy attack area	UAV i makes blue UAV j in its attack area	See (6)	0.3	0
7	Be occupied attack area	Blue UAV j makes UAV i in its attack area	Similar to (6)	-0.3	0
8	Hit the opponent	UAV i hits blue UAV j successfully	See (3)	0.8	2
9	Destroy the opponent	UAV i destroys blue UAV j	$B_j < 0$ after i's attack	1.5	5
10	Be attacked	UAV i is hit by blue UAV j	Similar to (3)	-0.9	0
11	Be destroyed	UAV i is destroyed by blue UAV j	$B_i < 0$ after j's attack	-1.6	0

a) The k satisfies $\forall k \in \Omega_r' \cup \Omega_b'$ and $k \neq i$, and $D_{\min}$ denotes the minimum safe distance.

3.3.2 *Event reward*

During close air combat, UAV i needs to trigger a series of characteristic events to beat UAV j, such as successfully attacking j, getting j into the advantage area, and destroying j [18]. The involved events, their corresponding trigger conditions and reward values are shown in Table 2. When UAV i occupies the advantage area relative to UAV j, it will receive the advantage area reward r_{adv}, which is defined as

$$r_{\text{adv}} = 0.6 \cdot \frac{D_{\text{adv,max}} - D}{D_{\text{adv,max}} - D_{\text{adv,min}}} + 0.4 \cdot \frac{\pi - \varphi_{\text{esp},ij}}{\pi}. \tag{20}$$

At each decision step, the event reward $r_{e,i}$ for UAV i is calculated using the following steps. First, $r_{e,i}$ is reset to zero. Then, based on the relative position and velocity relationships between the UAVs, when an event is triggered, the corresponding reward value will be accumulated on $r_{e,i}$. Finally, after traversing the event types shown in Table 2, $r_{e,i}$ obtained by UAV i at this decision step is obtained.

3.3.3 *Reward assignment for dense reward*

Credit assignment, an important issue in the field of MARL, refers to how to allocate contributions towards results among all the agents. Similarly, in multi-UAV air combat, evaluating the impact of different red UAVs' maneuver decisions on the outcome of the air combat is of great significance to encourage red UAVs to actively cooperate and jointly attack the blue side's UAVs. In this paper, a reward assignment method is adopted to solve the credit assignment problem. $r_{s,i}$ and $r_{e,i}$ are calculated at each decision step. Therefore, the dense reward $r_{\text{den},i}$ is defined as $r_{\text{den},i} = r_{s,i} + r_{e,i}$. The reward shaping is applied to $r_{\text{den},i}$. The process of reward assignment is outlined in Algorithm 2. Note that r_{den0} is the basic dense reward value for each red UAV.

The $r_{\text{den},i}$ is utilized to show the contribution of UAV i. If $r_{\text{den},i} > 0$, indicating a positive contribution, $r'_{\text{den},i}$ is assigned based on the number of UAVs that make positive contributions $|I_u|$ and the summation of positive contributions α. Therefore, if UAV i makes more positive contributions and α is larger, it will receive a larger $r'_{\text{den},i}$. Conversely, if UAV i makes a negative contribution or even causes damage, it will be punished accordingly, meaning $r'_{\text{den},i} < 0$. In this way, the red UAVs are encouraged to survive and collectively beat the blue UAVs.

3.3.4 *End-game reward*

The last type of reward is the end-game reward, denoted as $r_{\text{end},i}$, which is allocated to UAV i of the red side at the end of an episode based on its individual contributions towards the results of the confrontation. In other words, through the distribution of the end-game reward, red UAVs are encouraged to improve their decision-making capabilities in a targeted manner and actively participate in the air combat process. The winning condition during the training phase is to destroy all the opponent's UAVs. It is also recommended to complete the combat as soon as possible with less blood loss.

(1) The end-game reward for wining. First, the whole wining reward $r_{\text{win,all}}$ for the red side is obtained by

$$r_{\text{win,all}} = r_{\text{win0}} \cdot n_r \cdot \left(0.75 + 0.25 \cdot \frac{N_{\text{step}} - n_{\text{step}}}{N_{\text{step}}}\right), \tag{21}$$

Algorithm 2 Reward assignment on dense reward

```
Require: Dense reward set $\Phi_{r,\mathrm{den}} = \{r_{\mathrm{den},i} | i \in \Omega'_r\}$, the number of red side's UAVs $n_r$, damage flag $\mathrm{dam}_i$ $i \in \Omega'_r$;
Ensure: Assigned dense reward set $\Phi'_{r,\mathrm{den}} = \{r'_{\mathrm{den},i} | i \in \Omega'_r\}$;
 1: Set alive red UAV set $I_u = \emptyset$;
 2: for $i \in \Omega'_r$ do
 3:     if $\mathrm{dam}_i$ is True then
 4:         $r'_{\mathrm{den},i} = -r_{\mathrm{den0}} \cdot n_r - \min \Phi_{r,\mathrm{den}}$;
 5:     else
 6:         if $r_{\mathrm{den},i} > 0.01$ then
 7:             Add $i$ into $I_u$;
 8:         else
 9:             if $r_{\mathrm{den},i} > -0.01$ then
10:                 $r'_{\mathrm{den},i} = 0$;
11:             end if
12:         end if
13:     end if
14: end for
15: if $I_u \neq \emptyset$ then
16:     Set $\alpha = \sum_{i \in I_u} r_{\mathrm{den},i}$;
17:     for $i_u \in I_u$ do
18:         $r'_{\mathrm{den},i_u} = (r_{\mathrm{den0}} \cdot n_r + 0.003 \cdot |I_u|/n_r + 0.007 \cdot \alpha/n_r) \cdot r_{\mathrm{den},i_u}/\alpha$;
19:     end for
20: end if
```

where r_{win0} is the basic wining reward for every red UAV, N_{step} is the maximum number of decision steps, and n_{step} denotes the number of decision steps at the end.

Then, the contributions are considered. During an air combat episode, the red UAVs trigger various events. More contributions at the end of the confrontation come from UAVs that have triggered events highly conducive to winning, such as destroying an opponent. Therefore, for every episode, the score of each red UAV is accumulated according to Tabel 2. The score of UAV i at the end of the episode is denoted by β_i. A higher β_i deserves to be allocated more winning rewards. Note that $\beta = \sum_{i\in\Omega_r} \beta_i$, and if $\beta = 0$, β will be set as 1 to make sense of (22).

Finally, the whole wining reward is given to UAV i as $r_{\mathrm{end},i}$, which is calculated by

$$r_{\mathrm{end},i} = r_{\mathrm{win,all}} \cdot \left(\frac{w_{\mathrm{win1}}}{n_r} + 0.03 \cdot |\Omega'_r| + w_{\mathrm{win2}} \cdot \frac{\beta_i}{\beta} + w_{\mathrm{win3}} \cdot \frac{B_{r,i}}{B_{r,\mathrm{sum}}} \cdot \frac{B_{r,i}}{B_0} \right), \tag{22}$$

where w_{win1}, w_{win2} and w_{win3} denote weights, where summation is 1, $B_{r,i}$ is the remaining blood value of UAV i, $B_{r,\mathrm{sum}} = \sum_{i\in\Omega'_r} B_{r,i}$, and B_0 is the initial blood value.

(2) The end-game reward for losing. Similarly, when losing, each red UAV receives a negative end-game reward. The whole losing reward $r_{\mathrm{lose,all}}$ is obtained by

$$r_{\mathrm{lose,all}} = r_{\mathrm{lose0}} \cdot n_r \cdot \left(0.80 + 0.20 \cdot \frac{N_{\mathrm{step}} - n_{\mathrm{step}}}{N_{\mathrm{step}}} \right), \tag{23}$$

where $r_{\mathrm{lose0}} < 0$ is the basic losing reward. Then, $B_{r,i}$ and β_i are reshaped by

$$\begin{aligned} B'_{r,i} &= B_0 - B_{r,i} + 10, \\ \beta'_i &= \max_{i\in\Omega_r} \beta_i - \beta_i + 1. \end{aligned} \tag{24}$$

Finally, the adopted $r_{\mathrm{end},i}$ for UAV i is presented as

$$r_{\mathrm{end},i} = r_{\mathrm{lose,all}} \cdot \left(\frac{w_{\mathrm{lose1}}}{n_r} - 0.02 \cdot |\Omega'_r| + w_{\mathrm{lose2}} \cdot \frac{\beta'_i}{\max_{i\in\Omega_r} \beta'_i} + w_{\mathrm{lose3}} \cdot \frac{B'_{r,i}}{B_0} \right), \tag{25}$$

where w_{lose1}, w_{lose2} and w_{win3} are weights which summation is 1.

3.4 Training process

In this paper, the MAPPO algorithm and transfer method are adopted to train the actor network to obtain the red side's strategy. The training process can be simply divided into two stages: experience collection over a number of decision steps and sampling to update the network, as shown in Figure 4.

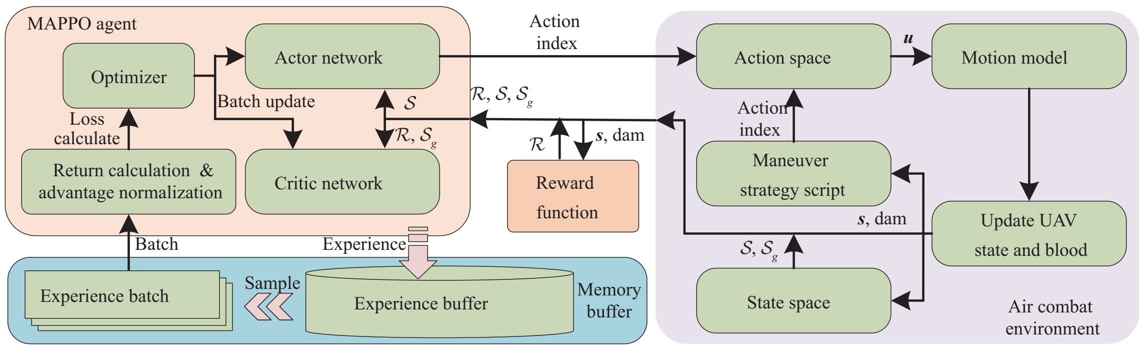

Figure 4 (Color online) Training process based on MAPPO in the UAV swarm air combat environment.

Table 3 Designed tips for training

Symbol	Description
C1	Random initialization under the same initial conditions: $x \in [1000, 3000]$, $y \in [1500, 3500]$, $z \in [1400, 3400]$, $v \in [60, 180]$, $\psi \in [0, 2\pi]$, $\theta = 0$
C2	Random initialization in the initial point set
C3	The blue UAV only performs the action with index 1
C4	The blue UAV uses the maneuver strategy script to make decision
C5	Transfer network parameters from the result of one-on-one air combat
C6	Apply the reward assignment

During the collection stage, the MAPPO agent is shared by red UAVs, while the maneuver strategy script is used to dictate the behaviors of the blue UAVs. Next, the actions of UAVs generalized by the MAPPO agent and script are executed, and the UAVs' states and blood values are updated according to the air combat scenario at each decision step. Then, $\mathcal{S}$, $\mathcal{S}_g$, actions, and the rewards given by the reward function are recorded and stored as experience in the experience buffer [28]. Once the experience buffer is full, the training process begins. Multiple experience batches, each with batch size B, are sampled from the buffer, and are then utilized to compute values, such as GAE advantage A and discounted return $\hat{R}$. Subsequently, the network parameter θ is updated by maximizing the objective [28]

$$L(\theta) = \frac{1}{Bn_r}\sum_{i=1}^{B}\sum_{k=1}^{n_r}\min\left(r_{i,k}, \mathrm{clip}(r_{i,k}, 1-\epsilon, 1+\epsilon)\right)A_{i,k} + \sigma\frac{1}{Bn_r}\sum_{i=1}^{B}\sum_{k=1}^{n_r} S\left[\pi_\theta(\mathcal{S}_{i,k})\right], \tag{26}$$

where $r_{i,k} = \frac{\pi_\theta(a_{i,k}|\mathcal{S}_{i,k})}{\pi_{\theta_{\mathrm{old}}}(a_{i,k}|\mathcal{S}_{i,k})}$ is the important sampling ratio for batch i and red UAV k. clip() is the clipping function, ϵ denotes the clipping parameter, S represents the policy entropy, and σ indicates the entropy coefficient hyperparameter. The network parameter ϕ is updated by minimizing the objective [28]

$$L(\phi) = \frac{1}{Bn_r}\sum_{i=1}^{B}\sum_{k=1}^{n_r}\max\begin{bmatrix}\left(V_\phi(\mathcal{S}_{g,i,k}) - \hat{R}_i\right)^2, \\ \left(\mathrm{clip}(V_\phi(\mathcal{S}_{g,i,k}), V_{\phi_{\mathrm{old}}}(\mathcal{S}_{g,i,k}) - \epsilon, V_{\phi_{\mathrm{old}}}(\mathcal{S}_{g,i,k}) + \epsilon) - \hat{R}_i\right)^2\end{bmatrix}. \tag{27}$$

The sheer complexity and stochastic nature of interactions among numerous autonomous agents in such swarm air combat environments amplify the computational demands, making direct training a potentially arduous and resource-intensive endeavor. Therefore, some training tips are proposed in this paper, as shown in Table 3, where C1 and C2 are the initialization schemes, C3 and C4 are the opponent's action selection schemes, and C5 and C6 are the strategies proposed in this paper. By combining these tips, one can generate training scenarios with various levels of complexity. The complex scenarios can be progressively derived by gradually relaxing conditions from simpler ones. The network parameters obtained from training in less complicated environments serve as initial parameters for subsequent training in more convoluted scenarios, thereby expediting the overall training process. For example, further training under C1 after training under C can lead to a faster training pace compared to starting the training directly under C1.

4 Experiment

In this section, we analyze the training results and conduct simulation tests of the proposed decision-making method within the designed air combat environment. Furthermore, ablation experiments are used to illustrate the contributions of the method's individual compositions.

4.1 Parameters setting

The combat environment's parameters are set as follows [31]. In (2), the parameters for constraints are set as $v_{\min} = 30$ m/s, $v_{\max} = 180$ m/s, $\theta_{\min} = -\pi/4$, $\theta_{\max} = \pi/4$, $n_{x\min} = -1$, $n_{x\max} = 2.5$, $n_{z\min} = -4$ and $n_{z\max} = 4$. The normalized reference value λ_0 is set as 5000 m for length-type components, π for angle-type components, and $v_{\max} - v_{\min}$ for velocity-type components. The parameters for attackable distance are set as $D_{\mathrm{att,min}} = 40$ m and $D_{\mathrm{att,max}} = 900$ m. The maximum attacking angle is $\varphi_{\mathrm{att,max}} = \pi/6$ and the maximum escaping angle is $\varphi_{\mathrm{esp,max}} = \pi/3$. For the blood parameters, they are set as $p_{\mathrm{att1}} = 0.1$, $p_{\mathrm{att2}} = 0.4$ and $p_{\mathrm{att}} = 0.8$. The blood thresholds are set as $B_0 = 300$, $B_1 = 51$, $B_2 = 21$ and $B_3 = 11$. For the advantage area, the settings are $D_{\mathrm{adv,min}} = 40$ m, $D_{\mathrm{adv,max}} = 1300$ m and $\varphi_{\mathrm{esp,area}} = \pi/3$. Similarly, $\varphi_{\mathrm{att,area}} = \pi/4$. The desired maximum distance D_s is set as 5000 m. The parameters related to time are set as $T_{\max} = 200$ s, $t_{\mathrm{step}} = 0.1$ s. The decision step is set as 0.5 s. For the reward function, the threshold values are set as $H_{\max} = 500$ m, $H_{\mathrm{adv}} = 300$ m, $H_{\mathrm{att}} = 100$ m and $H_{\min} = -300$ m. The weights in (12) are set as $w_\varphi = 0.15$, $w_d = 0.6$, $w_v = 0.1$ and $w_h = 0.15$. The basic rewards are set as $r_{\mathrm{den0}} = 0.01$, $r_{\mathrm{win0}} = 50$, and $r_{\mathrm{lose0}} = -50$. The main hyperparameters in MAPPO are set as learning rate 0.0003, GAE parameter 0.95, discount 0.99, number of batches 8, buffer size $B = 8192$, epoch 5, clip parameter $\epsilon = 0.15$ and the total number of training episode $N_{\mathrm{eps}} = 30000$.

When C4 is adopted, the training difficulty will rise sharply, making it mostly impossible to produce satisfactory results within $N_{\mathrm{eps}} = 30000$. Therefore, an incremental strategy is designed for the initial blood value B_{blue0} of the blue side while training. Specifically, when $N_{\mathrm{eps}} < 8000$, $B_{\mathrm{blue0}} = 50$; when $8000 \leqslant N_{\mathrm{eps}} < 12000$, $B_{\mathrm{blue0}} = 80$; when $12000 \leqslant N_{\mathrm{eps}} < 20000$, $B_{\mathrm{blue0}} = 100$; and when $N_{\mathrm{eps}} \geqslant 20000$, $B_{\mathrm{blue0}} = 300$.

4.2 Training results

4.2.1 *Training results of one-on-one air combat*

To facilitate the transfer of the network, training for one-on-one air combat is conducted first, involving only one UAV for each side, red and blue. The architectures of actor and critic networks are shown in Figure 3. The training process is structured into three stages: stage 1 (C2+C3+C6), stage 2 (C2+C4+C6), and stage 3 (C1+C4+C6), with every stage building upon the training results of the previous stage.

The training curves of the total rewards per episode are shown in Figure 5. It can be seen from the training progression that the agent's maneuver policy quickly converges and maintains stability in the simplest stage 1. Then, the agent acquires an effective winning policy during training in the slightly more complex stage 2 and gradually improves the learned policy in the most complex stage 3 while maintaining basic stability. Ultimately, through phased training, the agent acquires a winning policy against opponents initialized under identical conditions, consistently securing higher rewards and thereby maintaining a high win rate. This observation indicates that the proposed decision-making method for one-on-one air combat is rational and adaptive. Then, the training results are loaded for testing and one of the test examples is shown in Figure 6. It is evident that the blue UAV enjoys a significant altitude advantage at the outset, posing a threat to the red UAV. Both UAVs approach each other, endeavoring to create favorable attacking conditions to beat the other. The red UAV, however, utilizes a smaller turning radius and strategic climbs to reduce the horizontal distance to the blue UAV while maintaining similar attitudes, thereby avoiding entering the enemy's attacking area and mitigating the risk of being targeted. Subsequently, through more agile maneuvers, the red UAV manages to position itself behind the blue UAV and establishes a tactical advantage. The blue UAV attempts to quickly break away from the attack lock by lowering its altitude, while the red UAV takes corresponding actions to maintain its attack advantage. Finally, capitalizing on this advantageous positioning, the red UAV executes a chase and ultimately destroys its opponent. Therefore, in simple air combat scenarios, the proposed method can achieve the requirements effectively.

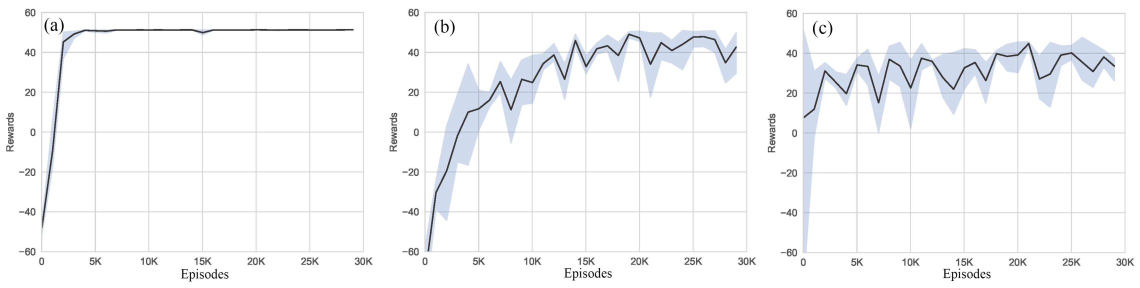

Figure 5 (Color online) Episode rewards for one-on-one air combat. (a) Stage 1: C2+C3+C6; (b) stage 2: C2+C4+C6; (c) stage 3: C1+C4+C6.

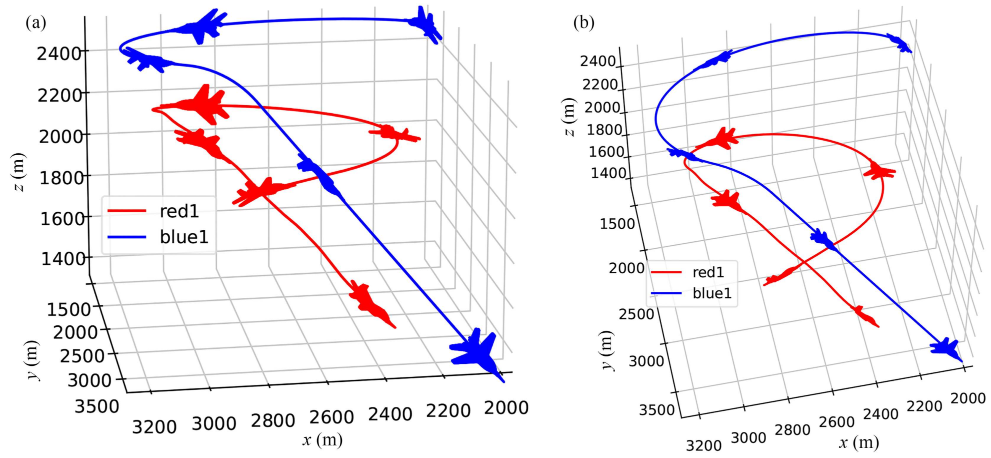

Figure 6 (Color online) Test result for one-on-one air combat. (a) Side view; (b) top view.

4.2.2 *Training results of swarm air combat*

Part of the actor network trained in one-on-one scenarios is transferred to help train the networks in swarm air combat according to Figure 3. During the training, the parameters of subnetwork C are frozen, meaning these parameters remain unchanged throughout the network updates. The training process is structured into three stages: stage A (C2+C3+C5+C6), stage B (C2+C4+C5+C6), and stage C (C1+C4+C5+C6). The training curves of these stages are shown in Figure 7. According to the training curves, it is evident that after the red UAVs complete their learning in the simple scenario, they can quickly identify the effective policy in the transitional scenario, showing steady improvement in performance. Subsequently, they can effectively maintain effective maneuvering policy in more complex scenario, despite experiencing a marginal dip in total rewards. The training results are loaded for testing and the confrontation outcomes are shown in Figure 8. Figure 8(a) illustrates a cooperative pursuit policy leading to the sequential destruction of blue UAVs; Figure 8(b) demonstrates how rapid turning maneuvers are employed to reverse disadvantageous situations and continue the pursuit; Figure 8(c) portrays the transition wherein, upon completion of an attack mission on a current target, the agent immediately engages in further attacks against the remaining blue UAVs. By analyzing the confrontation flight trajectories, it is clear that the red UAV swarm adeptly handles the blue UAV swarm's versatile offense, with all members actively participating in the confrontation. Notably, the red UAVs have also learned to secure victory by tactically forcing the blue UAVs to crash into the ground. This demonstrates that the proposed tips effectively prevent the emergence of “lazy agents” within the swarm.

4.3 Ablation experiments

4.3.1 *Training results for ablation experiments*

To validate the effectiveness of the design tips within the proposed maneuver decision-making method, ablation experiments are conducted to assess the effect of each tip on the training process. According to Table 3, three different combinations of designed tips are used in the experiments. Case I (C5+C6): this combination adopts the main designed tips for swarm air combat, and its training results are displayed in

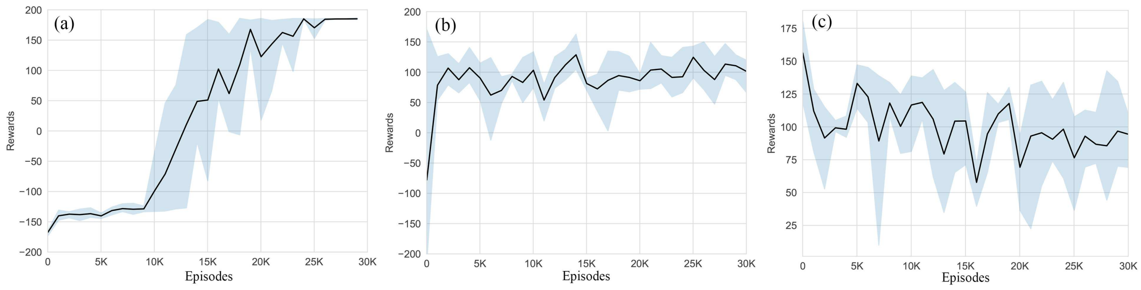

Figure 7 (Color online) Episode rewards for three-on-two air combat. (a) Stage A: C2+C3+C5+C6; (b) stage B: C2+C4+C5+C6; (c) stage C: C1+C4+C5+C6.

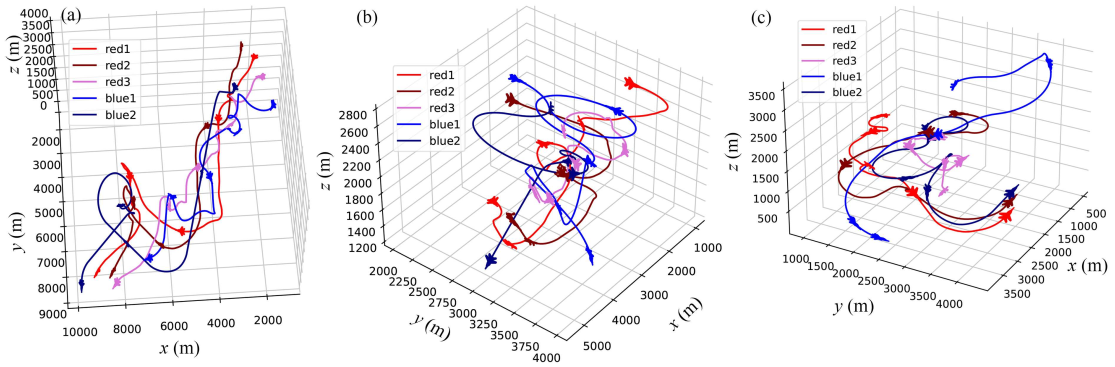

Figure 8 (Color online) Test results for three-on-two air combat. (a) Cooperative pursuit; (b) rapid turning and pursuit; (c) attacking remaining opponent.

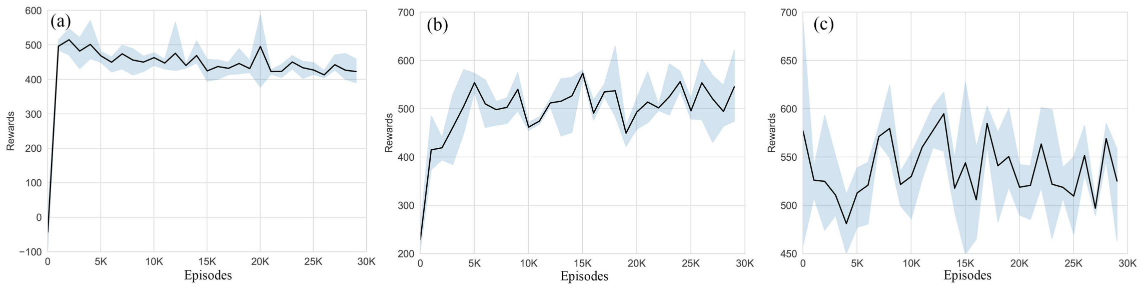

Figure 9 (Color online) Episode rewards for three-on-two air combat in case II. (a) C2+C3+C5; (b) C2+C4+C5; (c) C1+C4+C5.

Figure 7. Case II (C5 only): In this scenario, only C5 is used, meaning no reward assignment operation is performed. Case III (C6 only): Here, C5 is used, and subnetwork C shown in Figure 3 is not transferred but instead participates in network training and updating, with parameters changing iteratively. The training results of Cases II and III are shown in Figures 9 and 10, respectively. Since the reward assignment changes the reward values at each decision step, it is less meaningful to compare the reward values. However, it can still be discerned that the total rewards can be more stable when both transferring and reward assessment are employed, especially in complex scenarios.

Furthermore, the maneuver decision-making strategies trained for the three cases are implemented in three-on-two air combat confrontation, with multiple episodes of air combat conducted separately. The recorded winning rates are shown in Table 4, and the five types of results are defined as (28). Comparing Figures 7 and 10, it is evident that although Case I initially experiences an inhibitory effect when employing C2 resulting by transferring results from the one-on-one scenario, its rewards gradually increase, ultimately leading to the acquisition of a more robust policy. Additionally, when employing C1, the mean rewards of Case I show a trajectory of stability and manifest a clear convergence trend compared to those of Case III. Thus, despite the initial setbacks caused by transferring, which may temporarily hinder progress owing to other untrained parts of the actor network, it is eventually demonstrated that the transfer process proves beneficial. This is because the transfer simplifies the network layers to be trained, thereby yielding better overall training results. Comparing Figures 7 and 9, it becomes apparent

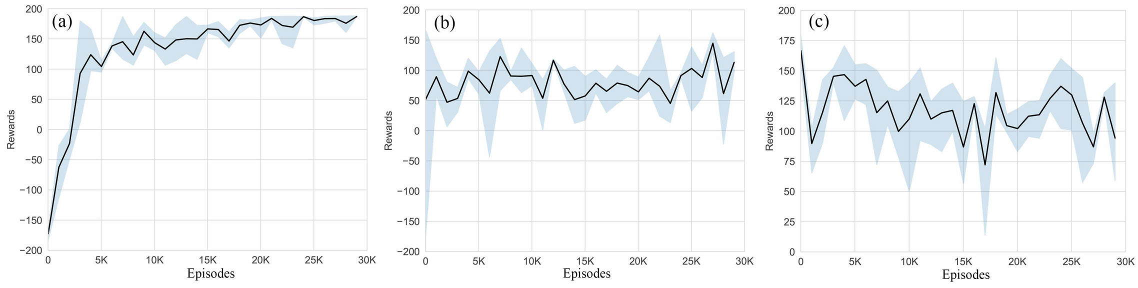

Figure 10 (Color online) Episode rewards for three-on-two air combat in case III. (a) C2+C3+C6; (b) C2+C4+C6; (c) C1+C4+C6.

Table 4 Results of confrontation with maneuver strategy script in three cases

Confrontation type	Winning rate (%)	Losing rate (%)	Drawing rate (%)	Loose winning rate (%)	Loose losing rate (%)
Case I	87.00	6.00	7.00	89.00	9.00
Case II	81.00	3.00	3.00	79.00	18.00
Case III	79.00	4.00	5.00	79.00	18.00

that the reward assignment tip leads to a slow increase and ultimate convergence of the reward curve for Case I in less intricate conditions. Conversely, for Case II, without this tip, the rewards initially reach a high level but subsequently witness a downward trend over the training period, failing to achieve convergence. Additionally, when employing C1, Case I demonstrates notably less fluctuation and a more poised progression relative to Case II. This demonstrates that the reward assignment tip can help to accelerate the learning process.

From Table 4, it is obvious that Case I achieves the largest winning rate and loose winning rate. This indicates that after the same training process, the proposed method with transferring and reward assignment can develop a more intelligent and effective maneuver decision-making policy. Thus, the designed tips are effective in promoting the active participation of each agent in air combat, thereby enhancing the overall winning rate of the swarm.

$$\text{Confrontation result:}\begin{cases}\text{winning}, & |\Omega_r'|>0 \text{ and } |\Omega_b'|=0,\\ \text{losing}, & |\Omega_r'|=0 \text{ and } |\Omega_r'|>0,\\ \text{drawing}, & |\Omega_r'|>0 \text{ and } |\Omega_b'|>0,\\ \text{loose winning}, & |\Omega_r'|-|\Omega_b'|\geqslant 2 \text{ or } (|\Omega_r'|>0 \text{ and } |\Omega_b'|=0),\\ \text{loose losing}, & |\Omega_b'|-|\Omega_r'|>0 \text{ or } (|\Omega_r'|=0 \text{ and } |\Omega_b'|>0).\end{cases} \tag{28}$$

4.3.2 *Confrontation test*

To further illustrate the effectiveness of the proposed maneuver decision-making method, the training results of Case I are tested against the results of Case II and Case III in a three-on-three UAV swarm air combat environment. The original Cases were conducted in a three-on-two scenario; therefore, some restrictions are imposed: when the number of surviving UAVs on the opponent's side is greater than 2, the two closest UAVs will be selected as observation objects for maneuver decision-making. The recorded results for 50 episodes are shown in Table 5. Note that the initial blood value of UAV is set to 100 to shorten the time of each episode and the loose wining's condition is defined as $|\Omega_r'|-|\Omega_b'|\geqslant 0$ or $(|\Omega_r'|>0$ and $|\Omega_b'|=0)$.

The results indicate that in Case I, the maneuver decision-making ability is stronger, even when trained under the same conditions. Through the network transferring tip, the learned policy becomes better equipped to handle enemy attacks, adopting a more conservative approach that can potentially reduce both winning and losing rates. Conversely, relying solely on reward assignment tip leads to a relatively aggressive learning strategy with a high winning rate but also a higher losing rate. Therefore, by using the designed tips in conjunction, a more balanced offensive and defensive strategy can be achieved.

Table 5 Results of confrontation between three cases

Confrontation type	Winning rate (%)	Losing rate (%)	Drawing rate (%)	Loose winning rate (%)	Loose losing rate (%)
Case I vs. Case II	14.00	7.00	29.00	22.00	19.00
Case I vs. Case III	25.00	19.00	6.00	25.00	19.00
Case II vs. Case III	17.00	24.00	9.00	19.00	24.00

5 Conclusion

In this paper, a maneuver decision-making method for UAV swarm short-range air combat based on the MARL algorithm is proposed. Key tips, the neural network transferring and the reward assignment, are adopted to accelerate the training process for swarm air combat maneuver decision-making using the MAPPO algorithm. Specifically, separate state spaces for local and global observation are designed for actor and critic networks. The actor network strategically incorporates the pre-trained network from a one-on-one air combat scenario. Additionally, the proposed reward function for MAPPO is divided into three parts, and a reward assessment tip is implemented to prevent individual UAVs in the swarm from becoming "lazy" and contributing minimally to the swarm air combat effort while still reaping similar or even greater rewards. Ultimately, the proposed method is validated through simulations and ablation experiments. The results indicate the reward assessment tip effectively guides the individuals to actively participate in air combat, ensuring their contribution. Concurrently, the network transferring operation leverages knowledge acquired in simpler scenarios to accelerate the training efficiency in more complex ones.

Our future work will focus on two key aspects. First, it will explore more realistic scenario designs, incorporating factors like wind disturbances and model uncertainties. Second, it will explore larger-scale UAV swarm air combat to overcome the effect of the number of participants on the network model and to develop a more flexible and intelligent air combat decision-making method.

Acknowledgements This work was supported by National Key R&D Program of China (Grant No. 2023YFC3011001) and National Natural Science Foundation of China (Grant Nos. U20B2071, 62350048, T2121003).

References

1 Fan S, Liu H H T. Multi-UAV cooperative hunting in cluttered environments considering downwash effects. Guid Navigat Control, 2023, 03: 2350004

2 Kong L, Reis J, He W, et al. On dynamic performance control for a quadrotor-slung-load system with unknown load mass. Automatica, 2024, 162: 111516

3 Li S, Shao X, Zhang W, et al. Distributed multicircular circumnavigation control for UAVs with desired angular spacing. Defence Tech, 2024, 31: 429–446

4 Kong L, Reis J, He W, et al. Experimental validation of a robust prescribed performance nonlinear controller for an unmanned aerial vehicle with unknown mass. IEEE ASME Trans Mechatron, 2024, 29: 301–312

5 Jiang F, Xu M, Li Y, et al. Short-range air combat maneuver decision of UAV swarm based on multi-agent transformer introducing virtual objects. Eng Appl Artif Intell, 2023, 123: 106358

6 Dong Y Q, Ai J L, Liu J Q. Guidance and control for own aircraft in the autonomous air combat: a historical review and future prospects. Proc Inst Mech Eng Part G J Aero Eng, 2019, 233: 5943–5991

7 Sun Z, Piao H, Yang Z, et al. Multi-agent hierarchical policy gradient for air combat tactics emergence via self-play. Eng Appl Artif Intelligence, 2021, 98: 104112

8 Kong W R, Zhou D Y, Zhang K, et al. Air combat autonomous maneuver decision for one-on-one within visual range engagement base on robust multi-agent reinforcement learning. In: Proceedings of the 16th International Conference on Control & Automation (ICCA), Singapore, 2020. 506–512

9 Yang Q, Zhang J, Shi G, et al. Maneuver decision of UAV in short-range air combat based on deep reinforcement learning. IEEE Access, 2019, 8: 363–378

10 Wang L, Wang J, Liu H, et al. Decision-making strategies for close-range air combat based on reinforcement learning with variable-scale actions. Aerospace, 2023, 10: 401

11 Li S, Wang Y, Zhou Y, et al. Multi-UAV cooperative air combat decision-making based on multi-agent double-soft actor-critic. Aerospace, 2023, 10: 574

12 Duan H, Li P, Yu Y. A predator-prey particle swarm optimization approach to multiple UCAV air combat modeled by dynamic game theory. IEEE CAA J Autom Sin, 2015, 2: 11–18

13 Huang C, Dong K, Huang H, et al. Autonomous air combat maneuver decision using Bayesian inference and moving horizon optimization. J Syst Eng Electron, 2018, 29: 86–97

14 Liu L, Zheng Y, Lu X, et al. Research on individual performance index of air cluster combat aircraft based on differential game theory. J Phys-Conf Ser, 2023, 2478: 102013

15 Liu Y P, Gao X, Shi J X, et al. Research on decision-making method of air combat embedded training based on extended influence diagram. In: Proceedings of Advances in Guidance, Navigation and Control. Lecture Notes in Electrical Engineering, Singapore, 2021

16 Jiandong Z, Qiming Y, Guoqing S, et al. UAV cooperative air combat maneuver decision based on multi-agent reinforcement learning. J Syst Eng Electron, 2021, 32: 1421–1438

17 Li Y, Shi J, Jiang W, et al. Autonomous maneuver decision-making for a UCAV in short-range aerial combat based on an MS-DDQN algorithm. Defence Tech, 2022, 18: 1697–1714

18 Zhu J, Kuang M, Zhou W, et al. Mastering air combat game with deep reinforcement learning. Defence Tech, 2024, 34: 295–312
19 Yuan X, Wang H, Yu W. A weighted mean field reinforcement learning algorithm for large-scale multi-agent collaboration. Guid Navigat Control, 2023, 03: 2350007
20 Li J N, Nie H, Chai T, et al. Reinforcement learning for optimal tracking of large-scale systems with multitime scales. Sci China Inf Sci, 2023, 66: 170201
21 Wang H, Wang J. Enhancing multi-UAV air combat decision making via hierarchical reinforcement learning. Sci Rep, 2024, 14: 4458
22 Luo D, Fan Z, Yang Z, et al. Multi-UAV cooperative maneuver decision-making for pursuit-evasion using improved MADRL. Defence Tech, 2024, 35: 187–197
23 Wang Z, Guo Y, Li N, et al. Autonomous collaborative combat strategy of unmanned system group in continuous dynamic environment based on PD-MADDPG. Comput Commun, 2023, 200: 182–204
24 Hu D, Yang R, Zhang Y, et al. Aerial combat maneuvering policy learning based on confrontation demonstrations and dynamic quality replay. Eng Appl Artif Intell, 2022, 111: 104767
25 Austin F, Carbone G, Falco M, et al. Automated maneuvering decisions for air-to-air combat. In: Proceedings of Guidance, Navigation and Control Conference, Monterey, 1987
26 Yang A W, Li Z W, Li B, et al. Air combat situation assessment based on dynamic variable weight. Acta Armamentarii, 2021, 42: 1553–1563
27 Zhan G, Zhang X, Li Z, et al. Multiple-UAV reinforcement learning algorithm based on improved PPO in ray framework. Drones, 2022, 6: 166
28 Yu C, Velu A, Vinitsky E, et al. The surprising effectiveness of PPO in cooperative, multi-agent games. 2021. ArXiv:2103.01955
29 Schulman J, Wolski F, Dhariwal P, at al. Proximal policy optimization algorithms. 2017. ArXiv:1707.06347
30 Zhu J W, Zhang H, Zhao S B, et al. Multi-constrained intelligent gliding guidance via optimal control and DQN. Sci China Inf Sci, 2023, 66: 132202
31 Li L T, Zhou Z M, Chai J J, et al. Learning continuous 3-DoF air-to-air close-in combat strategy using proximal policy optimization. In: Proceedings of IEEE Conference on Games (CoG), Beijing, 2022. 616–619