UAV Formation Beyond-Visual-Range Air Combat Decision Based on Multi-agent Reinforcement Learning

*JIANG Yufei, SHI Hanyue, ZHOU Yaoming**

School of Aeronautic Science and Engineering, Beihang University, Beijing 100191, P. R. China

(Received 10 June 2025; revised 26 November 2025; accepted 25 December 2025)

Abstract: With the rapid evolution of weapon systems towards precision and intelligence, unmanned aerial combat has increasingly transitioned from close-range engagements to beyond-visual-range (BVR) operations. This paper addresses the challenges of learning effective missile launch strategies in BVR air combat, where the long delay between weapon launch and target hit leads to sparse and delayed reward problems. This paper first extends the multi-agent proximal policy optimization (MAPPO) framework to incorporate expert rule-based launch control, resulting in MAPPO with launch constraints (MAPPO-LC). This method ensures that missile launch decisions satisfy tactical constraints on distance, altitude and timing while providing the learning process with a viable starting policy. Building upon this baseline, this paper introduces MAPPO with reward return (MAPPO-RR), a reward return mechanism that explicitly identifies missile launch and hit events as key decision nodes, and return the hit reward to the launch step. This reward redistribution method mitigates the delayed reward problem, and significantly accelerates policy convergence in multi-agent BVR scenarios. Experimental evaluation demonstrates that the MAPPO-RR algorithm achieves a win rate exceeding 75% and exhibits a sampling efficiency over 55% higher than that of the baseline method.

Key words: beyond-visual-range (BVR) air combat; collaborative decision-making; multi-agent reinforcement learning; delayed reward; reward reshaping

CLC number: V279 **Document code:** A **Article ID:** 1005-1120(2026)03-0386-14

0 Introduction

Autonomous air combat (AAC) technology refers to the capability of a fighter aircraft to acquire battlefield situational information through onboard sensing equipment and autonomously execute air combat maneuvers based on observations. This is achieved by generating tactical decisions and executing them through maneuver control actions. Among these, the decision-making module, tasked with generating optimal tactical actions in real-time under highly dynamic and complex air combat scenarios, constitutes the primary focus and challenge of AAC research. Decision-making algorithms in air combat can be categorized into four branches[1]: Game theory-based methods[2-3], heuristic algorithms[4-5], expert knowledge-based approaches[6-7], and neural network-based techniques.

With the advancement of reinforcement learning (RL) algorithms and their remarkable performance in domains such as Atari and StarCraft, an increasing number of researchers are exploring the application of RL to autonomous air combat decision-making. Hu et al.[8] proposed a novel dynamic prioritized replay method, enabling agents to learn tactical strategies from historical data, thereby enhancing the algorithm's convergence speed. Liu et al.[9] addressed low training efficiency in proximal policy optimization (PPO) by incorporating multi-head attention and phased reward mechanisms. Liu et al.[10] developed an air combat decision-making method based on multi-agent proximal policy optimization (MAPPO), specifically for beyond-visual-

*Corresponding author, E-mail address: zhouyaoming@buaa.edu.cn.

How to cite this article: JIANG Yufei, SHI Hanyue, ZHOU Yaoming. UAV formation beyond-visual-range air combat decision based on multi-agent reinforcement learning[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2026, 43(3): 386-399.

http://dx.doi.org/10.16356/j.1005-1120.2026.03.005

range (BVR) scenarios. To tackle the sparse reward problem in traditional RL, they designed a comprehensive reward function. Xu et al.[11] proposed a RL method combining virtual opponent modeling and value attention decomposition, addressing higher-order dynamics challenges and credit assignment problems while demonstrating superior performance in complex scenarios. Zhang et al.[12] presented an algorithm which uniquely combined prior-based training with prior-free execution to accelerate training, ensure robust target reacquisition. To capture temporal scale features in aerial combat observations, Yang et al.[13] integrated long short-term memory (LSTM) layers into neural networks, supplemented with delayed input stacking techniques, and further combined expert policies, curiosity-driven reward mechanisms, and curriculum learning to enhance adaptability and strategic decision-making.

Han et al.[14] proposed the prioritized population play with diversified partners (P3DP) method, which enhanced the robustness of unmanned aerial combat vehicle tactical maneuver decision-making against diverse opponent strategies through historical win-rate-based prioritized sampling, joint optimization of individual and population entropy, and a partner mechanism. Chen et al.[15] proposed an attention-based multi-agent actor-critic (AMAAC) algorithm for unmanned aerial vehicle (UAV) decision-making, and extending a missile hit probability-based actor-critic model to multi-UAV scenarios. A observations-of-fighters encoder (OFE) was introduced to extend the algorithm to UAV combat scenarios of different scales. Wang et al.[16] proposed a rule-driven decision-making method enhanced by evolutionary game theory, aiming to improve interpretability and competitiveness in one-versus-one (1v1) BVR air combat.

Current research indicates that the RL algorithm holds significant potential for application in multi-agent cooperative air combat decision-making. However, several issues remain in existing studies. First, the realism of simulation environments is often insufficient. To reduce computational complexity, many studies simplify aircraft models and onboard systems in simulations. For instance, some studies simplify air combat by using 2D environments[17], three-degree-of-freedom aircraft models[18], basic fire-control radar and automated attack missile[19]. Second, the primary weapon in BVR air combat scenarios is the medium-to-long-range missile. However, existing studies often handle missile launch using either fixed expert strategies or launch reward[17]. Methods based only on expert rules lack adaptability, while approaches relying on immediate launch rewards suffer from the delayed reward problem due to the long interval between launch and hit. To address these challenges, this paper presents the MAPPO with reward return (MAPPO-RR) algorithm.

The key contributions of this work are as follows.

(1) A comprehensive framework is presented to address challenges of autonomous UAV coordination in BVR air combat. To enhance the performance of multi-agent RL in BVR combat, expert knowledge-based cooperative rules are designed as adversarial strategies to guide RL training. We employ MAPPO as the foundational algorithm to construct a centralized training with decentralized execution (CTDE) reinforcement learning framework.

(2) A missile launch constraint strategy MAPPO with launch constraints (MAPPO-LC) is introduced into the original MAPPO framework. By determining launch permission based on distance, flight altitude and cooldown, MAPPO-LC enables the agent to perform reasonable offensive actions during early training. This approach overcomes the limitations of treating missile launch as a standard action, mitigating ineffective early-stage policies and facilitating the acquisition of positive rewards.

(3) MAPPO-RR, a reward reshaping mechanism is introduced to address the delayed reward problem by returning critical rewards to earlier decision points, thereby enhancing the agent's ability to learn key actions and improving overall training efficiency. The proposed algorithm achieves a stable win rate exceeding 75% against that of expert-level strategies, demonstrating significant improvements in both tactical success and training efficiency compared to baseline methods.

1 Simulation Environment

1.1 Combat model and scenarios

The BVR air combat simulation system designed in this study utilizes a hybrid Python and C++ programming approach. Aircraft and missile dynamics are encapsulated in C++, while the remaining components are implemented in Python. The aircraft model in this study is based on the JSBSim library. An open-source aircraft dynamics modeling software is developed in C++. It defines the six-degree-of-freedom full-motion equations for the F-16 fighter and is widely recognized and used in both academic and industrial sectors.

To obtain strategies suitable for real combat scenarios, a simulation environment that reflects realistic conditions is constructed, as shown in Fig.1. The multi-threaded simulation module enables parallel simulation training for different combat tasks. In this environment, agents interact with the simulation to obtain observational data. Ego aircraft data are gathered through sensors. Partner aircraft data are transmitted via data links, and enemy aircraft information is captured by radars. These three types of information are integrated to support agent decision-making.

The simulated battlefield is an unobstructed airspace. Each of red and blue teams consists of two unmanned F-16 fighters, with each aircraft carrying six AIM-120D medium-to-long-range air-to-air missiles. All aircraft are equipped with fire-control radars, electronic support measures and data links. At the start of combat, early warning aircraft provide both teams with situational information with noise. To ensure a fair evaluation of algorithmic performance, two teams are configured with identical maneuvering and combat capabilities. Consequently, the engagement outcomes are strictly dictated by their respective decision-making methods where an expert rule-based strategy versus a reinforcement learning agent. The goal for both teams is to maximize enemy aircraft destruction while ensuring self-alive to secure victory.

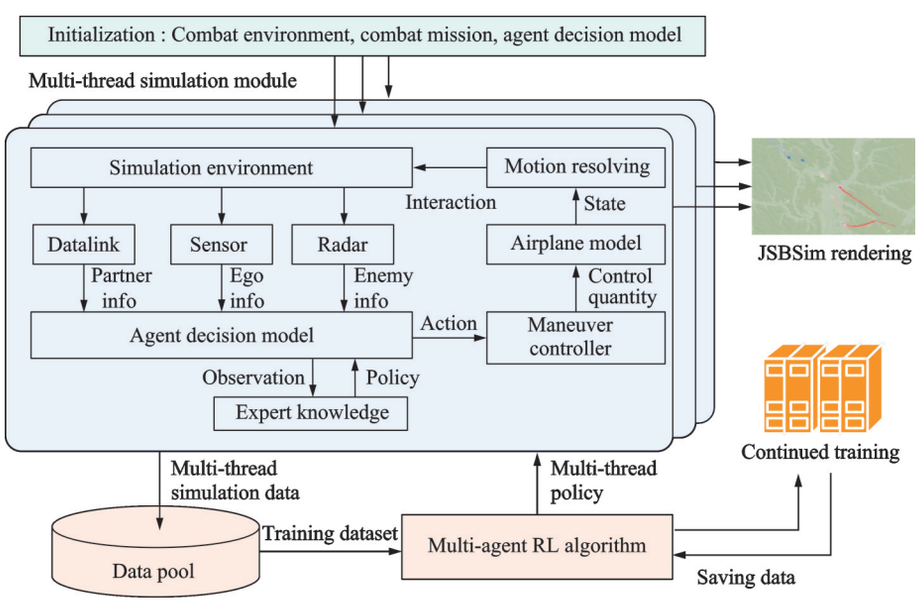


Fig.1 Framework of cooperative air combat simulation system

1.2 Expert opponent strategies for reinforcement learning

The cooperative attack-defense strategy designed in this paper is based on the basic theory of BVR air combat. The attack strategy consists of two parts: Missile launch and aircraft maneuvering. For the missile launch strategy, a variable launch altitude is used based on the distance to the enemy.

The launch command is given when the aircraft reaches the permissible launch altitude. For the maneuvering control strategy, it is divided into four phases: Attack, attack-to-defense switch, defense and defense-to-attack switch, as shown in Fig.2.

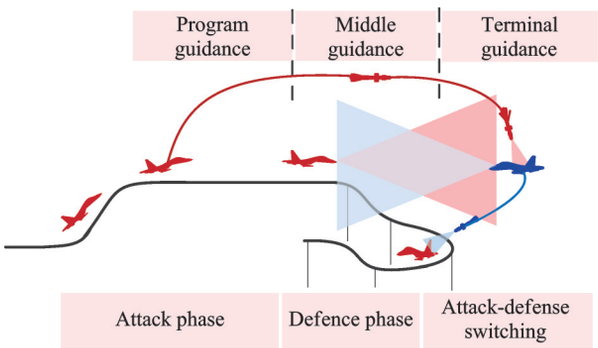


Fig.2 Schematic diagram of each phase of BVR air combat

During the attack phase, priority is given to maintain the aircraft at the missile launch altitude while executing maneuvering actions to obtain a favorable attack angle. The aircraft approaches the enemy target and executes a climb maneuver to increase the altitude, thereby expanding the missile's firing range. Subsequently, by pitching up, it performs a lofted missile launch for long-range attacks. Before missile lock-on, the aircraft descends to a low altitude to provide mid-course guidance for the missile.

The attack-defense switch is triggered by one of two conditions: (1) When the aircraft receives a missile warning; and (2) when it runs out of weapons. The dodge missile takes the highest priority in the decision-making process. Upon receiving a missile warning, the system immediately switches to the defense phase and performs an evasive maneuver, executing a barrel roll to turn and face away from the missile's direction to avoid interception.

In the defense phase, no missile launches are allowed. Instead, the aircraft maneuvers to increase distance from the enemy and attempts to approach from the side to regain positional advantage. After successfully evading the missile, the aircraft rapidly flies away from the enemy to restore a favorable tactical situation.

The defense-to-attack transition occurs when the maximum duration of the defense phase is reached and the aircraft still has remaining missiles. At this point, the system transitions back to the attack phase.

2 MAPPO-RR Algorithm Framework

2.1 MAPPO algorithm introduction

In multi-UAV BVR air combat decision-making, each UAV is modeled as an independent decision-making agent, and the air combat process is treated as a finite-horizon decision-making problem with incomplete situational information during engagements. This scenario can be formulated as a decentralized partially observable Markov process (Dec-POMDP)[19].

A decentralized Markov process can be represented as $\langle N, S, O, A, P, R, \gamma \rangle$, where $N=\{1, 2, \cdots, n\}$, O denotes the observation space of agents, and S describes the true state of the environment. At each time step, each agent selects an action $a_{i,t}$ according to its policy π_i and local observation $o_{i,t}$. $A=\{A_1, A_2, \cdots, A_n\}$ denotes the set of actions available to all agents, and A_i denotes the set of actions available to agent i. The joint action $u=\{a_1, a_2, \cdots, a_n\}$ denotes the combination of actions chosen by all agents at time t. The state transition function $P(s_{t+1}|a_t, s_t)$ denotes the probability that the next state is s_{t+1} given the current state s_t and the joint action a_t. R denotes the reward function, and γ is the discount factor, reflecting the importance of future rewards. The multi-agent decision policy is expressed as a joint policy function $\pi(u|s)=\prod_{i=1}^{n}\pi_i(a_i|s_i)$. During each policy update, it is essential to evaluate the rewards obtained by the strategy. Solving the Dec-POMDP problem means optimizing an objective function to identify the policy that $\pi(u|s)$ selects the optimal joint action u at state s.

In reinforcement learning, the actor-critic framework integrates policy evaluation and policy improvement, offering both robust performance and flexibility that effectively speed up algorithm convergence. Under this framework, we implement the actor and critic using two distinct neural networks, as shown in Fig.3. Both architectures consist of a feature extraction layer, a gated recurrent unit (GRU) for processing temporal dependencies, and a multilayer perceptron (MLP) to generate the final outputs.

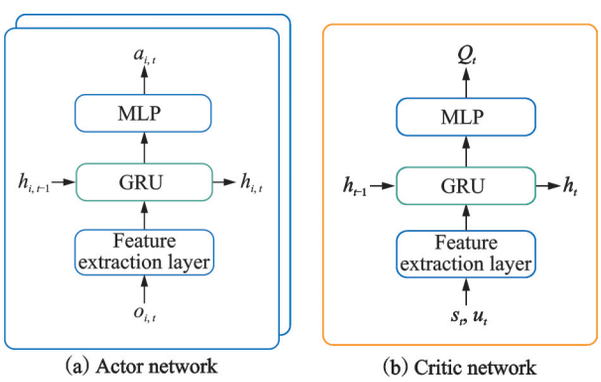


Fig.3 Algorithmic network structure

The actor network employs a policy function with the objective of maximizing the expected cumulative return, while the critic network uses a value function to estimate the value of the current policy. To alleviate the high variance associated with gradient estimation, an advantage function is introduced to quantify the relative benefit of a given state-action pair over the average performance.

$$A_\pi(o_t,a_t)=Q_\pi(o_t,a_t)-V_\pi(o_t) \tag{1}$$

PPO[20] is a RL method based on the actor-critic framework. It effectively addresses the challenge of selecting an appropriate learning rate during training. During updates, PPO computes the ratio of the action probabilities under the new policy $\pi_\theta(a_t|o_t)$ and old policy $\pi_{\theta_{\text{old}}}(a_t|o_t)$ to control the extent of policy updates. PPO restricts policy changes by introducing a clip function to constrain this ratio. The objective function of PPO is presented as

$$J_{\text{CLIP}}(\theta)=E_t[\min(r_t(\theta)A_t,\text{clip}(r_t(\theta),1-\varepsilon,1+\varepsilon)A_t)] \tag{2}$$

In this context, ε denotes the clip coefficient and the clip function constrains the ratio $r_t(\theta)$ between the new and the old policies within a predetermined interval $[1-\varepsilon,1+\varepsilon]$ to ensure the algorithm's convergence.

2.2 State space and action space

For state space, both global and local observations are used. The critic network employs a 244-dimensional global observation for action evaluation, while the actor network selects actions based on a 99-dimensional local observation.

For the 2v2 BVR air combat scenario, the global observation includes complete state information for all aircraft and missiles. Specifically, each aircraft ego state includes id, position, attitude, velocity, acceleration, radar and lock information, and remaining missiles. The missile data are recorded for 24 missiles with each missile information (ego id, target info, and lock info).

The local observation for each aircraft includes information such as partner information, enemy information, ego missiles, enemy missile and radar warning. Regarding enemy aircraft information, when the radar cannot track the enemy aircraft, the own aircraft receives status information with noise. If aircraft locks enemy through radar, it receives the real information.

For the action space, two agents share the same action space and adopt a centralized training with CTDE architecture. During training, a single policy network is utilized, while during execution, each agent makes independent decisions based on its own observations. The selected actions will serve as the top-level commands for the tactical controller, which then passed to the lower maneuver layers to control the aircraft's posture. As shown in Table 1, 11 maneuvering actions are designed, including level flight, climbing, diving, turning and dodge missiles, among others.

Table 1 Action space

Maneuver code	Maneuver description
Level flight	Keep speed at 1 Mach
Acceleration	Accelerate to 2 Mach
Climb	Climb at a 30° angle
Dive	Dive at a −30° angle
Missile evasion	Turn away from missile direction to evade
Sharp turn toward enemy	Perform a 6g horizontal turn toward the enemy target bearing
Turn toward enemy	Perform a 4g horizontal turn toward the enemy target bearing
Sharp turn away from enemy	Perform a 6g horizontal turn to face away from the enemy target bearing
Turn away from enemy	Perform a 4g horizontal turn to face away from the enemy target bearing
Turn tangential to enemy	Perform a 4g horizontal turn to a bearing perpendicular to the enemy target
Offset toward enemy	Perform a 4g horizontal turn to a bearing ±30° relative to the enemy target

2.3 Reward function

In the combat mission scenario presented in this paper, a fully cooperative environment is designed, where all agents share the same reward function. The reward function consists five components: Death penalty, missile hit reward, guidance reward, lock-on reward and missile warning reward. The reward of agent i at time step t is

$$R_t^{(i)} = R_{\text{death}}^{(i)} + R_{\text{hit}}^{(i)} + R_{\text{guid}}^{(i)} + R_{\text{lock}}^{(i)} + R_{\text{warn}}^{(i)} \tag{3}$$

The death rewards for the entire combat mission are designed as

$$R_{\text{death}}^{(i)} = \begin{cases} -20 & \text{Agent } i \text{ is shot down} \\ -100 & \text{Agent } i \text{ crashes} \\ 0 & \text{Otherwise} \end{cases} \tag{4}$$

To address the issue of sparse rewards, subjective rewards are assigned throughout the entire missile launch process. This encourages the aircraft to acquire advantageous positions for missile launches and assists the missile in locking onto the enemy through the guidance process

$$R_{\text{hit}}^{(i)} = \sum_{m \in \mathcal{M}_i} \left[\mathbb{I}_{\text{parent}(m)=i} \cdot r_{\text{kill}} + \mathbb{I}_{\text{guider}(m)=i} \cdot r_{\text{guid}} \right] \tag{5}$$

where $\mathcal{M}_i$ is the set of missiles that successfully hit a target during the episode; $r_{\text{kill}} = 40$ the reward for directly launching a missile that destroys an enemy; $r_{\text{guid}} = 40$ the reward for guiding a missile that destroys an enemy; $\mathbb{I}$ the indicator function; parent(m) the missile's launching aircraft ID; and guider(m) the aircraft to provide guidance for missile m.

For the guidance phase, to encourage the aircraft to fly towards enemy for missile guidance, rewards are given when the aircraft successfully guides the missile.

$$R_{\text{guid}}^{(i)} = \begin{cases} \delta_{\text{g}} & h_i < H_{\text{thr}} \text{ or } d_i > D_{\text{thr}}, \text{ guiding} \\ 0 & \text{Otherwise} \end{cases} \tag{6}$$

where h_i is the altitude of UAV i; $H_{\text{thr}} = 6\,000$ m; d_i the missile-target distance; $D_{\text{thr}} = 90\,000$ m and $\delta_{\text{g}} = 0.000\,5$.

If a missile successfully locks on a target, a reward is given, with an additional bonus for high speed.

$$R_{\text{lock}}^{(i)} = \delta_l + \begin{cases} \delta_v & v_{\text{m}} \geqslant 3M \\ 0 & \text{Otherwise} \end{cases} \tag{7}$$

where $\delta_l = 0.01$ is the base lock reward; v_{m} the missile velocity; M the Mach speed unit; and $\delta_v = 0.2$ the high-speed lock bonus.

For the dodge missile phase, if the number of missile warnings is smaller than that of the previous time step, it means that the missile evasion is successfully achieved through maneuvering, so a reward is given. If an incoming missile threat is detected, a penalty is proportionally applied to the current load factor.

$$R_{\text{warn}}^{(i)} += \begin{cases} \alpha_{\text{warn}} & M_{\text{prev}}^{(i)} > W^{(i)} \\ -\beta_{\text{warn}} \cdot \left(1 - \dfrac{|a_n^{(i)}|}{5}\right) & W^{(i)} > 0 \\ 0 & \text{Otherwise} \end{cases} \tag{8}$$

where $\alpha_{\text{warn}} = 2.0$ is the reward for a decrease in the number of active missile warnings; $M_{\text{prev}}^{(i)}$ the previous step's missile warning count; $W^{(i)}$ the current missile warning count; $\beta_{\text{warn}} = 0.02$ the penalty scaling factor; and $a_n^{(i)}$ the aircraft overload.

2.4 Missile launch constrain strategy

In a BVR air combat, due to the limited number of missiles, the number of decision-making chance for the agent to launch a missile is limited. If missile launches are treated solely as maneuver actions for exploration, the simulation inevitably results in multiple selections of the missile launch action. As a result, all missile launches may be wasted early in the episode, severely limiting the agent's ability to explore the optimal state-action space associated with successful combat outcomes. When given the limited number of missiles, the agent will struggle to learn how to effectively manage missile launches. Therefore, the missile launch condition needs to be designed to improve the learning efficiency and stability of the RL algorithm.

In previous studies, a fixed missile launch interval was typically used. This causes a problem that during periods when missile launches is cooling down, agent's action decisions cannot be responded or executed by the environment, which severely interferes with the calculation of the agent's action values, leading to a deviation in policy optimization and affecting the convergence of the policy. Additionally, the policy tends to be relatively fixed, making it easy for the opponent to find weakness.

$$h_{\min}(d_{\min})=\begin{cases}10.5 & d_{\min}>100\\ 0.1d_{\min}+1 & 65<d_{\min}\leqslant 100\\ 5.5 & 25<d_{\min}\leqslant 65\\ 0 & d_{\min}\leqslant 25\end{cases} \tag{9}$$

To overcome this problem, this paper designs a missile launch strategy based on expert knowledge, which called the MAPPO-LC algorithm. The strategy includes a launch cool-down time and permissible launch altitude that changes with distance, as it shown in Eq.(9). The closer the distance, the shorter the cool-down time and the lower the permissible launch altitude. Once the aircraft's state meets these constraints, it will launch missiles towards the enemy aircraft.

In the proposed MAPPO-LC method, missile launch decisions are limited by a set of constraints. A launch is permitted only when four conditions are simultaneously satisfied: (1) The elapsed time which is counted since the previous launch exceeds the predefined cool down interval, preventing excessive firing frequency; (2) no missiles are currently in the return or guidance state, ensuring resource availability; (3) the ego platform's altitude exceeds the minimum allowable launch altitude; (4) the number of launched missiles does not exceed the enemy number.

$$\text{shoot}_t=\mathbb{I}_{t-t_{\text{last}}>t_{\text{cool}}}\cdot\mathbb{I}_{N_{\text{missiles}}=0}\cdot\mathbb{I}_{h_{\text{ego}}>h_{\min}(d_{\min})}\cdot\mathbb{I}_{N_{\text{shoot}}<N_{\text{radar}}} \tag{10}$$

2.5 Delayed reward issue in missile launch

At the mission level in this study, the weapon, a medium-to-long-range air-to-air missile, introduces a significant delayed reward issue. In reinforcement learning, an agent typically receives immediate rewards after executing an action. However, in a BVR air combat, there exists a substantial delay between missile launch and target impact. During this interval, the missile must complete a series of complex maneuvers, launch, middle guidance and terminal guidance, before it can effectively destroy the enemy, at which point the hit reward is finally given. In classical reinforcement learning algorithms, the hit reward is primarily allocated to the state-action pair when the missile hits the target. Due to the discount factor in the reward back-propagation, the reward for missile launch is minimal, which makes it ineffective in providing meaningful feedback for timing the launch.

Based on the aforementioned problem, this paper proposes a missile hit reward back-propagation method through objective reward reshaping, combined with a missile launch constraint strategy, referred to as the MAPPO-RR algorithm. At the moment of missile launch, the system records the launcher's information and launch time. If the missile finally destroy the target, the reward corresponding to the launch time in the temporary data pool is modified, with a portion of the reward obtained at the hit moment being returned to the launch time, as shown in Fig.4. Compared to the baseline algorithm, the MAPPO-RR algorithm significantly accelerates the convergence speed during the training process.

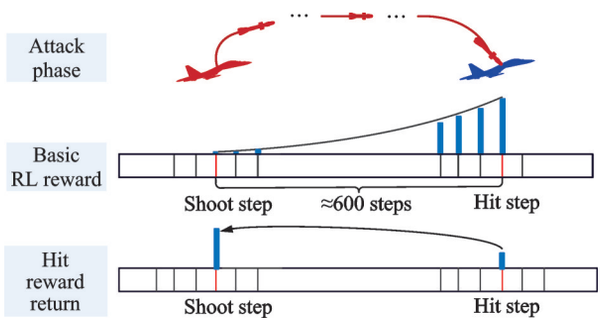


Fig.4 RL reward & hit reward return process

2.6 Hit reward return algorithm

MAPPO employs a global value function to promote coordination among individual PPO agents. Each agent generates actions based on its local observations and a shared policy to maximize the discounted cumulative reward. The critic, utilizing a value function based on global observations, evaluates the value of actions chosen by the actor network.

In the simulations presented in this paper, due to the identical aircraft configuration and performance, both agents share a single actor network. The data obtained by each agent through interaction with the environment is stored in a shared data pool for network training. The shared data pool helps improve the sample utilization efficiency.

The loss function for the actor network is defined as

$$\mathcal{J}(\theta)=\left[\frac{1}{Bn}\sum_{i=1}^{B}\sum_{k=1}^{n}\min\left(r_{\theta,i}^{(k)}A_i^{(k)},\mathrm{clip}\left(r_{\theta,i}^{(k)},1-\varepsilon,1+\varepsilon\right)A_i^{(k)}\right)\right]+\sigma\frac{1}{Bn}\sum_{i=1}^{B}\sum_{k=1}^{n}S\left[\pi_\theta\left(o_i^{(k)}\right)\right] \tag{11}$$

The loss function for critic network is

$$\mathcal{J}(\Phi)=\frac{1}{Bn}\sum_{i=1}^{B}\sum_{k=1}^{n}\max\left[\left(V_\Phi\left(s_i^{(k)}\right)-\hat{R}_i\right)^2,\left(\mathrm{clip}\left(V_\Phi\left(s_i^{(k)}\right),V_{\Phi_{\mathrm{old}}}\left(s_i^{(k)}\right)-\varepsilon,V_{\Phi_{\mathrm{old}}}\left(s_i^{(k)}\right)+\varepsilon\right)-\hat{R}_i\right)^2\right] \tag{12}$$

where $A_i^{(k)}$ denotes the advantage function; S the entropy of the policy; σ the policy entropy coefficient; R_i the discount reward; B the batch size; n the number of agents; and ε the clipping coefficient. The term $r_{\theta,i}^{(k)}=\pi_\theta\left(a_i^{(k)}\mid o_i^{(k)}\right)/\pi_{\theta_{\mathrm{old}}}\left(a_i^{(k)}\mid o_i^{(k)}\right)$ is the probability ratio between the current and previous policies. i and k represent the sample index within the mini-batch and the agent index, respectively. θ_{old} and Φ_{old} represent the parameters of the previous actor and critic networks, respectively.

The MAPPO-RR algorithm is developed based on the MAPPO algorithm to address the delayed reward problem in missile launching.

In a BVR air combat, long-range air-to-air missiles are characterized by a significant time lag between the launch and the target steps. For a missile m launched by agent i, let

$$\tau_m^{\mathrm{launch}}<\tau_m^{\mathrm{hit}},\quad \Delta_m\triangleq\tau_m^{\mathrm{hit}}-\tau_m^{\mathrm{launch}} \tag{13}$$

If the hit event get reward $R_{\mathrm{hit},m}^{(i)}$ at τ_m^{hit}, its expected contribution at the launch step is $\gamma^{\Delta_m}R_{\mathrm{hit},m}^{(i)}$, effectively eliminating the advantage signal at launch, making it difficult for the agent to learn missile launching or offensive maneuvers, as shown in Fig.4.

To enhance credit assignment at critical decision points, especially launch events, we apply reward reshaping at the trajectory level. We define a return switch for each hit event and launch event, redistributing part or all of the hit reward back to key time step in $[\tau_m^{\mathrm{launch}},\tau_m^{\mathrm{hit}}]$. Then the reshaped reward for agent i is shown as

$$\tilde{r}_t^{(i)}=r_t^{(i)}+\sum_{m\in\mathcal{H}_i}\mathbb{I}\left[t=\tau_m^{\mathrm{launch}}\right]R_{\mathrm{hit},m}^{(i)}-\sum_{m\in\mathcal{H}_i}\mathbb{I}\left[t=\tau_m^{\mathrm{hit}}\right]R_{\mathrm{hit},m}^{(i)} \tag{14}$$

where $\mathcal{H}_i$ is the set of missiles whose hits generate rewards for agent i.

$$\delta_t^{(i)}=\tilde{r}_t^{(i)}+\gamma V_\Phi\left(s_{t+1}\right)-V_\Phi\left(s_t\right)\quad i=1 \tag{15}$$

where $\tilde{r}_t^{(i)}$ is the reshaping reward. Advantage values near τ_m^{launch} are significantly increased, boosting sampling and updates for launch maneuvers. The pseudocode for the MAPPO-RR algorithm is as follows. And its framework is illustrated in Fig.5.

MAPPO-RR algorithm

Initial parameters for actor and critic network π,θ,Φ
while step $\leqslant$ max_step do
  Initial data pool $D=\{\}$
  For $b=1$ to buffer_size do
    Initial recurrent neural network net parameters
    For $t=1$ to T do
      for agent $i=\{1,2,\cdots,n\}$ do
        Actor network $\pi(o_t^i,h_{t-1,\theta}^i;\theta)$ produces action

distribution and RNN parameter

Critic network $V(s_t^i, h_{t-1,V}^i; \Phi)$ produces state value function and RNN parameter

select action a_t^i

end for

Perform actions a_t^i, receive r_t, s_{t+1}, o_{t+1}

If missile is launched, record launch time τ_m^{launch}

If missile hits, hit reward returns $\tilde{r}_t^{(i)}$

Add $[s_t, o_t, h_{t,\pi}, h_{t,V}, a_t, \tilde{r}_t^i, s_{t+1}, o_{t+1}]$ to D_{temp}

If terminate

Record the outcome of the battle

Store a total episode into the temporary data pool D_{temp}

break

end for

Compute advantage and discount reward

Save data to data pool $D = D \cup D_{\text{temp}}$

end for

For mini-batch $k = \{1, 2, \cdots, K\}$ do

Randomly select mini-batch data b from D

Update the parameters of the RNN network

end for

Updated network parameters through Adam optimizer

end while

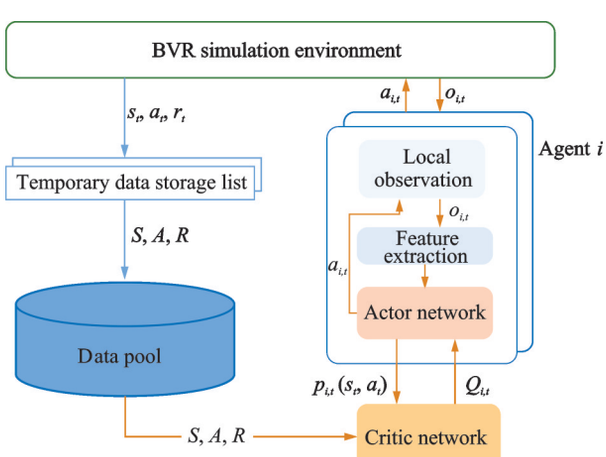


Fig.5 Reinforcement learning training framework

In the basic MAPPO framework, simulation data is stored in the data pool in the form of state-action pairs. Since this paper improves the reward reshaping for hits, a new data storage and extraction method is designed. State-action pairs are temporarily stored in a temporary list. When the simulation episode reaches the termination condition (e.g., one agent is completely destroyed) or the maximum simulation steps are reached, the data from the temporary list is then stored in the data pool as a complete episode. The size of the data pool is represented by the buffer size parameter, which denotes the number of complete episodes stored in the pool. During training, data are selected in the form of state-action pairs, and the mini-batch size represents the number of state-action pairs chosen from the data pool for each training step.

3 Simulation Results

3.1 Training parameter settings

The experimental environment is based on the Ubuntu 20.04 system, with the computer's CPU being an AMD Ryzen 9 5950X and the GPU being an NVIDIA RTX 4090. The modified algorithm introduced in this paper, which includes missile launch constraints, is referred to as MAPPO-LC. The improved algorithm that incorporates missile launch constraints and hit reward return is referred to as MAPPO-RR. During training, the blue side utilizes the multi-aircraft attack-defense strategy designed in Section 1.2, while the red side uses the MAPPO-LC, MAPPO-RR and the classical MAPPO algorithm. After training, a comparative analysis of the success rates and performance of each algorithm during the training process is conducted to validate the effectiveness of algorithms designed in this paper. Table 2 shows the hyperparameter settings for the RL algorithms.

Table 2 RL parameter setting

Parameter	Value
Parallel number	8
Episode length	4 500
Number of MLP layer	3
MLP layer dim	256
Learning rate	5e−4
Generalized advantage estimation discount factor	0.99
Clip parameter	0.025
PPO epoch	5
Hit reward return ratio	1

3.2 Simulation results

An evaluation is performed every five training cycles. During each evaluation, a 16 rounds of combat are conducted, and the average reward from these 16 rounds is recorded as the strategy's average reward. The success rate is calculated by dividing the number of victories by the total number of battles. The results of each evaluation round are plotted into reward function and success rate curves. These results are then smoothed using the sliding window method, with the smoothed outcomes depicted by the red curves in Fig.6.

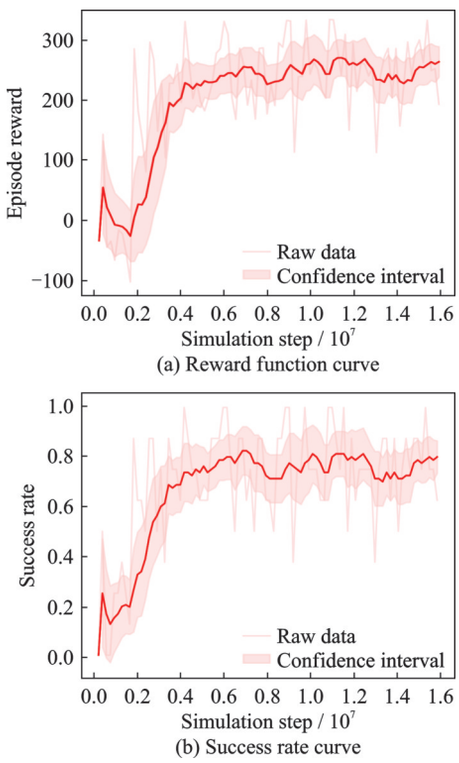


Fig.6 Training performance

As the number of training epoch increases, the total average per round shows an upward trend, indicating that the RL algorithm is learning a more effective combat strategy. The success rate also shows an upward trend, with the average win rate stabilizing above 75%. This fully demonstrates the effectiveness of the RL algorithm in BVR air combat decision-making. At the same time, the trend of the success rate curve is consistent with that of the reward function curve, indicating that the reward function designed in this paper is reasonable and can guide the agent to learn an effective air combat strategy.

Further ablation experiments are conducted, comparing the improved MAPPO-RR, MAPPO-LC, and MAPPO algorithms under the same parameters and reward functions. Results are shown in Fig.7, where the red curve is the win rate curve of the improved MAPPO-RR algorithm, the blue curve is the win rate curve of the improved MAPPO-LC algorithm, and the green curve is the win rate curve of the MAPPO algorithm with fixed launch intervals. Compared to the MAPPO algorithm with fixed launch intervals, the launch restriction strategy designed in this paper effectively improves the agent's win rate, and the win rate ultimately stabilizes above 70%. In contrast, the strategy with fixed launch intervals achieves a maximum win rate of around 45%, with large fluctuations in the win rate, making it difficult to converge. This fully verifies that the launch restriction strategy designed in this paper can effectively reduce the decision-making burden of the RL algorithm. The introduction of the prior knowledge-based launch restriction strategy enhances operational effectiveness and guides the agent to learn combat tactics in the BVR air combat environment.

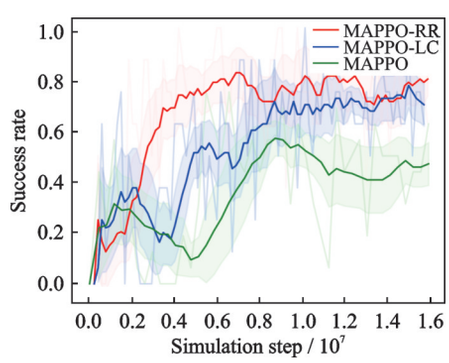


Fig.7 Comparison of the three algorithms' winning rate curves

When comparing the MAPPO-LC algorithm, which uses only missile launch restrictions, with the MAPPO-RR algorithm that includes reward feedback for hits, the MAPPO-LC algorithm reaches a win rate of over 70% after 9×10^6 training steps. In contrast, the improved MAPPO-RR algorithm,

with reward feedback for hits, achieves a win rate of over 75% after just 4×10^6 training steps and gradually converges. This demonstrates a 55% improvement in training speed compared to that of the algorithm without reward feedback for hits. Fig.8 presents the win rate heatmap from 16-round pairwise combat between algorithms, where EXPERT represents the expert strategy introduced in Section 1.2, and cell values indicate the win rate of row algorithms against column opponents.

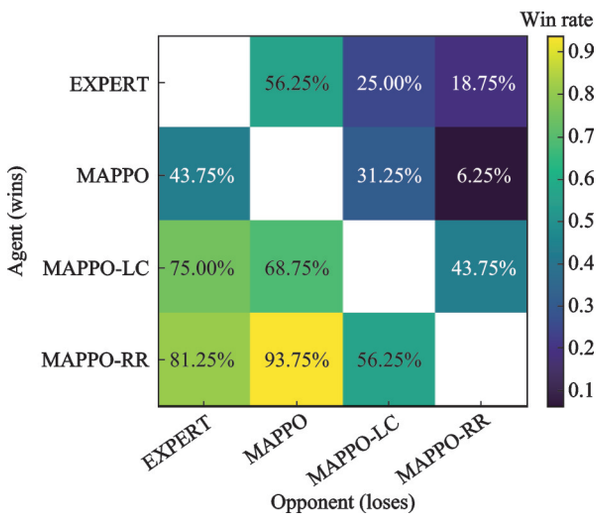


Fig.8 Head-to-head algorithm evaluation matrix

This fully demonstrates that the MAPPO-RR algorithm proposed in this paper effectively addresses the issue of delayed rewards, accelerates the convergence speed, and ultimately enables the agent to learn a combat strategy with a higher win rate.

In the simulation environment, the optimal strategy trained by the RL algorithm is compared with the fixed tactics for a 2v2 BVR air combat. The win rates from 16 simulation confrontations are shown in Table 3.

Table 3 Comparison of algorithm success rate

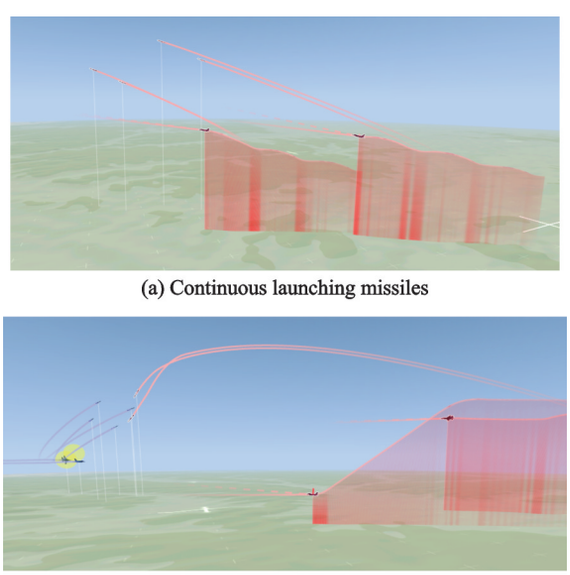
Algorithm	Final success rate/%
MAPPO	43.75
MAPPO-LC	75.00
MAPPO-RR	81.25

The above results fully demonstrate that the proposed MAPPO-RR algorithm is an improvement. In the face of sparse rewards and limited decision-making chances, the missile launch restriction design based on expert knowledge in the MAPPO-LC and the MAPPO-RR algorithm with reward feedback for hits improves the success rate and the convergence speed effectively.

3.3 MAPPO-RR strategy simulation performance analysis

In the simulation replay of the strategy learned by the reinforcement learning algorithm, the improved MAPPO algorithm in this paper effectively learns typical strategies for the BVR 2v2 multi-aircraft combat, including shot before sight, cooperative guidance and dodge missile.

(1) Shot before sight strategy. As shown in Fig.9, when the red side detects the enemy first, the agent climbs to gain altitude advantage, expands the attack range and attacks the enemy. As it approaches, the agent lowers its altitude for low-altitude guidance. The other red aircraft continues to launch missiles while providing mid-course guidance for those already in flight. This ensures that missiles successfully achieve terminal lock-on and successfully destroy the target.



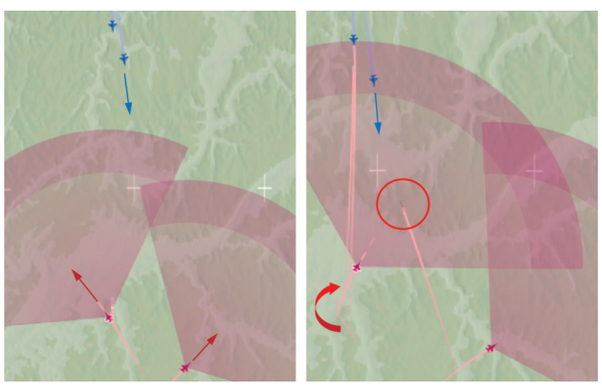
(a) Continuous launching missiles

(b) Mid-course guidance

Fig.9 Shot before sight strategy

(2) Cooperative guidance strategy. When encountering the enemy, the two red aircraft fly side-by-side to avoid a head-on engagement. The left red aircraft launches a missile to suppress the enemy on

its side, while the right red aircraft maneuvers and fires at the enemy on its side. As shown in Fig.10, when the enemy enters the radar range, the left red aircraft guides its missile and also coordinates guidance for the missile from the friendly aircraft. The red side successfully completes the BVR strike on two aircraft of the blue side without any losses.



(a) Fly side-by-side (b) Friendly aircraft relay guidance

Fig.10 Cooperative guidance strategy

(3) Dodge missile strategy. When the red side's first wave of attack fails to destroy the blue side, and the red side faces a missile launched by the blue side, the agent learns dodge missile strategy. Fig.11 shows the agent learns to reduce the altitude and performs a barrel roll to evade the missile by escaping from the missile's velocity vector direction. The decision-making logic and maneuver performance of the algorithm are clearly presented in the replay of the RL algorithm's effectiveness.

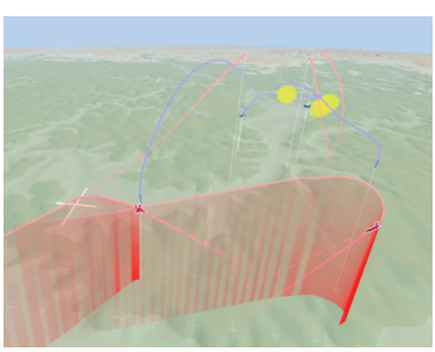


Fig.11 Dodge missile strategy

The simulation results, combined with the basic theory of the BVR air combat, provide a comprehensive explanation of the high success rate decision-making process, proving that the RL algorithm designed in this paper performs well in BVR air combat missions.

4 Conclusions

In response to the challenges of sparse rewards and a limited number of decision steps in the BVR air combat, this paper introduces a RL-based cooperative decision-making algorithm for BVR team engagements. To address the finite-decision-step problem, we propose an enhanced MAPPO-LC algorithm, which incorporates missile-launch cool down intervals and aircraft state as launch constraints. This design significantly accelerates convergence and improves win rates, offering a preliminary solution to decision-limited scenarios.

Furthermore, we develop a hit-reward objective reward return method MAPPO-RR, construct a complete episode simulation data storage mechanism. By identifying missile launch and hit events as key nodes, this paper propagates the reward from the time of hit back to the time of launch at the end of each simulation, thus mitigating the delayed-reward issue inherent in the prolonged decision interval between launch and impact.

Experimental results demonstrate that MAPPO-RR algorithm, with hit-reward return, achieves a stable win rate exceeding 75%, substantially higher than 43.7% achieved by the standard MAPPO, and accelerates convergence by 55%. Ablation studies further confirm the superior performance of MAPPO-RR in BVR air-combat tasks, outperforming MAPPO in both win rate and convergence speed.

References

[1] WANG X W, WANG Y H, SU X C, et al. Deep reinforcement learning-based air combat maneuver decision-making: Literature review, implementation tutorial and future direction[J]. Artificial Intelligence Review, 2023. DOI:10.1007/s10462-023-10620-2.

[2] LU B, RU L, HU S G, et al. Research on autonomous manoeuvre decision making in within-visual-range aerial two-player zero-sum games based on deep reinforcement learning[J]. Mathematics, 2024, 12(14): 2160.

[3] REN Z, ZHANG D, TANG S, et al. Cooperative maneuver decision making for multi-UAV air combat based on incomplete information dynamic game[J]. Defence Technology, 2023, 27: 308-317.

[4] WANG Y, ZHANG X W, ZHOU R, et al. Research on UCAV maneuvering decision method based on heuristic reinforcement learning[J]. Computational Intelligence and Neuroscience, 2022, 1: 1477078.

[5] JIANG Y, WANG D B, BAI T T, et al. Multi-UAV objective assignment using Hungarian fusion genetic algorithm[J]. IEEE Access, 2022, 10: 43013-43021.

[6] QIAN C X, ZHANG X B, LI L, et al. H3E: Learning air combat with a three-level hierarchical framework embedding expert knowledge[J]. Expert Systems with Applications, 2024, 245: 123084.

[7] CHEN Y X, LUO H, WANG G Q. Expert experience soft actor-critic for unmanned aerial vehicle dynamic target assignment[J]. Journal of Aerospace Information Systems, 2025, 22(5): 379-390.

[8] HU D Y, YANG R N, ZHANG Y, et al. Aerial combat maneuvering policy learning based on confrontation demonstrations and dynamic quality replay[J]. Engineering Applications of Artificial Intelligence, 2022, 111: 104767.

[9] LIU J Y, WANG G, FU Q, et al. Task assignment in ground-to-air confrontation based on multiagent deep reinforcement learning[J]. Defence Technology, 2023, 19: 210-219.

[10] LIU X X, YIN Y, SU Y Z, et al. A multi-UCAV cooperative decision-making method based on an MAPPO algorithm for beyond-visual-range air combat[J]. Aerospace, 2022, 9(10): 563.

[11] XU X J, WANG Y F, GUO X, et al. Multi-UAV air combat cooperative game based on virtual opponent and value attention decomposition policy gradient[J]. Expert Systems with Applications, 2025, 267: 126069.

[12] ZHANG J D, WANG D H, YANG Q M, et al. Loyal wingman task execution for future aerial combat: A hierarchical prior-based reinforcement learning approach[J]. Chinese Journal of Aeronautics, 2024, 37(5): 462-481.

[13] YANG M C, SHAN S Z, ZHANG W W. Decision-making and confrontation in close-range air combat based on reinforcement learning[J]. Chinese Journal of Aeronautics, 2025, 38(9): 103526.

[14] HAN H R, CHENG J, LV M L, et al. Augmenting the robustness of tactical maneuver decision-making in unmanned aerial combat vehicles during dogfights via prioritized population play with diversified partners[J]. IEEE Transactions on Aerospace and Electronic Systems, 2025, 61(5): 12892-12907.

[15] CHEN C, SONG T, MO L, et al. Scalable cooperative decision-making in multi-UAV confrontations: An attention-based multiagent actor-critic approach[J]. IEEE Transactions on Aerospace and Electronic Systems, 2025, 61(6): 15195-15209.

[16] WANG W F, RU L, LV M L, et al. Exploring hierarchical hybrid autonomous maneuvering decision-making architecture in beyond visual range air combat[J]. IEEE Transactions on Vehicular Technology, 2025, 74(10): 15491-15506.

[17] XIONG P S, LIU H, TIAN Y L, et al. Helicopter maritime search area planning based on a minimum bounding rectangle and K-means clustering[J]. Chinese Journal of Aeronautics, 2021, 34(2): 554-562.

[18] YIN Y F, GUO Y, SU Q R, et al. Task allocation of multiple unmanned aerial vehicles based on deep transfer reinforcement learning[J]. Drones, 2022, 6(8): 215.

[19] LI S W, JIA Y H, YANG F, et al. Collaborative decision-making method for multi-UAV based on multiagent reinforcement learning[J]. IEEE Access, 2022, 10: 91385-91396.

[20] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal policy optimization algorithms[EB/OL]. (2017-07-20). https://arxiv.org/abs/1707.06347.

Acknowledgement This work was supported by the National Natural Science Foundation of China (No.52272382).

Authors

The first author Ms. **JIANG Yufei** is currently pursuing the master's degree in aircraft design with Beihang University. Her current research interests include intelligent air combat and deep reinforcement learning.

The corresponding author Prof. **ZHOU Yaoming** received his Ph.D. degree from Beihang University. He is currently a professor and a Ph.D. supervisor at Beihang University. His research interests include intelligent decision-making of UAV, UAV path planning, and intelligent UAV design.

Author contributions Ms. **JIANG Yufei** designed the study, complied the models, conducted the analysis, interpreted the results, and wrote the manuscript. Mr. **SHI Hanyue** proposed innovative suggestions, participated in the

simulation modeling and design of air combat opponents work. Prof. **ZHOU Yaoming** provided guidance on the manuscript and experiments design. All authors commented on the manuscript draft and approved the submission.

Competing interests The authors declare no competing interests.

(Production Editor: WANG Jie)

基于多智能体强化学习的无人机编队超视距空战决策方法

姜雨菲，石含玥，周尧明

(北京航空航天大学航空科学与工程学院，北京 100191，中国)

摘要：随着无人机空战逐渐从近距格斗向超视距(Beyond-visual-range, BVR)作战演进，导弹发射与命中目标之间存在较长的时间间隔，导致多智能体强化学习在空战决策中面临严重的奖励稀疏与延迟奖励问题。为解决这一难题，本文提出了一种基于多智能体近端策略优化(Multi-agent proximal policy optimization, MAPPO)的改进算法。首先，引入基于专家规则的发射控制机制(MAPPO with launch constraints, MAPPO-LC)，通过约束导弹发射的距离、高度和冷却时间，限制智能体的导弹发射策略，有效缓解其在训练初期的无效探索；其次，在此基础上提出了一种带奖励回传机制(MAPPO with reward return, MAPPO-RR)的算法，该算法将导弹发射与命中目标时间作为关键节点，通过奖励重塑的方法，将最终的命中奖励按比例跨时间步回传至导弹发射时刻。仿真对抗实验表明，MAPPO-RR算法不仅能够促使智能体自主学习到先敌发射、协同制导等典型的多机协同战术，且在与专家策略的对抗中实现了超过75%的稳定胜率，其采样效率和策略收敛速度较MAPPO基线方法提升了55%。本文研究表明，将专家先验知识与奖励重塑机制结合，能够有效解决超视距空战中稀疏奖励和奖励滞后的问题，显著加速算法收敛并提升策略胜率。

关键词：超视距空战；协同决策；多智能体强化学习；延迟奖励；奖励重塑

研究亮点：

1. 提出专家规则的发射控制机制(MAPPO-LC)，创新性地将空战战术规则(距离、高度限制与冷却时间)转化为强化学习动作空间的约束，有效避免了模型在训练初期的盲目探索与资源浪费。

2. 提出一种带奖励回传机制(MAPPO-RR)的算法，旨在解决超视距空战中导弹从发射到命中过程中因时间跨度较长而导致的延迟奖励问题。该机制通过将关键节点的奖励信号回传至早期决策时刻，有效增强智能体对关键战术节点的决策学习能力，显著提升训练效率与策略收敛速度。

3. 改进后的算法收敛速度较传统基线算法提升了55%，胜率稳定在75%以上，且能够展现出协同制导，火力压制等空战战术。