IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 21 May 2024, accepted 19 June 2024, date of publication 27 June 2024, date of current version 4 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3419889

RESEARCH ARTICLE

Autonomous Air Combat Maneuver Decision-Making Based on PPO-BWDA

HONGMING WANG, ZHUANGFENG ZHOU, JUNZHE JIANG, WENQIN DENG, AND XUEYUN CHEN

School of Electrical Engineering, Guangxi University, Nanning 530004, China

Corresponding author: Xueyun Chen (20140043@gxu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 62061002.

ABSTRACT As Unmanned Combat Aerial Vehicle (UCAV) continue to play an increasingly pivotal role in modern aerial warfare, enhancing their intelligence levels is imperative for global military advancement. Despite notable progress in employing deep reinforcement learning for autonomous air combat maneuver decision-making, existing methods grapple with subpar performance, sluggish training, and susceptibility to local optima. Therefore, this paper proposes a new air combat maneuver decision algorithm based on Proximal Policy Optimization (PPO). Firstly, we establish a UCAV adversarial model and design a dual observation space. Secondly, we develop an Actor-Critic network based on Bidirectional Long Short-Term Memory (BiLSTM) and Multi-Head Self-Attention (MHSA), which better handles high-dimensional information with temporal correlations in air combat situations. Thirdly, we propose an action selection method based on Parallel Monte Carlo Tree Search with Watch the Unobserved (WU-PMCTS) to assist the algorithm in making more effective maneuver decisions. Fourthly, we design a Dynamic Reward Evaluation (DRE) method to dynamically adjust the weights of various rewards according to different adversarial situations, improving algorithm performance. Finally, we introduce an Advantage Prioritized Experience Replay (APER) to sample according to the sample advantage values, enhancing algorithm training efficiency. Experimental results from ablation and comparative experiments demonstrate the superiority of the proposed algorithm over PPO and other mainstream algorithms, with a 0.32 increase in average return and a 36% increase in win rate.

INDEX TERMS Unmanned combat aerial vehicle, deep reinforcement learning, autonomous air combat, maneuver decision-making, PPO, BiLSTM, MHSA, WU-PMCTS, DRE, APER.

I. INTRODUCTION

Unmanned Combat Aerial Vehicle (UCAV), which is drones designed for combat, represent a new era in aerial warfare [1]. In modern warfare, aerial military forces have significant advantages in intelligence, surveillance, reconnaissance, and combat operations [2]. Enhancing the intelligence of unmanned combat aircraft has become a critical pathway for nations to bolster their military capabilities and achieve military modernization. UCAV is capable of performing highly challenging flight missions that exceed the limits of human pilots, and there is a growing trend towards replacing manned fighter aircraft with these advanced unmanned systems.

Maneuver decision-making refers to the process of controlling combat aircraft to gain air superiority and pose a threat to enemy aircraft by considering maneuverability, current situations, and other relevant information [3]. In Beyond-Visual-Range (BVR) air combat, the core objective of maneuver decision-making is to expose enemy aircraft to the attack range of our air-to-air missiles as early as possible, while skillfully maneuvering to evade enemy missiles.

The associate editor coordinating the review of this manuscript and approving it for publication was Frederico Guimarães.

This aims to achieve the tactical goals of Within-Visual-Range (WVR) air combat, where maneuver decision-making focuses more on gaining a positional advantage behind the enemy and evading enemy pursuit. Particularly in high-intensity close-range dogfights, maneuver decision-making becomes critically important.

In recent years, the application of reinforcement learning methods in autonomous air combat decision-making has garnered extensive attention from scholars both domestically and internationally. Notably, Deep Reinforcement Learning (DRL) methods have achieved significant success in the field of intelligent decision-making. Reinforcement learning is a paradigm of machine learning where an agent learns strategies through continuous interaction with the environment, selecting actions that lead to optimal states and ultimately achieving decision-making goals. Deep learning, on the other hand, can discover patterns within vast amounts of data and use them for prediction and classification, combining feature representation and learning into a single algorithm. However, deep learning models perform poorly when data is insufficient. DRL combines the strengths of both reinforcement learning and deep learning, enabling the analysis of state data for autonomous decision-making. Since 2016, DRL methods have achieved a series of remarkable successes in intelligent decision-making. The AlphaGo series of algorithms [4], [5], [6] defeated human champions in the game of Go, shattering the belief that traditional artificial intelligence could not succeed in highly complex domains. AlphaFold v2 achieved groundbreaking results in protein structure prediction [7], and AlphaTensor made significant advances in fast matrix multiplication algorithms [8]. Additionally, AlphaStar [9] outperformed 99.8% of human players in StarCraft II, and OpenAI Five [10] defeated the reigning world champions in DOTA2 with a score of 2:0. Notably, in the ''AlphaDogfight Trials'' organized by the U.S. Defense Advanced Research Projects Agency in 2020, the Falco algorithm [11] achieved a decisive 5:0 victory over human pilots in 1v1 WVR air combat, demonstrating overwhelming superiority.

Due to the outstanding performance of DRL in real-time strategy games and high-dimensional decision-making tasks, DRL-based air combat maneuvering decisions have also made significant breakthroughs. In the study by [12], the Deep Q-Learning (DQN) algorithm [13] was employed for air combat intelligent agent decision-making, yielding satisfactory results. In [14], DQN was combined with Long Short-Term Memory (LSTM) neural networks [15], and Monte Carlo Tree Search (MCTS) rewards [16] were introduced for maneuver decision research. Additionally, [17] addressed the problem of UCAV covert approach in continuous state spaces using the Double DQN [18], which incorporates both a target network and a main network. In [19], the combination of LSTM and Dueling DQN [20] based on the improved Double DQN algorithm was applied to autonomous maneuvering decisions. This approach enhanced the memory function of the agent, accelerated its convergence speed, and achieved effective one-on-one confrontations with maneuvering agents. The aforementioned methods are all value function-based, requiring the estimation of action values corresponding to each action in the current state. As a result, the action space must be discretized, making it challenging to effectively handle air combat tasks in continuous action spaces.

In further research, the study in [21] employed the Actor-Critic (AC) framework for autonomous air combat decision-making, leveraging the advantages of both value-based and policy-based methods. The work in [22] utilized the Deep Deterministic Policy Gradient (DDPG) [23] method, an improvement of AC, successfully addressing the ''dimensional explosion'' problem in maneuver outputs of traditional air combat decision-making methods, and achieving continuous action outputs. To mitigate the impact of overestimation in value functions on the performance stability and convergence speed of the maneuver decision model, some scholars combined DDPG with Double DQN, creating the Twin Delayed Deep Deterministic Policy Gradient (TD3) [24] algorithm for training maneuver decision models. The study in [25] designed two experience replay buffers for TD3, one for successes and one for failures. During network updates, samples were taken from both buffers in specific proportions, improving sample utilization efficiency to some extent. These policy gradient-based methods are better suited for controlling continuous variables in air combat environments compared to value function-based methods. However, they still face challenges such as network overfitting and extended training times.

Recent studies have advanced the application of sophisticated reinforcement learning algorithms for autonomous air combat decision-making. For instance, the study in [26] utilized the Soft Actor Critic (SAC) algorithm, achieving effective tactical outcomes. Another work [27] introduced the Proximal Policy Optimization (PPO) algorithm [28] to address the challenges of continuous action spaces, learning close-combat strategies in an end-to-end manner from observational data. Additionally, [29] integrated an attention mechanism [30] into PPO, developing an attention model based on enemy threats to comprehensively account for the influence of multiple adversary aircraft. Considering the susceptibility of standard PPO algorithms, which utilize fully connected neural networks, to gradient explosion and vanishing gradients as network complexity increases, leading to potential training failures, [31] proposed a PPO algorithm combined with Long Short-Term Memory (LSTM) network. This approach significantly improved the model's convergence speed. To further enhance training sample selection and utilization efficiency, [32] introduced the FRE-PPO method, which combines final reward estimation with PPO. This method replaces the original advantage estimation function with a final reward estimation to increase training efficiency. These policy gradient-based methods generally offer superior applicability and more stable training processes

compared to value function-based approaches. However, their relatively simple network structures may be insufficient for effectively managing the complex, high-dimensional situational information inherent in air combat environments.

Furthermore, in exploring methods to train autonomous air combat decision-making models without relying on human expertise or dense rewards, literature [33] presents a maneuver decision approach based on MCTS. This method utilizes MCTS to identify the optimal action among seven fundamental maneuvers, aiming to maximize the value of the air combat advantage function. While verifying the feasibility of MCTS for maneuver decision-making, this approach operates within a discrete action space. Moreover, literature [34] introduces a maneuver decision method combining deep reinforcement learning with MCTS, employing MCTS in a continuous action space and leveraging neural network-guided self-play to enhance the capabilities of air combat agents. Additionally, literature [35] highlights the challenge of agents relying on random actions in the early stages of training, hindering their ability to gain rewards and learn effective decision-making. Consequently, the PPO-MCTS approach is proposed to address this issue. Despite the enhancements in decision-making capabilities offered by these MCTS-integrated methods, they still face challenges such as incomplete reward coverage and low training efficiency.

We have seen some progress in above researches, but challenges persist in air combat maneuver decision-making methods based on DRL:

- **Network Structure:** Most algorithms use fully connected layers in their network structure, which are inadequate for handling temporally correlated and high-dimensional information. Issues like gradient explosion and vanishing gradients are common, especially with rapidly changing air combat data, affecting the model's robustness and generalization.
- **Action Selection:** Random action sampling from a continuous space leads to ineffective decision-making in early training stages. Balancing ''exploration'' and ''exploitation'' is challenging, often resulting in local optima and increased training time, limiting the model's applicability in real-world scenarios.
- **Reward Function Design:** Fixed reward weights fail to comprehensively consider the combined impact of angle, distance, speed, and altitude, overlooking crucial features in different situations. Consequently, the learned strategies lack specificity and flexibility.
- **Experience Replay:** Uniform random sampling can lead to correlated data and uneven contributions to gradient learning, resulting in low sample utilization, prolonged training, convergence issues, and overfitting.

In view of the four problems, this paper aims to address the limitations of the aforementioned methods. We propose a novel air combat maneuver decision-making algorithm that combines PPO with a network architecture based on Bidirectional Long Short-Term Memory (BiLSTM) and Multi-head Self-Attention (MHSA) [30]. This algorithm integrates Parallel Monte Carlo Tree Search with Watch the Unobserved (WU-PMCTS) [36] and a Dynamic Reward Evaluation (DRE) method, employing Advantage Prioritized Experience Replay (APER) technology. Hereafter, we refer to the algorithm as PPO-BWDA. This paper main contributions are as follows:

1) We establish a UCAV adversarial model based on the kinematics and dynamics formulas in three-dimensional space and design a dual observation space.
2) We design a BiLSTM+MHSA structure to replace the traditional fully connected layers in the Actor-Critic network of the PPO algorithm. The BiLSTM+MHSA architecture facilitates flexible attention to key information in sequences, dynamically adjusting focal points to effectively handle highly time-correlated air combat situational data.
3) We propose an action selection method based on WU-PMCTS. Using a Gaussian distribution constructed from the mean actions output by the Actor and selecting actions using WU-PMCTS enables a better balance between ''exploration'' and ''exploitation'' during the decision-making process, resulting in more effective maneuver decisions.
4) We design a dynamic reward evaluation method, dynamically adjusting the weights of various reward functions according to different adversarial scenarios to comprehensively assess air combat situations. This dynamic reward evaluation approach allows UCAV to achieve higher rewards in response to different situations.
5) We design an advantage prioritized experience replay mechanism, determining the probability of sampling based on the size of the sample's advantage value to improve the training efficiency of the algorithm.
6) The results of ablation experiments validate the effectiveness of the proposed algorithm. Compared to PPO, the average return increased by 0.32, and the win rate increased by 36%. Comparative experiments demonstrate that the proposed algorithm's average return and win rate are significantly superior to those of other mainstream reinforcement learning algorithms, confirming its superiority.

II. RELATED WORK

A. PROXIMAL POLICY OPTIMIZATION

The Trust Region Policy Optimization (TRPO) [37] algorithm is a type of policy gradient method within the Actor-Critic framework. In policy gradient algorithms, when the network is a deep model, updating parameters along the policy gradient can lead to significant deterioration in performance if the step size is too large, thereby affecting training effectiveness. Therefore, TRPO improves policy updates by employing a monotonic maximum step size

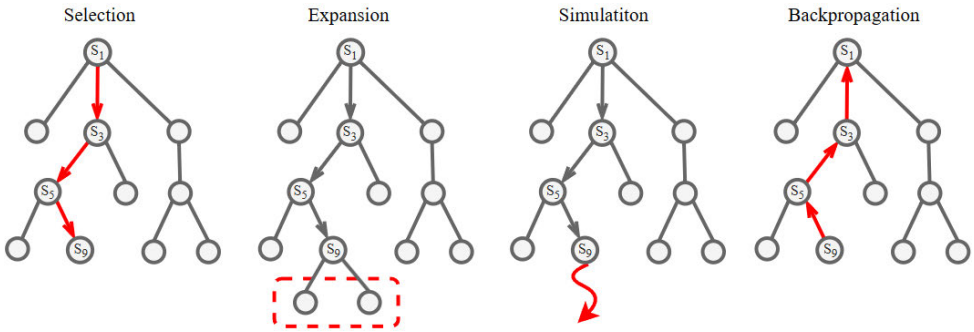


FIGURE 1. MCTS Process: Selection, Expansion, Simulation, and Backpropagation.

approach while using the KL divergence to represent a specific constraint that ensures the new policy is superior to the old policy:

$$\begin{cases} \max E_{a\sim\pi_{old}}\left[\dfrac{\pi(a|s)}{\pi_{old}(a|s)}\hat{A}(s,a)\right] \\ s.t.\overline{D}_{KL}^{\rho_{\pi_{old}}}(\pi_{old},\pi)\leq\delta \end{cases} \tag{1}$$

where $\pi_{old}(a|s)$ represents the probability of the old policy taking action a given state s, $\pi(a|s)$ represents the probability of the new policy taking action a given state s, D_{KL} denotes the KL divergence, and $\hat{A}(s,a)$ is the advantage function, expressed as:

$$\hat{A}(s,a) = (Q_{\pi_{old}}(s,a) - V_{\pi_{old}}(s)) \tag{2}$$

where $Q_{\pi_{old}}(s,a)$ represents the value obtained by taking action a in state s, and $V_{\pi_{old}}(s)$ represents the average value in state s. Therefore, the advantage function $\hat{A}(s,a)$ measures how much better or worse the current state and action are compared to the average level. The larger the value of the advantage function, the better the current state and action.

Proximal Policy Optimization (PPO) is an algorithm optimized based on TRPO. The PPO algorithm addresses the issue of excessively large updates in Policy Gradient algorithms while reducing the complexity of solving the algorithm, making it easier to implement through coding. It offers better convergence and stability compared to previous deep reinforcement learning methods. PPO can be divided into PPO-penalty and PPO-clip, with the latter being more widely used. The objective function of the PPO-clip algorithm is:

$$\mathcal{L}^{\text{CLIP}}(\theta) = \hat{\mathbb{E}}\left[\min\left(\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)}\hat{A}(s,a),\right.\right. \\ \left.\left.\text{clip}\left(\frac{\pi(a|s)}{\pi_{\text{old}}(a|s)},1-\varepsilon,1+\varepsilon\right)\hat{A}(s,a)\right)\right] \tag{3}$$

where ε is a hyperparameter. The clip function is a constraint function that ensures the ratio of the new policy to the old policy is constrained within $[1-\varepsilon, 1+\varepsilon]$. If the ratio is less than $1-\varepsilon$, it takes the value $1-\varepsilon$; if it is greater than $1+\varepsilon$, it takes the value $1+\varepsilon$. This constraint ensures that the new policy does not update beyond a predetermined range, avoiding excessive policy changes while preserving the well-performing parts of the original policy to prevent instability and divergence.

B. MONTE CARLO TREE SEARCH

MCTS is a tree search strategy that balances historical returns and future exploration. The core idea of MCTS is to construct a search tree that represents all possible actions and their outcomes in a game. In this paper, the MCTS search process consists of four stages: selection, expansion, evaluation, and backpropagation, as illustrated in Figure 1.

1) **Selection:** Starting from the root node, recursively select the optimal child node using an exploration algorithm until reaching a leaf node. The selection phase employs the UCT formula:

$$UCT = \underset{s'\in C(s)}{\arg\max}\left\{\frac{Q_{s'}}{N_{s'}} + c\sqrt{\frac{\log(N_s)}{N_{s'}}}\right\} \tag{4}$$

where $N_{s'}$ represents the number of simulations conducted on the current node, $Q_{s'}$ indicates the total score obtained from simulations on the current node. N_s denotes the number of simulations on the parent node, $C(s)$ represents the set of all child nodes of s, and c is an adjustable parameter that controls the level of exploration.

2) **Expansion:** In each search iteration, after the selection phase reaches a leaf node, if the node has not been explored, it needs to be expanded.
3) **Simulation:** Starting from the expanded node, a simulation is run until the end of the game.
4) **Backpropagation:** After obtaining the simulation result, the parent nodes are continuously updated backward.

When dealing with large search spaces, traditional MCTS incurs high computational costs due to the need for sequential execution. In 2019, Liu et al. [36] proposed a parallelization technique for MCTS called Watch the Unobserved in UCT (WU-UCT), which is conceptually similar to tree parallelization [38]. This method achieves linear speedup with minimal performance loss. The algorithm uses a virtual visit count, M_s, to balance exploration and exploitation. M_s

represents the total number of times node s has been accessed without simulating (or evaluating) the leaf nodes along that path, thus preventing redundant exploration across multiple threads during parallel execution. The exploration algorithm used by WU-UCT in the selection phase is shown in the equation below:

$$WU-UCT = \arg\max_{s' \in C(s)} \left\{ \frac{Q_{s'}}{N_{s'}} + c\sqrt{\frac{2\log(N_s + M_s)}{N_{s'} + M_{s'}}} \right\} \quad (5)$$

where $M_{s'}$ represents the number of visits to node s' along the path where the node has been searched but not yet evaluated at the leaf node. If one of the search threads accesses a node s', then the value of M for that node will be incremented accordingly. As shown in Equation(5), since M of s' increases, the exploration value calculated through the exploration equation will decrease, reducing the likelihood of the next search thread revisiting it. This effectively prevents multiple threads from parallelly accessing the same tree node. WU-UCT can greatly speed up the tree search under the premise of guaranteeing the performance of tree search as much as possible, and reasonably balance the relationship between exploration and utilization in the search process.

III. PROPOSED METHOD

A. UCAV AIR COMBAT MODEL

To accurately describe the flight trajectory characteristics and motion features of a UCAV, a detailed three-degree-of-freedom point mass dynamics model is used for the research and simulation of the UCAV's autonomous decision-making mechanism. To ensure the smooth progress of the research, this paper establishes the aircraft kinematic and dynamic model based on the following assumptions:

- Assume that the Earth's rotation and curvature have no impact on the UCAV's motion, and treat the ground coordinate system as an inertial coordinate system.
- Assume that gravitational acceleration is constant, neglecting the effects of dimensional and altitude changes on acceleration.
- Assume that atmospheric turbulence and gusts have no effect on the UCAV's motion.
- Assume that the UCAV's mass remains constant and treat it as a controllable point mass.
- Assume that the UCAV is always in a state of moment equilibrium, ignoring the process of attitude changes caused by short-term moment imbalances.
- Assume that the UCAV performs non-banked, skid-free maneuvers in three-dimensional space.

Based on these assumptions and the fundamental principles of dynamics, the dynamic model of the UCAV in the ground coordinate system [39] is established, as shown in Figure 2.

In Figure 2, v represents the current velocity direction of the UCAV, v' represents the projection of v on the xoy plane. γ is the angle between v' and v, representing the pitch angle of the UCAV, with the positive direction being the nose-up

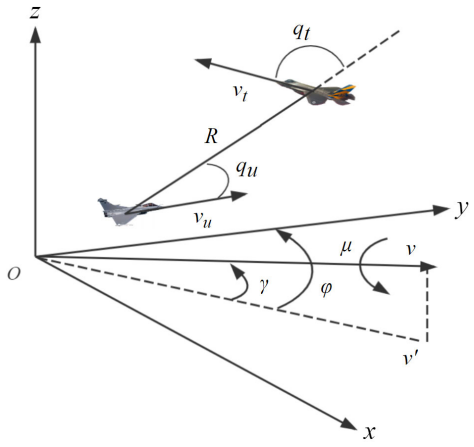


FIGURE 2. UCAV air combat model.

direction of the aircraft. φ is the angle between v' and the oy axis, representing the yaw angle of the UCAV, with the positive direction being the right yaw direction of the aircraft. μ is the angle between the body coordinate system axis and the plumb plane containing the body coordinate system's y-axis, indicating the roll angle of the UCAV in the positive direction of the airplane's tilt to the right. The kinematic equations of the UCAV can thus be derived as follows:

$$\begin{cases} \dot{x} = v\cos\gamma\cos\varphi \\ \dot{y} = v\cos\gamma\sin\varphi \\ \dot{z} = v\sin\gamma \end{cases} \quad (6)$$

where $\dot{x}$, $\dot{y}$, and $\dot{z}$ represent the rates of change of velocity along each axis, respectively. In the same coordinate system, the dynamics equations of the UCAV can be expressed as:

$$\begin{cases} \dot{v} = g(n_x - \sin\gamma) \\ \dot{\gamma} = \dfrac{g}{v}(n_y\cos\mu - \cos\gamma) \\ \dot{\varphi} = \dfrac{g}{v\cos\gamma}n_y\sin\mu \end{cases} \quad (7)$$

where g represents the gravitational acceleration under normal gravity conditions, n_x is along the direction of the aircraft's flight velocity, usually referred to as the longitudinal load factor or axial load factor, used to change the magnitude of the aircraft's flight velocity; n_y is perpendicular to the aircraft's motion plane, typically referred to as the normal load factor, used to change the direction of the aircraft's flight velocity. This dynamic equation can clearly and intuitively describe the changes in the aircraft's attitude. From Equations (6) and (7), it can be clearly inferred the relationship between the normal load factor, the axial load factor, and the roll angle around the velocity axis and the aircraft's motion.

B. RELATIVE POSITION AND OBSERVATION SPACE

During air combat decision-making, it is crucial to understand the relative positions and velocities of both the own and target aircraft, as well as their respective lead angles. The position

coordinates of the own UCAV are $P_u = \{x_u, y_u, z_u\}$, and those of the target UCAV are $P_t = \{x_t, y_t, z_t\}$. The vector R denotes the position of the own UCAV relative to the target. V_u is the velocity vector of the own aircraft, and V_t is the velocity vector of the target aircraft. V_{ut} represents the velocity of the own aircraft relative to the target. The lead angle q_u is the angle between the own aircraft's velocity vector and the relative position vector, while the lead angle q_t is the angle between the target aircraft's velocity vector and the extension of the relative position vector. The formulas for calculating q_u and q_t are as follows:

$$\begin{cases} R = P_t - P_u \\ V_{tu} = V_t - V_u \\ q_u = \arccos \dfrac{P_{tu} \cdot V_u}{\|P_{tu}\| \cdot \|V_u\|}, 0 \le q_u \le \pi \\ q_t = \arccos \dfrac{P_{tu} \cdot V_t}{\|P_{tu}\| \cdot \|V_t\|}, 0 \le q_t \le \pi \end{cases} \tag{8}$$

In typical observation space design, all situational information from both the own S_u and target S_t forces is directly used as observation data, leading to redundant information and increased difficulty in training the decision-making network. This paper introduces a dual observation space O, which combines relative situational information with individual feature information, effectively reducing the redundancy of observation data. The observation space O consists of situational information from both sides O_a and relative situational information O_r, as shown in the following equation:

$$O = \begin{cases} O_a = \{\mu_u, \gamma_u, \varphi_u, v_u, \mu_t, \gamma_t, \varphi_t, v_t\} \\ O_r = \{\Delta D, \Delta D_x, \Delta D_y, \Delta D_z, q_u, q_t\} \end{cases} \tag{9}$$

where ΔD represents the relative distance, and ΔD_x, ΔD_y, and ΔD_z are its components along the x, y, and z axes, respectively.

C. ACTOR-CRITIC NETWORK

Air combat maneuver decision-making aims to achieve the optimal solution during combat, ensuring self-safety while swiftly defeating target aircraft. This task is complex and challenging, requiring the comprehensive utilization of temporally sequenced status information acquired by aircraft and the implementation of advanced technologies and algorithms for intelligent decision-making. Therefore, to enable aircraft to fully leverage the observed temporal information during combat, we have designed an improved Actor-Critic network structure based on BiLSTM and MHSA for air combat maneuver decision-making.

1) BiLSTM

LSTM networks, a variant of Recurrent Neural Networks, address the long-term dependency issue by introducing gate units, thus mitigating gradient vanishing. BiLSTM networks integrate forward and backward LSTMs, processing sequence information in both directions. Their outputs

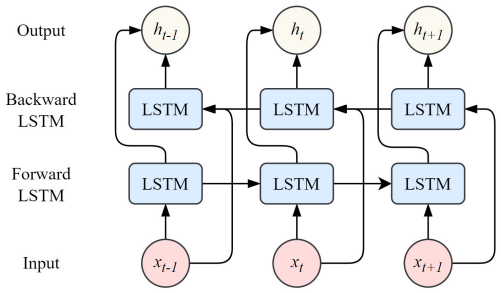


FIGURE 3. The BiLSTM deep neural network. Where x represent the input, h_t represent the output hidden state.

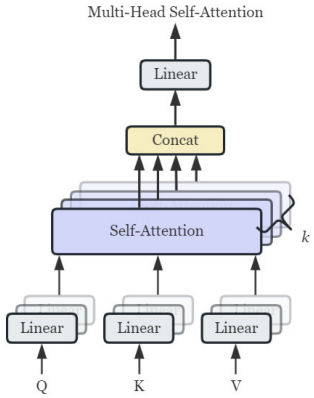


FIGURE 4. The structure of MHSA. Where matrices Q, K and V represent the query, key, and value respectively. They are obtained through different linear transformations of the same matrix.

are concatenated, effectively capturing historical sequence information while also focusing on future moments. The structural diagram is depicted in Figure 3.

Incorporating BiLSTM into the Actor-Critic network enables the network to leverage both forward and backward temporal information concurrently, thereby fostering a more holistic comprehension of variations in flight status. Such capability is paramount for capturing the swiftly evolving dynamic environment of air combat.

2) MHSA

The attention mechanism is widely used in deep learning models to selectively focus on relevant information while disregarding irrelevant details, effectively enhancing model performance. MHSA enhances the model's capability to process sequential data by introducing multiple heads on top of the self-attention mechanism. Each head learns different attention weight distributions, capturing diverse relationships and features, thereby strengthening the modeling of observation sequences. The structural diagram is depicted in Figure 4.

Incorporating MHSA into the Actor-Critic network enables adaptive attention to each observation dimension. The model dynamically adjusts attention to focus on crucial information for the current decision, enhancing flexibility. It can selectively attend to different observation dimensions at different time steps, thus improving the model's perception

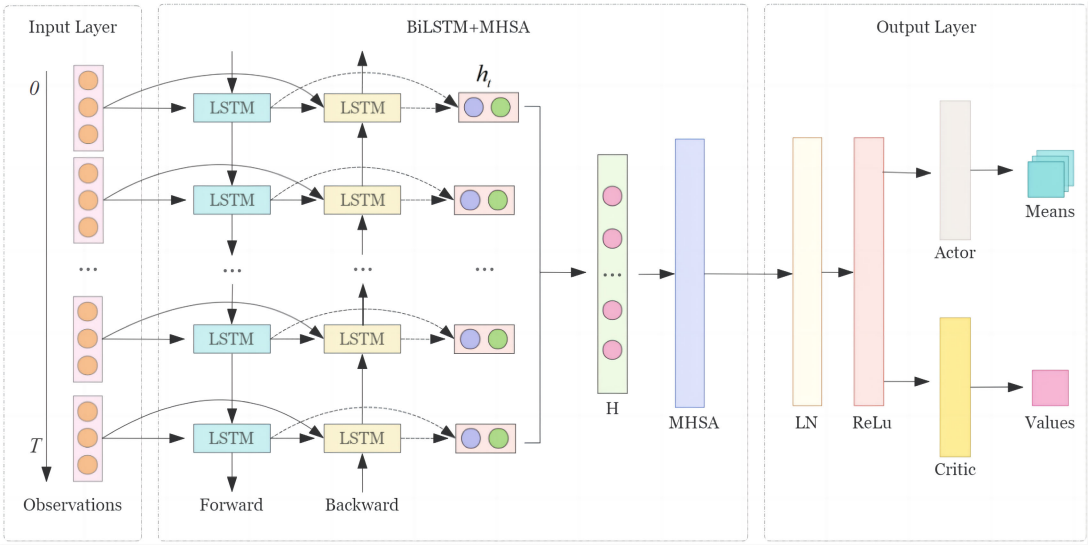


FIGURE 5. The overall structure of actor-critic network. The actor-critic network takes observations as input and outputs the mean action values and predicted state values, providing valuable guidance for WU-PMCTS in terms of both value and policy.

of dynamic battlefield scenarios. This assists the algorithm in learning and decision-making more effectively.

3) OVERALL STRUCTURE OF ACTOR-CRITIC NETWORK

The overall structure of the Actor-Critic network is illustrated in Figure 5. The network takes the observations O at the current time step as input and outputs the mean action values and the predicted values of states. Initially, the Actor-Critic network processes the observation information from time step 0 to T, followed by forward LSTM and backward LSTM processing, as shown in equation:

$$\begin{cases} \overrightarrow{h_t} = LSTM(\overrightarrow{x_t}, \overrightarrow{h_{t-1}}) \\ \overleftarrow{h_t} = LSTM(\overleftarrow{x_t}, \overleftarrow{h_{t+1}}) \\ h_t = \overrightarrow{h_t} + \overleftarrow{h_t} \end{cases} \tag{10}$$

where $\overrightarrow{x_t}$ and $\overrightarrow{h_t}$ represent the forward input and output hidden state of the LSTM at time step t, respectively, $\overleftarrow{x_t}$ and $\overleftarrow{h_t}$ represent the backward input and output hidden state of the LSTM at time step t, respectively, and h_t represents the combined output hidden state of the forward and backward LSTMs at time step t. It contains the feature information at the current time step t as well as the bidirectional information before and after time step t.

Then, concatenate the output hidden states h_t at each time step to obtain the output matrix $H = [h_0, h_1, h_2, \ldots, h_T]$. The output H is input into MHSA to compute the output of each self-attention head, and then the outputs of all heads are concatenated and linearly mapped to obtain the MHSA output H_s, as shown in the following equation:

$$\begin{cases} H^* = \text{softmax}\left(\dfrac{(HW^Q)(HW^K)^T}{\sqrt{d_k}}\right)HW^V \\ H_s = \text{Concat}(H_1{}^*, H_2{}^*, H_3{}^*, \ldots, H_k{}^*)W^O \end{cases} \tag{11}$$

where W^Q, W^K and W^V are linear mapping matrices for query, key, and value, respectively, and $\sqrt{d_k}$ is the scaling factor.

After the MHSA output, layer normalization is applied to enhance the stability of the network. Finally, the ReLU activation function and fully connected layers are used to obtain the output action means for the Actor head and the output state prediction values for the Critic head. This Actor-Critic network architecture design enables more effective capture of crucial temporal information and features in the observation sequence, such as the relative positions, velocities, and altitudes of both own and target UCAVs. It enhances the algorithm's perception and decision-making capabilities in dynamic battlefield scenarios.

D. WU-PMCTS

From the previous section, we understand that inputting the current observation into the Actor-Critic network yields the mean action and predicted value of the state as outputs. In traditional approaches, actions are directly randomly sampled from a Gaussian distribution constructed using the mean. However, this method may lead to reduced returns. Hence, we have designed a method that combines the output of the Actor-Critic network with WU-PMCTS. WU-PMCTS integrates root parallelization MCTS with WU-UCT. In this approach, actions are sampled from a Gaussian distribution formed by the mean output of the Actor network. Subsequently, these sampled actions, along with the value predicted by the Critic, guide WU-PMCTS to obtain the most valuable actions. This method is then employed for self-play training. Following this, information such as actions, rewards, and the next observation is stored in a replay buffer for subsequent training. The overall process is shown in Figure 7.

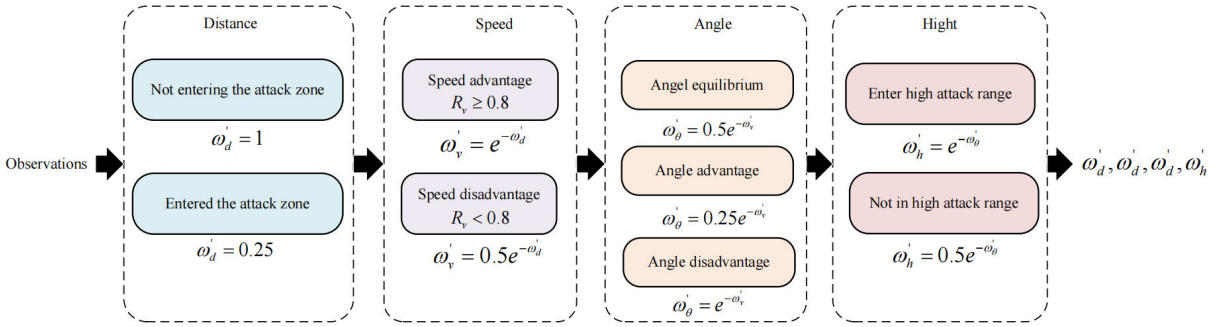


FIGURE 6. The dynamic reward evaluation method. Where ω_d', ω_v', ω_θ', and ω_h' represent dynamic reward weights.

In WU-PMCTS, each node represents a state S, and the edges of the node represent the actions taken from that state. Each node stores a set (Q, N, M, P), where N represents the number of visits to the node, Q represents the total estimated value of the node, M represents the number of virtual visits to the node, and P represents the prior probability of the node obtained from a Gaussian distribution. The tree search process is guided by the Actor and Critic networks. Starting from the initial state S_1, the observation of the state is input into the Actor network to obtain the output action values. The action values are considered as the mean of a Gaussian distribution. A Gaussian distribution is constructed by combining the action mean with a diagonal covariance matrix, and then K actions are sampled from this distribution for tree search. The state's estimated value V is directly provided by the Critic network. The exploration equation used in the selection phase of WU-PMCTS is as follows:

$$a_t = \arg\max_{s' \in C(s)} \left\{ \frac{Q_{s'}}{N_{s'}} + cP\sqrt{\frac{2\log(N_s + M_s)}{N_{s'} + M_{s'}}} \right\} \tag{12}$$

The tree search process is guided by the Actor and Critic networks. Starting from the initial state S_1, the observation value of the state is input into the Actor network to obtain the output action value. The action value is regarded as the mean of the Gaussian distribution. By constructing a Gaussian distribution with a diagonal covariance matrix and action mean, K actions are sampled from this distribution for tree search. The estimated value of the node state, V, is directly given by the Critic network.

When the tree search reaches a leaf node, for unexpanded leaf nodes, Gaussian action sampling is performed based on the action mean output by the Actor network. However, the number of sampled actions for non-root nodes is less than that for the root node, to increase exploration depth and avoid redundant simulations of similar sampled actions. During the simulation phase, the estimated value of the state is directly provided by the Critic network. When a certain number of steps are reached or the terminal state is encountered, the tree search proceeds with backpropagation to update the node's visit count N, virtual visit count M, and total estimated value Q. Finally, the action of the node with the highest value is selected as the action for policy selection. This action selection method yields higher rewards compared to random selection, thereby enhancing training performance.

Using WU-PMCTS for action selection enables the agent to achieve higher rewards compared to randomly selecting actions from a Gaussian distribution. This implies that the agent can obtain better strategies, and employing WU-PMCTS significantly accelerates the tree search speed compared to conventional MCTS while ensuring optimal tree search performance as much as possible.

E. REWARD FUNCTION

In air combat decision-making, the objective of maneuvering is to gain a favorable attack position. By using the instantaneous spatial and temporal conditions between the UCAV and the target as reward signals, and constructing corresponding air combat reward functions to evaluate the UCAV's positional advantages, the decision system can select the appropriate maneuvers, thus enhancing the UCAV's combat superiority. Traditional air combat environment rewards typically include dense rewards such as distance, speed, and angle rewards. The comprehensive air combat reward value is derived through fixed weighting of these rewards, or by using sparse terminal or event rewards. However, these reward evaluation methods fail to fully and accurately reflect the UCAV's confrontation situation and the current overall air combat status.

To better address different confrontation scenarios, we have developed a dynamic reward evaluation (DRE) method. This approach enables UCAV to learn and make decisions more effectively based on the rewards provided by the environment.

1) DISTANCE REWARD

Relative distance is a crucial parameter in air combat. During confrontations, if the distance between the two parties is too great, the probability of a successful weapon engagement decreases, while if the distance is too close, safety issues arise. Therefore, within the safe operational constraints, there exists an optimal attack range $[d_{\min}, d_{\max}]$. For our aircraft, the distance reward function can be formulated as shown

in Equation:

$$R_d = \begin{cases} e^{-\alpha(\frac{d-d_{\max}}{d_{\max}})}, d > d_{\max} \\ 1, d_{\min} \le d \le d_{\max} \\ e^{-\beta(\frac{d_{\min}-d}{d_{\min}})}, d < d_{\min} \\ 0, d \le d_s \end{cases} \tag{13}$$

where d represents the scalar distance between the two aircraft, $d_{\max}$ is the upper limit of the attack range, $d_{\min}$ is the lower limit of the attack range, and α and β are adjustable penalty coefficients. When the distance between our aircraft and the target aircraft is within the attack range, the distance advantage is maximized. When the distance between the two aircraft is outside the attack range, it is necessary to quickly move closer to the attack range.

2) VELOCITY REWARD

In air combat, speed is also an important factor affecting the battle situation. Higher speed grants the aircraft more initiative, allowing it to enter the attack range more quickly during an attack and making it easier to approach or evade the opponent. However, if the speed is too high, the aircraft's maneuverability will be greatly reduced. In actual close-range air combat, our optimal speed will vary in real time based on the target's speed and relative distance. Therefore, the desired speed v^* is introduced and defined as:

$$v^* = \begin{cases} v_t, d_{\min} \le d \le d_{\max} \\ v_t + (v_{\max} - v_t)(1 - e^{-(\frac{d-d_{\max}}{d_{\max}})}), d > d_{\max} \\ v_t + (v_{\max} - v_t)(1 - e^{-(\frac{d_{\min}-d}{d_{\min}})}), d < d_{\min} \end{cases} \tag{14}$$

where $v_{\max}$ is the maximum scalar speed of the UCAV, and v_t is the scalar speed of the target aircraft. For our aircraft, when the distance between our aircraft and the target aircraft is greater than or less than the attack range $[d_{\min}, d_{\max}]$, our aircraft should accelerate to shorten the distance to the opponent. The desired speed v^* lies between v_t and $v_{\max}$, and its actual value is determined by the distance d. When the aircraft is within the attack range $[d_{\min}, d_{\max}]$, v^* equals the target speed v_t to maintain the distance advantage. The ideal speed is always greater than or equal to the opponent's speed to maintain a relative advantage. Therefore, the speed scalar reward function can be defined as the ratio between the actual speed of the aircraft and the ideal speed:

$$R_v = \frac{v_u}{v^*} e^{\frac{-2|v^*-v_u|}{v^*}} \tag{15}$$

where v_u represents the scalar speed of our aircraft. The smaller the deviation between the speed of our aircraft and the ideal speed, the greater the speed advantage. This means that the UCAV can reach the attack range more quickly and form favorable attack conditions in a shorter amount of time. When $v_u = v_t$, the speed advantage is maximized.

3) ANGLE REWARD

In air combat, an aircraft is in an advantageous position when it is chasing another aircraft, in a disadvantageous position when it is being chased by another aircraft, and in a balanced state when both aircraft are flying in opposite or same directions. Therefore, this paper calculates the angle reward function based on the lead angle between the two aircraft as shown in the following equation:

$$R_\theta = \frac{2\pi - q_u - q_t}{2\pi} \tag{16}$$

From the equation above, it can be observed that the smaller the lead angle of our aircraft, the greater our attack advantage, while the smaller the lead angle of the target aircraft relative to ours, the greater the attack disadvantage of the target. When $q_u = q_t = 0$, the angle advantage of our aircraft reaches its maximum at 1, forming a chasing situation from our side to the target. Conversely, when $q_u = q_t = \pi$, the angle advantage of the target reaches its maximum, and our angle advantage is 0, indicating a chasing situation from the target to us. Additionally, based on the lead angles of the two aircraft in the air combat process, the relative posture situations between the two aircraft can be simplified into three categories: relative advantage, relative disadvantage, and mutual balance, as shown in Table 1 below.

TABLE 1. The relative posture situations own and target UCAVs.

Our Lead Angle q_u	Target's Lead Angle q_t	Situational Relationship
$0° \le q_u < 90°$	$0° \le q_t < 90°$	Our UCAV has advantage
$0° \le q_u < 90°$	$90° < q_t \le 180°$	Balanced situation
$90° \le q_u \le 180°$	$0° \le q_t \le 90°$	Balanced situation
$90° \le q_u \le 180°$	$90° < q_t \le 180°$	Target's UCAV has advantage

4) ALTITUDE REWARD

In air combat, when the relative flight altitude is higher than the target's, the UCAV gains potential energy advantage. Aircraft positioned at higher altitudes can seize better attacking opportunities and utilize altitude superiority for diving, rapidly entering the attack zone, or withdrawing from combat. Therefore, in practical air combat, the optimal attack altitude range is delineated, and the corresponding altitude reward function is defined as follows:

$$R_h = \begin{cases} 1, h_{\min} < \Delta h < h_{\max} \\ e^{-\frac{(\Delta h - h_{\min})^2}{2(h_{\max}-h_{\min})^2}}, \Delta h < h_{\min} \\ e^{-\frac{(\Delta h - h_{\max})^2}{2(h_{\max}-h_{\min})^2}}, \Delta h > h_{\max} \end{cases} \tag{17}$$

The formula represents the altitude difference between our aircraft and the target aircraft, where $h_{\min}$ and $h_{\max}$ denote the upper and lower bounds of the optimal attack altitude range, respectively. When our UCAV is relatively higher than the target and the altitude difference falls within the optimal attack altitude range, it maximizes its air combat advantage. However, if the altitude difference is too large or too small, the aircraft needs to adjust its altitude towards the attack area.

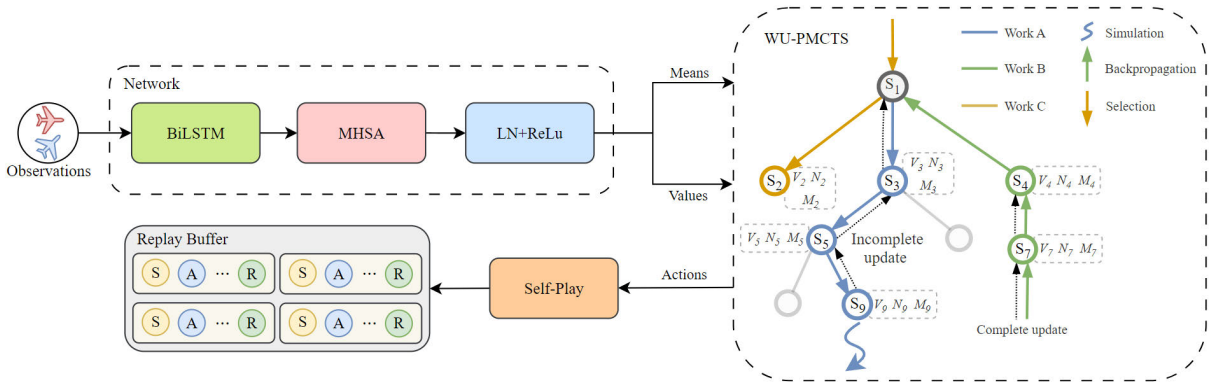


FIGURE 7. The overall process of action selection using WU-PMCTS. In WU-PMCTS, branches of different colors represent parallel tree searches.

5) DYNAMIC REWARD EVALUATION

The DRE method adjusts reward weights in real-time according to changes in the battlefield situation, enabling the UCAV to gain a more comprehensive understanding of the battlefield environment and make more precise maneuver decisions. DRE subdivides different dynamic weighting coefficients for each reward function based on 24 types of combat situations. The dynamic weighting coefficient of the previous level will affect the dynamic weighting coefficient of the next level, with a larger coefficient at the previous level resulting in faster decay of the subsequent dynamic weighting coefficients. The DRE method is illustrated in Figure 6.

In air combat, UCAV must first determine whether they have entered the attack zone. If a UCAV has not entered the attack zone, it should prioritize quickly reaching this zone. Therefore, the dynamic weight of the distance reward ω'_d is determined by whether the aircraft is in the attack zone. The basis for a UCAV's rapid entry into the attack zone is its speed advantage. If the speed advantage is significant (greater than 0.8), the UCAV can quickly enter the attack zone, thus the dynamic weight of the speed reward ω'_v is influenced by the speed advantage. Once the UCAV has entered the attack zone and maintained a speed advantage, it can focus more on adjusting its angle and altitude to gain a superior position. The dynamic weight of the angle reward ω'_θ is determined by the lead angles of both sides and can be categorized into mutual parity, our advantage, and our disadvantage. The dynamic weight of the altitude reward ω'_h is based on whether the UCAV is within the optimal attack altitude range.

After selecting four dynamic reward weights ω'_d, ω'_v, ω'_θ, and ω'_h, normalization is performed to obtain the actual reward weights ω_d, ω_v, ω_θ, and ω_h, as shown in Equation:

$$[\omega_d, \omega_\theta, \omega_v, \omega_h] = \frac{[\omega'_d, \omega'_\theta, \omega'_v, \omega'_h]}{\omega'_d + \omega'_\theta + \omega'_v + \omega'_h} \tag{18}$$

and we have:

$$\omega_d + \omega_\theta + \omega_v + \omega_h = 1 \tag{19}$$

Therefore, the comprehensive reward evaluation value is:

$$R_{total} = \omega_d R_d + \omega_\theta R_\theta + \omega_v R_v + \omega_h R_h \tag{20}$$

From the above Equation (20), it can be seen that when the comprehensive reward evaluation value is large, the UCAV is in an advantageous position and can better attack the target. Conversely, when the comprehensive reward evaluation value is low, the UCAV is in a disadvantageous position and is more likely to be attacked by the target. The DRE method mentioned above can dynamically adjust the reward function weights according to different combat situations. This helps the UCAV make better maneuvering decisions during training based on the current situation and avoids issues where the UCAV might fall into local optima or experience inefficient training due to incomplete considerations.

F. ADVANTAGE PRIORITIZED EXPERIENCE REPLAY

In reinforcement learning algorithms, after obtaining training samples, selecting an appropriate number of samples from these samples for network training is a crucial step. Traditional sampling methods usually rely solely on random uniform sampling. However, to improve training efficiency and effectiveness, we have designed a more refined method called APER for sample selection. This method leverages the property of the advantage function in reinforcement learning, which measures the superiority of taking a certain action in a given state relative to the average level.

Specifically, for each batch of samples, we can calculate their corresponding advantage values. Samples with higher advantage values indicate that they contribute more to policy improvement compared to other samples. Therefore, these samples should be prioritized for sampling during training. The equation for sample selection is as follows:

$$P(i) = \frac{(e^{A(i)})^\rho}{\sum_{j=1}^{N} (e^{A(j)})^\rho} \tag{21}$$

where $A(i)$ represents the advantage value of the i-th sample, ρ indicating the magnitude of the advantage function's

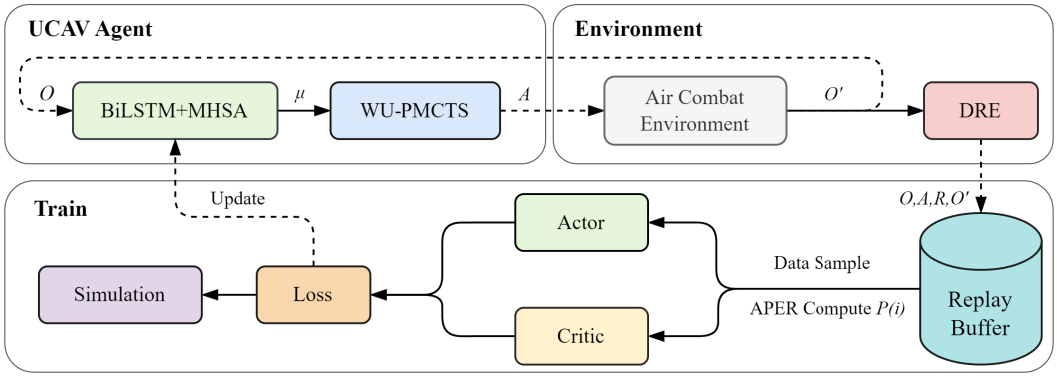


FIGURE 8. The overall flowchart of PPO-BWDA algorithm.

influence in the sampling probability, $P(i)$ represents the probability of the i-th sample being sampled by dividing the advantage value of the i-th sample by the total advantage value. When $\rho = 0$, uniform sampling of samples is performed, and when $\rho = 1$, sampling is done entirely according to the advantage function probability value. The APER method is based on the advantage function and adjusts the sampling probability to focus more on samples with higher advantage values. The probability of each sample being selected can be calculated by the following formula. This sampling method can effectively increase the utilization of samples, ensure that the network focuses more on samples that play an important role in improving the policy during training, thereby accelerating training convergence and improving the overall performance of the policy. The overall train loop and algorithm flow of our PPO-BWDA algorithm is shown in the figure 8 and Algorithm 1.

IV. EXPERIMENT AND ANALYSIS

To validate the performance improvements of our proposed PPO-BWDA algorithm compared to the baseline PPO and other mainstream algorithms, we first describe the experimental setup and evaluation methods in this section. Following this, we conduct ablation and simulation experiments to demonstrate the advantages of the various improvements introduced in this paper. Finally, we perform comparative and simulation experiments to establish the superiority of the proposed algorithm over current mainstream algorithms.

A. EXPERIMENTAL SETTINGS

During each round of adversarial training, the initial positions, speeds, and angles of our UCAV and the target aircraft are randomly initialized from the ranges specified in Table 2. The total number of training epochs for the agent is set to 5000. The Replay Buffer stores a batch size of 2048 episodes, with a mini_batch size of 64 episodes. Each batch size undergoes 10 samplings of mini_batches, and each mini_batch involves 5 network updates. The simulation time step is set to 0.1s, meaning both sides select their actions simultaneously every 0.1s. The maximum number of decisions per confrontation is 1250. In the algorithm, the clipping parameter is set to 0.2, the discount factor is 0.99, and the initial learning rate is 3×10^{-4}, decaying to 0 by the end of the training.

In air combat scenarios, we need to define the conditions for the end of a round and determine the outcome. Taking our aircraft as an example, our goal is to occupy a favorable attack angle and maneuver to the target's rear, ensuring that the target falls within our attack range and finds it difficult to escape quickly. Simultaneously, we must avoid falling into the target's attack range.

Specifically, the victory conditions can be quantified using the following criteria:

- $q_u \leq 30°$ and $q_t \leq 60°$
- $d_{\min} \leq d \leq d_{\max}$
- $h_{\min} \leq h \leq h_{\max}$
- $v_u = v^*$

If these conditions are met, it is recorded as a victory for our UCAV. Conversely, if the target UCAV meets its respective conditions, it is recorded as a victory for the target. If the maximum number of decision steps is reached without a clear victory, the round is recorded as a draw.

B. ABLATION EXPERIMENT

In the ablation experiments, our UCAV adopted four different algorithms for maneuver decision-making: PPO, PPO-BW, PPO-BWD, and PPO-BWDA. The target UCAV used a pre-trained PPO algorithm for maneuver decision-making. Specifically, PPO-BW represents the PPO algorithm enhanced with a BiLSTM + MHSA Actor-Critic network structure and WU-PMCTS for action selection. PPO-BWD includes the additions of DRE on top of PPO-BW, while PPO-BWDA further incorporates APER.

1) EXPERIMENT EVALUATION

In Figure 9(a), we focus on the average return obtained by the agent during the training process. The solid line represents the mean of the average return, while the shaded area represents the standard deviation. In air combat maneuver

Algorithm 1 PPO-BWDA Algorithm

Input: Initialize Actor network parameters θ^0, Critic network parameters ω^0, hyperparameters, and environment (with dual posture observations, Replay Buffer, DRE and APER)
Output: Actor network parameters θ, Critic network parameters ω
while *Epochs* < E **do**
  Reset environment get ovservations O_t;
  for *steps* = 1 **to** T **do**
    Inputs O_t to Actor network to get mean values μ;
    Construct Gaussian distribution with mean and covariance matrix output by μ;

$$N(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{(} - \frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu))$$

    Use WU-PMCTS to select our agent actions $a_{t,o}$ from $N(x \mid \mu, \Sigma)$;

$$a_t = \arg\max_{s' \in C(s)} \left\{ \frac{Q_{s'}}{N_{s'}} + cP\sqrt{\frac{2\log(N_s + M_s)}{N_{s'} + M_{s'}}} \right\}$$

    Get target UCAV actions $a_{t,t}$ by O_t;
    Execute $(a_{t,u}, a_{t,t})$ in environment to get DRE reward r_t, next observation O_{t+1} and *done*;

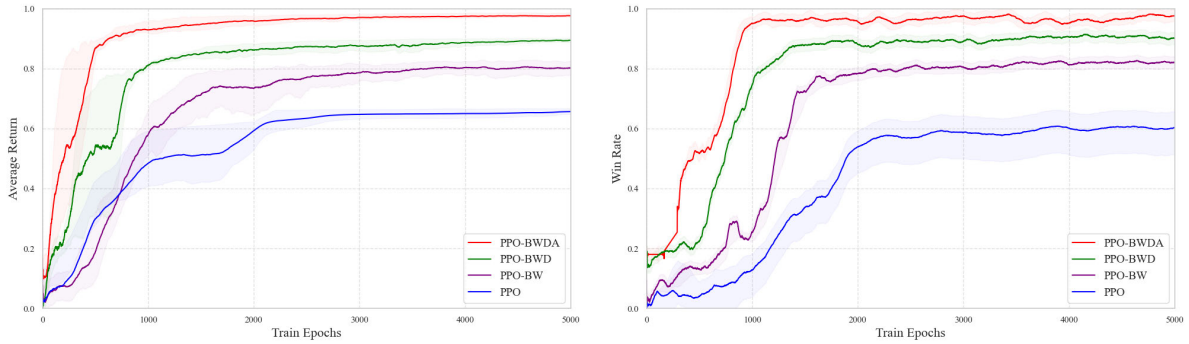
$$r_t = \omega_d R_d + \omega_\theta R_\theta + \omega_v R_v + \omega_h R_h$$

    Store $(O_t, a_{t,u}, r_t, O_{t+1})$ in Replay Buffer;
    if *done* $\neq 0$ **then**
      break;
    if *Replay Buffer size* $\geq$ *batch size* **then**
      for *k_epochs* = 1 **to** K **do**
        Epochs += 1;
        Calculate the advantage values A_t for each sample;
        Sample mini batch $(O_t, a_{t,u}, r_t, O_{t+1})$ using APER;
        for *update_per* = 1 **to** N **do**
          Compute the surrogate objective for the sample:

$$L(\theta) = \min(r_t(\theta) \cdot A_t, clip(, 1-\varepsilon, 1+\varepsilon)A_t), r_t(\theta) = \frac{\pi(a_t|O_t)}{\pi_{old}(a_t|O_t)}$$

          Update Actor network parameters: $\theta \leftarrow \theta + \alpha \nabla_\theta L(\theta)$
          Update Critic function parameters: $\omega \leftarrow \omega - \beta \nabla_\omega \left(V_\omega(s_t) - (r_t + \gamma V_\omega(s_{t+1}))\right)^2$
      Clear Replay Buffer;

decision-making, the value of the average return reflects whether the agent has learned reasonable and effective strategies. The experimental results show that the original PPO algorithm takes up to 2000 epochs to converge, with a convergence value of 0.66. Additionally, at the early stages of training, the PPO algorithm exhibits random decision-making, causing significant fluctuations in the standard deviation of the shaded area. In contrast, PPO-BW shows a significant improvement over the original PPO algorithm in both convergence speed and convergence value, converging after 1500 epochs with an average return of 0.81. This improvement is due to PPO-BW leveraging the improved BiLSTM + MHSA network structure and the WU-UCT based PMCTS action selection mechanism, enabling the agent to learn and choose appropriate maneuver strategies more accurately, thereby accelerating the training process and increasing the average return value. Furthermore, PPO-BWD introduces DRE on top of PPO-BW, allowing the agent to more comprehensively evaluate the current battlefield situation and adjust its decision-making strategy accordingly. As a result, PPO-BWD shows a more significant improvement in both convergence speed and convergence value, converging after 1200 epochs with an average return of 0.94, further proving the importance of dynamic situation assessment for air combat maneuver decision-making. Finally, PPO-BWDA incorporates APER, which further optimizes the agent's learning process. During training, it demonstrated the highest convergence speed and value. This proves that APER allows



(a) Average return of ablation experiments

(b) Win rate of ablation experiments

FIGURE 9. Ablation experiments results.

TABLE 2. Experimental parameters.

Parameter	Value	Parameter	Value
x	[0, 30] km	γ	$[-\pi/4, \pi/4]$
y	[0, 30] km	φ	$[-\pi, \pi]$
z	[0, 30] km	$d_{\max}$	2 km
v	[150, 400] m/s	$d_{\min}$	1 km
n_x	[-2, 2]	$h_{\max}$	1 km
n_y	[-6, 6]	$h_{\min}$	0.1 km
μ	$[-\pi, \pi]$	g	9.81 m/s^2

the agent to utilize the data from the experience replay buffer more effectively, accelerating the training process and increasing the average return. PPO-BWDA converged after 1000 training episodes, achieving an average return of 0.98, which is 0.32 higher than the final convergence value of the PPO algorithm.

Moreover, to comprehensively evaluate the effectiveness of the improved algorithms in solving maneuver decision-making problems, the changes in the win rate recorded during training are shown in Figure 9(b), which is the number of wins per epoch divided by the total number of epochs. As the number of training iterations increases, the win rate of the improved PPO-BW, PPO-BWD, and PPO-BWDA rises faster and reaches higher values compared to the PPO algorithm, which is consistent with the trend in the average return. This phenomenon further verifies that our improved techniques enable the algorithm to learn more reasonable and effective maneuver strategies during training, and the decision-making ability of the agent gradually enhances. Ultimately, the improved PPO-BW, PPO-BWD, and PPO-BWDA algorithms achieve win rates of 82%, 90%, and 97%, respectively, while the final win rate of PPO is only 61%. This indicates that agents trained with the improved algorithms can more effectively handle different situations in confrontations, making more optimized decisions and achieving more victories.

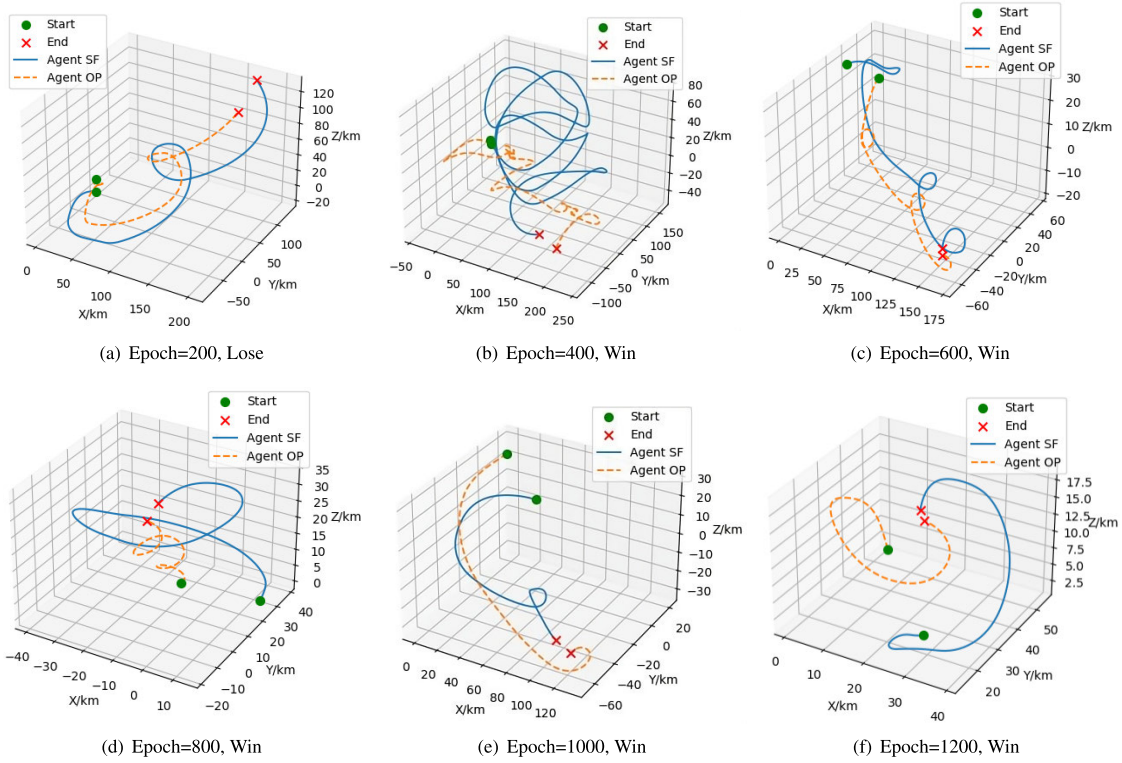
2) SIMULATION EXPERIMENT

To further demonstrate the superiority of our proposed algorithm over the benchmark algorithm PPO in learning decision-making for space station maneuvers.These evaluations compared agents trained using the PPO-BWDA and PPO algorithms over epochs ranging from 0 to 1200, as shown in Figures 10. In these simulations, Agent SF represents our UCAV, and Agent OP represents the target's UCAV. The green circles denote the starting positions of the UCAV, while the red crosses indicate their ending positions. In Figures 10(a), (b) and 11(a), (b), in the early stages of training (at epochs 200 and 400), the UCAV guided by the PPO algorithm exhibited actions akin to random exploration, indicating a lack of learned effective attack strategies. In contrast, the UCAV guided by the PPO-BWDA algorithm was already engaging in mutual entanglement with the target UCAV, although it had not yet overcome the opponent.

As training progressed, as depicted in Figures 10(c), (d) and 11(c), (d) at epochs 600 and 800, the UCAV trained with the PPO-BWDA algorithm began to learn attack strategies and adjust its posture. It attempted to maneuver behind the target UCAV to gain a positional advantage, although the chosen attack routes were suboptimal and often led to futile circling. Conversely, the UCAV trained with the PPO algorithm was only starting to learn how to engage with the target UCAV.

By the time training reached epochs 1000 and 1200, as shown in Figures 10(e), (f) and 11(e), (f), the UCAV trained with the PPO-BWDA algorithm demonstrated a more effective selection of attack strategies based on the target UCAV's position. It swiftly occupied favorable positions and easily defeated the enemy agent. Meanwhile, the UCAV trained with the PPO algorithm, although able to engage the target UCAV, still failed to secure victory.

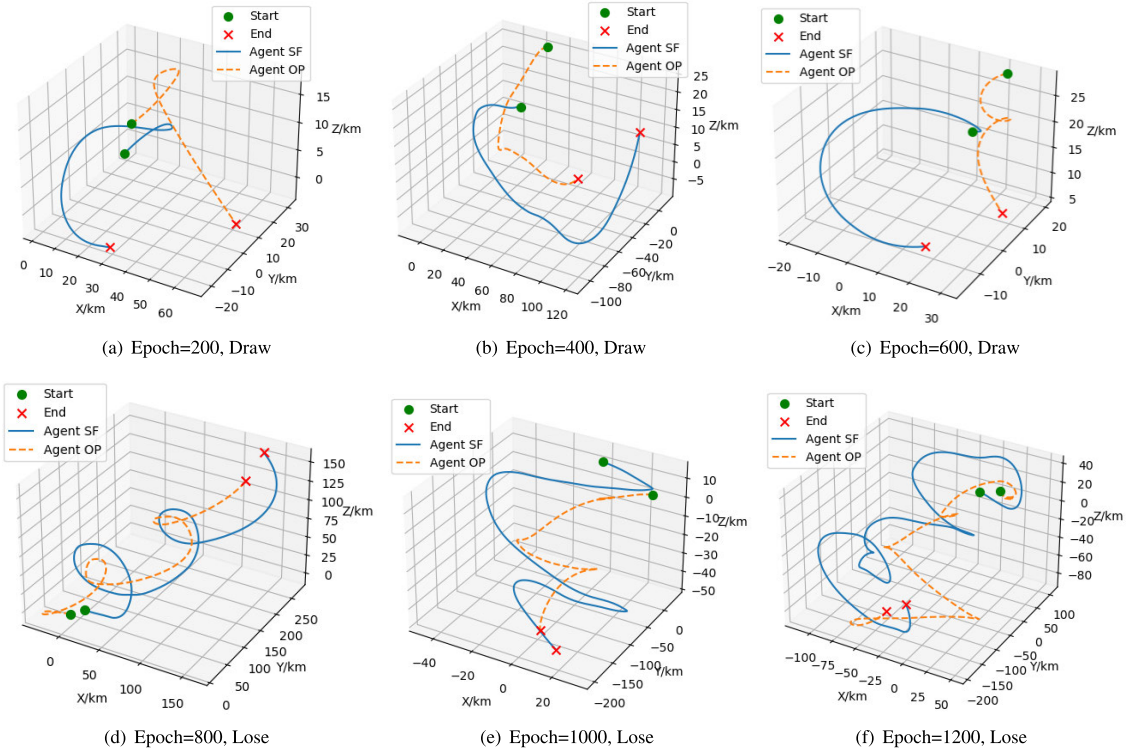
The results of the ablation experiments and simulation adversarial experiments demonstrate the effectiveness of our



(a) Epoch=200, Lose (b) Epoch=400, Win (c) Epoch=600, Win

(d) Epoch=800, Win (e) Epoch=1000, Win (f) Epoch=1200, Win

FIGURE 10. The simulation results of PPO-BWDA over 0-1200 epochs.

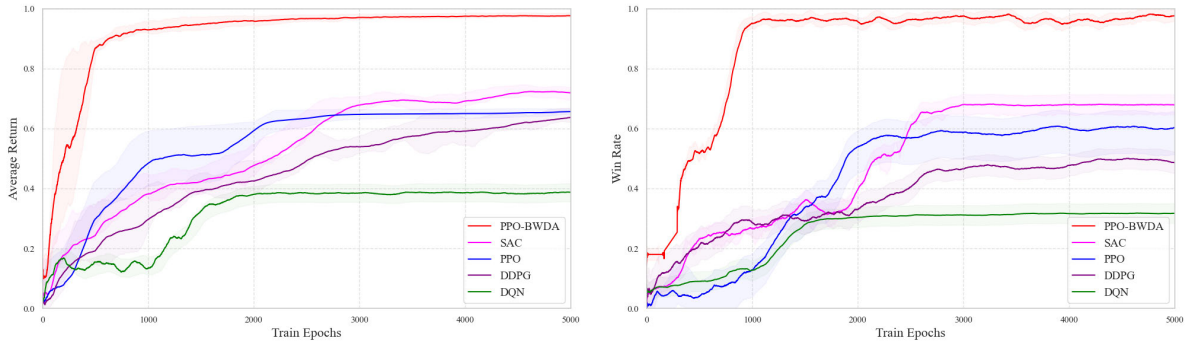


(a) Epoch=200, Draw (b) Epoch=400, Draw (c) Epoch=600, Draw

(d) Epoch=800, Lose (e) Epoch=1000, Lose (f) Epoch=1200, Lose

FIGURE 11. The simulation results of PPO over 0-1200 epochs.

improvement techniques in enhancing the decision-making capability of the algorithm and accelerating the convergence of the training process. The PPO-BWDA algorithm benefits from the introduction of an improved Actor-Critic



(a) Average return of comparison experiments

(b) win rate of comparison experiments

FIGURE 12. Comparison experiments results.

network structure, WU-PMCTS, DRE, and APER, each contributing to its performance to varying degrees. These advancements are significant for improving the autonomous decision-making capabilities and combat efficiency of air combat systems. Additionally, they provide new insights and methods for research in related fields.

C. COMPARISON EXPERIMENT

1) EXPERIMENT EVALUATION

In the comparative experiments, we introduced state-of-the-art algorithms widely used in air combat decision-making, namely SAC, DDPG, and DQN, into our air combat environment and evaluated their performance. The average return and win rate of the agents were recorded and are shown in Figure 12.

The DQN algorithm performed the worst, with both the Average Return and win rate curves growing slowly throughout the training process, eventually stabilizing at 0.39 and 32%, respectively. This indicates that DQN is not effective in learning complex air combat tasks.

The DDPG algorithm showed a slow increase in both Average Return and win rate curves before 1000 epochs, with a significant improvement between 1000 and 1500 epochs, eventually stabilizing at 0.63 and 49%, respectively. Although there was some improvement over DQN, it was still suboptimal, achieving an Average Return of 0.63.

For the SAC algorithm, the average return and win rate curves stabilized at 0.72 and 68%, respectively, slightly higher than PPO but still not reaching the level of PPO-BWDA. In contrast, our PPO-BWDA algorithm demonstrated the best performance, with both average return and win rate rising rapidly from the start. After approximately 5000 training epochs, PPO-BWDA exhibited significant advantages in both convergence speed and final convergence values. The PPO-BWDA algorithm outperformed the compared DDPG, SAC, and DQN algorithms in terms of both convergence speed and final values.

The experiments results further validate the superiority of our proposed PPO-BWDA in air combat maneuver decision-making. The performance results of ablation and comparison experiments are illustrated in table 3.

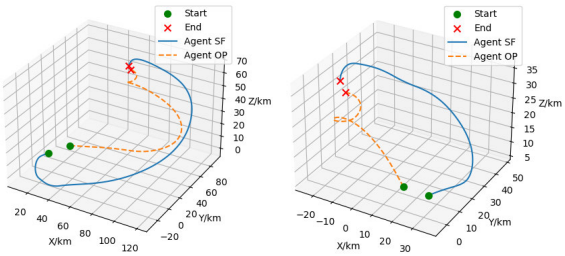
TABLE 3. Performance comparison of different algorithms.

Algorithm Name	Average Return	Win Ratios	Lose Ratios	Draw Ratios
DQN	0.39	32%	59%	9%
DDPG	0.63	49%	42%	9%
SAC	0.72	68%	35%	7%
PPO	0.66	61%	33%	6%
PPO-BW	0.81	82%	13%	5%
PPO-BWD	0.94	90%	7%	3%
PPO-BWDA	0.98	97%	2%	1%

2) SIMULATION EXPERIMENT

To further validate the advantages of our algorithm in air combat maneuver decision-making, we conducted simulation adversarial experiments using the improved PPO-BWDA algorithm against PPO, SAC, DDPG, and DQN. As shown in Figure 13, in (a) and (b), the agent trained with our algorithm quickly devised attack strategies and rapidly approached adversary agents trained by DQN and PPO, respectively. After gaining advantages in speed, altitude, and distance, our agent adjusted its posture to defeat the adversary. During confrontations with the SAC algorithm and the DDPG algorithm in (c) and (d), our agent employed different strategies to counter the actions of the target agents, ultimately defeating them. The results of these simulation adversarial experiments against PPO, SAC, DDPG, and DQN demonstrate that our improved PPO-BWDA algorithm not only exhibits excellent performance during training but also shows significant advantages over other algorithms in head-to-head confrontations, verifying the effectiveness and reliability of our approach.

The results of the comparison experiments and simulation adversarial experiments demonstrate the effectiveness and superiority of our proposed PPO-BWDA algorithm



(a) PPO-BWDA vs DQN (b) PPO-BWDA vs PPO

(c) PPO-BWDA vs DDPG (d) PPO-BWDA vs SAC

FIGURE 13. Simulation results with different algorithm.

in addressing air combat decision-making problems. The improvements show rapid convergence and higher final convergence values, providing strong support for enhancing the autonomous decision-making capabilities of air combat systems.

V. CONCLUSION

This study addresses the limitations of current deep reinforcement learning methods in air combat maneuver decision-making by proposing an improved approach, the PPO-BWDA algorithm based on PPO. Firstly, we established a UCAV adversarial framework, which includes an aircraft motion model, dynamic equations, and dual observation spaces. We then introduced our designed BiLSTM + MHSA network structure to enable the algorithm to more effectively handle the observed situational information in air combat. Additionally, we proposed an action selection method based on WU-PMCTS, allowing the algorithm to choose more optimal actions. Furthermore, we used our designed DRE to provide more comprehensive situational rewards for different adversarial scenarios, and implemented APER to enhance training efficiency through sample selection.

Finally, ablation experiments demonstrated that each improvement in our proposed PPO-BWDA algorithm contributed to performance enhancements to varying degrees. The final convergence values for Average Return and win rate were 0.32 and 36% higher than those of PPO, respectively. Comparative experiments effectively demonstrated the superiority of our proposed algorithm over current mainstream algorithms in terms of decision-making capability and training convergence speed. This indicates that our approach can provide significant technical support for applying deep reinforcement learning in autonomous air combat decision-making.

However, our algorithm also has some limitations. For instance, it is more time-consuming compared to some traditional methods, requiring substantial time to train the agent. Moreover, its training effectiveness is less satisfactory in more complex environments involving missile engagements or multi-agent confrontations. These are areas for future improvement of the PPO-BWDA algorithm. Future work will further explore the applicability of the algorithm in broader air combat scenarios and conduct more in-depth optimizations and validations in line with actual combat requirements.

REFERENCES

[1] V. Chamola, P. Kotesh, A. Agarwal, N. Gupta, and M. Guizani, "A comprehensive review of unmanned aerial vehicle attacks and neutralization techniques," *Ad Hoc Netw.*, vol. 111, Feb. 2021, Art. no. 102324.

[2] C. Hao, H. Jian, L. Quan, Z. Sihang, and Z. Zhongjie, "Research progress and prospects of autonomous air combat maneuver decision technology," *Control Theory Appl./Kongzhi Lilun Yu Yinyong*, vol. 40, no. 12, 2023.

[3] X. Lei, D. Shilin, T. Shangqin, H. Changqiang, D. Kangsheng, and Z. Zhuoran, "Beyond visual range maneuver intention recognition based on attention enhanced tuna swarm optimization parallel BiGRU," *Complex Intell. Syst.*, vol. 10, no. 2, pp. 2151–2172, Apr. 2024.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[5] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.

[6] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, Dec. 2018.

[7] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, and A. Bridgland, "Highly accurate protein structure prediction with AlphaFold," *Nature*, vol. 596, no. 7873, pp. 583–589, Aug. 2021.

[8] A. Fawzi, M. Balog, A. Huang, T. Hubert, B. Romera-Paredes, M. Barekatain, A. Novikov, F. J. R. Ruiz, J. Schrittwieser, G. Swirszcz, D. Silver, D. Hassabis, and P. Kohli, "Discovering faster matrix multiplication algorithms with reinforcement learning," *Nature*, vol. 610, no. 7930, pp. 47–53, Oct. 2022.

[9] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, and J. Oh, "Grandmaster level in StarCraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, Nov. 2019.

[10] OpenAI et al., "Dota 2 with large scale deep reinforcement learning," 2019, *arXiv:1912.06680*.

[11] A. P. Pope, J. S. Ide, D. Micovic, H. Diaz, D. Rosenbluth, L. Ritholtz, J. C. Twedt, T. T. Walker, K. Alcedo, and D. Javorsek, "Hierarchical reinforcement learning for air-to-air combat," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2021, pp. 275–284.

[12] Q. Yang, J. Zhang, G. Shi, J. Hu, and Y. Wu, "Maneuver decision of UAV in short-range air combat based on deep reinforcement learning," *IEEE Access*, vol. 8, pp. 363–378, 2020.

[13] V. Mnih, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[14] D. Hu, R. Yang, J. Zuo, Z. Zhang, J. Wu, and Y. Wang, "Application of deep reinforcement learning in maneuver planning of beyond-visual-range air combat," *IEEE Access*, vol. 9, pp. 32282–32297, 2021.

[15] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 1–9.
[16] R. Coulom, "Efficient selectivity and backup operators in Monte-Carlo tree search," in *Proc. Int. Conf. Comput. Games*. Springer, 2006, pp. 72–83.
[17] H. Jin, D. Yong, and G. Zhenlong, "Stealth engagement strategy for unmanned aerial vehicles based on double deep Q network," *Electron. Opt. Control*, vol. 27, no. 7, pp. 52–57, 2020.
[18] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, 2016, pp. 1–7.
[19] H. Dongyuan, Y. Rennong, Z. Jialiang, Z. Wanze, Z. Yu, and Z. Qiang, "Intelligent maneuver decision for unmanned combat aircraft based on LSTM-dueling DQN," *Tactical Missile Technol.*, no. 6, pp. 97–104, 2021.
[20] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.
[21] S. Chu, Z. Hui, W. Yuan, Z. Huan, and H. Jin, "Autonomous UAV maneuver decision method based on reinforcement learning," *Firepower Command Control*, vol. 44, no. 4, p. 142, 2019.
[22] Q. Yang, Y. Zhu, J. Zhang, S. Qiao, and J. Liu, "UAV air combat autonomous maneuver decision based on DDPG algorithm," in *Proc. IEEE 15th Int. Conf. Control Autom. (ICCA)*, Jul. 2019, pp. 37–42.
[23] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.
[24] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1587–1596.
[25] G. Wanchun, X. Wujie, Y. Hui, and D. Wenhan, "Unmanned aerial vehicle anti-pursuit maneuver decision based on improved double delay deep deterministic policy gradient method," *J. Air Force Eng. Univ.*, vol. 22, no. 4, pp. 15–21, 2021.
[26] B. Li, S. Bai, B. Meng, S. Liang, and Z. Li, "Autonomous UAV air combat decision algorithm based on SAC algorithm," *Command Control Simul.*, vol. 44, no. 5, pp. 24–30, 2022.
[27] L. Li, Z. Zhou, J. Chai, Z. Liu, Y. Zhu, and J. Yi, "Learning continuous 3-DoF air-to-air close-in combat strategy using proximal policy optimization," in *Proc. IEEE Conf. Games (CoG)*, Aug. 2022, pp. 616–619.
[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
[29] W. Tang, Y. Sun, and Q. Yang, "A reinforcement learning algorithm for 2v2 close-range air combat," *Tactical Missile Technol.*, vol. 1, pp. 120–130, 2022.
[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.
[31] Q. Yan, Z. Baoqi, Z. Jie, and L. Zhongkai, "Autonomous guidance method for unmanned aerial vehicle close-range air combat based on ppo algorithm," *Electron. Opt. Control*, vol. 30, no. 1, pp. 8–14, 2023.
[32] H. Zhang, Y. Wei, H. Zhou, and C. Huang, "Maneuver decision-making for autonomous air combat based on FRE-PPO," *Appl. Sci.*, vol. 12, no. 20, p. 10230, Oct. 2022.
[33] X. He, X. Jing, and C. Feng, "Air combat maneuver decision based on MCTS method," *J. Air Foreign Eng. Univ.*, vol. 18, pp. 36–41, 2017.
[34] H. Zhang, H. Zhou, Y. Wei, and C. Huang, "Autonomous maneuver decision-making method based on reinforcement learning and Monte Carlo tree search," *Frontiers Neurorobot.*, vol. 16, Oct. 2022, Art. no. 996412.
[35] Z. Hong-Peng, "Maneuver decision-making through proximal policy optimization and Monte Carlo tree search," 2023, *arXiv:2309.08611*.
[36] A. Liu, J. Chen, M. Yu, Y. Zhai, X. Zhou, and J. Liu, "Watch the unobserved: A simple approach to parallelizing Monte Carlo tree search," 2018, *arXiv:1810.11755*.
[37] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1889–1897.
[38] G. M. B. Chaslot, M. H. Winands, and H. J. van Den Herik, "Parallel Monte-Carlo tree search," in *Proc. Comput. Games: 6th Int. Conf. (CG)*, Beijing, China. Springer, 2008, pp. 60–71.
[39] J. Hu, L. Wang, T. Hu, C. Guo, and Y. Wang, "Autonomous maneuver decision making of dual-UAV cooperative air combat based on deep reinforcement learning," *Electronics*, vol. 11, no. 3, p. 467, Feb. 2022.

HONGMING WANG received the B.E. degree from Guangxi University, China, in 2022, where he is currently pursuing the master's degree with the College of Electrical Engineering. His current research interests include reinforcement learning and autonomous maneuver decision-making in air combat.

ZHUANGFENG ZHOU received the B.E. degree from Guangxi University, China, in 2022, where he is currently pursuing the master's degree with the College of Electrical Engineering. His current research interests include deep reinforcement learning and intelligent decision-making in drones.

JUNZHE JIANG received the B.E. degree from Zhengzhou University, China, in 2022. He is currently pursuing the M.S. degree with Guangxi University. His research interests include optimization, multi-agent systems, and deep reinforcement learning.

WENQIN DENG received the B.E. degree from Guangxi University, China, in 2018, where she is currently pursuing the M.S. degree. Her research interests include semantic segmentation, computer vision, and deep learning.

XUEYUN CHEN is currently an Associate Professor and a Ph.D. Supervisor with the School of Electrical Engineering, Guangxi University. His research interests include target detection and recognition in remote sensing image, computer go, face detection, and automatic driving.