Hindawi
International Journal of Aerospace Engineering
Volume 2023, Article ID 3657814, 20 pages
https://doi.org/10.1155/2023/3657814

Research Article

Autonomous Maneuver Decision of UCAV Air Combat Based on Double Deep Q Network Algorithm and Stochastic Game Theory

Yuan Cao, Ying-Xin Kou, Zhan-Wu Li, and An Xu

Aviation Engineering College, Air Force Engineering University, Xi'an 710038, China

Correspondence should be addressed to Ying-Xin Kou; kgykyx@hotmail.com

Received 9 January 2022; Revised 27 December 2022; Accepted 28 December 2022; Published 16 January 2023

Academic Editor: Jacopo Serafini

Aiming at the problem that unmanned combat aerial vehicle (UCAV) is difficult to quickly and accurately perceive situation information and make maneuvering decision autonomously in modern air combat, which is easily affected by complex factors, a maneuvering decision algorithm of UCAV combined with deep reinforcement learning and game theory is proposed in this paper. Firstly, through the UCAV dynamics model and maneuver library, a reasonable air combat situation assessment model and advantage reward function are established, and the sample data of situation assessment indicators are constructed using the structure entropy weight method. Secondly, the convolutional neural network (CNN) is used to process the high-dimensional continuous situation features of UCAV in air combat, eliminate the correlation and redundancy between situation features, and train the neural network to approximate the action-value function. Then, the double deep Q network (DDQN) algorithm in reinforcement learning (RL) is introduced to train the agent by the interaction with the environment and combined with Minimax algorithm in stochastic game theory to solve the optimal value function in each specific state, and the optimal maneuver decision of UCAV is obtained. Air combat simulation results show that UCAV can choose maneuvers autonomously under different situations and occupy a dominant position quickly by this method, which greatly improves the combat effectiveness of UCAV.

1. Introduction

The rapid development of UCAV technology and the gradual maturity of artificial intelligence technology make UCAV autonomous combat become one of the main ways of future air combat [1]. In air combat, UCAV can make a large overload action close to the ultimate strength of the aircraft body structure without considering human factors, greatly improving the maneuver ability and combat ability of UCAV. Due to the influence of such adverse factors as narrow communication bandwidth, low transmission rate, and long delay of long distance network, UCAV operation must be separated from the guidance control of ground base station to a certain extent and have independent intelligent decision-making ability. In this paper, the deep reinforcement learning algorithm in artificial intelligence is combined with the game theory in military operations research, and an intelligent air combat maneuver decision model is established to solve the autonomous decision problem in UCAV air combat.

In the ever-changing air combat, UCAV model is nonlinear and the flight trajectory of enemy UCAV is characterized by uncertainty, randomness, and real-time performance. In an uncertain environment, the air situation information needed to be collected is extremely complex, and the traditional methods cannot achieve rapid and accurate air combat decision. Therefore, how to select accurate and effective air combat maneuver strategy which is beneficial to our combat based on UCAV situation information of both sides has become an important research direction at present. Researchers have carried out relevant studies and achieved many research results (including manned aircraft and UCAV), which can be generally divided into three categories. The first category is the maneuver decision based on the basic maneuver library. The establishment of the maneuver action library was systematically studied and summarized in literature [2]. In literature [3], the maneuvering library design, control application, and maneuvering identification of basic motion library are studied, and the problems existing in

maneuvering decision based on motion library are elaborated in detail. The second category is maneuver decision based on optimization algorithm. In literature [4], the air combat is regarded as a Markov process, and the air combat situation is calculated by Bayesian theory, and the weight of maneuver decision factors is adjusted adaptively to make the objective function more reasonable and ensure the advantage of UCAV. In literature [5], although genetic algorithm has better robustness and search ability and is suitable for solving large-scale optimization problems, it cannot solve problems without displaying objective function, so an improved genetic algorithm is adopted to model UCAV maneuvering decision. Although the above methods have the advantages of simple principle, easy to implement, accurate evaluation results, and persuasiveness, the decision-making model has the problem of complicated reasoning process, which wastes a lot of time to search for optimization, resulting in slow response speed of UCAV, which is difficult to meet the requirements of real time in practical application. The third category is maneuver decision based on artificial intelligence, including artificial neural network, expert system, and deep reinforcement learning algorithm. In literature [6], an evolutionary expert system tree method is proposed to study air combat decision-making, which solves the problem that the traditional expert system method cannot cope with unexpected situations, but the simulation environment is relatively simple. In literature [7], adaptive neural network technology is used to design PID controller, which has strong tracking accuracy for highly maneuvering targets. However, neural network method requires a large number of air combat samples, resulting in insufficient learning samples. Compared with neural network and expert system, deep reinforcement learning combines the perceptual ability of deep learning with the decision-making ability of reinforcement learning in a general form and achieves direct control from original input to output through end-to-end learning. Since its proposal, it has made substantial breakthroughs in many aspects that require perception of high-dimensional raw input data and decision control, and it is more suitable for air combat maneuver decision-making. In literature [8], an intelligent decision-making method for air combat maneuvering based on heuristic reinforcement learning is proposed, in which the relatively optimal air combat maneuvering decision sequence is calculated by "trial and error" with the external environment, and the neural network method is used to learn the process of reinforcement learning to realize the real-time dynamic iterative calculation of air combat decision sequence. In literature [9], it is proposed to use deep Q network to analyze the optimal control instructions of UCAV, and UCAV can independently evaluate battlefield situation and make tactical decisions in different situations. However, the above two papers lack the theoretical research on UCAV air combat game, and the deep Q network used will cause the problem of high estimation of Q value in the learning process, resulting in unstable algorithm performance.

In air combat, UCAVs are not only required to have accurate and timely maneuver strategies but also to have effective resource allocation schemes. In particular, in the multi-UCAV cooperative operation, the fast and correct air combat maneuver decision is the premise to ensure the victory of the war, but the target and fire resource allocation is a particularly important part after the threat assessment. There is a distinct difference between a multi-UCAV air combat and a single-UCAV air combat. The difference is that in the face of multiple enemy targets, the target and firepower are allocated optimally according to our resources. The purpose of target assignment is to select targets to attack or avoid from a number of enemy targets within the range that our aircraft can search. Collaborative target assignment means that UCAVs transmit target location and other information to ground command and control system through data link, and the system allocates targets according to the allocation principle based on battlefield situation.

Target and fire allocation is a complex combinatorial optimization problem with many constraints. Traditional solutions include implicit enumeration and linear programming. However, with the increase of the problem dimension, the solution of the problem will consume a lot of time, so it is not conducive in solving complex problems. Some intelligent algorithms, such as genetic algorithm and particle swarm optimization algorithm, are widely used in the research field of resource allocation. In literature [10], a joint design of UAV-USV-UUV network for collaborative underwater target search is proposed, and DQN algorithm is proposed to solve the proposed target search problem. The final simulation results show that considering the balance between energy consumption and interconnectedness of the system, the scheme is suitable for underwater target search and has a high success rate. In literature [11], a heterogeneous AUV auxiliary information collection system is proposed, whose goal is to maximize the energy efficiency of nodes considering AUV trajectory, resource allocation, and information age. Finally, the simulation results verify the effectiveness and superiority of the proposed strategy. In literature [12], a knowledge-based RL method is proposed, which uses domain knowledge to compress the state space that the agent needs to explore, so as to improve the convergence speed of the algorithm. Specifically, the inertial law of aircraft and the free space signal attenuation law are used to guide the efficient detection of UAVS in state space. Finally, the simulation results show that the proposed algorithm is superior to the modelless RL algorithm in terms of convergence speed and average reward. The above three literatures are studying the allocation methods of computing and transmission resources, which have high reference value.

To improve the autonomy and accuracy of UCAV maneuvering decisions, this paper proposes a UCAV maneuver decision algorithm that combines deep reinforcement learning and game theory. Firstly, the advantage reward function of air combat is constructed by UCAV dynamics model and maneuvering decision control library. Then, the indicator weight of the air combat situation assessment model is established by using the structure entropy weight method. Secondly, the Minimax algorithm in stochastic game theory is combined with DDQN learning algorithm in deep reinforcement learning, and the optimal

function value in different states is solved by linear programming, and the neural network is trained to approximate the value function. Finally, the optimal maneuver strategy is obtained according to the output of Minimax-DDQN network. This algorithm can reduce the complexity of artificial manipulation, ensure the superiority of the calculation results, and improve the decision-making ability and response speed of UCAV in air combat. The effectiveness and feasibility are verified by simulation experiments.

2. Deep Reinforcement Learning Based on Value Function

2.1. Deep Learning. Deep learning (DL) is a branch of machine learning algorithm that uses multiple processing layers with complex structures or multiple nonlinear transformations to abstract data at a high level [13]. DL models are usually composed of multiple layers of nonlinear operation units, which automatically learn abstract feature representation from a large amount of training data by taking the output of the lower layer as the input of the higher layer, so as to discover distributed features of the data. Similar to shallow neural network, deep neural network can also provide modeling for complex nonlinear systems, and its higher structural level provides a higher level of abstraction for the model, thus improving the model capability.

The enrichment of training resources and the improvement of computing performance provide sufficient conditions for the wide application of convolutional neural networks (CNN) [14]. Compared with traditional neural networks, CNN effectively reduces algorithm complexity through weight sharing and pooling, which has a good effect on data feature extraction. It can be seen from Figure 1 that CNN used in this paper includes one input layer, two convolution layers, one pooling layer, two fully connected layers, and one output layer. The data is converted into compact intermediate classes in the input layer to reduce dimension and remove redundant information. In the convolution layer, the input neurons of each neural network are connected with the local input of sample data to extract local features. The dimension of input feature samples is effectively reduced in the pooling layer and the size of data space is also reduced, so the number of parameters is reduced and overfitting is controlled to some extent. The input data of each dimension in the convolution layer is connected with the output in the full connection layer, and the output threshold is determined by activation function. The features selected by the upper layer are taken as the input features of the next layer in the CNN, and different features are captured by different convolutional kernels in the way of weight sharing, so as to continuously optimize and integrate the features and finally obtain the optimal features.

2.2. Reinforcement Learning. Reinforcement Learning (RL) is a Markov decision process (MDP) in which the agent and the environment interact with each other and learn the optimal strategy through "trial and error" [15]. For model-free reinforcement learning problems, the Markov decision process is described by a 4-tuple: state space S, action space A, state transition probability P, and reward function R. Policy π is defined as a mapping from the state space to the action space. The interaction process of reinforcement learning is as follows. At time t, the agent through policy π applies an action a_t to the environment. After the action a_t is executed, the state is transferred from s_t to the next state s_{t+1} with probability $p^{a_t}_{s_t,s_{t+1}}$, and the agent obtains the reward r_t from the environment. The purpose of reinforcement learning is to maximize long-term cumulative rewards by constantly adjusting strategies. Therefore, the state value function $V^\pi(s)$ and the action-value function $Q^\pi(s,a)$ are defined to measure the quality of each state or each action.

In the process of air combat as shown in Figure 2, the UCAV flight parameters of our side and the enemy are calculated to form the air combat environment state description including UCAV angle, altitude, speed and other situation information, and output to the agent. At the same time, according to the current flight status of our UCAV and the enemy, the advantage reward function proposed in this paper is used to evaluate the reward value and output it to the agent. In the spirit of trying to maximize the reward value as much as possible, the agent makes an action and applies the action to the air combat environment through UCAV. Reinforcement learning agent realizes self-learning by continuously interacting with air combat environment through reward value, flight status, and action, dynamically updating and evaluating maneuvering strategies, and making subsequent maneuvering actions tend to be optimal.

In air combat, it is assumed that the reward r of each action is added up from time t to time T, which is the sum of the reward in the state transition process. Considering that the further reward value should contribute less to the value function of the current state, it must be multiplied by a discount factor γ. The sum of the rewards is as follows.

$$R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}, \tag{1}$$

where $\gamma \in [0, 1]$ is used to measure the effect of future reward values on cumulative reward values.

Action-value function $Q^\pi(s,a)$ refers to executing action a in the current state s and following policy π until the end. The cumulative reward obtained by the agent in this process can be expressed as

$$Q^\pi(s,a) = E[R_t | s_t = s, a_t = a, \pi]. \tag{2}$$

A policy is called optimal if the expected rewards of π^* are greater than or equal to those of all other policies. The action-value function of the optimal policy is

$$Q^*(s,a) = \max_{\pi^*} E[R_t | s_t = s, a_t = a, \pi]. \tag{3}$$

According to the Bellman formula, the action-value function can be shown as $Q(s,a) = r + \gamma Q^*(s', a')$, where s' represents the next state, a' represents the best action in

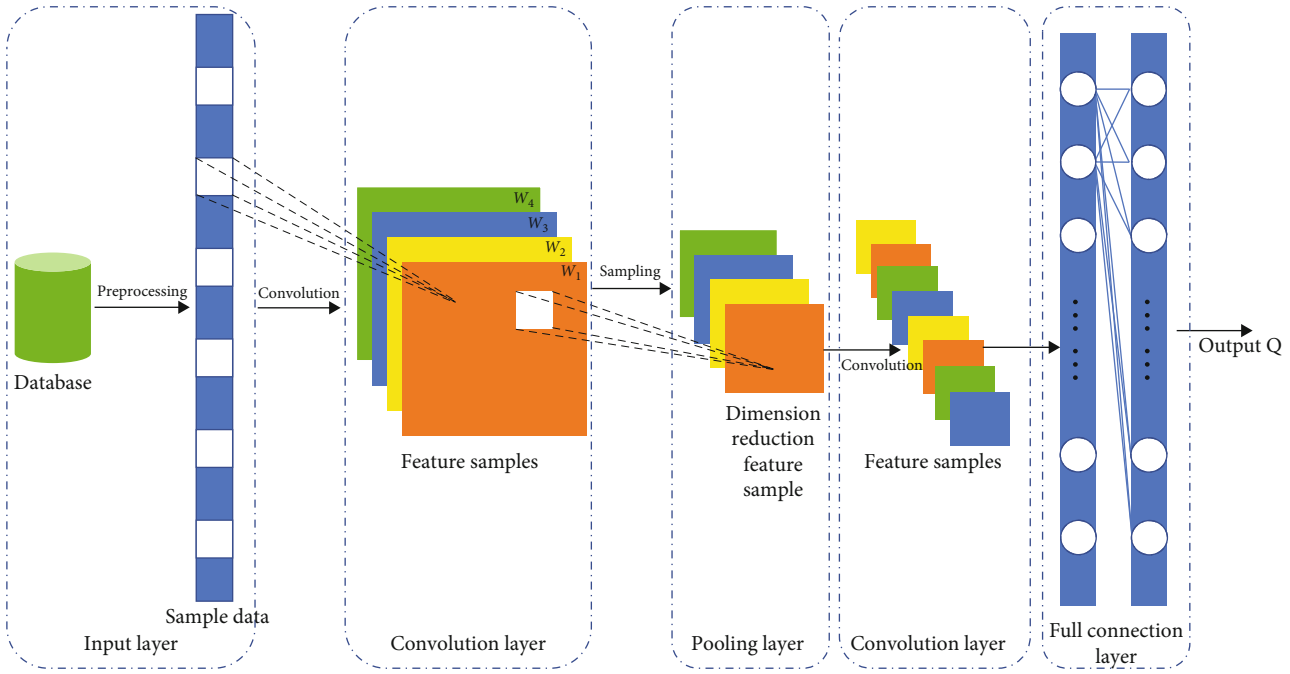


Figure 1: CNN architecture.

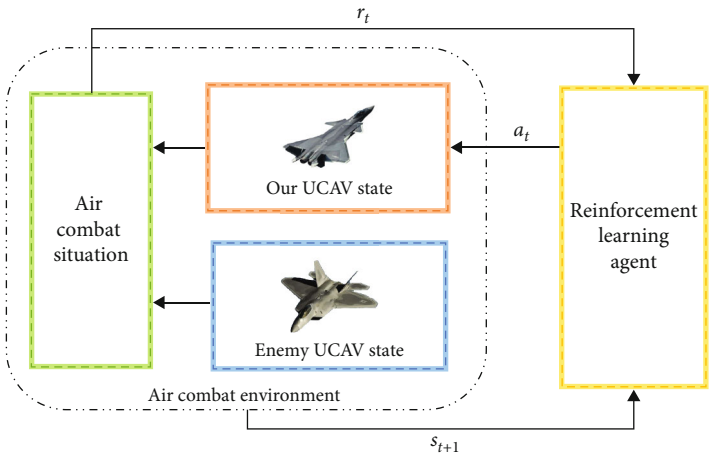


Figure 2: Air combat maneuver strategy framework based on reinforcement learning.

the next state, and r represents the reward of a action to the next state. The optimal Q value function solved is

$$Q_{i+1}(s,a) = E_{s' \in S}\left[r + \gamma \max_{a'} Q_i\left(s', a'\right) | s, a\right], \quad (4)$$

when $i \longrightarrow \infty$, $Q_i \longrightarrow Q^*$. Through continuous iteration, the action-value function will eventually converge, so as to obtain the optimal policy: $\pi^* = \arg \max_{a \in A} Q^*(s,a)$.

2.2.1. Deep Q-Learning Network. Q-learning is an important method widely used in reinforcement learning. Mnih et al. [16, 17] proposed a deep Q network (DQN) model by combining convolutional neural network with Q-learning algorithm in traditional reinforcement learning. DQN model based on the neural network adapts to the unlabeled sample data of reinforcement learning and adjusts its own parameters from the environmental information to seek the optimal strategy to satisfy the maximum reward. According to formula (4), when the number of actions tends to infinity, the expectation also tends to the real optimal value function Q^*. In DQN, each Q value is estimated by neural network:

$$Q(s, a; \theta) \approx Q^*(s,a), \quad (5)$$

where θ is the connection weight of neurons in DQN. There is a difference between the estimated Q value obtained by Bellman formula and the predicted Q value by neural

network. Therefore, a loss function is introduced to minimize the difference:

$$L(\theta_i) = E_{(s,a,r,s')}\left[(y_i - Q(s, a\,; \theta_i))^2\right], \tag{6}$$

where target function $y_i = E_{s' \in S}[r + \gamma \max Q(s', a'\,; \theta_i^-)|s, a]$ and Q network is

$$Q(s, a\,; \theta_i) = r + \gamma \sum_{s' \in S} P\left(s'|s, a\right) \max_{a \in A} Q\left(s', a\,; \theta_i\right). \tag{7}$$

Target network and online Q networks are both deep convolutional neural networks, which have the same structure but different built-in parameters. The optimal solution is obtained by minimizing the loss function $L(\theta_i)$ by gradient descent method, and the weight θ_i^- of the target network is updated with the weight θ_i of Q network after N iteration, while θ_i^- remained unchanged in the middle process.

2.2.2. Double Deep Q-Learning Network. Due to the air combat environment is nonstationary and the online DQN updates are highly correlated, the algorithm is fundamentally unstable, which may cause overestimation attributing to the selection and evaluation of actions in deep Q-learning are based on the weight θ_i^- of target function network. If the overestimated Q values are not evenly distributed, it will be difficult to obtain the optimal solution. Van Hasselt proposed a double deep Q network algorithm based on double Q-learning algorithm [18]. The core idea of double deep Q network is as follows. Deconstruct the action selection and evaluation of target function network in DQN, construct two different weights θ_i and θ_i^-, where θ_i is used to select the action with the maximum Q value and θ_i^- is used to estimate the value of the greedy-policy and updates the weights of the online network θ_i. The two weights separate action selection from policy evaluation and reduce the risk of overestimating Q value. The target function of double deep Q-learning is as follows:

$$y_t^{\text{DoubleDQN}} = r + \gamma Q\left(s', \arg \max_a Q\left(s', a\,; \theta_i\right); \theta_i^-\right), \tag{8}$$

where θ_i is the online network weight and θ_i^- is the target network weight.

3. Stochastic Game Theory

Game theory, as a modern scientific system, studies how decision-makers make decisions when all aspects of the decision-making subject interact and the theory about the balance of the decision, which has been widely used in military research. The UCAV air combat studied in this paper is a game process in a certain airspace under the premise of obeying certain safety rules. In the process of the game, the gain of one party inevitably leads to the loss of the other party, and the sum of the gain and loss of the two parties is zero, so the game type is two-person zero-sum game.

The air combat problem studied in this paper is a continuous process, while the traditional game is a single step, more like a discrete process. Therefore, it is necessary to expand from traditional game to the stochastic game. It can be considered that stochastic game [19] is the promotion of game theory in traditional Markov decision process (MDP).

A stochastic game can be represented by a six-tuple $\{n, S, (a_i)_{i=1}^n, (r_i)_{i=1}^n, P, \gamma\}$. n represents the number of agents in the game, and S is the state space of the game. After the agent makes an action, the state will change, and its evolution is Markov. a_i represents the action of the agent i, and $\prod_{i=1}^n a_i$ is the action space of all agents. r_i represents the reward obtained by the agent i in the next state s' after the action a_i taken by state s. P represents the probability that the agent i will move from s to s' after the state s takes an action a_i. γ represents the discount factor.

In a stochastic game, the reward of agent in the next state depends on the current state and the current actions of all agents. According to the theory of Nash equilibrium [20], it is necessary to find a Nash equilibrium strategy $\pi_i \in \Pi_i$ to solve the stochastic game, indicating that the agent i chooses action a under state s to maximize the future reward value. The Nash equilibrium strategy of the multiagent stochastic game is $(\pi_1^*, \cdots \pi_n^*)$, and $\forall s \in S$, $i = 1, \cdots, n$, satisfy the following formula:

$$V_i(s, \pi_1^*, \cdots, \pi_i^*, \cdots \pi_n^*) \geq V_i(s, \pi_1^*, \cdots, \pi_i, \cdots \pi_n^*). \tag{9}$$

$V_i(s, \pi_1^*, \cdots, \pi_i^*, \cdots \pi_n^*)$ is the state value function, simply written as $V_i^*(s)$. $Q_i^*(s, a_1, \cdots a_n)$ is the action-value function. According to Bellman formula,

$$\begin{aligned} &\sum_{a_i \in \prod_{i=1}^n a_i} Q_i^*(s, a_1, \cdots, a_n)\pi_1^*(s, a_1) \cdots \pi_i^*(s, a_i) \cdots \pi_n^*(s, a_n) \\ &\quad \geq \sum_{a_i \in \prod_{i=1}^n a_i} Q_i^*(s, a_1, \cdots, a_n)\pi_1^*(s, a_1) \cdots \pi_i(s, a_i) \cdots \pi_n^*(s, a_n). \end{aligned} \tag{10}$$

In this paper, the Minimax method is used to construct linear programming to solve Nash equilibrium strategy in each specific state S. The formula of Minimax algorithm for two-agent zero-sum game is as follows:

$$V = \max_{\pi \in PD(A)} \min_{a_2 \in A} \sum_{a_1 \in A} Q_i^*(a_1, a_2)\pi_i, \tag{11}$$

where $PD(A)$ is the policy space and A is the optional action space of two agents. The significance of formula (11) is that each agent maximizes the expected reward value in the worst situation during the game with the opponent. The reward value of the stochastic game is expressed in the following form:

$$R_z = \begin{bmatrix} r_{11} & \cdots & r_{1j} \\ \vdots & \ddots & \vdots \\ r_{i1} & \cdots & r_{ij} \end{bmatrix}, \tag{12}$$

where the element r_{ij} of matrix R_z represents the reward value obtained by the z agent when the first agent selects action i and the second agent selects action j. In a zero-sum game, the two agents are perfectly competitive against each other, so $R_1 = -R_2$. $p_i(i = 1, 2, \cdots, n)$ is defined to represent the probability that the first agent selects action i, and $q_j(j = 1, 2, \cdots, m)$ is defined to represent the probability that the second agent selects action j. For the first agent, the following linear programming can be obtained:

$$\begin{cases} \max\limits_{p_i} V_1, \\ r_{11}p_1 + \cdots + r_{i1}p_i \geq V_1, \\ \vdots \\ r_{1j}p_1 + \cdots + r_{ij}p_i \geq V_1, \\ \sum\limits_{i=1}^{n} p_i = 1, \\ p_i \geq 0. \end{cases} \tag{13}$$

Similarly, since there is $R_1 = -R_2$, the linear programming of the Nash equilibrium strategy of the second agent can be obtained:

$$\begin{cases} \min\limits_{q_j} V_2, \\ r_{11}q_1 + \cdots + r_{1j}q_j \leq V_2, \\ \vdots \\ r_{i1}q_1 + \cdots + r_{ij}q_j \leq V_2, \\ \sum\limits_{j=1}^{m} q_j = 1, \\ q_j \geq 0. \end{cases} \tag{14}$$

The Nash equilibrium strategy can be obtained by solving the linear programming of formulas (13) and (14).

4. Air Combat Maneuver Game Modeling Based on Uncertain Environment

4.1. UCAV Flight Dynamics Model and Maneuver Library

4.1.1. UCAV Flight Dynamics Model. In the process of studying the maneuvering decision of UCAV, the dynamic model of UCAV in Cartesian coordinate system is adopted, which takes normal overload, track angle, and heading angle as the aircraft control variables. In order to simplify the research complexity, the angle of attack and sideslip angle during UCAV flight are not considered. The ground coordinate system is regarded as the inertial coordinate system, ignoring the influence of the earth's rotation and revolution and ignoring the change of the earth's curvature. The dynamics model [21] is as shown.

$$\begin{cases} \dot{x} = v\cos\gamma\cos\psi, \\ \dot{y} = v\cos\gamma\sin\psi, \\ \dot{z} = v\sin\gamma, \\ \dot{v} = g(n_x - \sin\gamma), \\ \dot{\gamma} = \dfrac{g}{v}(n_z\cos\mu - \cos\gamma), \\ \dot{\psi} = \dfrac{g}{v\cos\gamma}n_z\sin\mu, \end{cases} \tag{15}$$

where x, y, and z are the coordinate locations of UCAV in the coordinate system, respectively. v represents the speed. γ represents the track angle, and ψ represents the heading angle. g represents the acceleration of gravity. μ represents the roll angle around the velocity vector. n_x is the overload in the velocity direction, and n_z is the overload in the pitch direction. $\dot{x}$, $\dot{y}$, $\dot{z}$, $\dot{v}$, $\dot{\gamma}$, and $\dot{\psi}$ are the time derivatives of corresponding physical quantities, respectively.

4.1.2. UCAV Maneuver Library. The UCAV maneuver library refers to the optional set of UCAV maneuver decisions in air combat, which contains the maneuver actions or maneuver sequences that can be selected in air combat. The maneuver library is mainly divided into two categories: the first is the typical maneuver library based on air combat maneuver tactics, and the second is the basic maneuver library based on UCAV operation mode. The typical maneuvers include vertical somersaults, horizontal roll maneuvers, stall cobra maneuvers, high yo-yo, and low yo-yo, but these typical maneuvers are all made up of basic maneuvers. Therefore, UCAV maneuvers are selected from the basic maneuver library, and 7 basic maneuver actions designed by NASA scholars are used [22], which are (1) uniform and straight flight, (2) maximum acceleration direct flight, (3) maximum deceleration of direct flight, (4) maximum overload climb, (5) maximum overload dive, (6) maximum overload right turn, and (7) maximum overload left turn. These are shown in Figure 3.

In this paper, based on basic maneuver actions and formula (15), action command $[\dot{v}, \gamma, \psi]$ is used to control UCAV, and a candidate action library for autonomous combat decision-making is established.

(1) Control instructions for uniform and straight flight:

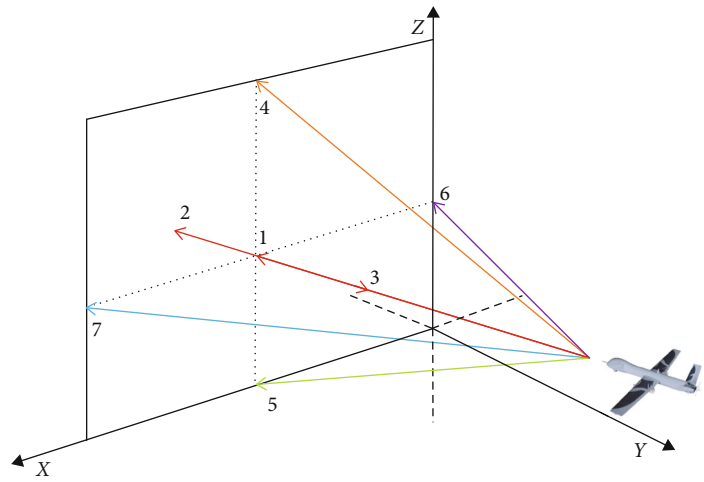
$$[\dot{v}, \gamma, \psi] = [0, 0, 0]. \tag{16}$$

(2) Control instructions for maximum acceleration direct flight:

$$[\dot{v}, \gamma, \psi] = [\dot{v}_{\max}, 0, 0]. \tag{17}$$

(3) Control instructions for maximum deceleration of direct flight:

$$[\dot{v}, \gamma, \psi] = [-\dot{v}_{\max}, 0, 0]. \tag{18}$$



1. Uniform and straight flight
2. Maximum acceleration direct flight
3. Maximum deceleration of direct flight
4. Maximum overload climb
5. Maximum overload dive
6. Maximum overload right turn
7. Maximum overload left turn.

Figure 3: UCAV basic maneuver library.

(4) Control instructions for maximum overload climb:

$$[\dot{v}, \gamma, \psi] = [\dot{v}_{\max}, \gamma_{\max}, 0]. \quad (19)$$

(5) Control instructions for maximum overload dive:

$$[\dot{v}, \gamma, \psi] = [\dot{v}_{\max}, -\gamma_{\max}, 0]. \quad (20)$$

(6) Control instructions for right turn for maximum overload:

$$[\dot{v}, \gamma, \psi] = [\dot{v}_{\max}, 0, \psi_{\max}]. \quad (21)$$

(7) Control instructions for maximum overload left turn:

$$[\dot{v}, \gamma, \psi] = [\dot{v}_{\max}, 0, -\psi_{\max}]. \quad (22)$$

$\dot{v}$ is the tangential acceleration instruction of UCAV, $\dot{v}_{\max}$ and $-\dot{v}_{\max}$ are maximum acceleration and maximum deceleration, respectively. γ is the track angle inclination instruction. $\gamma_{\max}$ and $-\gamma_{\max}$ are the climb and dive with the maximum inclination that the UCAV body can withstand, respectively. ψ is the heading angle inclination instruction. $\psi_{\max}$ and $-\psi_{\max}$ are the maximum angle of deflection right and left turn, respectively (the right side of the flight direction of UCAV is positive and the left side is negative).

$[\dot{v}, \gamma, \psi]$ is an aircraft maneuvering action command, and $\mathbf{\Lambda}$ is set as a collection of action commands for UCAV. Each action a_i in the maneuver library is composed of several maneuvering action commands. It can be seen that the action set A consists of a set of discrete action commands, which is a subset of the action command set. In order to get close to the actual flight state, the flight acceleration $\dot{v}$ of the control output is limited to $[-2g, 3g]$, the track angle γ is limited to $[-30\circ, 30\circ]$, and the heading angle ψ is limited to $[-70\circ, 70\circ]$.

4.2. Air Combat Situation Assessment and Advantage Reward Function

4.2.1. Air Combat Situation Assessment. The process of air combat maneuver decision is to choose the most beneficial maneuvering action based on the situation information. The situation information of our UCAV and the enemy UCAV is regarded as the reward and punishment signal, and the corresponding air advantage function is constructed. It can make the decision system choose the right maneuver and improve the advantage of our UCAV to the enemy UCAV in air combat. Therefore, air combat situation assessment is particularly important, which affects the combat effectiveness of UCAV weapons and ultimately affects the results of air combat [23]. In this paper, the air combat situation evaluation function of angle situation, distance situation, height situation, and speed situation is designed, and

the comprehensive situation is obtained by weighting these four parts, which is used to evaluate the advantages and disadvantages of our UCAV.

(1) Angle Situation Indicator. In air combat, the launching condition of air-to-air missile is more demanding, especially the launching angle. If our UCAV has an angle advantage over the enemy, then our probability of winning will increase dramatically. The angle situation function is as follows:

$$Q_\gamma = \begin{cases} \frac{3}{2}\cos\gamma + 1 - \frac{\sqrt{3}}{2}, |\gamma| < \frac{\pi}{6}\,\text{rad}, \\ \frac{6}{\pi}\left(\frac{\sqrt{3}}{2} - 1\right)\left(|\gamma| - \frac{\pi}{3}\right), \frac{\pi}{6}\,\text{rad} \le |\gamma| \le \frac{\pi}{2}\,\text{rad}, \end{cases}$$

$$Q_\psi = \begin{cases} \frac{3}{2}\cos\psi + 1 - \frac{\sqrt{3}}{2}, |\psi| < \frac{\pi}{6}\,\text{rad}, \\ \frac{5}{3} - \frac{4}{\pi}|\psi|, \frac{\pi}{6}\,\text{rad} \le |\psi| < \frac{5\pi}{12}\,\text{rad}, \\ 5 - \frac{12}{\pi}|\psi|, \frac{5\pi}{12}\,\text{rad} \le |\psi| < \frac{\pi}{2}\,\text{rad}, \\ -1, \frac{\pi}{2}\,\text{rad} \le |\psi| < \pi\,\text{rad}, \end{cases} \tag{23}$$

where Q_γ is track angle situation function and Q_ψ is the heading angle situation function. (x_a, y_a, z_a) and (x_b, y_b, z_b) are the three-dimensional coordinates of our UCAV and the enemy UCAV, respectively, and our line of sight vector $\mathbf{R}_{\text{LOS}} = (x_b - x_a, y_b - y_a, z_b - z_a)$.

(2) Distance Situation Indicator. The main factor affecting distance situation is the range of air-to-air missile. This paper assumes that the detection range of airborne radar of UCAV must be greater than the range of missile, so the distance situation function is

$$Q_R = \begin{cases} 1, R < R_m, \\ e^{-(R-R_m)^2/2\sigma^2}, R_m \le R, \end{cases} \tag{24}$$

where R_m is the missile range. σ is the standard deviation. R is the straight-line distance between the two UCAVs. When the enemy UCAV is within the range of the missile, the distance situation value is always 1. In other cases, the further the enemy UCAV is from our UCAV, the smaller the situation value.

(3) Height Situation Indicator. In the process of air combat, the relative height of two UCAV will affect their gravitational potential energy. UCAV with higher height is more likely to transform potential energy advantage into kinetic energy advantage. The height situation function is as follows:

$$Q_H = \begin{cases} 0.1, H < -5\,\text{km}, \\ 0.5 + 0.1H, -5\,\text{km} \le H < 5\,\text{km}, \\ 1, H \ge 5\,\text{km}, \end{cases} \tag{25}$$

where H is the height difference between our UCAV and the enemy UCAV. The higher the flight height of UCAV, the greater the situation value. When the relative height is greater than 5 km, the situation value of height is 1.

(4) Speed Situation Indicator. The flight speed of UCAV affects the maneuver ability in air combat and the initial speed of launching air-to-air missiles. The speed situation function is as follows:

$$Q_V = \begin{cases} 0.1, v_i < 0.6v_j, \\ -0.5 + v_i/v_j, 0.6v_j \le v_i \le 1.5v_j, \\ 1, v_i > 1.5v_j, \end{cases} \tag{26}$$

where v_i and v_j are the flight speeds of our UCAV and enemy UCAV, respectively.

(5) Comprehensive Situation Indicator. Comprehensively considering angle indicator, distance indicator, height indicator, and speed indicator, the comprehensive situation indicator is solved by linear weighting method:

$$Q = w_1 Q_\gamma + w_2 Q_\psi + w_3 Q_R + w_4 Q_H + w_5 Q_V, \tag{27}$$

where $w_i (i = 1, 2, 3, 4, 5)$ represents the weight of each situation value, reflecting the proportion of each situation value in the comprehensive situation.

4.2.2. The Structure Entropy Weight Method. When the objective weighting method is used to determine the weight of enemy UCAV situation index, the error of parameters measured by airborne sensors will lead to inaccurate weight values. Therefore, it is necessary to combine subjective weighting method to make the determination of index weight more objective and reasonable [24]. A weighting method combining subjective and objective called "structure entropy weighting method" [25, 26] is used in this paper. This method combines Delphi method and fuzzy analysis method. Firstly, experts are used to evaluate the importance of indicators subjectively, and then, structure entropy weight method is used to objectively and quantitatively analyze the subjective evaluation values of experts. Finally, entropy value is calculated and "blind degree" analysis is carried out to obtain reasonable indicator weights. The specific steps of the structure entropy weight method are as follows.

Step 1. Collect expert opinions and form a "typical ranking."

Consult with k well-known experts in air combat and review the information related to air combat to determine the importance of each situational assessment indicator, as shown in Table 1, and finally, form a "typical ranking." Each expert evaluated the importance of each situation

Table 1: Expert opinion evaluation form.

Expert number	Q_γ	Q_ψ	Q_R	Q_H	Q_V
Expert 1	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
⋮	⋮	⋮	⋮	⋮	⋮
Expert i	a_{i1}	a_{i2}	a_{i3}	a_{i4}	a_{i5}
⋮	⋮	⋮	⋮	⋮	⋮
Expert k	a_{k1}	a_{k2}	a_{k3}	a_{k4}	a_{k5}

assessment indicator and obtained the evaluation matrix $\mathbf{A}$ $(\mathbf{A} = a_{ij})_{k\times 5}$ for each indicator, where $i = 1, 2 \cdots, k$ and $j = 1, 2, \cdots, 5$. a_{ij} represents a qualitative ranking obtained by the ith expert after evaluating the jth situation indicator.

Step 2. "Blindness" analysis of "typical ranking."

Because the experts participating in the survey have deviations in their understanding of the importance of situation assessment indicators, the ranking of the importance of situation assessment indicators might be different from the real situation, resulting in noise and uncertainty in the data. It is necessary to correct the result of "typical ranking" of experts and use the entropy weight method in information theory to calculate the entropy value of each indicator, so as to reduce the uncertainty of "typical ranking" of experts and make the importance of each indicator judged by experts more real.

For the qualitative and quantitative transformation of the ranking number a_{ij} in Table 1, the membership function of the qualitative ranking transformation is defined as

$$\mu(a_{ij}) = \frac{\ln(m - a_{ij})}{\ln(m-1)}, \tag{28}$$

where m is the conversion parameter amount, $m = j + 2$, $m = 7$. Through formula (28), the ranking matrix $\mathbf{A}$ can be quantitatively transformed into the corresponding membership matrix $\mathbf{B}$, where $b_{ij} = \mu(a_{ij})$.

The "unanimous view" of k experts on the indicator μ_j is called the average degree of awareness, which is recorded as b_j, so

$$b_j = \frac{(b_{1j} + b_{2j} + \cdots + b_{kj})}{k}. \tag{29}$$

"Knowledge blindness" is the uncertainty caused by the expert's cognition of the factor μ_j, denoted as T_i; then,

$$T_i = \left| \frac{[(\max(b_{1j}, b_{2j}, \cdots, b_{kj}) - b_j) + (\min(b_{1j}, b_{2j}, \cdots, b_{kj}) - b_j)]}{2} \right|. \tag{30}$$

The overall knowledge of the k experts on the indicator μ_j is recorded as x_j,

$$x_j = b_j(1 - T_j),\ x_j > 0. \tag{31}$$

The evaluation vector $\mathbf{X} = (x_1, x_2, \cdots, x_5)$ of all the experts on each indicator.

Step 3. Normalization.

In order to get the weight of the indicator μ_j, normalize $x_j = b_j(1\text{-}T_j)$; let

$$w_j = \frac{x_j}{\sum_{j=1}^{5} x_j}. \tag{32}$$

Obviously, $w_j(j = 1, 2, \cdots, 5) > 0$, $\sum_{j=1}^{5} w_j = 1$. $\mathbf{W} = [w_1, w_2, \cdots, w_5]$ is the weight vector of the indicator set $\mathbf{Q} = [Q_\gamma, Q_\psi, Q_R, Q_H, Q_V]$ formed by the situation assessment indicator. $\mathbf{W}$ is the overall judgment of the consistency of the importance of the indicator set $\mathbf{Q}$ by k "expert opinions," which conforms to the wishes or cognition of the k expert groups.

4.2.3. The Advantage Reward Function. According to the instantaneous air situation of UCAV in the air combat, the forward attack capability of UCAV infrared air-to-air missile at close range is taken into consideration [27], and the advantage reward function is established in this paper. The advantage reward function is used as the criterion of reward and punishment to help the agent choose the best maneuver and improve the air advantage of UCAV over the enemy UCAV. When the enemy UCAV is in the missile attack area of our UCAV, our UCAV takes the advantage, as shown in Figure 4.

There are 4 conditions for our UCAV to achieve air advantage. (1) The radar detection range R_r and the range R_f of air-to-air missile are both greater than the straight-line distance between the two aircraft R. (2) The velocity direction of our UCAV and the track angle γ and heading angle ψ of line of sight vector $\mathbf{R}_{\mathrm{LOS}}$ are both within the range of $(-(\pi/6)\mathrm{rad}, \pi/6\mathrm{rad})$. (3) The height difference between our UCAV and the enemy UCAV H is greater than or equal to zero. (4) The comprehensive situation value Q_a of our UCAV must be greater than that Q_b of the enemy UCAV. When the above four conditions are met at the same time, it can be judged that our UCAV gains the advantage and obtains the reward value in the stochastic game process, which can be expressed as follows.

$$\begin{cases} R < R_r, R < R_f, \\ |\psi| < \dfrac{\pi}{6}\mathrm{rad}, |\gamma| < \dfrac{\pi}{6}\mathrm{rad}, \\ H \geq 0, \\ Q_b < Q_a. \end{cases} \tag{33}$$

4.3. Stochastic Game Modeling. In this paper, the UCAV air combat between the enemy and ours is modeled as a two-person zero-sum game. According to Section 3, a six-tuple $\{n, S, (a_i)_{i=1}^n, (r_i)_{i=1}^n, P, \gamma\}$ needs to be determined in

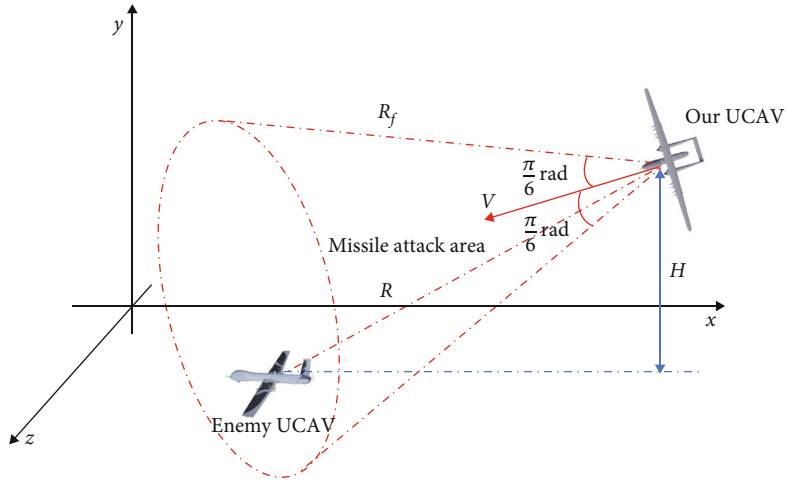


Figure 4: UCAV advantage area.

stochastic game, which is used to construct stochastic game model in air combat.

(1) Number n: number of UCAV on the enemy and our side, $n = 2$

(2) State space S: the state characteristics of UCAV can be determined according to the factors that affect UCAV air combat situation. It is mainly composed of coordinate information (x, y, z), track angle γ, and heading angle ψ of UCAV of the enemy and our party, so

$$S = (x_a, y_a, z_a, \gamma_a, \psi_a, x_b, y_b, z_b, \gamma_b, \psi_b). \tag{34}$$

(3) Action space $(a_i)_{i=1}^n$: it can be seen from Section 4.1.2 that there are 7 basic actions in the UCAV maneuver action library. Each action a_i in the maneuver library corresponds to a set of control values $[\dot{v}, \gamma, \psi]$. Thus, action space **A** consists of a discrete set of action values

(4) Reward function $(r_i)_{i=1}^n$: in the stochastic game, the action-value function of Markov process is used to represent the reward, and $Q(s, a, b)$ is used to represent the action-value function when our UCAV chooses the action a under state s and the enemy UCAV chooses the action b. Rewards are evaluated according to the advantage reward function in Section 4.2.3 for the current state s. If our UCAV is dominant, the reward value r is 1; if the enemy UCAV is dominant, the reward value r is −1; otherwise, r is 0

(5) Transition probability P: under the action (a_1, b_2) combined by our UCAV action a_1 and enemy UCAV action b_1, the current state s moves to the next state s' probability

(6) Discount factor γ: the discount factor is selected from $[0, 1]$, which is generally about 0.9

5. Air Combat Maneuver Game Strategy Based on DDQN Algorithm

5.1. The Value Function of the Stochastic Game. The stochastic game is a dynamic game process with state transition probability involving multiple agents. The game process is divided into several game stages. At the beginning of each stage, the agent selects its own strategy and obtains corresponding rewards determined by the current state and strategy. Then, proceed to the next stage randomly according to the probability distribution and agent strategy. In the new state, the previous strategy selection process is repeated and the game continues. All the rewards obtained by agents in stochastic game are generally summed up by the current reward value and the expected reward value. Because in the game environment, the expected reward value will be affected by the strategy of opponent, and in the process of air combat, it is generally impossible to predict the maneuver of the enemy aircraft, so this paper adopts Minimax algorithm to select the optimal strategy of stochastic game. The core idea of Minimax algorithm is it is assumed that the enemy agent has perfect decision-making ability, so every decision of the enemy will put us into the worst situation, and we hope to find the most beneficial strategy in this situation. The value function of MDP represents the expected discount return obtained by the optimal strategy and the state value function $V(s)$ and action-value function $Q(s, a)$ as follows:

$$\begin{cases} V(s) = \sum_{a \in A} \max_{b \in B} Q(s, a, b)\pi_a, \\ Q(s, a, b) = R(s, a, b) + \gamma \sum_{s' \in S} P\left(s' | s, a, b\right) V\left(s'\right), \end{cases} \tag{35}$$

where $R(s, a, b)$ is the current reward value of action a and b, $P(s'|s, a, b)$ is the probability of state s transfer to s' after

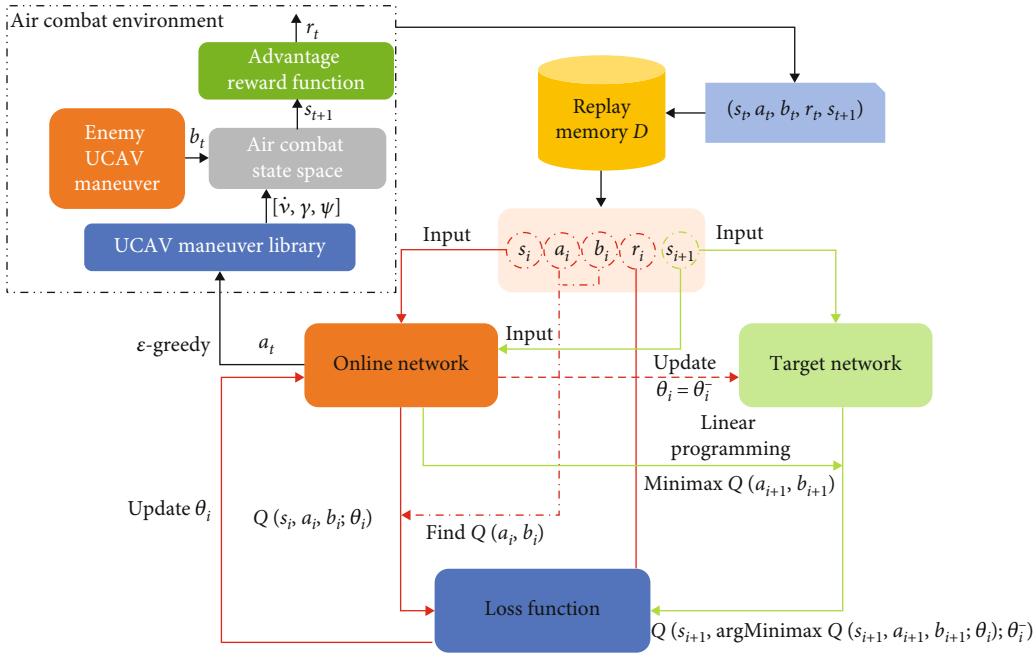


Figure 5: Minimax-DDQN training process diagram.

action a and b are selected, A and B are optional action spaces, and π_a is policy set. The solution of state value function is a dynamic programming iterative process. In order to maximize the action-value function of the current state, the current reward value plus the action-value function that maximizes the next state is required.

According to the Minimax algorithm in the stochastic game, the optimal state value function $V(s)$ under state s is

$$V(s) = \max_{\pi \in PD(A)} \min_{b \in B} \sum_{a \in A} Q(s, a, b)\pi_a, \tag{36}$$

where $PD(A)$ is the discrete probability distribution composed of a series of actions and A and B are the optional action spaces of our UCAV and enemy UCAV, respectively. Based on the Q-learning algorithm, the temporal-difference (TD) [28] error is used to continuously adjust $Q(s, a, b)$ so that the state value of the network output is constantly approaching the true value. Then, the Minimax method in Section 3 is used to construct a linear program to solve the optimal strategy and optimal value function.

5.2. The Establishment of Deep Reinforcement Model for Air Combat. In air combat, the maneuver actions of our UCAV and enemy UCAV are characterized by continuity. When the action space is infinitely continuous, the deep reinforcement model based on Q-learning [29] becomes more complex due to the dynamic nature of transition probability. In order to solve this problem, the TD method in Q-learning is used to constantly approximate the current action-value function. The action-value function based on Minimax algorithm can be obtained by combining formula (36).

$$Q(s, a) = r + \gamma V\left(s'\right). \tag{37}$$

In order to overcome the difficulty in determining the transfer probability P, the action-value function $Q(s, a)$ is updated by adding step size factor α. There is an increment between the Q value calculated by this method and the original Q value, and the size of the updated Q value is adjusted by the step factor. The combined formulas (36) and (37) are as follows.

$$Q(s, a, b) = Q(s, a, b) + \alpha\left(r + \gamma V\left(s'\right) - Q(s, a, b)\right). \tag{38}$$

Compared with traditional Q-learning, Minimax-Q method combines the idea of stochastic game. The maximum value in Q-learning is replaced by Minimax value, and $V(s')$ is obtained by linear programming method. Finally, the optimal strategy under Nash equilibrium condition is obtained.

The Q neural network constructed in this paper consists of input layer, hidden layer, and output layer. The input of the neural network is the situation information of UCAV on both sides, and the output is the maneuver action value that our UCAV can select in the state s. Meanwhile, the UCAV keeps interacting with the environment to get the optimal action so that UCAV can make operational decisions autonomously and improve its intelligence degree. In the process of UCAV air combat decision-making, the convolutional neural network is firstly used to calculate the long-term discount expectation of each state action by approximating the action-value function. Then, the Q network is used as the evaluation basis to iterate all maneuvers in different states and finally select an optimal maneuver action. In order to solve the problem of instability or even nonconvergence when deep neural network approximates the action-value function, the DDQN algorithm adopted in this paper introduces an experience replay mechanism

Algorithm: UCAV air combat maneuver based on Minimax-DDQN algorithm
1. Initialize the memory replay unit D and limit of memory storage.
2. Initialize online Q network and target Q network, and randomly generate parameters θ_i and θ_i^-.
3. **for**episode = **1**, 2, $\cdots\cdots$, M**do**
4. Initialize the UCAV situation information of the enemy and our side to observe the current state s_t
5. **for**step = **1**, 2, $\cdots\cdots$, T**do**
6. Select one of the seven moves in the basic maneuver library at random with the probability of ε
7. Otherwise, select action $a_t = \arg \max_{a\in A} \min_{b\in B} Q(s_t, a, b; \theta_i)$
8. Execute the action a_t
9. Observe reward r_t and next state s_{t+1} are obtained
10. Stored data sample $(s_t, a_t, b_t, r_t, s_{t+1})$ is in D
11. **end for**
12. Take out a group of sample $(s_t, a_t, b_t, r_t, s_{t+1})$ from D randomly.
13. Set $y_i = \begin{cases} r & \text{for terminal } s_{i+1} \\ r + \gamma Q(s_{i+1}, \arg \max_{a_{i+1}\in A} \min_{b_{i+1}\in B} Q(s_{i+1}, a_{i+1}, b_{i+1}; \theta_i); \theta_i^-) & \text{for nonterminal } s_{i+1} \end{cases}$
14. Update the online network using gradient descent based on the loss function $L(\theta_i) = E_{(s_i,a,b,r,s_{i+1})}[(y_i - Q(s_i, a_i, b_i; \theta_i))^2]$
15. Update the target Q network every C turn, set parameter $\theta_i = \theta_i^-$, and gradually decrease ε value until 0.1.
16. **end for**

Algorithm 1: Minimax-DDQN algorithm pseudocode.

[30]. For each time step t, the state, action, reward, and next state obtained by our UCAV's interaction with the environment are stored in the database and a replay memory sequence is formed. During the training, a small batch of experience samples is randomly extracted from the memory replay sequence each time, and the network parameter θ is updated using the stochastic gradient descent (SGD) algorithm. The samples for training deep networks are usually required to be independent of each other. Therefore, the experience replay mechanism reduces the correlation between data by repeatedly sampling historical data and increases the efficiency of data use, thus improving the stability of the algorithm.

5.3. Minimax-DDQN Network Training Process. Reinforcement learning is the process of constantly learning information and modifying strategies in order to achieve the desired reward. The feature selection results of deep convolutional neural network are used as input information of reinforcement learning, and the network is continuously trained to calculate the Q value. DDQN optimizes the loss function through the reward value of environmental feedback. The loss function is to compare the estimated Q function obtained by Bellman equation with the Q value predicted by neural network and obtain a difference, which is used to update the network. The DDQN algorithm is a kind of reinforcement learning algorithm, which can be used to solve the problem of unstable algorithm performance caused by overestimation of action value of Q-learning. DDQN deconstructs the action selection and action evaluation in DQN, which is understood as a copy of network parameters based on DQN as the target network. Therefore, DDQN defines two networks with the same structure but different internal parameters. The advantages and disadvantages of ε greedy algorithm are evaluated by online network parameter θ_i so that the strategy is constantly close to the optimal and the action is selected. The Q value function is evaluated by the target network parameter θ_i^-. The two sets of parameters separate action selection from strategy evaluation and reduce the risk of overestimating Q value. After the current parameters of the target network are fixed for a period of time, the latest parameters of the online network are passed

Table 2: Expert opinion on situation assessment indicators.

Expert	Situation assessment indicators				
	Q_γ	Q_ψ	Q_R	Q_H	Q_V
1	6	8	3	4	2
2	5	7	1	6	3
3	7	9	1	6	3
4	4	9	2	7	5
5	8	9	3	6	4
6	7	8	2	6	5
7	6	9	2	3	4
8	7	8	2	5	4
9	6	9	3	7	4
10	5	8	1	4	3

Table 3: Comparison of weight of AHP and structure entropy method in air combat.

Situation indicators	Weight	AHP	Structure entropy
Track angle	w_1	0.1011	0.2197
Heading angle	w_2	0.3721	0.2856
Distance	w_3	0.1424	0.1064
Height	w_4	0.1189	0.2181
Speed	w_5	0.2655	0.1702

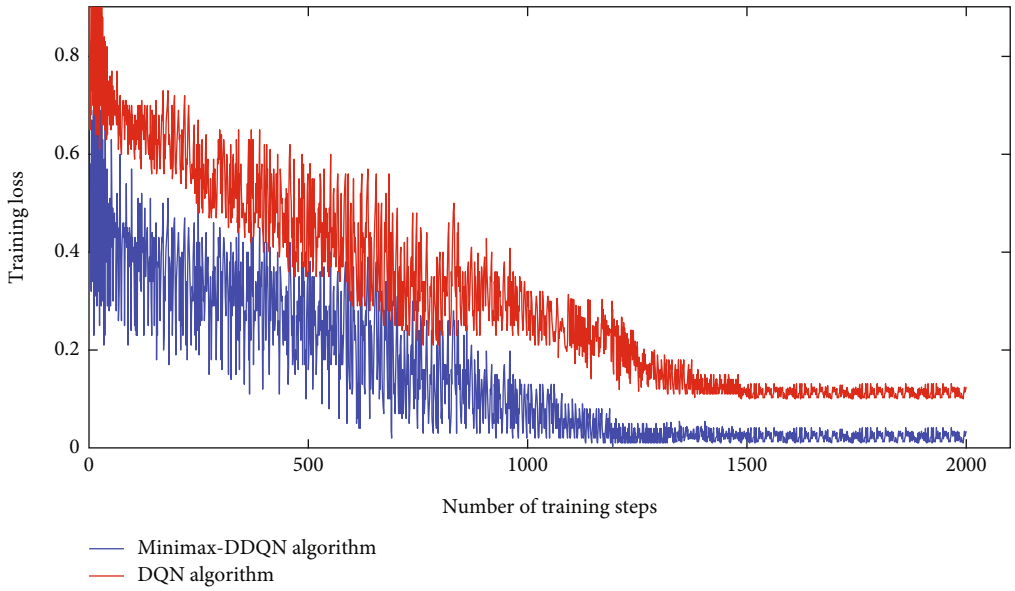


Figure 6: Neural network training loss variation diagram.

to the target network. Since the optimal value function $V(s)$ under state s is obtained by Minimax algorithm in Section 5.1, the deep double Q-learning target value function in Section 2.2.2 is improved to obtain

$$y_t^{\text{DoubleDQN}} = r + \gamma Q\left(s', \arg \max_{a\in A} \min_{b\in B} Q\left(s', a, b; \theta_i\right); \theta_i^-\right). \tag{39}$$

Then, according to formula (6) in Section 2.2.1, the loss function of Minimax-DDQN is

$$L(\theta_i) = E_{(s,a,r,s')}\left[\left(r + \gamma Q\left(s', \arg \max_{a\in A} \min_{b\in B} Q\left(s', a, b; \theta_i\right); \theta_i^-\right) - Q(s, a; \theta_i)\right)^2\right]. \tag{40}$$

The specific training process of Minimax-DDQN is shown in Figure 5.

In summary, the algorithm steps of Minimax-DDQN are given.

Step 1. Initialize the memory replay unit D. Set the upper limit of memory storage 10^6 group data according to the complexity and diversity of air combat situation.

Step 2. Initialize online Q network and target Q network, and randomly generate parameters θ_i and θ_i^-, respectively.

Step 3. Initialize the UCAV situation information of the enemy and our side to observe the current states$_t$.

Step 4. Perform T simulations and training. During each training,

(1) an action is randomly selected from the 7 actions in the basic maneuver library with the probability of ε, and the action $a_t = \arg \max_{a\in A} \min_{b\in B} Q(s_t, a, b; \theta_i)$ is predicted by online network with the probability of $1 - \varepsilon$

(2) perform the action a_t and apply the control quantity $[\dot{v}, \gamma, \psi]$ to UCAV to observe the next state s_{t+1}. The reward r_t is obtained through the advantage reward function

(3) data sample $(s_t, a_t, b_t, r_t, s_{t+1})$ is stored in memory replay unit D

(4) judge whether the air combat round is over. If it is not, loop Step 4. If it is, jump out Step 4

Step 5. Randomly take out a group of sample $(s_i, a_i, b_i, r_i, s_{i+1})$ from memory replay unit D.

Step 6. If s_{i+1} is the terminal, then $y_i = r_i$. If s_{i+1} is not the terminal, the linear programming method is first used to obtain the Minimax state value $V(s_{i+1})$, and then, the target Q value is calculated by Bellman equation:

$$y_i = r + \gamma Q\left(s_{i+1}, \arg \max_{a_{i+1}\in A} \min_{b_{i+1}\in B} Q(s_{i+1}, a_{i+1}, b_{i+1}; \theta_i); \theta_i^-\right). \tag{41}$$

Step 7. Update the online network using gradient descent based on the loss function $L(\theta_i) = E_{(s_i,a,b,r,s_{i+1})}[(y_i - Q(s_i, a_i, b_i; \theta_i))^2]$.

Step 8. Update the target Q network every C turn, set parameter $\theta_i = \theta_i^-$, and gradually decrease ε value until 0.1.

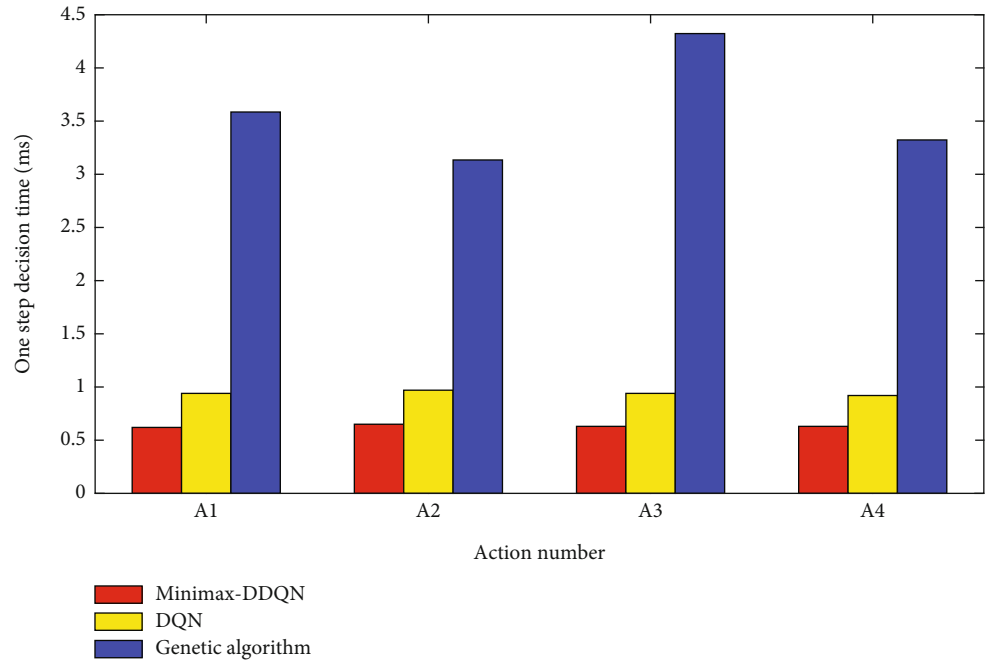


Figure 7: One-step decision time comparison.

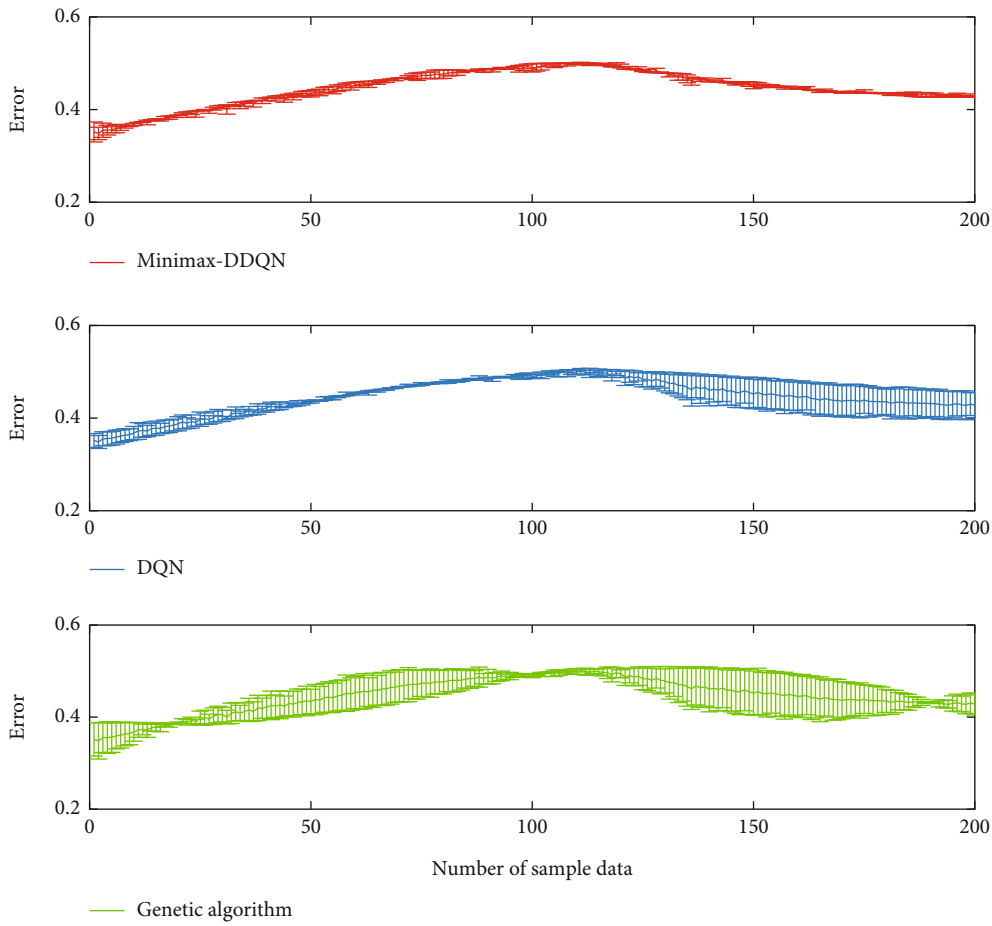


Figure 8: Comparison of three algorithms' accuracy.

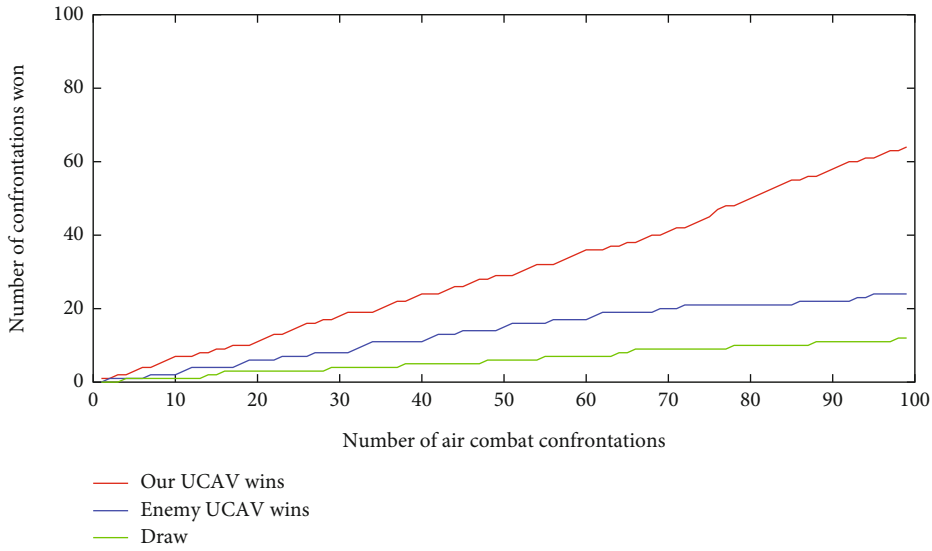


Figure 9: Minimax-DDQN and DQN air combat win time comparison.

Step 9. Repeat Step 3 to Step 8. The number of training cycles is M.

The simulation process of Minimax-DDQN algorithm is described by pseudocode in Algorithm 1.

6. Experimental Simulation and Verification

6.1. Optimization and Verification of Situation Weights Based on Structure Entropy Weight Method. In air combat situation assessment, angle, distance, height, and speed should be considered comprehensively, and the comprehensive situation indicator should be solved by linear weighting method. The situation assessment indicator weights more close to actual combat are helpful to the accuracy of the evaluation advantage reward function and the reasonable reward value. Therefore, this paper introduces a structure entropy weighting method based on the combination of subjective and objective weighting methods. Through consultation with 10 experts and scholars in the field, the importance of each situation indicator was evaluated (important degree is divided into 9 levels, level 9 is the most important, level 1 is the least important), and expert opinions are obtained as shown in Table 2.

By using the structure entropy weight method, the comprehensive weight of evaluation by many experts is used to reduce the one-sidedness of a single expert in the analytic hierarchy process (AHP) method and avoid the subjective judgment of experts. The comparison results are shown in Table 3.

It can be seen from Table 2 that the weight ranking result using the structure entropy weight method is $w_2 > w_1 > w_4 > w_5 > w_3$, and the weight ranking result of the AHP is $w_2 > w_5 > w_3 > w_4 > w_1$. Through consulting experts, it is known that our UCAV has the advantage of track angle and heading angle, which can lead the enemy UCAV to be in the attack area of our air-to-air missile and can provide our UCAV with better missile launch conditions. In addition, our UCAV with a high advantage can find and lock the enemy UCAV more easily, thus taking advantage of the opportunity to improve the probability of air-to-air missile hit. Therefore, the angle situation indicator and height situation indicator have great influence in the evaluation, so the weight value is high. In contrast, the weight ranking results obtained by the structure entropy weight method are more consistent with the conclusions of the consulting experts, indicating that this method is more reasonable.

Table 4: Initial state of UCAV air combat maneuver simulation.

Number	UCAV	$X(m)$	$Y(m)$	$Z(m)$	$V(km/h)$	ψ	γ	μ
1	Our	1200	3260	350	340	−45°	5.5°	0
	Enemy	1500	2600	300	360	−45°	8.3°	0
2	Our	1200	2500	1800	350	−90°	10.2°	0
	Enemy	1200	3000	2000	400	−90°	11.5°	0
3	Our	6460	2000	500	300	135°	4.7°	0
	Enemy	7500	2000	350	320	135°	8.1°	0

6.2. Simulation Experiment of Minimax-DDQN Network. In order to verify the algorithm combining deep reinforcement learning and stochastic game proposed in this paper, the airspace size of UCAV flight simulation is 5 km × 5 km × 5 km. Parameter settings of the Minimax-DDQN algorithm are as follows. A two-layer fully connected feed-forward neural network is used as the online Q network, which has 10 input nodes and 7 output nodes. The network has two hidden layers; the unit size is 1000 and 500, respectively. Sigmoid function is selected as the activation function in the output layer, namely, $S(x) = 1/(1 + e^{-x})$. The number of training cycles is $M = 1000$, and the maximum number of decisions in a single cycle is $T = 500$, and each decision cycle in the training environment is 0.5 s. Discount factor $\gamma = 0.95$ in

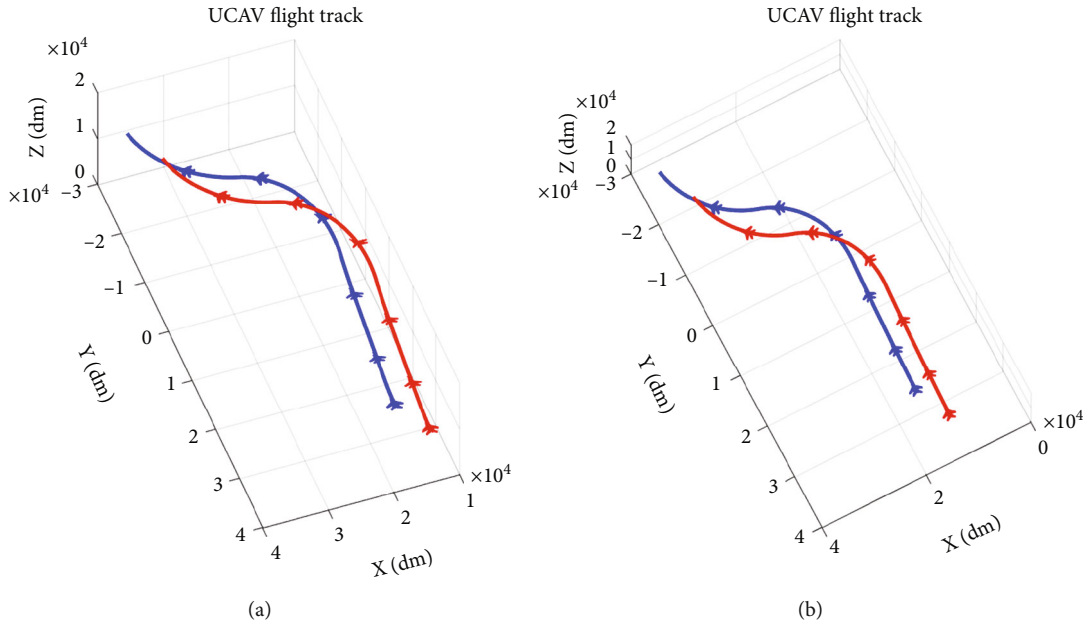


Figure 10: UCAV flight trajectory of the first air combat simulation. (a) Main view of the flight trajectory. (b) Top view of flight trajectory.

deep reinforcement learning. The learning rate of neural network is lr = 0.0005, and the capacity of memory replay unit is 10^6. When the memory unit is full, the latest group of data will cover the first group of data in the memory unit. 1000 training samples are selected from the memory unit to train the neural network, and target network parameters are updated every 3000 steps.

In the simulation, our UCAV selects maneuver strategy according to Minimax-DDQN algorithm, while the enemy UCAV adopts DQN algorithm to select maneuver strategy. During a round, if our UCAV occupies the advantage area, our UCAV gets reward value $r = 1$. If the enemy occupies the advantage area, our UCAV gets reward value $r = -1$. In other cases, our UCAV reward value is $r = 0$. The training loss obtained by Minimax-DDQN algorithm and DQN algorithm after 2000 steps of training is shown in Figure 6. As can be seen from the figure, as the number of training steps increases, the convergence rate of training loss of Minimax-DDQN network is faster than that of DQN algorithm. The Minimax-DDQN network becomes stable at 1200 steps, while the DQN network becomes stable at 1500 steps. In addition, Minimax-DDQN network finally converges to 0.02, and DQN network converges to 0.12. In Minimax-DDQN algorithm, Minimax value function $V(s)$ obtained by linear programming is used to update the Q value in the neural network, which will make the neural network have higher generalization ability and learning efficiency. Action selection and action estimation are separated in DDQN network, which avoids overestimation and improves the accuracy of algorithm. Therefore, Minimax-DDQN algorithm is significantly more efficient than DQN algorithm.

Air combat has very strict real-time requirements for maneuver decision-making. Therefore, it is necessary to test the real-time performance of the maneuver decision model. In this paper, a total of 4 groups of experiments are carried

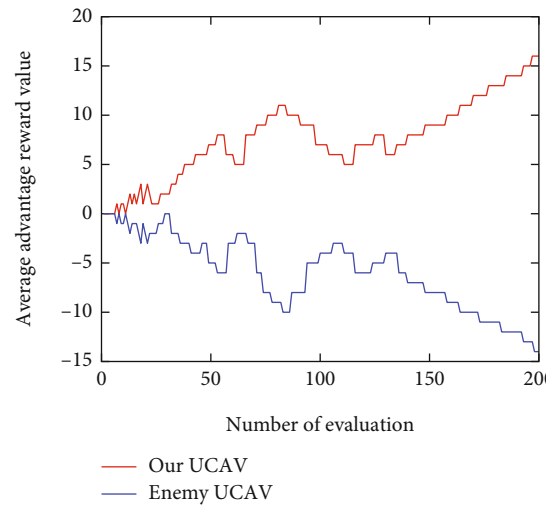


Figure 11: The average reward value of the first air combat simulation.

out. In each experiment, an accurate one-step decision time is obtained by averaging 1000 decision times. The decision time of Minimax-DDQN algorithm, DQN algorithm, and genetic algorithm is compared, and the results are shown in Figure 7. As can be seen from the figure, the one-step decision time of Minimax-DDQN algorithm is maintained at about 0.6 ms, while the decision time of DQN algorithm and genetic algorithm is about 0.9 ms and 3.5 ms. The one-step decision time of Minimax-DDQN algorithm is faster than that of DQN algorithm. In Minimax-DDQN algorithm, convolutional neural network is used to process the high-dimensional continuous situation features of UCAV in air combat and compress the state space to be explored, thus improving the decision-making speed of the algorithm.

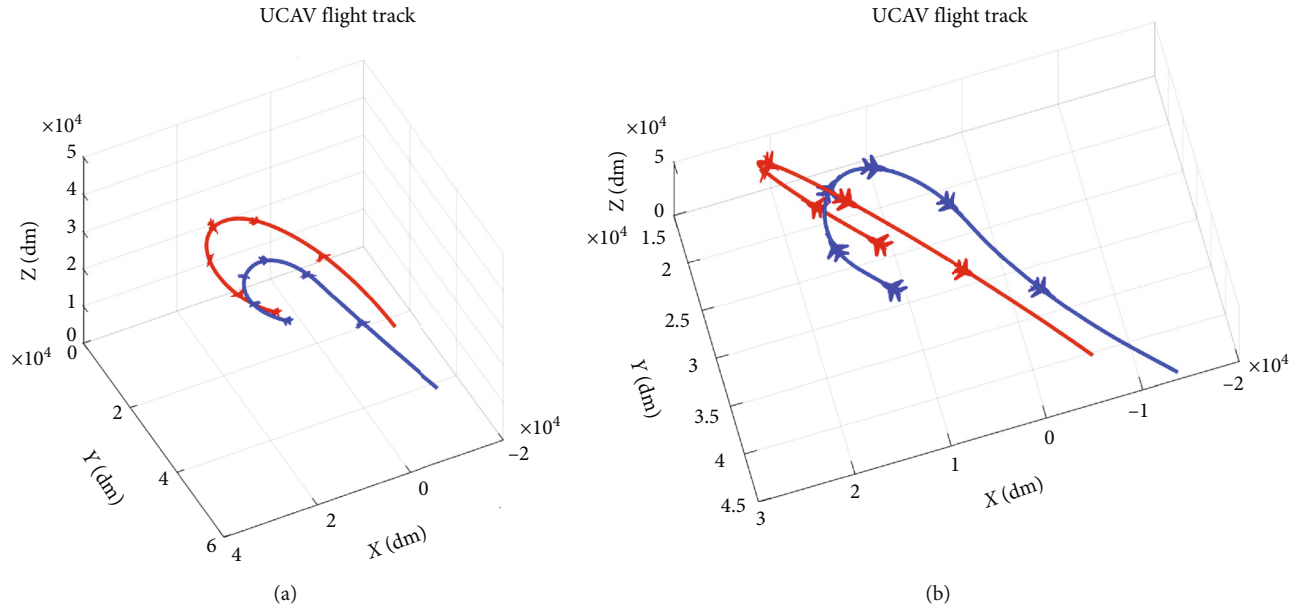


Figure 12: UCAV flight trajectory of the second air combat simulation. (a) Main view of the flight trajectory. (b) Top view of flight trajectory.

The decision time of Minimax-DDQN algorithm is also obviously better than that of genetic algorithm. Because the genetic algorithm needs a lot of loop iterative calculation and the real-time performance is more difficult to meet the requirements of air combat decision, it can be seen that the real-time performance of Minimax-DDQN algorithm is better than that of other commonly used air combat decision algorithms.

The three algorithms are used to fit the flight situation value function of a random segment of enemy UCAV, and Figure 8 is obtained. It can be seen that Minimax-DDQN algorithm has better accuracy than DQN algorithm and genetic algorithm. In the DQN algorithm, the same function is used for the selection and evaluation of actions, so the overestimation that the estimated value is larger than the real value occurs, resulting in the algorithm treating the suboptimal as the optimal. However, the double-Q-learning in Minimax-DDQN algorithm decouples selection from evaluation. The merits and demerits of the ε-greedy algorithm are evaluated and the action is selected through the online network, and the value function is estimated through the target network. Therefore, Minimax-DDQN algorithm can solve the observed overestimation problem and improve the accuracy of the algorithm.

In order to reflect the training effect in real time, the detection is conducted every 20 training steps, and the UCAV with the advantage area is judged as the winner. By comparing the Minimax-DDQN algorithm with the DQN algorithm, the respective strategies are obtained for game confrontation, and the game winning result is finally obtained as shown in Figure 9.

With the increase of training times, our UCAV using Minimax-DDQN algorithm wins more times than the enemy UCAV using DQN algorithm, and the probability of winning increases from 40% to 60% in Figure 9. As a result, Minimax-DDQN can make more accurate and effective decisions than DQN, guiding the UCAV to occupy advantage situation positions.

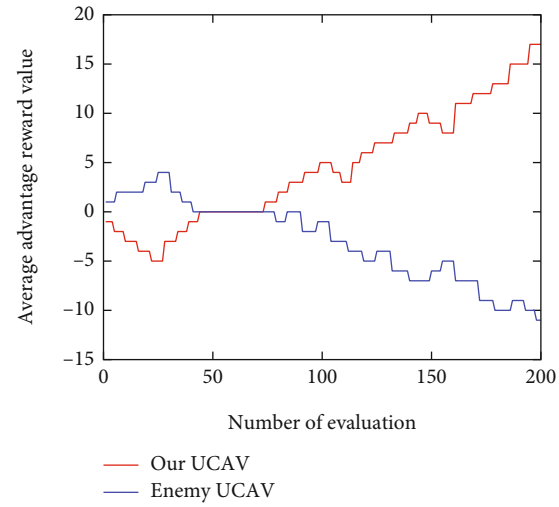


Figure 13: The average reward value of the second air combat simulation.

After the algorithm training, a large number of air combat game simulation are carried out. Before each simulation, the initial position, velocity, track angle, and heading angle of UCAV are randomly generated. Three typical simulation results are selected for analysis in this paper. The initial UCAV status of air combat maneuver simulation is shown in Table 4.

As shown in Figure 10, the initial position of our UCAV (red) is behind the enemy UCAV (blue) in the first air combat training scenario. At this time, our UCAV is in the dominant area, but the enemy UCAV wants to escape the threat of our UCAV by maneuvering, so the enemy UCAV quickly

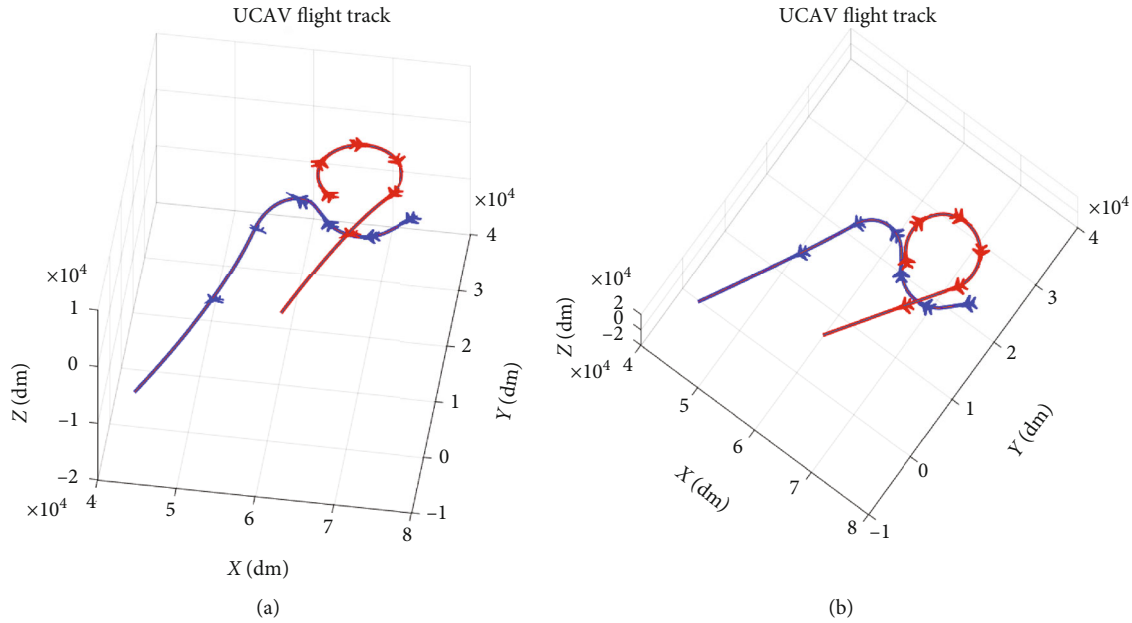


Figure 14: UCAV flight trajectory of the third air combat simulation. (a) Main view of the flight trajectory. (b) Top view of flight trajectory.

climbs up and turned left and then dives down again. In response to a series of maneuvers, our UCAV can maintain a similar angle and altitude to the enemy UCAV and accelerate in pursuit of the enemy UCAV. In the end, our UCAV keeps the enemy UCAV within missile range and continues to maintain our UCAV advantage.

In the first air combat simulation, the reward value of UCAV of the enemy and our side is evaluated every 20 rounds, as shown in Figure 11, to evaluate the times of UCAV in the advantage area. At the beginning of the air combat, our UCAV is in the rear and higher than the enemy UCAV, so the average advantage reward of our UCAV gradually increased and gradually consolidated its advantage position. However, the enemy UCAV tries to maneuver away from our UCAV, which results in a slight decrease in the average advantage reward, and eventually the average advantage reward continues to rise as our UCAV catches up with the enemy UCAV. It can be seen that the advantage reward function established in this paper is completely consistent with the simulation results, and Minimax-DDQN algorithm is faster than DQN algorithm in learning speed and generalization performance, which can better realize the selection of maneuver actions in air combat.

The second UCAV flight trajectory is shown in Figure 12. The initial position of enemy UCAV is behind our UCAV, and our UCAV is at a disadvantage. As the enemy UCAV is in a higher position, our UCAV climbs quickly to the right and watches the maneuver strategy of the enemy UCAV. At this time, the enemy UCAV also begins to turn to the right, and our UCAV soon occupies the height advantage, quickly converts the potential energy into kinetic energy for a dive attack, occupying the dominant position.

Since our UCAV is at a disadvantage in the initial state, the advantage reward of enemy UCAV is greater than our UCAV. During our UCAV maneuver climb, our UCAV is temporarily free of enemy threat, so the average advantage reward for both sides is 0. Until our UCAV heads into the enemy UCAV and gains a height advantage, advantage value of our UCAV gradually becomes positive and keeps rising, as shown in Figure 13.

In the third air combat simulation, the initial position of the enemy UCAV is behind our UCAV, but our UCAV is slightly higher. Enemy UCAV tries to quickly climb up and close to our UCAV. Meanwhile, our UCAV quickly takes a maneuver action of turning to the right. In the face of the pursuit of the enemy UCAV, our UCAV circles behind the enemy UCAV and wins the victory. UCAV flight trajectory is shown in Figure 14.

As shown in Figure 15, although the initial position of enemy UCAV is behind our UCAV, the enemy UCAV is lower than our UCAV. Therefore, the enemy UCAV is not in the advantage area, and the advantage reward value is 0. Until the enemy UCAV climbs quickly and gains an advantage, our UCAV advantage reward is negative and the enemy UCAV advantage reward is positive. Subsequently, our UCAV makes maneuver to gradually transform the disadvantages into advantages. At this time, our UCAV advantage reward value also gradually becomes positive.

It can be seen from the above simulation experiments that Minimax-DDQN algorithm has better decision-making ability than DQN algorithm. First, a reasonable situation assessment model and advantage reward function are established in this paper. The structure entropy weight method is used in the model, which makes the model objectively evaluate the behavior of enemy UCAV. Second, the DDQN algorithm combines the idea of stochastic game. The maximum value in Q-learning is replaced with Minimax value in Minimax theory, and the optimal strategy is obtained under Nash equilibrium condition by using linear

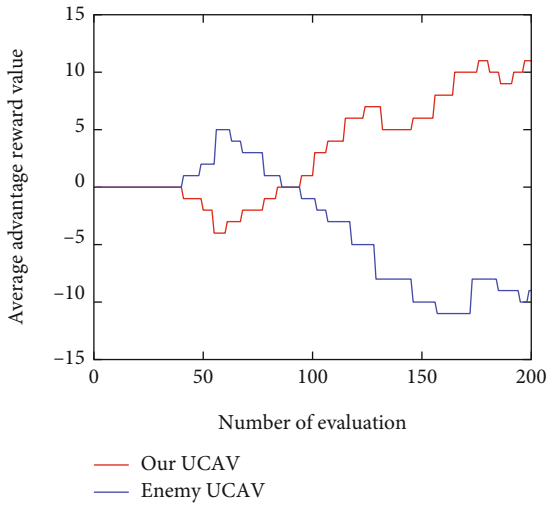


Figure 15: The average reward value of the third air combat simulation.

programming method. Assuming the enemy UCAV has a high level of decision-making ability, our UCAV can choose the most profitable action when the enemy UCAV forces our UCAV benefits to decrease. The Minimax algorithm is used to maximize the benefits in the worst situation. Then, faced with the problem that the algorithm is difficult to converge quickly in the high-dimensional state space, the convolutional neural network is used to process the high-dimensional continuous situation features of UCAV in air combat, eliminate the correlation and redundancy among the situation features, and thus improve the convergence speed of the algorithm. Finally, two different Q networks are used to perform the action selection and estimation process in the DDQN algorithm, thus solving the overestimation problem in DQN algorithm and improving the accuracy of the algorithm. Based on these four points, our UCAV using Minimax-DDQN algorithm can defeat the enemy UCAV using DQN algorithm in the simulation experiment.

7. Conclusion

In this paper, a UCAV maneuver decision algorithm combining deep reinforcement learning and stochastic game theory is proposed based on the characteristics of modern intelligent air combat. The main conclusions are as follows.

(1) Based on UCAV dynamics model and maneuver action library, a reasonable air combat advantage reward function is established in this paper. The situation assessment model is constructed by using the structure entropy weight method, which not only contains the subjective experience and professional knowledge of experts but also can maintain the objective authenticity, so the sample has high credibility

(2) Convolutional neural network is used to process the high-dimensional continuous situation features of UCAV in air combat, which can effectively eliminate the correlation and redundancy between situation features and constantly approach the action value function

(3) The DDQN algorithm in reinforcement learning is introduced to train the agent in interaction with the environment, and the Nash equilibrium strategy is solved by combining with the Minimax algorithm in game theory, and the optimal maneuver decision of UCAV is obtained

Finally, the simulation results show that Minimax-DDQN algorithm can effectively improve the autonomous combat capability of UCAV and has higher intelligence degree and stronger generalization performance than DQN algorithm. The simulation results have high engineering reference value and provide effective support for mission planning and firepower allocation.

Data Availability

The data used to support the findings of this study were related to secrets. Requests for data will be considered by the corresponding author in the future if necessary.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This paper's funding is supported by the graduate student funds of Air Force Engineering University.

References

[1] W. Yang, "Development of future fighters," *Acta Aeronautica et Astronautica Sinica*, vol. 41, no. 6, article 524377, 2020.

[2] K. Q. Zhu and Y. F. Dong, "Study on the design of air combat maneuver library," *Aeronautical Computer Technique*, vol. 31, no. 4, pp. 50–52, 2001.

[3] Y. W. Zhong, J. R. Liu, L. Y. Yang, and G. Z. Shen, "Maneuver library and integrated control system for autonomous close-in air combat," *Acta Aeronautica et Astronautica Sinica*, vol. 29, no. s1, pp. 114–121, 2008.

[4] H. U. Changqiang, D. O. Kangsheng, H. U. Hanqiao, T. A. Shangqin, and Z. H. Zhuoran, "Autonomous air combat maneuver decision using Bayesian inference and moving horizon optimization," *Journal of Systems Engineering and Electronics*, vol. 29, no. 1, pp. 86–97, 2018.

[5] J. Xie, Q. Yang, S. Dai, W. Wang, and J. Zhang, "Air combat maneuver decision based on reinforcement genetic algorithm," *Journal of Northwestern Polytechnical University*, vol. 38, no. 6, pp. 1330–1338, 2020.

[6] X. Wang, W. J. Wang, K. P. Song, and M. W. Wang, "UAV air combat decision based on evolutionary expert system tree," *Ordnance Industry Automation*, vol. 38, no. 1, pp. 42–47, 2019.

[7] C. Rosales, C. S. Miguel, and F. G. Rossomando, "Identification and adaptive PID control of a hexacopter UAV based on

neural networks," *International Journal of Adaptive Control and Signal Processing*, vol. 33, no. 1, pp. 74–91, 2019.

[8] J. Zuo, R. Yang, Y. Zhang, Z. Li, and M. Wu, "Intelligent decision-making in air combat maneuvering based on heuristic reinforcement learning," *Acta Aeronautica et Astronautica Sinica*, vol. 38, no. 10, article 321168, 2017.

[9] Q. M. Yang, J. D. Zhang, G. Q. Shi, J. Hu, and Y. Wu, "Maneuver decision of UAV in short-range air combat based on deep reinforcement learning," *IEEE Access*, vol. 8, no. 13, pp. 363–378, 2020.

[10] W. Wei, J. Wang, Z. Fang, J. Chen, Y. Ren, and Y. Dong, "3U: joint design of UAV-USV-UUV networks for cooperative target hunting," *IEEE Transactions on Vehicular Technology*, vol. 11, no. 9, pp. 1–6, 2022.

[11] Z. Fang, J. Wang, J. Du, X. Hou, Y. Ren, and Z. Han, "Stochastic optimization-aided energy-efficient information collection in Internet of underwater things networks," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 1775–1789, 2022.

[12] Z. Li, Y. Lu, X. Li, Z. Wang, W. Qiao, and Y. Liu, "UAV networks against multiple maneuvering smart jamming with knowledge-based reinforcement learning," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12289–12310, 2021.

[13] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[14] E. A. Smirnov, D. M. Timoshenko, and S. N. Andrianov, "Comparison of regularization methods for ImageNet classification with deep convolutional neural networks," *AASRI Procedia*, vol. 6, no. 25, pp. 89–94, 2014.

[15] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*, vol. 9, no. 5, 1998MIT Press, 1998.

[16] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Playing Atari with deep reinforcement learning," in *Proceeding of the Workshops at the 26th Neural Information Processing Systems 2013*, pp. 201–220, Lake Tahoe, USA, 2013.

[17] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2019.

[18] H. V. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proceedings of the 13th AAAI Conference on Artificial Intelligence*, pp. 2094–2100, Phoenix, Arizona, USA, 2016.

[19] K. Sallhammar, S. J. Knapskog, and B. E. Helvik, "Using stochastic game theory to compute the expected behavior of attackers," in *Symposium on Applications & the Internet Workshops IEEE Computer Society*, pp. 102–105, Trento, Italy, 2005.

[20] M. L. Bertotti and M. Delitala, "FROM discrete kinetic and stochastic game theory to modelling complex systems in applied sciences," *Mathematical Models and Methods in Applied Sciences*, vol. 14, no. 7, pp. 1061–1084, 2004.

[21] P. Williams, "Three-dimensional aircraft terrain-following via real-time optimal control," *Journal of Guidance Control & Dynamics*, vol. 30, no. 4, pp. 1201–1206, 2007.

[22] F. Austin, G. Carbone, M. Falco, H. Hinz, and M. Lewis, "Automated maneuvering decisions for air-to-air combat," in *Proceedings of the Guidance, Navigation and Control Conference*, pp. 659–669, Washington, USA, 1987.

[23] J. S. McGrew, J. P. How, B. Williams, and N. Roy, "Air-combat strategy using approximate dynamic programming," *Journal of Guidance Control and Dynamics*, vol. 33, no. 5, pp. 1641–1654, 2010.

[24] Y. Cao, Y. X. Kou, A. Xu, and Z. F. Xi, "Target threat assessment in air combat based on improved glowworm swarm optimization and ELM neural network," *International Journal of Aerospace Engineering*, vol. 2021, Article ID 4687167, 19 pages, 2021.

[25] Q. Y. Cheng, "Structure entropy weight method to confirm the weight of evaluating index," *Systems Engineering-Theory and Practice*, vol. 30, no. 7, pp. 1225–1228, 2010.

[26] I. Kojadinovic and J. I. Marichal, "Entropy of bi-capacities," *European Journal of Operational Research*, vol. 178, no. 1, pp. 168–184, 2007.

[27] Q. Yang, Y. Zhu, J. Zhang, S. Qiao, and J. Liu, "UAV air combat autonomous maneuver decision based on DDPG algorithm," in *2019 IEEE 15th International Conference on Control and Automation (ICCA)*, Edinburgh, UK, 2019.

[28] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, no. 1, pp. 9–44, 1988.

[29] C. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[30] J. Long and S. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine Learning*, vol. 8, no. 3-4, pp. 293–321, 1992.