machines

MDPI

Article

Air Combat Maneuver Decision Method Based on A3C Deep Reinforcement Learning

Zihao Fan [1], Yang Xu [2,3], Yuhang Kang [4] and Delin Luo [1,*]

[1] School of Aerospace Engineering, Xiamen University, Xiamen 361102, China
[2] School of Civil Aviation, Northwestern Polytechnical University Xian, Xi'an 710072, China
[3] Yangtze River Delta Research Institute of NPU, Suzhou 215400, China
[4] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China
* Correspondence: luodelin1204@xmu.edu.cn

Abstract: To solve the maneuvering decision problem in air combat of unmanned combat aircraft vehicles (UCAVs), in this paper, an autonomous maneuver decision method is proposed for a UCAV based on deep reinforcement learning. Firstly, the UCAV flight maneuver model and maneuver library of both opposing sides are established. Then, considering the different state transition effects of various actions when the pitch angles of the UCAVs are different, the 10 state variables including the pitch angle, are taken as the state space. Combined with the air combat situation threat assessment index model, a two-layer reward mechanism combining internal reward and sparse reward is designed as the evaluation basis of reinforcement learning. Then, the neural network model of the full connection layer is built according to an Asynchronous Advantage Actor–Critic (A3C) algorithm. In the way of multi-threading, our UCAV keeps interactively learning with the environment to train the model and gradually learns the optimal air combat maneuver countermeasure strategy, and guides our UCAV to conduct action selection. The algorithm reduces the correlation between samples through multi-threading asynchronous learning. Finally, the effectiveness and feasibility of the method are verified in three different air combat scenarios.

Keywords: deep reinforcement learning; UCAV; maneuver decision; A3C; asynchronous mechanism

check for updates

Citation: Fan, Z.; Xu, Y.; Kang, Y.; Luo, D. Air Combat Maneuver Decision method Based on A3C Deep Reinforcement Learning. *Machines* **2022**, *10*, 1033. https://doi.org/l10.3390/machines10111033

Academic Editor: Dan Zhang

Received: 6 October 2022
Accepted: 3 November 2022
Published: 5 November 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the development of information technology and the great progress of artificial intelligence, autonomous systems of agents are becoming more and more common, and autonomous maneuvering decision-making of unmanned combat aircraft vehicles (UCAVs) has also become an important issue of current research. UCAVs are capable of autonomous flight to varying degrees: they can be controlled autonomously by a computer or remotely by a human operator [1]. Facing the increasingly complex air combat environment and battlefield situation, UCAVs will face increasingly complex tasks and challenging environments in various practical applications [2].

Domestic and overseas scholars have achieved some progress from extensive research on the decision making problem of air combat maneuvering decision. They have proposed many decision-making theories and methods, which can be approximately classified into three categories according to different solving ideas: methods based on expert knowledge [3–5], methods based on game theory [6–9] and methods based on heuristic learning [10–13]. In [4], combining genetic methods and expert systems, a fast response autonomous maneuver decision model is established based on the maneuver library. In [6], the idea of the differential game is used for the air combat game confrontation problem to obtain the optimal air combat model. In [9], an autonomous decision-making method is proposed, which combines matrix game and genetic algorithms. The improved matrix game algorithm is used to determine the approximate range of the optimal strategy of the UCAVs, and the genetic algorithm is used to search for the optimal strategy in the range. In [12], a

neural network is trained using those samples obtained from simulations, and the future situation is predicted according to current information to select the optimal action. The algorithm can achieve victory in air combat through fewer actions. In [13], the genetic fuzzy tree method is applied to control the flight of a UCAV in an air combat mission. Among the above methods, the expert system method relies on prior knowledge and lacks flexible and effective adjustment methods for complex and changeable battlefield environments. The matrix game method is not suitable for maneuvering decision-making in large action spaces. The deep neural net requires many training samples. The reinforcement learning method does not rely on prior knowledge, but faces the problem of the dimensional explosion of complex state spaces. To solve the above problems, deep reinforcement learning has emerged and gradually become a hot topic in recent years[14–16]. Reinforcement learning does not require training samples and optimizes action strategies by interacting with the environment through trial and error. Deep neural networks effectively solve the problem of dimension explosion of complex state space in reinforcement learning. Since the proposal of deep reinforcement learning, many outstanding algorithms have emerged, which have been successively used in the field of maneuver decision-making of UCAVs, such as DQN [17,18], DDQN [19,20] and DDPG [21,22], etc.

This paper proposes an autonomous maneuver decision model based on an asynchronous advantage actor-critic (A3C) algorithm. The study [23] proposes an asynchronous variant method of the reinforcement learning algorithm, which can effectively break the correlation of inputs caused by online reinforcement learning algorithms, and has stable performance in neural network training without consuming a large number of storage resources. In this paper, the method is applied to the autonomous decision-making of UCAVs in air combat. The main contributions of this paper are as follows: (a) the A3C algorithm is innovatively applied to the air combat maneuver decision problem. Compared with the deep reinforcement learning methods based on the experience pool mechanism, the multi-thread asynchronous mechanism breaks the input correlation to avoid the resource waste caused by the experience replay and speed up the training speed. (b) In response to the problem of the sparse reward in reinforcement learning algorithms, a two-layer reward mechanism combining internal reward and sparse reward is established according to the air combat situation threat assessment index model to promote faster convergence of this algorithm. (c) Simulation results show the performance of the model against the three enemy maneuver strategies of straight flight, hovering flight and DQN algorithm, and verify the feasibility and effectiveness of the model in autonomous maneuver decision-making.

2. Modeling of UCAVs Flight Maneuvers

2.1. UCAV Maneuver Model

To simplify the complexity of the air combat environment, some experimental assumptions are made for the design of the UCAV flight maneuver model: the influences of air resistance and flow velocity are not considered, the mass of UCAVs is constant, and the sideslip angle and angle of attack during a flight are not considered. The acceleration of gravity does not change with the change of flight altitude, and the effect of earth curvature is overlooked.

Based on the above assumptions, the ground inertial coordinate system is established as shown in Figure 1, where the X-axis points to the true north direction of the ground coordinate system, the Y-axis points to the true east direction, and the Z-axis points vertically upward. $\vec{v}_r$ is the velocity vector of the red UCAV, ψ, θ and φ are the yaw angle, pitch angle and roll angle of the UCAV respectively, $\vec{v}_b$ is the velocity vector of the blue UCAV, d is the distance between the two UCAVs, ATA is the deviation angle, representing the angle between the speed of the red UCAV and the distance vector between the two UCAVs, AA is the detachment angle, representing the angle between the speed of the blue UCAV and the distance between the two UCAVs.

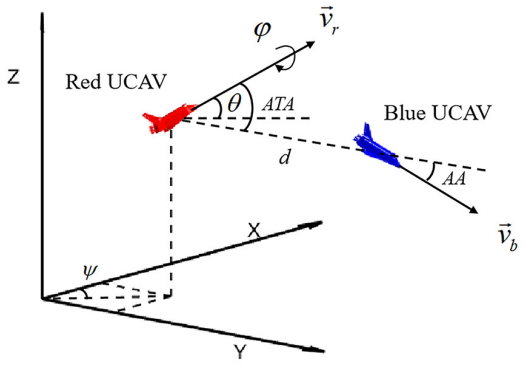

Figure 1. The state description of UCAVs in the three-dimensional coordinate system.

2.1.1. Kinematic Model

On the basis of the above-ground inertial coordinate system, the kinematic formula of UCAV in 3D space is established as follows [24]:

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

2.1.2. Kinetic Model

Because the study of the paper focuses on maneuver decision-making and trajectory generation, and does not involve complex aerodynamic parameters and attitude changes in the six-DOF model, the dynamic model is established by using normal and tangential overload as follows:

$$\begin{cases} \dot{v} = g(N_x - \sin\theta) \\ \dot{\psi} = \frac{gN_z\sin\varphi}{v\cos\theta} \\ \dot{\theta} = \frac{g}{v}(N_z\cos\varphi - \cos\theta) \end{cases} \tag{2}$$

where N_x is the tangential overload of UCAVs, in the direction of flight speed, and can provide power for the UCAV to move forward, N_z is the normal overload of UCAVs, in the direction perpendicular to the speed of flight, and can provide lift for the UCAV. Lastly, g is the gravity acceleration.

2.2. Air Combat Situation Assessment and Judgment of Victory and Defeat

In the game confrontation of air combat, the warring parties need to evaluate the current air combat situation to analyze the advantageous positions for attack and the inferior areas, which are easy to be attacked by the enemy.

This paper introduces the Antenna Train Angle (*ATA*) and Aspect Angle (*AA*) to describe the basic situation in air combat. According to the geometric relationship of the two aircraft situation, the formula is as follows:

$$d = \sqrt{(x_b - x_r)^2 + (y_b - y_r)^2 + (z_b - z_r)^2} \tag{3}$$

$$ATA = \arccos\frac{(x_b - x_r)\cos\psi_r\cos\theta_r + (y_b - y_r)\sin\psi_r\cos\theta_r + (z_b - z_r)\sin\theta_r}{d} \tag{4}$$

$$AA = \arccos\frac{(x_b - x_r)\cos\psi_b\cos\theta_b + (y_b - y_r)\sin\psi_b\cos\theta_b + (z_b - z_r)\sin\theta_b}{d} \tag{5}$$

In an air battle, the side colliding with the rear belongs to the dominant side, and the side that is rear-ended belongs to the disadvantaged side, and two planes facing each other

or against each other belong to the state of mutual equilibrium. Assuming that the red UCAV is our fighter plane, the air combat situation relationship is stipulated as follows. R is 1 means our fighter is victorious, and R is -1 means our fighter is defeated:

$$R = \begin{cases} 1 & ATA < \frac{\pi}{6} \text{ and } AA < \frac{\pi}{6} \text{ and } d < 1000 \\ -1 & ATA > \frac{5\pi}{6} \text{ and } AA > \frac{5\pi}{6} \text{ and } d < 1000 \end{cases} \quad (6)$$

3. Air Combat Maneuver Decision Scheme

3.1. Introduction to Deep Reinforcement Learning

As shown in Figure 2, through continuous interaction with the environment using the trial-and-error method, reinforcement learning can obtain the optimal strategy for a specific task to maximize the expected cumulative payoff of the task.

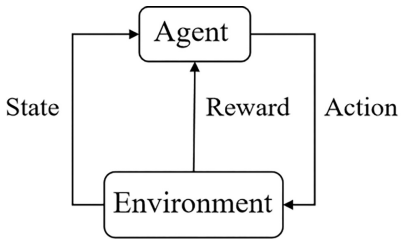

Figure 2. Basic principles of reinforcement learning.

Traditional reinforcement learning methods are used to solve some simple prediction and decision-making tasks, and they are helpless for the problems of high-dimensional state and action space. Deep learning has been a hot topic in recent years and is extensively used in classification and regression problems in various fields. Deep neural networks have good feature representation ability and can be used to solve the problem in which the high-dimensional state and value functions are difficult to represent in reinforcement learning.

3.2. A3C Reinforcement Learning Decision Model

3.2.1. Overall Framework

The A3C asynchronous framework [25] of the UCAV maneuvering decision model is shown in Figure 3. It is mainly composed of a global network and several workgroups, each of which occupies a thread corresponding to an independent UCAV and has its network model to interact with an independent environment. After each UCAV collects a certain amount of data by interacting with the corresponding environment, the gradient about the loss function of the neural net is calculated in the thread and is put into the global network for updating, and then its neural network parameters are replaced with the global network's parameters. The final trained global network can output the optimal policy of UCAVs in the current air combat environment.

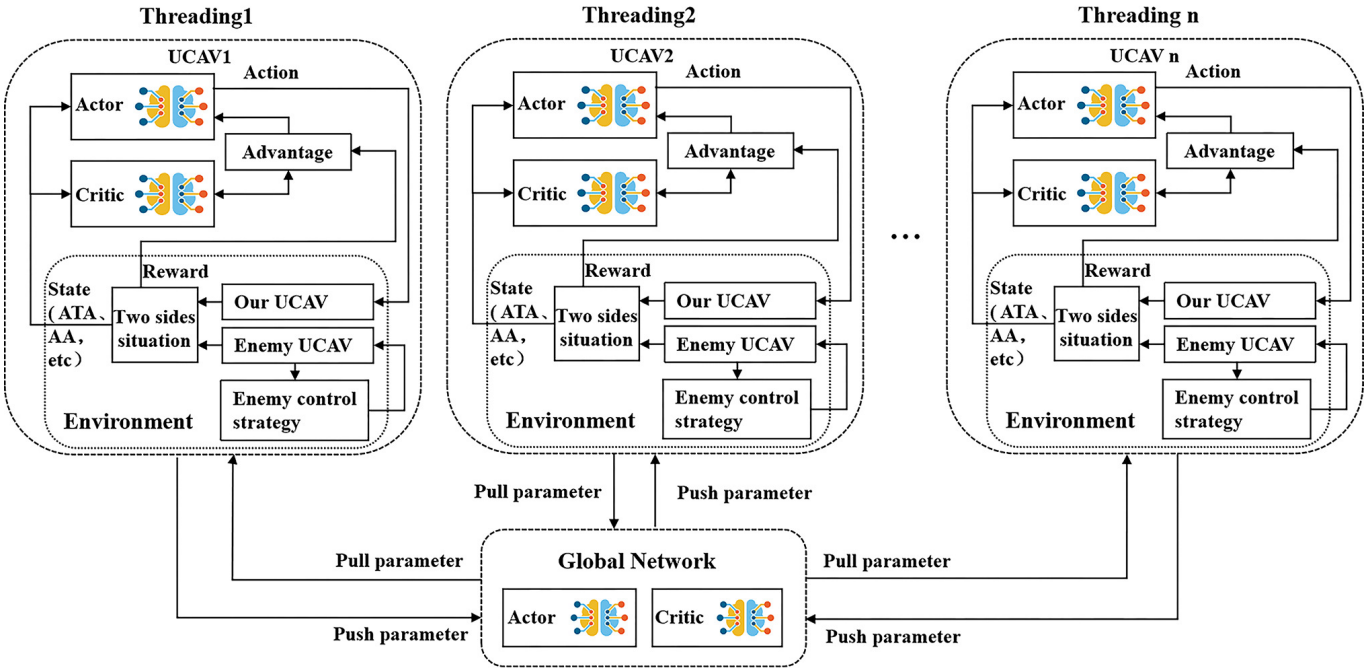

Figure 3. The A3C deep reinforcement learning asynchronous framework.

3.2.2. A3C Related Principles

The A3C deep reinforcement learning method adopts actor-critic (AC) architecture combined with an asynchronous learning mechanism. Then multiple agents can interact with independent environments to learn strategies. Therefore, A3C belongs to an actor-critic framework-based optimization algorithm. Compared with offline learning algorithms, which use the empirical replay mechanism to disrupt the sample correlation, the A3C algorithm collects the sample data asynchronously through the multi-thread mechanism to avoid the waste of resources caused by the empirical replay mechanism.

The AC algorithm combines the value function method and policy gradient method, with actor and critic networks fitting the individual policy function and state value function, respectively. The actor selects the action performed by the agent according to the current state information, and the critic evaluates the quality of the action selected by the actor by calculating the value function, and guides the actor to select an action in the next stage, as shown in Figure 4.

In the real air combat environment transformation process, the state transformation model of the environment will be too complex to be modeled. Therefore, it is necessary to simplify the state transformation model of reinforcement learning reasonably when the reinforcement learning model is used for UCAV autonomous maneuver decision-making problem. It is assumed that the probability of transitioning to the next state is only related to the current state, and has nothing to do with the previous state; that is, the Markov property of environmental state transformation is assumed to follow the Markov Decision Process (MDP). MDP is normally represented in the following six-tuple form

$$< S, A, P, R, \gamma, G >$$

where S is the state space of the environment, A is the action space of the agent, P is the state transition probability in the MDP, R is the instantaneous return obtained by the agent after executing action A, γ is reward decay factor, $\gamma \in [0, 1]$, and G is the subsequent cumulative discount reward from a certain point in the MDP.

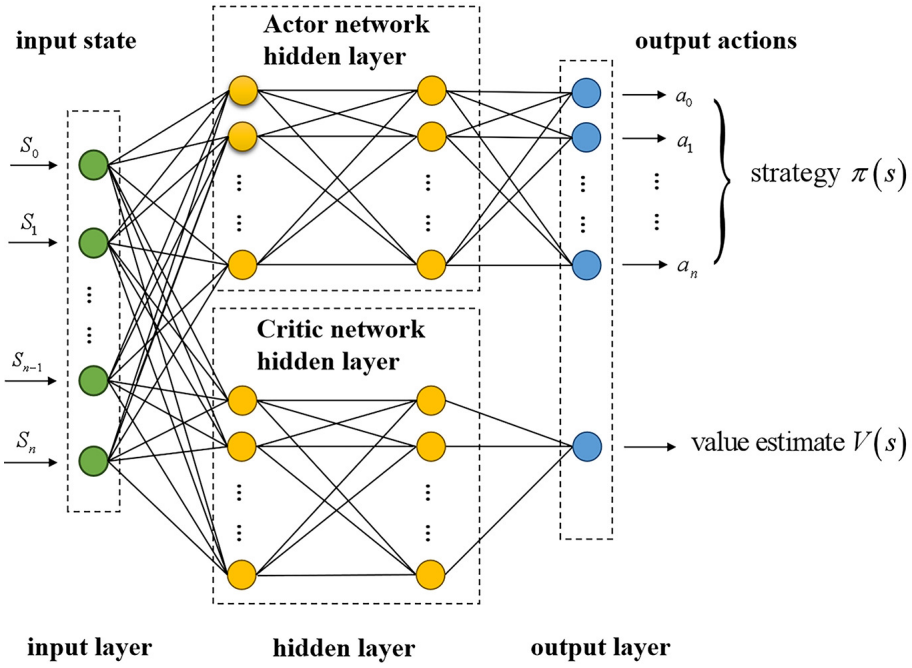

Figure 4. The Actor–Critic network frame.

The strategy π of an individual under the Markov hypothesis is expressed as:

$$\pi(a|s) = P(A_t = a|S_t = s) \tag{7}$$

According to the Bellman equation, the state value function V_π is expressed as:

$$\begin{aligned} V_\pi(s) &= \mathrm{E}_\pi(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots | S_t = s) \\ &= \mathrm{E}_\pi(R_{t+1} + \gamma V_\pi(S_{t+1}) | S_t = s) \end{aligned} \tag{8}$$

The deep neural net is used to approximate the above two functions, and the policy function is approximated as follows:

$$\pi_\theta(s, a) \approx \pi(a|s) \tag{9}$$

The state value function is approximated as follows:

$$V(s, \omega) \approx V_\pi(s) \tag{10}$$

The n-step sampling is used to accelerate convergence, and the formula of n-step return is as follows:

$$G(t, n) = R_t + \gamma R_{t+1} + \ldots + \gamma^{n-1} R_{t+n-1} + \gamma^n V(s\prime) \tag{11}$$

Using n-step return instead of Q-value function, the dominance function is obtained as follows:

$$\begin{aligned} A(s, t) &= Q(s, a) - V(s, \omega) \\ &= G(t, n) - V(s, \omega) \end{aligned} \tag{12}$$

The cross-entropy policy function is used as the loss function of the policy network, and the parameter update formula of the policy network is:

$$d\theta \leftarrow d\theta + \nabla_\theta log \pi_\theta(s_t, a_t) A(s, t) + c \nabla_\theta H(\pi(s_t, \theta)) \tag{13}$$

where $H(\pi(s_t, \theta))$ is the entropy of the strategy, which is used to ensure that the strategy can be fully explored and encourage agents to explore further to avoid falling into local optima, c is the influence factor of entropy, and α is the learning ratio of the actor network.

Taking the advantage function as the critical evaluation point, the loss function of the value network is obtained as:

$$L_c(\omega) = \mathrm{E}\left[\left(R_t + \gamma R_{t+1} + \ldots + \gamma^{n-1} R_{t+n-1} + \gamma^n V(s\prime) - V(s, \omega)\right)^2\right] \tag{14}$$

Therefore, the parameter update formula of the value network is:

$$d\omega \leftarrow d\omega + \frac{\partial\left(R_t + \gamma R_{t+1} + \ldots + \gamma^{n-1} R_{t+n-1} + \gamma^n V(s\prime) - V(s, \omega)\right)^2}{\partial\omega} \tag{15}$$

where β is the learning rate of the critic network.

3.3. State Space and Action Space Descriptions

3.3.1. State Space

To adequately reflect the air combat situation information of the two sides in an air battle, state variables are set to describe the air combat situation, and they are used as the inputs of two networks for learning. Therefore, the more numerous the state variable values are, the more detailed the situation information description will be, but the calculation amount of network learning will be increased. Considering that the state transition effects of various actions are very different when the pitch angles of the UCAV are different, the pitch angles of the two UCAVs are added to the state space as state variables. In this paper, the following 10 state variables are selected, and each variable is normalized as the network input:

$$x = [AA, ATA, d, \Delta H, \Delta v, \theta_r, \theta_b, z_r, v_r, \beta] \tag{16}$$

where, ΔH is the height difference between the two UCAVs, Δv is the speed difference between the two UCAVs, θ_r is the pitch angle of our UCAV, θ_b is the pitch angle of the enemy UCAV, z_r is the height of our UCAV, v_r is the speed of our UCAV, β is the angle between the speeds of the two UCAVs, which can be expressed by the formula as follows:

$$\beta = \arccos(\cos\psi_r \cos\theta_r \cos\psi_b \cos\theta_b + \sin\psi_r \cos\theta_r \sin\psi_b \cos\theta_b + \sin\theta_r \sin\theta_b) \tag{17}$$

3.3.2. Action Space

The UCAV maneuver library of air combat can be approximately classified into two levels. The first level is the typical tactical maneuver library, and the other level is the basic maneuver library. The tactical maneuver library includes tumbling tactics, cobra tactics, high-speed dive tactics, etc. The traditional basic maneuver library includes steady straight, acceleration straight, deceleration straight, left turn, right turn, up pull, and down dive. Figure 5 shows these seven basic maneuvers. The set of basic maneuvers was proposed by the National Aeronautics and Space Administration (NASA). It includes the most commonly used basic maneuvers of UCAVs [26], and can be combined to form new composite actions to form advanced tactical maneuvers.

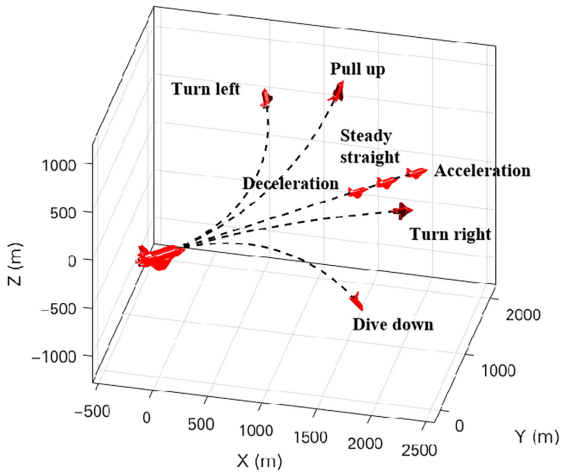

Figure 5. Seven basic maneuvers.

The A3C algorithm is applicable to discrete and continuous action spaces. The focus of this paper is to study the autonomous maneuver decision of UCAVs. It is not necessary to use too advanced tactical actions, and the seven basic maneuver libraries in the basic maneuver library can meet the experimental requirements.

Assuming that tangential overload, method overload and roll angle are constant when a maneuver is selected, seven different combinations of $[N_x, N_y, \varphi]$ are set according to the UCAV dynamics equation, and seven basic maneuvers are coded, that is, the triplet $[N_x, N_y, \varphi]$ is used as the control variables of seven discrete maneuvers in the experimental simulation. Table 1 lists the triplet code of seven basic actions.

Table 1. The triplet code of seven basic actions.

Basic Actions	Tangential Overload	Method Overload	Roll Angle
steady straight	0	1	0
deceleration	2	1	0
acceleration	−1	1	0
turn left	0	8	$-\pi/3$
turn right	0	8	$\pi/3$
pull up	0	8	0
dive down	0	−8	0

3.4. Reward Function

The setting of the reward function is a crucial link in reinforcement learning. The agent can complete the learning process by constantly trying different actions. Meanwhile, the system evaluates each step of the agent and gives a reward or punishment. In this paper, the air combat reward mechanism of UCAV is designed by combining external reward and internal reward. The external reward is the sparse reward. There is a sparse reward at the end of each episode; that is, it is awarded when our UCAV wins and is penalized when our UCAV is defeated or the two sides draw, so the reward function for external rewards R_e can be defined as:

$$R_e = \begin{cases} 20 & ATA<\frac{\pi}{6} \text{ and } AA<\frac{\pi}{6} \text{ and } d<1000 \\ -20 & \left(ATA>\frac{5\pi}{6} \text{ and } AA>\frac{5\pi}{6} \text{ and } d<1000\right) \text{ or } step==STEP \end{cases} \tag{18}$$

where *step* is the current step of the episode's time series and $STEP$ is the maximum length of steps specified in the episode.

Relying only on the sparse reward, it is difficult for agents to receive positive feedback and correct strategy evaluation in the process of random exploration. To solve this problem,

an internal reward is set to stimulate exploration, and an air situation assessment model is established by using the situation information of UCAV in the air combat environment, that is, an internal reward function R_i is defined, which is composed of air combat situation information such as direction, speed, relative altitude and distance.

Negative rewards are given when UCAV is in an unfavorable state. To avoid flying too high or too low, and also to prevent overspeed or stalling during flight, the reward function is set as follows:

$$R_p = \begin{cases} -10 & h < 300 \text{ or } h > 20000 \\ -10 & v < 50 \text{ or } v > 500 \\ 0 & \text{others} \end{cases} \tag{19}$$

In the air combat process, the angle relationship between the two engaged UCAVs is a crucial factor in judging the situational advantage of one over the other. In particular, the deviation angle ATA and detachment angle AA can largely affect the result of air combat. The value ranges of ATA and AA are defined within $[0, \pi]$. The smaller ATA and AA are, the better our air combat situation is. Thus, the angle rewards can be defined as:

$$R_a = 1 - \frac{ATA + AA}{\pi} \tag{20}$$

The distance reward function is set to maintain a certain safe distance between UCAVs:

$$R_d = \begin{cases} e^{\frac{d-d_{\min}}{d_{\min}}} & d < d_{\min} \\ 1 & d_{\min} \le d < d_{\max} \\ e^{-\frac{d-d_{\max}}{d_{\max}}} & d > d_{\max} \end{cases} \tag{21}$$

where: $d_{\min}$ is the nearest safe distance between the two UCAVs, and $d_{\max}$ is the furthest expected distance between the two UCAVs.

Combined with the air battle situation assessment model, the height threat index and speed threat index of both sides are given as follows:

$$T_h = \begin{cases} 1 & \Delta H < -\Delta H_m \\ 0.55 - \frac{0.45 \times \Delta H}{\Delta H_m} & -\Delta H_m \le \Delta H < \Delta H_m \\ 0.1 & \text{others} \end{cases} \tag{22}$$

$$T_v = \begin{cases} 0.1 & v_b < 0.6 v_r \\ -0.5 + \frac{v_b}{v_r} & 0.6 v_r \le v_b < 1.5 v_r \\ 1 & \text{others} \end{cases} \tag{23}$$

where, ΔH_m is the optimal air combat altitude difference, v_r is our UCAV's speed, and v_b is the enemy UCAV's speed.

When the air combat threat index is high, our UCAV is in an unfavorable situation and should be punished accordingly. Therefore, the altitude reward function and speed reward function are established as follows:

$$R_h = 1 - T_h \tag{24}$$

$$R_v = 1 - T_v \tag{25}$$

In order to make the UCAV learn to win with fewer steps, an extra step reward function R_s is set. According to the above air combat situation reward functions, the internal reward function in a 3D environment is defined as follows:

$$R_i = \omega_a R_a + \omega_d R_d + \omega_h R_h + \omega_v R_v + R_s + R_p \tag{26}$$

where, $\omega_a, \omega_d, \omega_h, \omega_v$ is the weight coefficient of each air combat situation information.

4. Experimental Simulation and Discussion

4.1. Experimental Parameter Setting

In the simulation, it is assumed that our UCAV is the red UCAV. The A3C maneuver decision algorithm is used for maneuver guidance of our UCAV, and the enemy UCAV is the blue UCAV. At the beginning of each episode, the two UCAVs are initialized to the given initial state, and the two UCAVs decide at the same time, every 0.5 s, which is called an execution step. If the winner is not decided within 200 steps, the episode will be judged as a tie. When evaluating the internal reward function after each maneuver selection, the influence degree of angle, distance, altitude, and speed on the situation is considered comprehensively, so the weight parameters of four situation information in the internal reward function are set as 0.45, 0.25, 0.15 and 0.15, respectively.

In this paper, three simulation conditions are designed to verify the effect of the A3C maneuver decision model. The first two groups set the enemy UCAV as fixed strategies, and do steady flight and spiral climb, respectively. The third group sets the trained DQN algorithm as the maneuver strategy of the enemy UCAV. To speed up the calculation, the starting position coordinate of the red UCAV is set as (0, 0, 3000), the initial speed is 250 m/s, and the initial pitch angle is 0 degrees in simulation. The parameters of two UCAVs' initial states are shown in Table 2.

Table 2. The initial states of two UCAVs in three groups of experiments.

Condition	Enemy Strategy		Stating Coordinate (m)	Speed (m/s)	Yaw Angle (Degree)	Pitch Angle (Degree)
1	steady straight	our UCAV	(0, 0, 3000)	250	180	0
		enemy UCAV	(1000, 5000, 3000)	200	0	0
2	spiral climb	our UCAV	(0, 0, 3000)	250	180	0
		enemy UCAV	(3000, 3000, 2800)	250	0	15
3	DQN algorithm	our UCAV	(0, 0, 3000)	250	180	0
		enemy UCAV	(3000, 3000, 3000)	200	0	0

Through the processing of the Actor and Critic network, the real-time state of UCAVs is mapped to the probability of different actions being selected and the value of the current state. To fully exploit the hidden features of a given input state, it is necessary to design an appropriate deep neural network structure. Since this experiment does not involve high-dimensional data such as screen images, a forward fully-connected network is used as the network structure of the Actor and Critic. After continuous attempts and comparisons of neural networks with different layers and neurons, the parameters of the network are finally set, as shown in Table 3.

Table 3. Actor–Critic Network parameters.

Network	Network Layer	Neurons	Activation Function	Learning Rate
Actor	input layer	10	-	0.0001
	hidden layer1	128	ReLu	
	hidden layer2	64	ReLu	
	output layer	7	Softmax	
Critic	input layer	10	-	0.0005
	hidden layer1	64	ReLu	
	hidden layer2	32	ReLu	
	output layer	1	-	

4.2. Algorithm Flow

In each group of simulation experiments, five UCAVs are set as our reinforcement learning working group to learn independently from the fixed maneuvering enemy UCAV environment. The reward decay factor is set to 0.9. Each thread updates the global network every 20 steps, and then the net's parameters in this thread are updated to the global neural net's parameters. Because the A3C algorithm uses an asynchronous multi-threading mechanism, the algorithm flow focuses on listing the flow of any thread. The algorithm flow is as shown in Algorithm 1. In this algorithm, a multi-core CPU is used to create multi-threads. Each thread has a virtual UCAV that interacts with the environment independently. The action is selected according to the output strategy, and the return is calculated, and local gradient updates are accumulated by formulas 13 and 15. These accumulated gradients do not update the local network parameters of the current thread, but are sent to the global network for parameter updates. Finally, the UCAV can perform the optimal action through the trained global network.

Algorithm 1 : Autonomous Maneuver Decision of UCAV based on A3C

Input: The state feature dimension S, the action dimension A, the global network parameters θ and ω, the neural network parameters θ', ω' of the current thread, the maximum number $EPISODE$ of iterations, the maximum length $STEP$ of one episode's time series within a thread. The reward decay factor γ and the number N of n-step rewards. The entropy coefficient c and the learning rates α, β.

Output: The global network parameters θ and ω that have been updated.

Initialize the time series step $T = 1$ globally shared
for $episode = 1 \rightarrow EPISODE$ **do**
  Initialize the thread-private time series step $t = 1$
  Reset Actor and Critic network gradient updates $d\theta' = 0, d\omega' = 0$
  Synchronizes parameters $\theta' = \theta, \omega' = \omega$
  Initializes the state s_t
  while $t <= STEP$ and s_t is not the termination state and T mod N != 0 **do**
    Select action a_t based on the output strategy $\pi_\theta(s_t, a_t)$ of the Actor network
    Perform action a_t
    Receive reward r_t and next state s_{t+1}
    Update steps $T \leftarrow T + 1$, $t \leftarrow t + 1$
  end while
  Calculate the return $G(t, t)$ for the state of the last time series:

$$G(t,t) = \begin{cases} 0 & \text{terminalstate} \\ V(s_t, \omega') & \text{others} \end{cases}$$

  for $i = t - 1 \rightarrow 1$ **do**
    Calculate the return for each moment:

$$G(t,i) = r_i + \gamma G(t, i+1)$$

    Accumulate local gradient updates $d\theta'$ of the Actor network:

$$d\theta \leftarrow d\theta + \nabla_\theta log \pi_\theta(s_t, a_t) A(s,t) + c \nabla_\theta H(\pi(s_t, \theta))$$

    Accumulate local gradient updates $d\omega'$ of the Critic network:

$$d\omega \leftarrow d\omega + \frac{\partial (G(t,i) - V(s,\omega))^2}{\partial \omega}$$

  end for
  Update the global neural net's parameters $\theta = \theta + \alpha d\theta'$ and $\omega = \omega + \beta d\omega'$
end for

4.3. Analysis of Experimental Results

In this paper, the hardware environment is Intel(R) Core(TM) I5-8400 CPU @ 2.80 Ghz, 16G random access memory, and the software environment is Python language and TensorFlow deep learning open source framework. Since A3C adopts an asynchronous learning mechanism, the exploration and learning effect of each UCAV is far from the same when interacting with the environment. Therefore, every 100 episodes are selected as a training stage, and the win rate, average episode reward and average episode step number of our UCAV are counted. The algorithm iteration is finished until the episode reward and episode step number converge.

The simulation results of the first group of experiments are given. Figure 6 shows the reward convergence of our UCAV. At the beginning of the training, the episode reward is very low, and there is a tendency for random fluctuations, which indicates that our UCAV is almost in a random exploration state at the early stage of the training. Around the 300th training stage, the reward begins to converge and eventually converges to close to the value of 0. Figure 7 shows the statistical trend curve of episode steps, and Figure 8 shows the change curve of our UCAV win rate. The enemy UCAV is steady and straight and our UCAV is in the exploration stage in the early stage, so both sides have been tied, and the step number of the episode is kept at the maximum number of steps. It is not until the 50th training stage that our UCAV begins to learn the tactics to defeat the enemy UCAV. During this period, it still keeps exploring, but the win rate gradually increases and the average number of episodes gradually decreases. At about the 350th training stage, the number of episodes tends to be stable, converging to about 47 steps, and the win rate of our UCAV also converged to about 0.95.

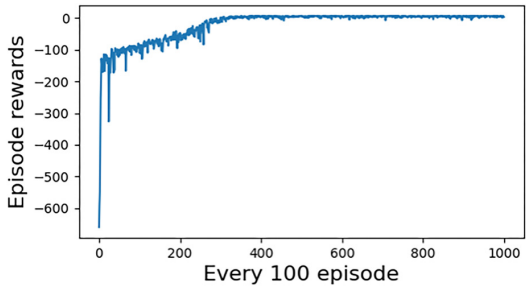

Figure 6. Our UCAV reward change curve in case 1.

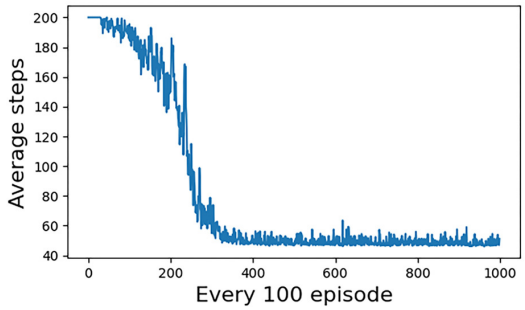

Figure 7. The change curve of turn steps in case 1.

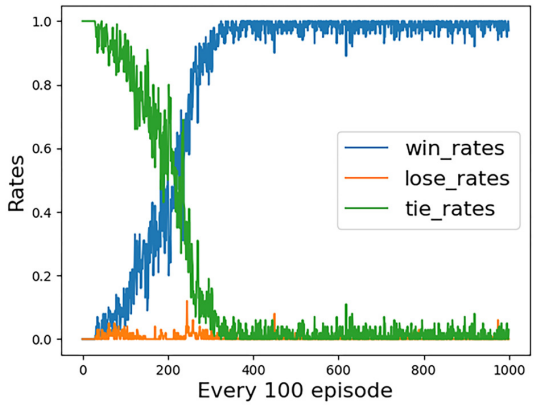

Figure 8. Our UCAV win rate change curve in case 1.

To show our UCAV learning situation in the training process more visually, the trajectory of two UCAVs in certain episodes in the middle and late training period is selected in the first experiment as shown in Figures 9 and 10. After a certain stage of training, the network parameters of the Actor–Critic have been preliminarily formed, which can purposefully guide our UCAV to make favorable actions, such as raising the height to gain a height advantage, going around behind the enemy UCAV to gain an angle advantage, and narrowing the distance to win air battle. In the later stage of training, the network parameters of the Actor–Critic continue to be updated and optimized, and our UCAV has learned the strategy to win at a faster speed. Figure 11 shows the changes in ATA, AA and distance d in the late training period. When the distance is less than 1000 m and the deviation angle and detachment angle are less than $\frac{\pi}{6}$, our UCAV wins.

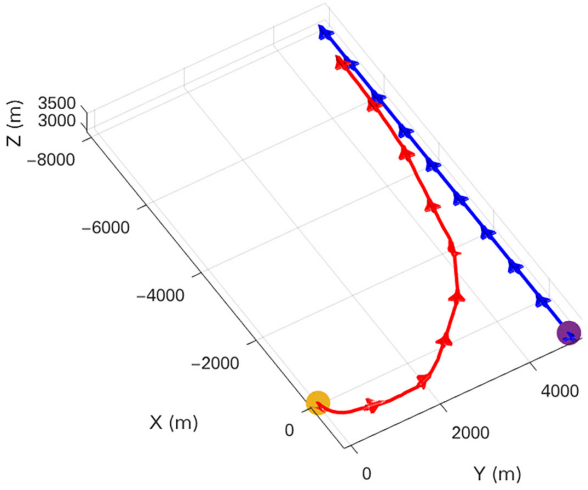

Figure 9. The two UCAV tracks in the middle training period in case 1.

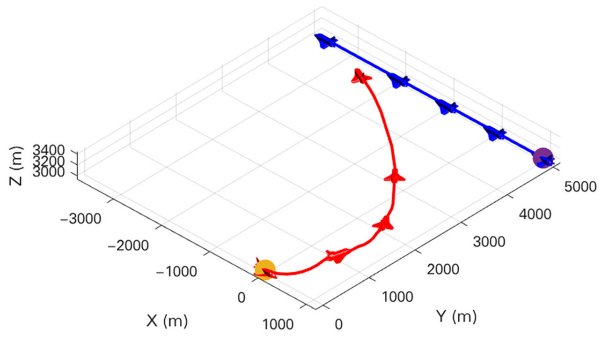

Figure 10. The two UCAV tracks in the later training period in case 1.

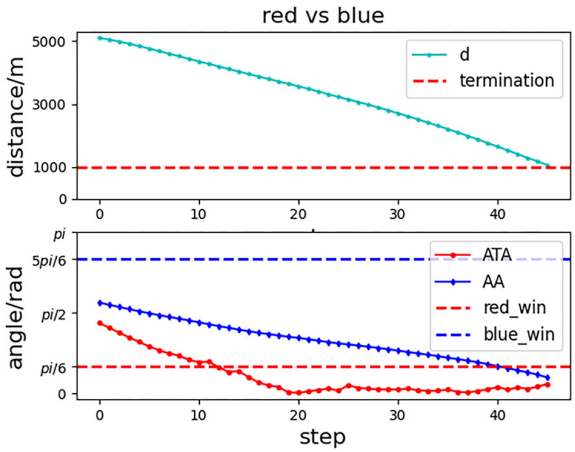

Figure 11. The changes of *ATA*, *AA* and *d* in the later training period in case 1.

In the second experiment, the enemy UCAV takes the maneuver of circling upward. Our reward and win rates converge more slowly than in the first experiment. As shown in Figures 12 and 13, the win rate of our UCAV starts to increase in the 120th training stage, and the reward and win rate converge in the 520th training stage. Figures 14 and 15 show the flight trajectories of two UCAVs in the middle and late training episodes of the second experiment. It can be seen that our UCAV has learned the winning strategy in the middle training period. Our UCAV can enter the hovering range of the enemy UCAV during the air battle, follow behind the enemy UCAV and keep closing the distance, and win the air battle. In the late period of training, our UCAV shows higher maneuvering ability and has learned to quickly intercept the enemy UCAV with a smaller turning radius. Figure 16 shows the statistical chart of ATA, AA and d in the late training period.

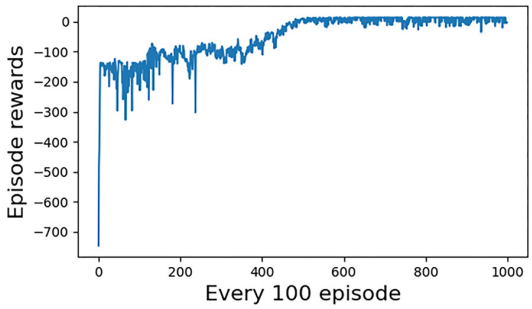

Figure 12. Our UCAV reward change curve in case 2.

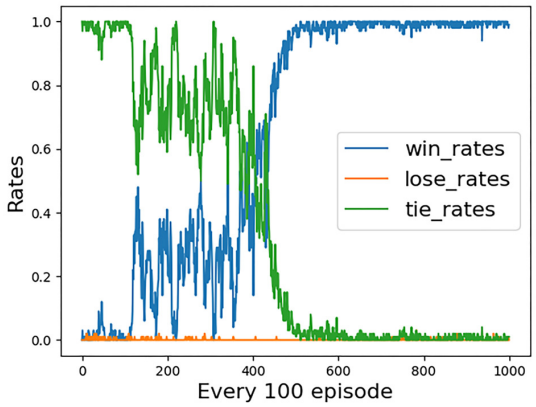

Figure 13. Our UCAV win rate change curve in case 2.

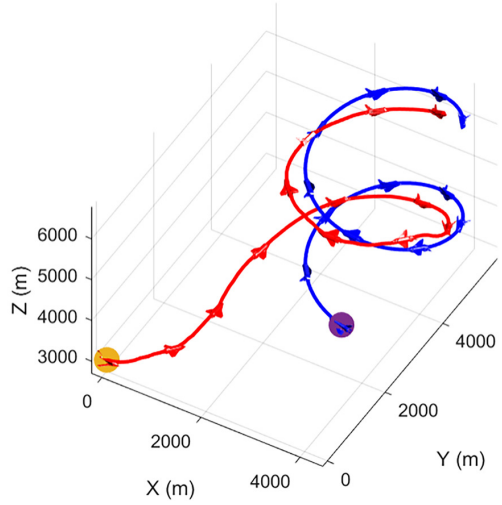

Figure 14. The two UCAV tracks in the middle training period in case 2.

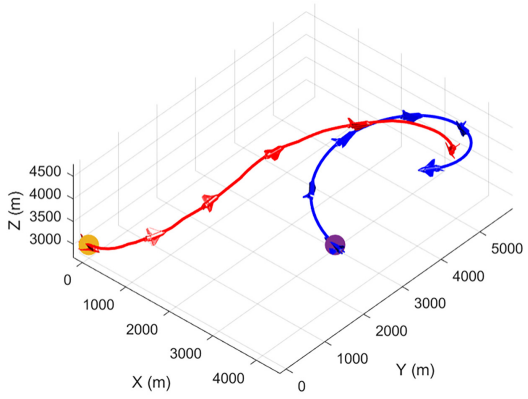

Figure 15. The two UCAV tracks in the later training period in case 2.

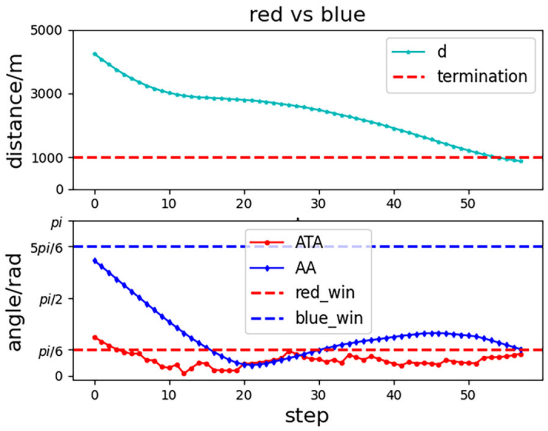

Figure 16. The changes of *ATA*, *AA* and *d* in the later training period in case 2.

In the third experiment, the enemy UCAV uses a DQN algorithm as the maneuver strategy to select actions. The DQN algorithm has been trained for 100,000 episodes against the greedy algorithm and enabled the UCAV to confront others independently. The win rate of our UCAV in the training process is shown in Figure 17. It can be seen that, with the progress of the training stage, the win rate gradually increases and finally converges to about 0.7. The movement track of two UCAVs in one episode in the late training period is selected as shown in Figure 18. In this episode, our UCAV primarily narrows the distance

with the enemy UCAV; meanwhile, the enemy UCAV tries to fly upward to increase the altitude advantage. Our UCAV quickly implements Flick Half Roll tactical action to win from outside the track of the enemy UCAV to behind the enemy UCAV. Figure 19 shows the statistical chart of the change rule of *ATA*, *AA* and *d* of this episode.

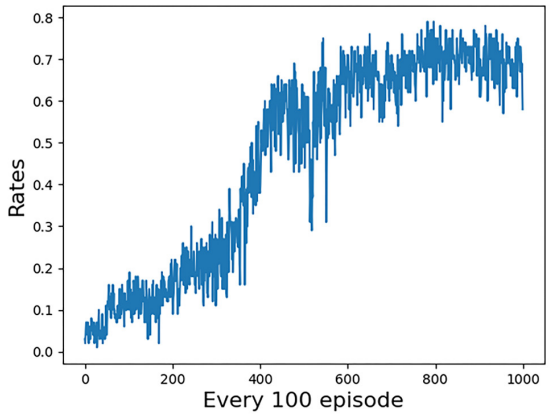

Figure 17. Our UCAV win rate change curve in case 3.

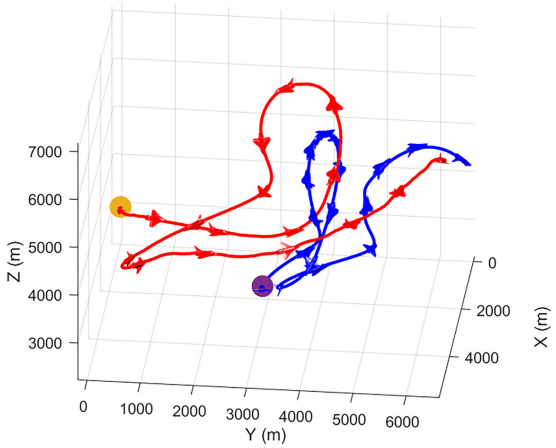

Figure 18. The two UCAV tracks in the later training period in case 3.

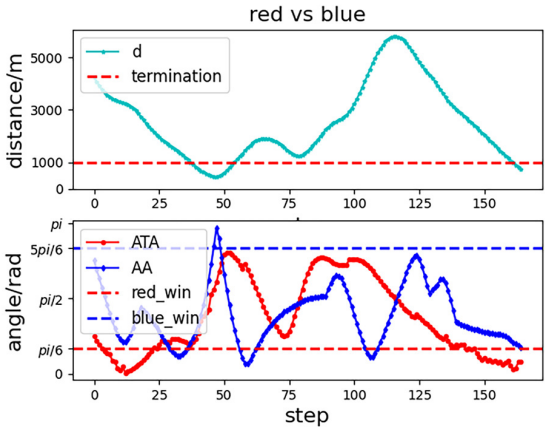

Figure 19. The changes of *ATA*, *AA* and *d* in the later training period in case 3.

In the first two cases, the enemy UCAV adopts a fixed strategy, and our A3C algorithm can quickly catch up with the enemy UCAV and quickly win after a period of training. It can be seen that the decision scheme proposed in this paper can effectively guide a UCAV to make maneuver decisions. In the third case, our UCAV equipped with the A3C

algorithm and the enemy UCAV equipped with the DQN algorithm are engaged in a fight. After training, our UCAV can suppress the enemy UCAV and converge to a higher winning rate, which indicates that the application of the A3C algorithm in the UCAV autonomous maneuver decision-making problem can be more effective.

5. Conclusions

In this paper, an asynchronous framework of an Actor–Critic network based on the advantage function is built to study the autonomous maneuver decision of UCAVs. To calculate the state transition and estimate the situation in an air combat confrontation, a physical model of UCAV is established, and a reward function mechanism combining sparse reward and internal reward is designed. The continuous state space and discrete action space based on control variables are set up. Then the neural network and multi-threading model are built to realize the asynchronous advantage AC framework where the UCAV of each thread learns from the environment independently and the parameters of the global AC network are updated regularly. Finally, the model is trained under three scenarios. When the enemy UCAV performs fixed maneuvers, our UCAV equipped with the A3C algorithm shows excellent autonomous maneuvering decision-making ability, and when the enemy UCAV adopts the DQN algorithm, our UCAV can still converge the reward and achieve a high winning rate. Therefore, the effectiveness and feasibility of the UCAV autonomous maneuver decision-making method based on A3C deep reinforcement learning are verified by observing the rewards, win rates and confrontation trajectories.

Of course, there are also some improvements in this paper. For example, the fixed initial position can be replaced with a random initial position, which can make the algorithm more fully trained. In addition, the seven candidate actions can be further subdivided, which can make the UCAV maneuver more flexible, so as to make a more beautiful adversarial maneuver trajectory. In future work, we will further study the above improvement points.

Author Contributions: Conceptualization, D.L. and Z.F.; methodology, D.L.; software, Z.F. and Y.X.; validation, D.L., Z.F., Y.K. and Y.X.; formal analysis, Y.K.; investigation, Y.X.; resources, Y.K.; writing—original draft preparation, Z.F.; writing—review and editing, D.L., Y.K. and Y.X.; visualization, Z.F.; funding acquisition, D.L. and Y.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China under Grant No.61673327, and Basic Research Programs of Taicang, 2021 under Grant No.TC2021JC28, Fun damental Research Funds for the Central Universities under Grant No.G2021KY05116 and No.G2022WD010 26, and Industrial Development and Foster Project of Yangtze River Delta Research Institute of NPU, Taicang under Grant No.CY20210202.

Data Availability Statement: Data sharing is not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Azar, A.T.; Koubaa, A.; Mohamed, N.A. Drone Deep Reinforcement Learning: A Review. *Electronics* **2021**, *10*, 999–1028. [CrossRef]
2. Zhang, Y.; Luo, D.L. Editorial of Special Issue on UAV Autonomous, Intelligent and Safe Control. *Guid. Navig. Control.* **2022**, *1*, 1–6. [CrossRef]
3. Burgin, G.H. *Improvements to the Adaptive Maneuvering Logic Program*; NASA: Washington, DC, USA, 1986.
4. Wang, X.; Wang, W.; Song, K. UAV Air Combat Decision Based on Evolutionary Expert System Tree. *Ordnance Ind. Autom.* **2019**, *38*, 42–47.
5. Fu, L.; Xie, F.H.; Meng, G.L. An UAV Air-combat Decision Expert System based on Receding Horizon Contro. *J. Beijing Univ. Aeronaut. Astronaut.* **2015**, *41*, 1994–1999.
6. Gracia, E.; Casbeer, D.W.; Pachter, M. Active Target Defence Differential Game: Fast Defender Case. *IET Control. Theory Appl.* **2017**, *11*, 2985–2993. [CrossRef]
7. Hyunju, P. Differential Game Based Air Combat Maneuver Generation Using Scoring Function Matrix. *Int. J. Aeronaut. Space Sci.* **2016**, *17*, 204–213.

8. Ha, J.S.; Chae, H.J.; Choi, H.L. A Stochastic Game-theoretic Approach for Analysis of Multiple Cooperative Air Combat. In Proceedings of the 2015 American Control Conference (ACC), Chicago, IL, USA, 1–3 July 2015; pp. 3728–3733.
9. Deng, K.; Peng, H.Q.; Zhou, D.Y. Study on Air Combat Decision Method of UAV Based on Matrix Game and Genetic Algorithm. *Fire Control Command Control* **2019**, *44*, 61–66. +71.
10. Chao, F.; Peng, Y. On Close-range Air Combat based on Hidden Markov Model. In Proceedings of the 2016 IEEE Chinese Guidance, Navigation and Control Conference, Nanjing China, 2–14 August 2016; pp. 688–695.
11. Li, B.; Liang, S.Y.; Chen, D.Q.; Li, X.T. A Decision-Making Method for Air Combat Maneuver Based on Hybrid Deep Learning Network. *Chin. J. Electron.* **2022**, *31*, 107–115.
12. Zhang, H.P.; Huang, C.Q.; Xuan, Y.B. Maneuver Decision of Autonomous Air Combat of Unmanned Combat Aerial Vehicle Based on Deep Neural Network. *Acta Armamentarii* **2020**, *41*, 1613–1622.
13. Ernest, N.; Carroll, D.; Schumacher, C. Genetic Fuzzy based Artificial Intelligence for Unmanned Combat AerialVehicle Control in Simulated Air Combat Missions. *J. Def. Manag.* **2016**, *6*, 1–7.
14. Kim, J.; Park, J.H.; Cho, D.; Kim, H.J. Automating Reinforcement Learning with Example-Based Resets. *IEEE Robot. Autom. Lett.* **2022**, *7*, 6606–6613. [CrossRef]
15. Wang, H.T.; Yang, R.P; Yin, C.S.; Zou, X.F.; Wang, X.F. Research on the Difficulty of Mobile Node Deployment's Self-Play in Wireless Ad Hoc Networks Based on Deep Reinforcement Learning. *Wirel. Commun. Mob. Comput.* **2021**, *11*, 1–13.
16. Kurzer, K.; Schörner, P.; Albers, A.; Thomsen, H.; Daaboul K.; Zöllner, J.M. Generalizing Decision Making for Automated Driving with an Invariant Environment Representation using Deep Reinforcement Learning. In Proceedings of the 2021 IEEE Intelligent Vehicles Symposium (IV), Nagoya, Japan, 11–17 August 2021; Volume 39, pp. 994–1000.
17. Yang, Q.M.; Zhang, J.D.; Shi, G.Q.; Hu, J.W.; Wu, Y. Maneuver Decision of UAV in Short-range Air Combat Based on Deep Reinforcement Learning. *IEEE Access* **2020**, *8*, 363–378. [CrossRef]
18. Hu, D.Y.; Yang, R.N.; Zuo, J.L.; Zhang, Z.; Wu, J.; Wang, Y. Application of Deep Reinforcement Learning in Maneuver Planning of Beyond-Visual-Range Air Combat. *IEEE Access* **2021**, *9*, 32282–32297. [CrossRef]
19. Yfl, A.; Jpsa, B.; Wei, J.A. Autonomous Maneuver Decision-making for a UCAV in Short-range Aerial Combat Based on an MS-DDQN Algorithm. *Def. Technol.* **2021**, *18*, 1697–1714.
20. Etin, E.; Barrado, C.; Pastor, E. Counter a Drone in a Complex Neighborhood Area by Deep Reinforcement Learning. *Sensors* **2020**, *20*, 2320.
21. Zhang, Y.T.; Zhang Y.M.; Yu, Z.Q. Path Following Control for UAV Using Deep Reinforcement Learning Approach. *Guid. Navig. Control* **2021**, *1*, 2150005. [CrossRef]
22. Wang, L.; Hu, J.; Xu, Z. Autonomous Maneuver Strategy of Swarm Air Combat based on DDPG. *Auton. Intell. Syst.* **2021**, *1*, 1–12. [CrossRef]
23. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A. Asynchronous Methods for Deep Reinforcement Learning. In Proceedings of the International Conference on Machine Learning, PMLR, New York, NY, USA, 19–24 June 2016; Volume 48, pp. 1928–1937.
24. Williams, P. Three-dimensional Aircraft Terrain-following Via Real-time Optimal Control. *J. Guid. Control Dyn.* **1990**, *13*, 1146–1149. [CrossRef]
25. Pan, Y.; Wang, W.; Li, Y.; Zhang, F.; Sun Y.; Liu D. Research on Cooperation Between Wind Farm and Electric Vehicle Aggregator Based on A3C Algorithm. *IEEE Access* **2021**, *9*, 55155–55164. [CrossRef]
26. Austin, F.; Carbone, G.; Falco, M. Automated Maneuvering Decisions for Air-to-air Combat. *AIAA J.* **1987**, *87*, 656–659.