symmetry

MDPI

*Article*

# Three-Dimensional Path Planning of UAVs in a Complex Dynamic Environment Based on Environment Exploration Twin Delayed Deep Deterministic Policy Gradient

**Danyang Zhang, Xiongwei Li *, Guoquan Ren, Jiangyi Yao, Kaiyan Chen and Xi Li**

Shijiazhuang Campus, Army Engineering University, Shijiazhuang 050003, China
* Correspondence: lxwys@aeu.edu.cn

**Abstract:** Unmanned Aerial Vehicle (UAV) path planning research refers to the UAV automatically planning an optimal path to the destination under the corresponding environment, while avoiding collision with obstacles in this process. In order to solve the problem of 3D path planning of UAV in a dynamic environment, a heuristic dynamic reward function is designed to guide the UAV. We propose the Environment Exploration Twin Delayed Deep Deterministic Policy Gradient (EE-TD3) algorithm, which combines the symmetrical 3D environment exploration coding mechanism on the basis of TD3 algorithm. The EE-TD3 algorithm model can effectively avoid collisions, improve the training efficiency, and achieve faster convergence speed. Finally, the performance of the EE-TD3 algorithm and other deep reinforcement learning algorithms was tested in the simulation environment. The results show that the EE-TD3 algorithm is better than other algorithms in solving the 3D path planning problem of UAV.

**Keywords:** path planning; symmetrical; coding mechanism; dynamic reward function



**Citation:** Zhang, D.; Li, X.; Ren, G.; Yao, J.; Chen, K.; Li, X. Three-Dimensional Path Planning of UAVs in a Complex Dynamic Environment Based on Environment Exploration Twin Delayed Deep Deterministic Policy Gradient. *Symmetry* **2023**, *15*, 1371. https://doi.org/10.3390/sym15071371

Academic Editor: Zhixun Su

Received: 15 June 2023
Revised: 29 June 2023
Accepted: 1 July 2023
Published: 5 July 2023



## 1. Introduction

With the development of Unmanned Aerial Vehicle (UAV) technology, UAVs are becoming more and more widely used in military and civilian fields. Because of their small size and strong mobility, military UAVs are often used in low-altitude or ultra-low-altitude raid operations on the battlefield. In the military application of UAVs, path planning is one of its important contents. UAV path planning can help UAVs complete tasks in complex environments, such as area search [1], terrain reconnaissance [2], formation flight [3], etc. [4–6]. At the same time, many factors need to be considered, such as terrain, obstacles, weather, flight speed, etc., so it is relatively complex and challenging.

So far, there has been a lot of research on UAV path planning methods around the world, which are usually based on rules or optimization algorithms, such as A* algorithm [7], Dijkasta algorithm [8], and the artificial potential field method [9]. The main ideas of algorithms such as the A* algorithm and the Dijkasta algorithm are usually based on Heuristic Search and graph theory. Some index is used as a Heuristic Function, such as distance, cost, valuation, etc. To evaluate every possible path and find the best one. In other words, all possible paths should be displayed in the space, otherwise the optimal solution cannot be found by such methods. Such algorithms can find the optimal path to a certain extent, but in practical applications, path planning problems often involve complex environments, dynamic obstacles, and other situations, which requires the path planning algorithm to quickly adapt to changing environments and the ability to learn and adapt strategies. The artificial potential field method represents a class of algorithms, and the representative methods include simulated annealing method [10], charge method [11], flow function method [12], etc. One of these methods is the flow function method, which is actually an extension of the artificial potential field method. It plans its path based on the potential field between the drone and the external environment (obstacles and targets).

while the existing artificial potential field method is mainly applied to the 2D environment, and the shape of the obstacles also has many limitations. Most of the previous studies on path planning have solved the problem of path planning in a 2D environment [13]. Compared with a complex 3D environment, a simple 2D path planning method cannot effectively identify obstacles and targets. So, it is necessary to solve the path planning problem of UAVs in 3D environments. In previous studies, many scholars were also concerned about the study of path planning in 3D environments [14–17]. However, most of these studies are conducted in a single static environment, without considering the dynamic complex environmental constraints.

Deep reinforcement learning (DRL) is an advanced technology that has achieved remarkable achievements in robot control and agent behavior in recent years. It can independently learn effective behavior strategies. It can therefore update the strategy in a complex 3D environment by interacting with the environment constantly, and gradually adapt to the changes of the environment [18]. The deep neural network can process high-dimensional input data, so it can enable the agent to obtain rich information from the surrounding environment, and facilitate the generation of corresponding action strategies. In addition, the strong expression ability of the deep neural network can be effectively located in the characteristics of nonlinear and dynamic environment, which provides good help for the path planning research in a 3D environment [19,20]. At the same time, by introducing noise in the training, it can better adapt to the changes and uncertainties of the environment. Therefore, DRL is widely used in the solution of the UAV path planning problem and implements the purpose of effectively completing the path planning in complex 3D environments.

In order to solve the problem of the UAV path in a complex 3D environment more effectively, we propose a model based on the Environment Exploration Twin Delayed Deep Deterministic Policy Gradient (EE-TD3) algorithm and combine it with the unmanned aerial raid task model. The simulation results show that the algorithm model can plan the optimal flight path for UAVs in complex 3D battlefield environments, and it has relatively fast convergence compared to previous research. The main work of this article is as follows:

1. This paper analyzes the task requirement and the battlefield environment, and designs the dynamic reward function of the system, solves the sparse reward problem in the traditional path planning, and accelerates the convergence speed of the algorithm;
2. An environmental information exploration coding mechanism is proposed to explore and identify the surrounding environment of UAVs and enter it into the algorithm model, and then output the optimal control information to solve the path planning problem;
3. The 3D environment model of the low-altitude raid task of the UAV is constructed and the process of the combination of the environmental model and the algorithm model is introduced in detail, and the simulation experiment shows that the algorithm model has better performance.

To summarize, this paper proposes a path planning algorithm based on deep reinforcement learning in a 3D environment and has significant implications for research in relevant directions. Firstly, the dynamic reward function can provide reference for other scholars. Secondly, the environment exploration method can be generalized to other deep reinforcement learning algorithms as a general method. Moreover, there are few studies relevant to complex dynamic 3D environments, and this paper also provides reference cases for such scenarios.

The main content of the rest of this article is as follows: Section 2 introduces the relevant work. Section 3 illustrates the establishment of the 3D battlefield environment model. Section 4 explains the principle of the EE-TD3 algorithm model. Section 5 shows the simulation results and analyzes the results. Section 6 presents the conclusion of this paper.

## 2. Related Work

So far, many researchers have used traditional intelligent algorithms to study UAV autonomous path planning, such as ant colony optimization algorithm, FPA algorithm, GA algorithm and so on. In recent years, with the continuous development of deep reinforcement learning (DRL) technology, more and more researchers have applied it to solve the autonomous path planning of UAV.

In terms of traditional algorithms, Chen J. et al. [21] proposed a formula based on mixed integer linear programming to find the optimal flight path for UAVs by fully searching the path space. At the same time, an original algorithm based on clustering is designed to divide different regions into clusters, allowing for the acquisition of the optimal point-to-point path for the UAV. C. Lamini et al. [22] proposed an improved crossover operator to solve the path planning problem in a static environment by using the genetic algorithm (GA). The crossover operator can also avoid local convergence, provide a feasible path with better fitness value than its parent operator, and make the algorithm converge faster. Ee Soong Low et al. [23] used the flower pollination algorithm (FPA) to improve the initialization of the Q value in Q-learning algorithm. By comparing the experimental evaluation under a different obstacle layout environment, it is shown that the Q-learning algorithm converges faster when FPA is used to improve the initial Q value.

In terms of DRL algorithm, Yang Yang et al. [24] used the Deep Q network (DQN) algorithm in this paper, which combined the Q-learning algorithm, experiential playback mechanism, and deep neural network in time to generate target Q values to solve the path planning problem of multiple robots. In Ref. [25], G. Lin et al. introduced a collision-free path planning method based on deep reinforcement learning, which is fast and robust. Firstly, the recurrent neural network is used to record and process the state information observed by the robot, and then the depth deterministic strategy gradient algorithm (DDPG) is used to predict the collision-free path of the current state. In the training experiment phase, the simulation environment is developed, and its parameters are randomized so that the cyclic DDPG algorithm can be generalized to real world scenarios. P. Chen et al. [26] proposed a path planning method based on the soft actor-critic (SAC) algorithm to solve the dynamic obstacle avoidance problem of Mechanical arm. In order to avoid moving obstacles in the environment while conducting real-time path planning, a comprehensive reward function of dynamic obstacle avoidance and target method is designed. To solve the problem of low sample utilization caused by random sampling, priority experience playback (PER) was used to change sample weights and improve the sampling efficiency. Finally, simulation experiments are carried out, and the experimental results show that this method can effectively avoid moving obstacles in the environment, and the success rate of path planning task is relatively high.

Recently, many researchers have tried to use the TD3 algorithm to solve the UAV path planning problem. Lei He et al. [27] proposed a path planner based on deep learning in this paper, which is used for the autonomous flight of quadrotor UAV in an unknown environment. The path navigation problem is modeled as a Markov decision process (MDP), and the path planner is trained in a simulated environment using a deep reinforcement learning (DRL) method using the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm. Finally, a real-world flight test is carried out to show that the path planner trained in the simulation has good performance and can be directly applied to the real environment. S. Zhang et al. [28] proposed an improved Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm. In order to make the UAV respond to the change of environment, the observed environment information is added to the Actor-Critic network input, and the double-flow Actor-Critic network structure is proposed to extract the environment information features. The performance of the algorithm is evaluated by simulation, and the experimental results show that the proposed method can effectively complete the autonomous path planning task in the multi-obstacle environment, which reflects the effectiveness of the proposed method. Although the above research shows the application ability of the TD3 algorithm in solving path planning problems,

its performance in a complex dynamic battlefield environment still needs to be studied further. Since TD3 algorithm has the advantages of strong ability to deal with large state space information and fast convergence speed, this paper proposes a path planning algorithm model for UAV based on environment exploration mechanism and TD3 algorithm in three-dimensional environment.

## 3. Three-Dimensional Battlefield Environment Model

A 3D space with a length and width of 50 km and a height of 1 km is established in a 3D cartesian coordinate system. In this 3D space, the UAV is tasked with taking low-altitude surprise attacks on radar positions 50 km away. In order to avoid enemy radar reconnaissance, the UAV must fly at an altitude of less than 1 km. During flight, the UAV needs to autonomously avoid static obstacles, such as mountains and ground buildings. Due to the low altitude of the UAV, it is also affected by the dynamic obstacles of random movement such as low flying birds and civil aircraft. Therefore, the UAV should also respond to dynamic obstacles in a timely manner during flight.

As shown in Figure 1, the 3D battlefield environment includes UAV $U$, low-altitude bird group $B$, mountain $M$, and radar $R$. An UAV can fly autonomously in the battlefield environment, and the position information of the UAV can be expressed as follows:

$$U_{(x,y,z)} = [x, y, z] \tag{1}$$

In Formula (1), $x \in (0,\ 50\ \text{km})$, $y \in (0,\ 50\ \text{km})$, $z \in (0,\ 1\ \text{km})$.

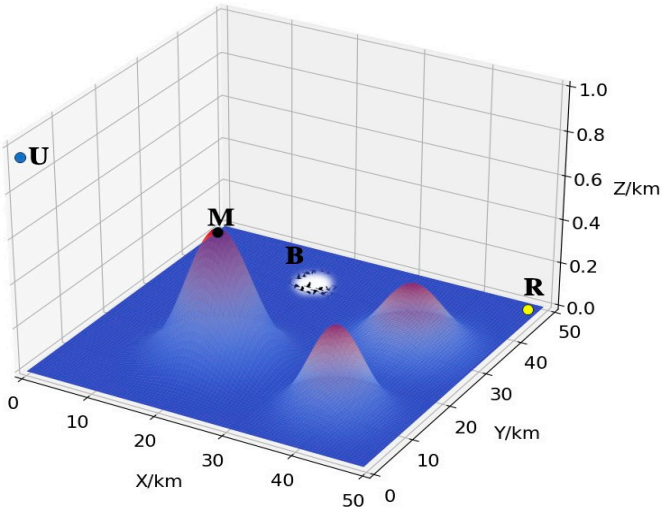


**Figure 1.** Battlefield environment diagram.

The velocity $v$ of the UAV is divided according to the direction of the 3D rectangular coordinate system, $v_x \in (0,\ 100\ \text{m/s})$, $v_y \in (0,\ 100\ \text{m/s})$, $v_z \in (-3,\ 3\ \text{m/s})$. The real-time location $U_i$ of the UAV can be expressed as:

$$U_i = [x_i, y_i, z_i] = \left[\sum_i (v_{xi} * \Delta t), \sum_i (v_{yi} * \Delta t), \sum_i (v_{zi} * \Delta t)\right] \tag{2}$$

The initial position of the UAV $U_0$ can be expressed as:

$$U_0 = [0\ \text{km}, 0\ \text{km}, 1\ \text{km}] \tag{3}$$

The coordinates of the vertices of mountain $M$ can be expressed as:

$$M = [x_m, y_m, z_m] \tag{4}$$

The height coordinate $z_m$ of a high mountain $M$ can be represented by the horizontal coordinates $x_m$ and $y_m$, as follows [29]:

$$z_m = a * e^{-(\frac{y-y_{m_0}}{b})^2-(\frac{x-x_{m_0}}{c})^2} \tag{5}$$

where, $x_{m_0}$ and $y_{m_0}$ are the horizontal coordinates of the lowest center of the mountain $M$, $a$ is the height coefficient of the mountain, and the changes in $b$ and $c$ can change the size of the mountain.

The UAV may encounter dynamic obstacles such as birds and civil low-altitude aircraft during low-altitude flight. It is assumed that these dynamic obstacles are randomly generated in an area with a height below 300 m and the initial center position is expressed as $B_{(x_0,y_0,z_0)}$. The dynamic real-time position $B_{(x,y,z)}$ can be expressed as:

$$B_{(x,y,z)} = [x_B, y_B, z_B] = \left[x_0 + v_{Bx} * \Delta t,\ y_0 + v_{By} * \Delta t,\ z_0 + v_{Bz} * \Delta t\right] \tag{6}$$

In Formula (6), the initial position $x_0 \in (0,\ 50\ \text{km})$, $y_0 \in (0,\ 50\ \text{km})$, $z_0 \in (0,\ 0.3\ \text{km})$, and dynamic obstacles to movement speed $v_{Bx} \in (-10,\ 10\ \text{m/s})$, $v_{By} \in (-10,\ 10\ \text{m/s})$, $v_{Bz} \in (-2,\ 2\ \text{m/s})$.

In order to ensure flight safety, the UAV should maintain a safe distance of more than 50 m between bird group $B$, and $d_{safe}$ can be expressed as:

$$d_{safe} = |U_i - B_{(x,y,z)}| = \sqrt{(x_{t_i} - x_B)^2 + (y_{t_i} - y_B)^2 + (z_{t_i} - z_B)^2} \geq 0.05\ \text{km} \tag{7}$$

The position of radar $R$ can be expressed as:

$$R = [50\ \text{km},\ 50\ \text{km},\ 0\ \text{km}] \tag{8}$$

The maximum detection radius of the radar is 40 km, but due to the influence of earth curvature, detection angle, ground obstacles, ground clutter, and other factors, the ultra-low altitude target below 1 km is difficult to be detected by radar. The probability of the UAV being detected by radar will change with the change of the distance $d$ between the two and the UAV altitude $h$. When the UAV flies too low, the radar will regard it as entering the low altitude blind zone and cannot detect it. Assuming that the radar blind zone is an airspace with a height of less than 300 m, the radar detection probability model can be expressed as follows.

$$P = \begin{cases} 0 & d > 40\ \text{km} \\ 0 & h < 0.3\ \text{km} \\ 1 & d \leq 40\ \text{km},\ h \geq 1\ \text{km} \\ \frac{1}{0.01+2e^{-5h+5}} - \frac{d^2}{3200} + \frac{1}{2} & d \leq 40\ \text{km}, 0.3\ \text{km} \leq h \leq 1\ \text{km} \end{cases} \tag{9}$$

The radar detection probability diagram can be obtained from Formula (9), as shown in Figure 2.

Figure 2 shows that the $d$-axis represents the Euclidean distance between the UAV and the radar, the $h$-axis represents the UAV flying altitude, and the $p$-axis represents the probability of the UAV being detected by the radar. Further analysis shows that the UAV flying altitude below 300 m can use the blind zone to avoid radar detection. In the range of flight altitude $h \in (0.3\ \text{km},\ 1\ \text{km})$, the probability of being detected by a radar increases as the UAV flies higher and closer to the distance.

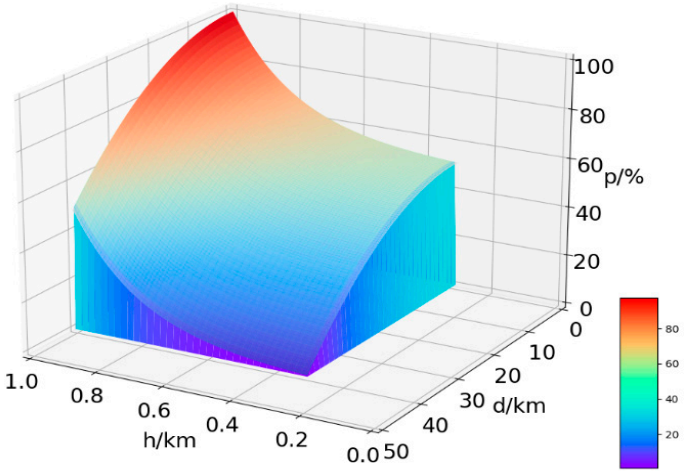


**Figure 2.** Radar detection probability diagram.

The UAV carries air-to-ground missiles with a maximum firing range of 8 km, and the mission is considered complete when the UAV safely reaches the position 8 km away from the radar. The radar positions are equipped with anti-aircraft missile weapon systems (including guidance radars and anti-aircraft missiles), which have a maximum range of 25 km. If the UAV is within the radar detection area and the maximum range of the anti-aircraft missile, the anti-aircraft missile will lock on to the UAV and attack the launch. Therefore, in order to prevent being targeted by anti-aircraft missiles, the UAV should avoid entering the radar detection area.

## 4. UAV Path Planning Algorithm Model

In this section, we introduce the development of the deep reinforcement learning algorithm, and improve the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm, so that it can better solve the path planning problem in UAV low-altitude raid mission. There are two improvements. The first point is that the environment exploration Twin Delayed Deep Deterministic Policy Gradient (EE-TD3) algorithm model is proposed, which mainly divides and compresses the regional state information near the UAV in the 3D environment, detects the state of the regions in different directions and encodes it with binary numbers, and uses the encoded environmental state value as the input of the algorithm model. This real-time status information can improve the UAV's perception of the environmental information. The second point is that a heuristic dynamic reward function is designed to solve the sparse reward problem in traditional path planning methods, so that the UAV can get the corresponding reward according to the current real-time state, and the convergence speed of the algorithm model in the training process is accelerated.

### *4.1. Deep Reinforcement Learning Algorithm Model*

Deep Reinforcement learning (DRL) is an end-to-end learning method that combines deep learning and reinforcement learning, which can simultaneously have the big data processing ability of deep learning (DL) and the decision-making ability of reinforcement learning (RL) [30]. DRL has achieved great results in the autonomous control of agents, which can effectively solve the problems of UAVs in traditional path planning. Among the existing DRL algorithm models, some algorithms are suitable for solving discrete action space problems, such as the DQN algorithm, and some algorithms are suitable for solving continuous action space problems, such as the DDPG algorithm and TD3 algorithm. DDPG is called Deep Deterministic Policy Gradient, where Deep refers to the deep network structure, Deterministic refers to deterministic selection, where the purpose is to help the policy gradient avoid random selection and output specific action values. Policy Gradient refers to the policy gradient algorithm, which can randomly select actions in a continuous action space based on the learned policy [31].

As shown in Figure 3, the DDPG algorithm adopts the end-to-end learning mode, takes the initial state information of the agent as the input, and the action strategy value $\mu(s_t)$ which is calculated by the algorithm as the output result. At the same time, random noise is added to the action policy to obtain the final output action. At the beginning of the task, the UAV outputs an action based on the current state $s_t$, which is evaluated by the designed reward function to evaluate the value of the output action, so as to obtain the feedback reward $r_t$ from the environment. In this way, it can be concluded that the optimal action in the current state gives a positive reward, and vice versa gives a negative reward. In this process, the current state information, the action, the reward, and the state information of the next time ($s_t$, $a_t$, $r_t$, $s_{t+1}$) are stored in the experience pool. When the experience pool is full, sample data is randomly selected from it for training, and the action strategy is constantly adjusted during the training process. In order to enhance the stability and accuracy of the algorithm, the gradient descent method is used to update the iterative network parameters. The DDPG algorithm is used to subdivide the network structure into an online network and target network under the framework of the AC algorithm. The online network includes the Actor online network and Critic online network, which can complete real-time output actions, evaluate actions, and update the network parameters. The target network includes the Actor target network and Critic target network, which are used to update the Actor online network and Critic online network. The target network and the online network have the same neural network architecture and initialization parameters. However, during the training process, the parameters of the Actor target network and the Critic target network are updated in a soft changing way, which enhances the stability of the training process compared to directly copying the parameters of the online network. The update process can be expressed by the following formula [32]:

$$\begin{cases} \theta^{Q'} = \tau\theta^{Q} + (1-\tau)\theta^{Q'} \\ \theta^{\mu'} = \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \end{cases} \tag{10}$$

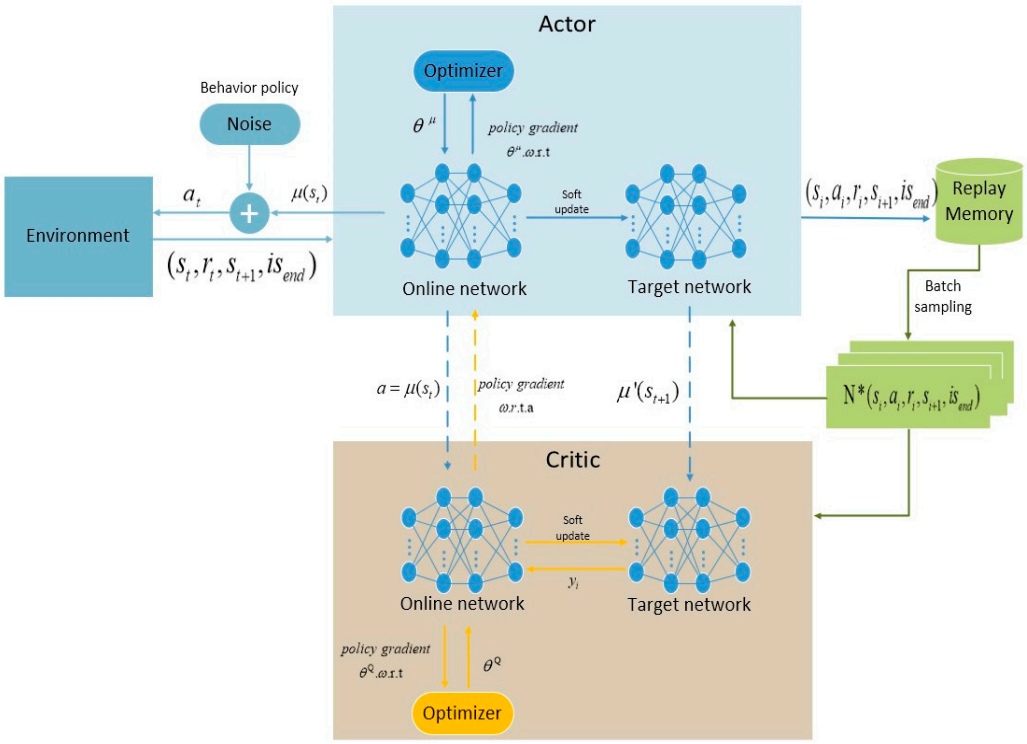


**Figure 3.** Model diagram of the DDPG algorithm.

In Equation (10), $\theta^{\mu}$ and $\theta^{Q}$ represent the parameters of the Actor-Critic online network, $\theta^{\mu'}$ and $\theta^{Q'}$ represent the parameters of the Actor-Critic target network, and $\tau \in (0,1)$ is the update coefficient.

TD3 algorithm is an improved algorithm based on the DDPG algorithm. TD3 algorithm introduces three key skills:

The first is to add smoothing noise to the action value output by the Actor target network, which can make the estimation more accurate, but no noise is added to the final action value output by the Actor network. The TD3 algorithm minimizes the error in the Bellman equation by calculating the mean square, and simultaneously learns two Q-functions $Q_{\varphi_1}$ and $Q_{\varphi_2}$. In addition, the action used to form the Q-Learning objective is based on the target policy $\mu_{\theta_{targ}}$, but a smoothing noise $\epsilon \sim N(0,\sigma)$ is also added to each dimension of the action, and the specific formula can be expressed as:

$$a'(s') = clip\left(\mu_{\theta_{targ}}(s') + clip(N(0,\widetilde{\sigma}), -c,\ c),\ a_{Low},\ a_{High}\right) \tag{11}$$

In Formula (11), $\widetilde{\sigma}$ is the standard deviation when calculating the normal distribution, and the larger the value is, the more noise is added.

The second is the clipped double Q-learning method. As shown in Figure 4, two independent Critic networks are used to estimate the $Q$ value, and the smaller $Q$ value is selected for updating when calculating the target $Q$ value, which can effectively solve the problem of overestimation of the $Q$ value in the DDPG algorithm. This way, the target policy becomes smoother and can avoid the problem of incorrect peaks that may be generated in the DDPG algorithm. The target policy can quickly use the peak and then produce brittleness or incorrect behavior. The specific formula for updating the $Q$ value can be expressed as follows:

$$y(r,s') = r + \gamma \min_{i=1,2} Q_{\varphi_i,targ}\ (s',\ a'(s')) \tag{12}$$

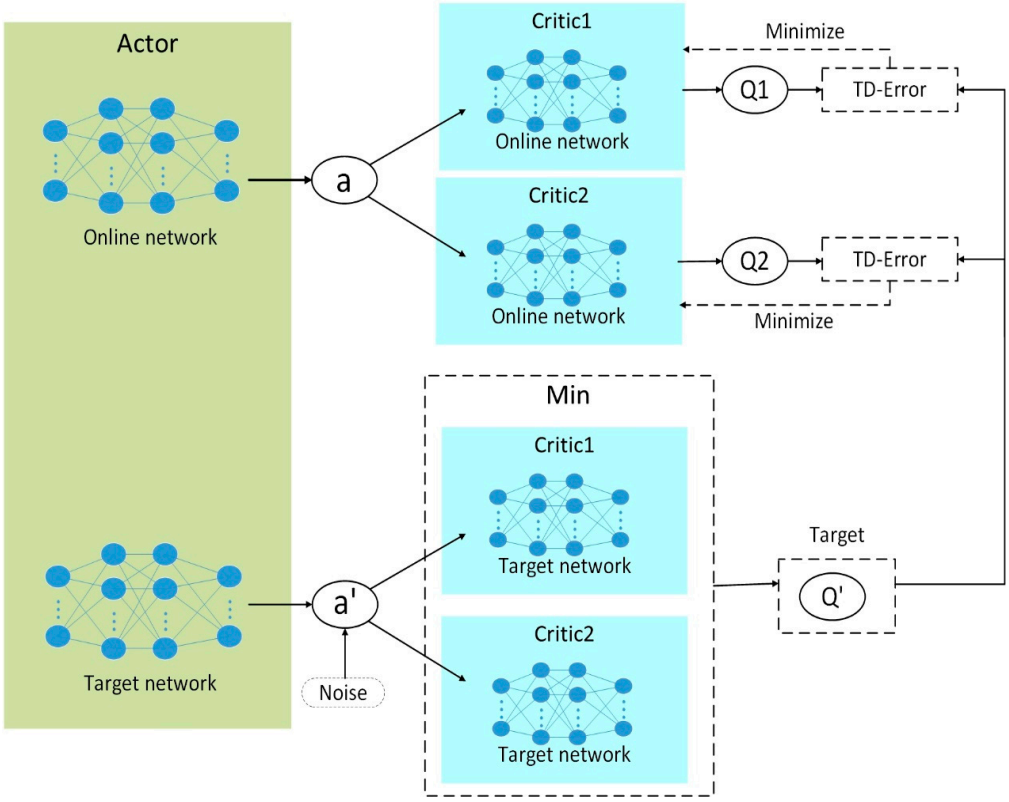


**Figure 4.** Clipped double Q-Learning method.

Then, both Q-value networks learn by regressing to this target value. The specific formula can be expressed as:

$$L(\varphi_i,\ D) = N^{-1} \sum_{(s,a,r,s',d)\sim D} \left(Q_{\ \varphi_i\ (s,\ a)} - y(r,\ s',\ d)\right)^2 \tag{13}$$

Thirdly, the Actor network in the TD3 algorithm is updated with a delay strategy, and the Critic network is updated normally. Therefore, the update frequency of the Critic network is faster than that of the Actor network, which can greatly reduce the error caused by the same frequency update and help to stabilize the training process. The pseudocode for the TD3 algorithm can be expressed as Algorithm 1:

**Algorithm 1:** TD3 algorithm.

1. Initialize critic networks $Q_{\theta_1}$, $Q_{\theta_2}$, and actor network with random parameters $\theta_1$, $\theta_2$, $\varnothing$
2. Initialize target networks $\theta_1' \leftarrow \theta_1$, $\theta_2' \leftarrow \theta_2$, $\varnothing' \leftarrow \varnothing$
3. Initialize replay buffer B
4. **for** $t = 1$ **to** $T$ do
5. Select action with exploration noise $a \sim \pi_{\varnothing}(s) + \epsilon$, $\epsilon \sim N(0,\sigma)$ and observe reward $r$ and new state $s'$
6. Store transition tuple $(s,a,r,s')$ in Replay memory
7. Sample mini-batch of $N$ transitions $(s,a,r,s')$ from Replay memory
8. Compute target action $\widetilde{a} \leftarrow \pi_{\varnothing'}(s') + \epsilon, \epsilon \sim clip(N(0,\widetilde{\sigma}), -\mathrm{c}, \mathrm{c})$
9. Compute target Q value $y \leftarrow r + \gamma_{i=1,2}^{min} Q_{\theta_i'}(s', \widetilde{a})$
10. Update critics $\theta_i \leftarrow argmin_{\theta_i} N^{-1} \sum (y - Q_{\theta_i}(s,a))^2$
11. **if** $t$ mod $d$ **then**
12. Update $\varnothing$ by the deterministic policy gradient:
13. $\nabla_{\varnothing} J(\varnothing) = N^{-1} \sum \nabla_a Q_{\theta_1}(s,a)|_{a=\pi_{\varnothing}(s)} \nabla_{\varnothing} \pi_{\varnothing}(s)$
14. Update target networks:
15. $\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i'$
16. $\varnothing' \leftarrow \tau\varnothing + (1-\tau)\varnothing'$
17. **end if**
18. **end for**

### *4.2. Environment Exploration Twin Delayed Deep Deterministic Policy Gradient Algorithm Model*

In this section, the environment exploration coding mechanism and the heuristic dynamic reward function are proposed, which can be combined with the TD3 algorithm model to obtain the Environment Exploration Twin Delayed Deep Deterministic Policy Gradient (EE-TD3) algorithm model. This model can effectively solve the problem of complex environmental information in previous research and the sparse reward problem in traditional path planning research. In the next section, we will use this algorithm model to try to solve the UAV autonomous path planning problem through simulation experiments.

#### 4.2.1. Coding Mechanism for Environmental Exploration

Most of the previous path planning studies take the current position information $(x,y,z)$ of the UAV as the input of the algorithm model, take action according to the output action value, and then perform interactive learning by obtaining rewards from the environment. When the training model reaches convergence, the UAV can take the corresponding correct action at any position until the task is completed. However, when there are dynamic obstacles in the environment, because the trajectory of the obstacles is random, it is impossible to judge the direction of the obstacles only by using the current position information of the UAV as input, and it is difficult to effectively realize the purpose of avoiding obstacles. In order to make the UAV reach the destination efficiently in a complex and dynamic environment, the algorithm model must receive the environmental state information of the area near the UAV, and immediately identify the position of the

obstacles and make decisions to avoid them. Therefore, this paper designs a symmetrical environment detection mechanism to encode the 3D environment information in the area near the UAV, so that the UAV can have better obstacle avoidance ability.

In modern warfare, the UAV is equipped with various sensors to detect the surrounding environment. It is assumed that the UAV is equipped with sensors that can detect the state of the region near itself, and the maximum action distance of the sensor is 100 m. With the information fed back by these sensors, the UAV can detect whether there are obstacles in the nearby area. Then, the binary number (1 or 0) is used to represent the existence or absence of obstacles, and a set of current environmental state information array can be obtained by state detection coding. This array is used as the input of the algorithm model, so that the algorithm model can make the right decision to control the UAV action according to the environmental state information in the nearby area.

As shown in Figure 5, the spherical area near the UAV in the 3D environment is divided. Since the spherical area has symmetry and the current 3D environment is a continuous space, infinite times of division can theoretically be carried out, but it will cause an increase in the amount of training calculation, so our scheme adopts limited times of division. It can be seen from Figure 5 that the UAV attachment area is divided according to the same Angle, where the Angle in the horizontal direction and the vertical direction is $\beta$, and the number of segmented areas is $n$. In the process of flight, a set of n-dimensional array $S_{UAV}$ is obtained by dividing and encoding the regional environment information detected by the UAV each time.

$$S_{UAV} = [s_0, s_1, s_2, s_3, \ldots\ldots, s_n], s_i \in [0,\ 1] \tag{14}$$

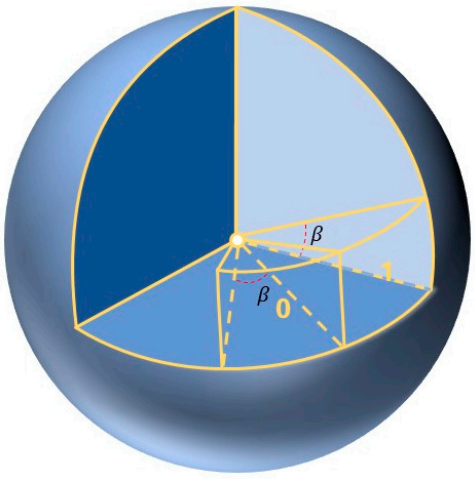


**Figure 5.** Illustration of the exploration of the environmental information.

Then, $S_{UAV}$ is used as the input of the algorithm model, and the UAV takes the next action according to the action output of the algorithm model. The environmental information exploration coding mechanism can be regarded as the compression coding of the state information of the area near the UAV, which simplifies the complex environmental information with interference and facilitates the algorithm model processing. In order to further verify the applicability of this method, we will select two cases $\beta = 45^\circ$ and $\beta = 30^\circ$ for experiments in the next section.

#### 4.2.2. Heuristic Dynamic Reward Function

The reward function also known as the immediate reward is the key to the DRL technique to solve the problem. In the previous traditional reinforcement learning algorithms, the reward rules of navigation are relatively simple, generally only considering the reward of reaching the goal position when completing the task, and the punishment after hitting

the obstacle. However, due to the relatively large 3D environment state space in this path planning task, if the UAV is only rewarded when completing the task and collision, and there is no reward in other states, the effective reward cannot be obtained in time, and the algorithm model will be difficult to converge in the training process. This leads to the problem of sparse reward [33]. The training process can be accelerated by introducing a continuous reward function to guide the exploration. After many attempts, we designed a heuristic dynamic reward function. The specific formula can be expressed as follows.

$$R = \begin{cases} 1 - \frac{d_{t+1}}{\eta}, d_t > d_{t+1} \\ \frac{d_{t+1}}{\eta} - 1, d_{t+1} \geq d_t \end{cases} \tag{15}$$

In Equation (15), $\eta$ is the reward coefficient and $\frac{d_{t+1}}{\eta} \in (0, 1)$, $d_t$ is the distance between the UAV and the radar position in the current state, and $d_{t+1}$ is the distance between the UAV and the target in the next state. The analysis of Equation (12) shows that when the UAV performs the current action, if the next state is closer to the target, the positive reward will be obtained, and the closer the distance to the target, the greater the positive reward value will be obtained. If the next state is further away from the goal, it receives a negative reward, and the further away from the goal, the greater the negative reward value. When the distance between the UAV and the target is less than 8 km, it is considered to have completed the mission and a positive reward of 300 is obtained. When the distance between the UAV and the dynamic obstacle is less than a safe distance or the UAV collides with the static obstacle, a negative reward of $-300$ is obtained and the environment is reset.

## 5. Simulation Experiment and Result Analysis

In this experiment, a battlefield simulation environment was constructed to verify the performance of the EE-TD3 algorithm, and the environment was set up with radar detection areas, mountains as static obstacles, and random low-altitude dynamic obstacles. The DDPG algorithm, TD3 algorithm and EE-TD3 algorithm will be verified in this battlefield simulation environment. All the experiments are conducted on a computer with Intel(R) Core(TM) i7-10700 CPU and NVIDIA GeForce RTX3060Ti GPU. Python3.9 is also used as the interpreter of the project program. Additionally, pytorch-1.12.1 is used as a deep learning framework to build neural networks.

### *5.1. Experimental Parameter Settings*

The setting of hyperparameters can affect the convergence effect of the algorithm model, which plays a key role in the effectiveness of the final experimental results. The main hyperparameters in the deep reinforcement learning model are: deep neural network structure parameters, learning rate $\alpha$, discount factor $\gamma$, experience pool storage capacity $R$, sampling number $N$, and target network soft update coefficient $\tau$.

The three algorithms in this experiment are added noise, but as the training goes on, the model gradually tends to converge. If the noise value is too large, it may produce local oscillation and make the model difficult to converge. Therefore, the noise attenuation factor $k \in (0, 1)$ is set in this experiment. During the training process, whenever the UAV reaches the end point to complete a task, the noise value is multiplied by the noise attenuation factor $k$ to reduce the noise and to accelerate the convergence speed of the model. Since the DDPG algorithm, TD3 algorithm, and EE-TD3 algorithm all adopt the Actor-Critic structure, the learning rates $\alpha_a$ and $\alpha_c$ of the Actor network and Critic network are important hyperparameters. When $\alpha_a$ is set to 0.0001 for multiple attempts, the model training effect is stable. Since the Critic network is responsible for evaluating the behavior of the model, and there are four independent neural networks in the Critic module in the TD3 algorithm and EE-TD3 algorithm, $\alpha_c$ has a great influence on the training effect of the algorithm model. We set nine different $\alpha_c$ for training. The convergence of the algorithm model in different environments is compared by analyzing the results. In addition, we

also conducted performance test experiments on all algorithm models and analyzed the performance effects of each algorithm model to obtain the optimal algorithm model.

The specific hyperparameter settings for this experiment are shown in Table 1.

**Table 1.** Hyperparameter settings.

| Hyperparameters | Symbol | Value |
|---|---|---|
| Hidden layers | - | 2 |
| Hidden layer units | - | 512 |
| Max episodes | - | 3000 |
| Max steps per episode | - | 500 |
| Actor network learning rate | $\alpha_a$ | 0.0001 |
| Discount factor | $\gamma$ | 0.99 |
| Replay buffer size | $R$ | 6400 |
| Batch size | $N$ | 256 |
| Soft update rate | $\tau$ | 0.005 |
| Noise attenuation factor | $k$ | 0.999 |

### *5.2. Analysis of Experimental Results*

#### 5.2.1. Experimental Results of DDPG Algorithm Model

Figure 6 shows the training effect achieved by the DDPG algorithm in the simulated battlefield environment after setting the learning rate $\alpha_c$ of the module network for nine different critics. The analysis results show that although some cases can converge, the DDPG model converges relatively well when $\alpha_c$ is 0.0005, and the model tends to converge after 2050 rounds. In general, the DDPG algorithm model has unstable convergence and obvious oscillation under different learning rates, and the effect is not ideal for solving the UAV path planning problem.

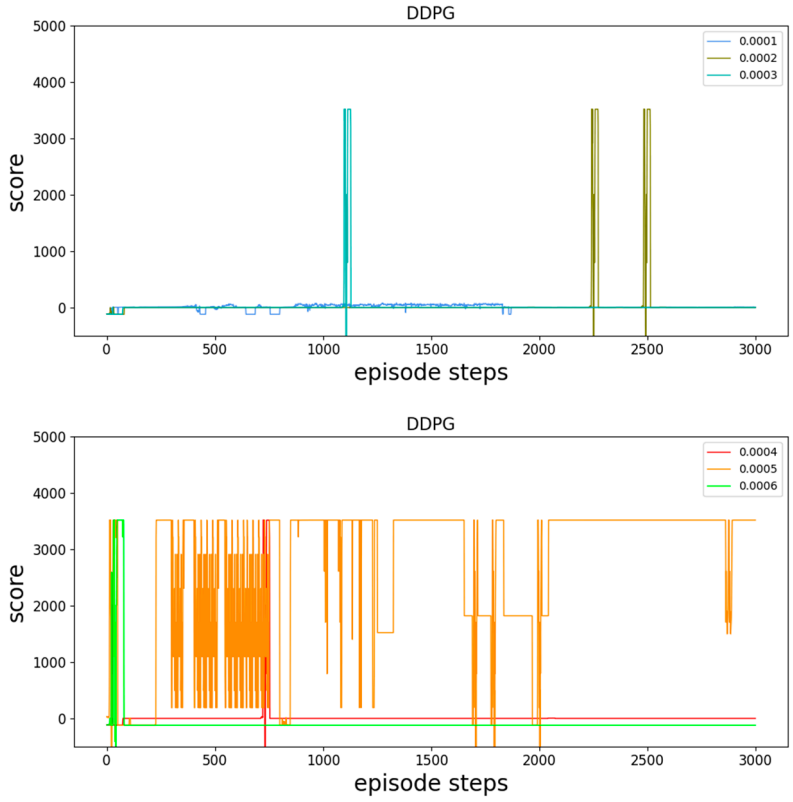


**Figure 6.** *Cont.*

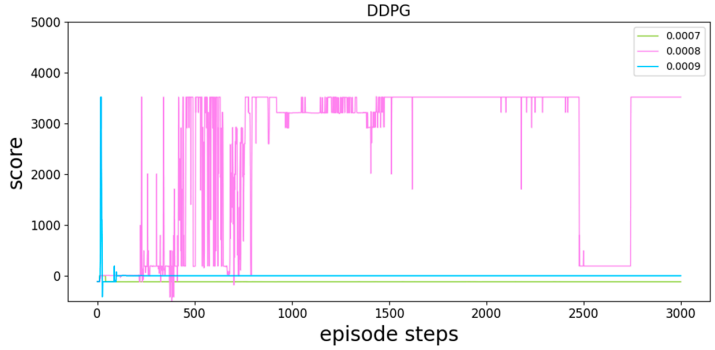


**Figure 6.** Training results of the DDPG algorithm model.

### 5.2.2. Experimental Results of the TD3 Algorithm Model

Figure 7 shows the training results of the TD3 algorithm in the battlefield simulation environment with nine Critic network learning rates $\alpha_c$.The analysis results show that the TD3 algorithm model tends to converge at 1550 rounds when $\alpha_c$ is 0.0006, and the TD3 algorithm model tends to converge at 1320 rounds when $\alpha_c$ is 0.0008. There is a certain degree of oscillation after these two results tend to converge, but the overall situation is convergent. Through further analysis it can be seen that the TD3 algorithm model has faster convergence speed and smaller oscillation amplitude than the DDPG algorithm model, so the TD3 algorithm model is more suitable for solving the UAV path planning problem.

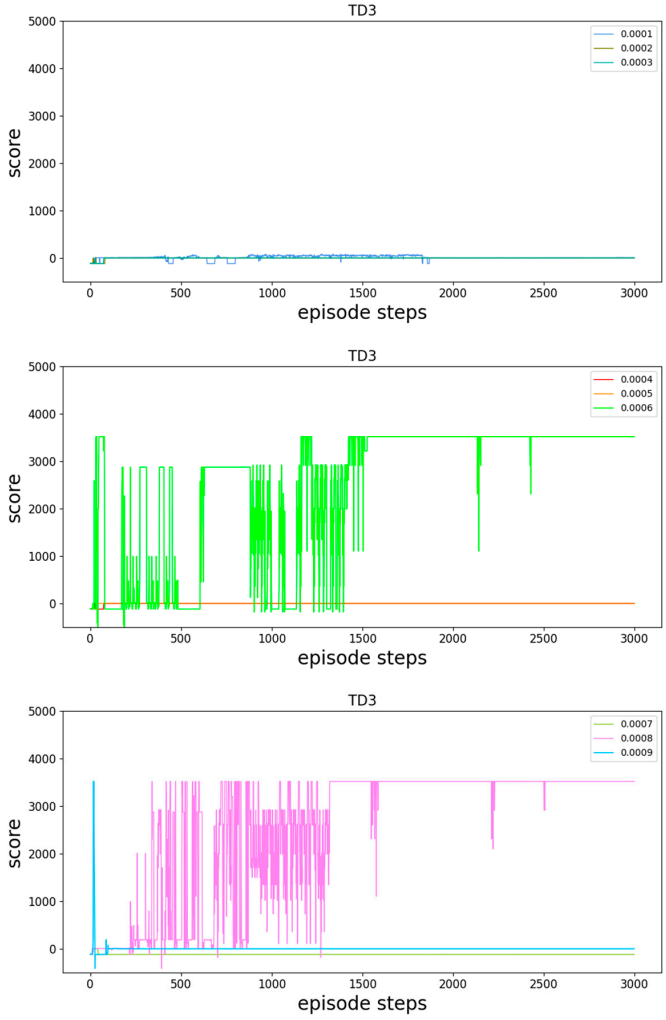


**Figure 7.** Training results of the TD3 algorithm model.

5.2.3. Experimental Results of the EE-TD3 Algorithm Model

In order to verify the applicability of the EE-TD3 algorithm, we selected two environmental region segmentation angles for experiments, and the specific experimental results are as follows:

(1) $\beta = 45^\circ$

Figure 8 shows the training effect achieved by the EE-TD3 algorithm in a simulated battlefield environment ($\beta = 45^\circ$) with nine Critic network learning rates $\alpha_c$. The analysis results show that the EE-TD3 algorithm model starts to converge at 1150 rounds when $\alpha_c$ is 0.0003, and the EE-TD3 algorithm model starts to converge at 1070 rounds when $\alpha_c$ is 0.0005. When $\alpha_c$ is 0.0007, the EE-TD3 algorithm model starts to converge at 1385 rounds. When $\alpha_c$ is 0.0009, the EE-TD3 algorithm model starts to converge in 1760 rounds. These four results tend to convergence; there is a small oscillation, but the overall situation is still convergent. At the same time, when $\beta = 45^\circ$, the optimal result of the EE-TD3 algorithm model has faster convergence speed and smaller oscillation amplitude than that of the TD3 algorithm model. Additionally, from the convergence success rate of the nine experiments, the EE-TD3 algorithm model is also higher than the TD3 algorithm model.

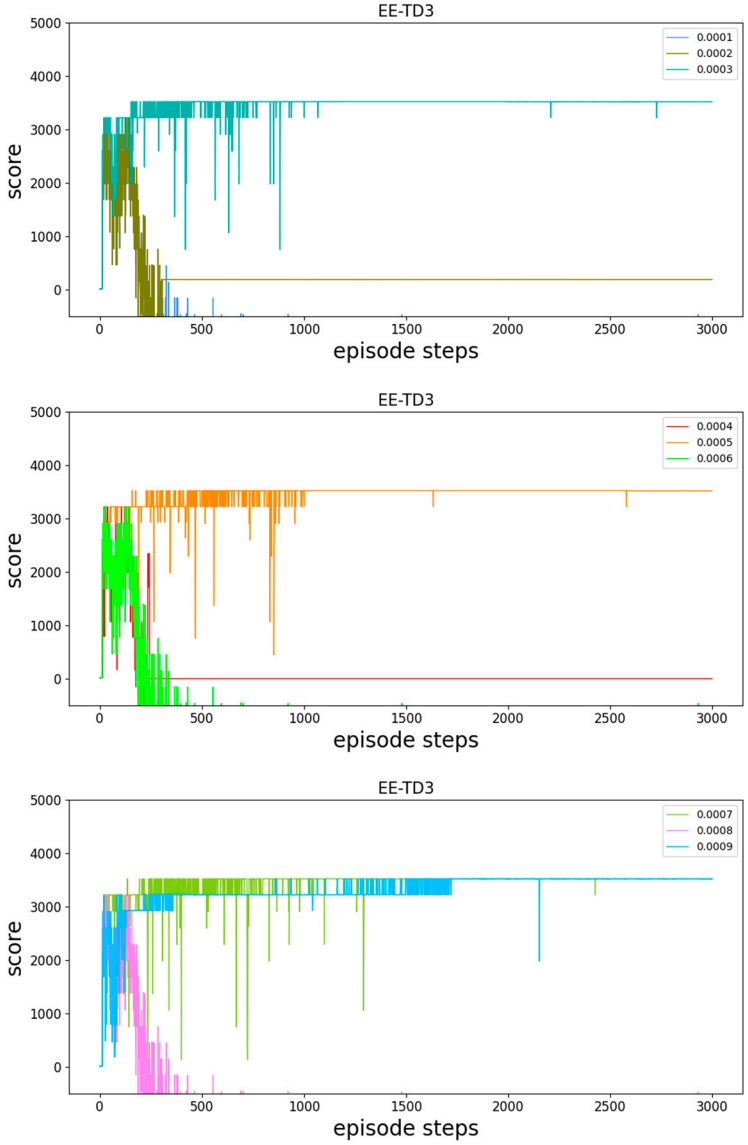


**Figure 8.** Training results of the EE-TD3 algorithm model ($\beta = 45^\circ$).

(2) $\beta = 30^\circ$

Figure 9 shows the training effect achieved by the EE-TD3 algorithm ($\beta = 30^\circ$) in a simulated battlefield environment with nine different Critic module network learning rates $\alpha_c$ set. The analysis results show that the EE-TD3 algorithm model begins to converge at 890 rounds when $\alpha_c$ is 0.0002, the EE-TD3 algorithm model begins to converge at 2270 rounds when $\alpha_c$ is 0.0003, and the EE-TD3 algorithm model begins to converge at 930 rounds when $\alpha_c$ is 0.0004. When $\alpha_c$ is 0.0005, the EE-TD3 algorithm model begins to converge at 2323 rounds. When $\alpha_c$ is 0.0009, the EE-TD3 algorithm model begins to converge at 2285 rounds. Five results of the EE-TD3 algorithm model converge when $\beta = 30^\circ$. At the same time, it can be seen that the optimal result of the EE-TD3 algorithm model when $\beta = 30^\circ$ is faster than the optimal result of the EE-TD3 algorithm model when $\beta = 45^\circ$. In addition, from the convergence success rate of these nine experiments, the training effect of the EE-TD3 algorithm model when $\beta = 30^\circ$ is also better. In general, for the UAV autonomous path planning problem, the EE-TD3 algorithm model is generally better than the TD3 algorithm and DDPG algorithm.

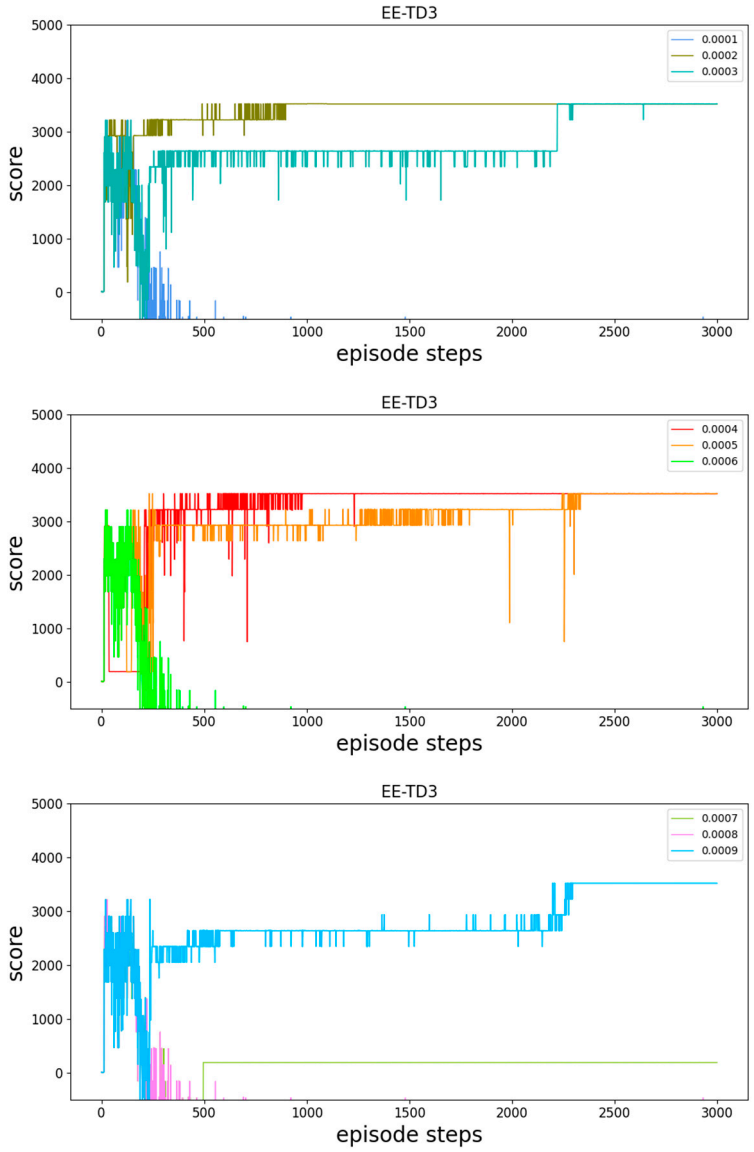


**Figure 9.** Training results of the EE-TD3 algorithm model ($\beta = 30^\circ$).

5.2.4. Experimental Results of the Algorithm Performance Test

To further verify the actual performance of the EE-TD3 algorithm proposed in this paper, we conducted comparative experiments on all the algorithm models that have converged. In the experiment, we conducted 1000 rounds of independent experiments for the 3 algorithm models, and obtained the task success rate and average path length of each algorithm model.

Figures 10 and 11 represent the task success rate and average path length of each algorithm model in the performance testing experiment, respectively. We can see from this that the performance of the DDPG algorithm is relatively poor, and the performance of the TD3 algorithm and EE-TD3 algorithm is relatively good. Further analysis shows that the performance of EE-TD3 algorithm is better as the environment region is segmented smaller. Therefore, the EE-TD3 algorithm model is the optimal choice in this experiment.

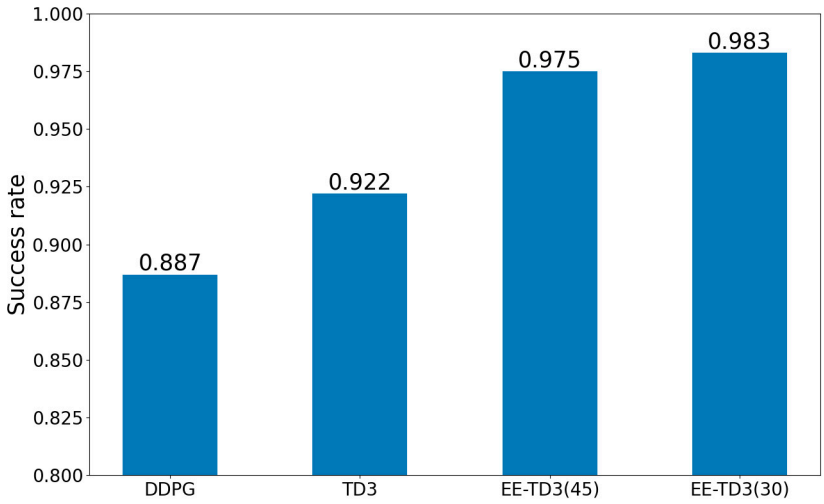


**Figure 10.** The success rate of the algorithm model.

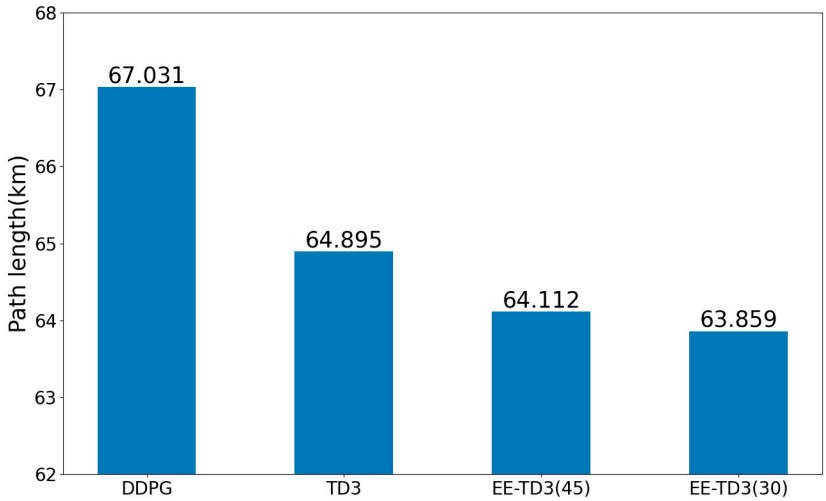


**Figure 11.** Average path length of the algorithm model.

## 6. Conclusions

This paper proposes the Environment Exploration Twin Delayed Deep Deterministic Policy Gradient (EE-TD3) algorithm for UAV autonomous path planning in complex 3D environments. First, the mission process of the UAV in 3D battlefield simulation environment is modeled, and the modeling of low-space 3D dynamic battlefield simulation environment is also completed. Then, a dynamic reward function with enlightening

guidance is designed to solve the problem of sparse rewards in the tradition, which can allow the algorithm model to converge faster. On the basis of these early work, the DDPG algorithm was simulated by the td3 algorithm and the EE-TD3 algorithm, the noise attenuation factor was added to accelerate the convergence of the model, and nine different learning rates were set up. Based on the experimental results, the convergence rate of the optimal results, and the success rate of the model convergence, we found that the model training effect of the EE-TD3 algorithm is superior to the TD3 algorithm model and the DDPG algorithm model. Finally, the performance test of the algorithm is performed, and the performance effect of the EE-TD3 algorithm model is the best, and the smaller the Angle of the area segmentation, the better the performance of the algorithm.

The encoding mechanism of the environmental exploration state in the EE-TD3 algorithm model changes the input mode of the algorithm model. In the future work, the input mode of the environment information can be further changed to make it obtain environmental information efficiently and promote the algorithm model to make better decisions. In addition, the EE-TD3 algorithm can be combined and compared with other deep reinforcement learning algorithms, such as DQN algorithm and SAC algorithm.

**Author Contributions:** Conceptualization, D.Z.; methodology, D.Z. and K.C.; software, D.Z.; validation, D.Z., X.L. (Xiongwei Li) and G.R.; formal analysis, K.C. and D.Z.; investigation, G.R.; resources, X.L. (Xiongwei Li), K.C., J.Y. and X.L. (Xi Li); data curation, X.L. (Xi Li); writing original draft preparation, D.Z.; writing—review and editing, D.Z., J.Y., G.R., X.L. (Xiongwei Li), K.C. and X.L. (Xi Li); visualization, D.Z. and J.Y.; supervision, G.R., X.L. (Xiongwei Li) and K.C.; project administration, G.R.; funding acquisition, K.C. and X.L. (Xiongwei Li). All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the National Natural Science Foundation of China, grant number 62071483.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare that they have no conflict of interest.

## References

1. Tomic, T.; Schmid, K.; Lutz, P.; Domel, A.; Kassecker, M.; Mair, E.; Grixa, I.L.; Ruess, F.; Suppa, M.; Burschka, D. Toward a fully autonomous UAV: Research platform for indoor and outdoor urbansearch and rescue. *IEEE Robot. Autom. Mag.* **2012**, *19*, 46–56. [CrossRef]
2. Stevens, R.; Sadjadi, F.; Braegelmann, J.; Cordes, A.; Nelson, R. Small unmanned aerial vehicle (UAV) real-time intelligence, surveillance and reconnaissance (ISR) using onboard pre-processing. In Proceedings of the Automatic Target Recognition XVIII, Orlando, FL, USA, 16–20 March 2008; Volume 6967.
3. Sira-Ramírez, H.; Castro-Linares, R.; Puriel-Gil, G. An active disturbance rejection approach to leader-follower controlled formation. *Asian J. Control* **2014**, *16*, 382–395. [CrossRef]
4. Wu, C.; Shi, S.; Gu, S.; Zhang, L.; Gu, X. Deep reinforcement learning-based content placement and trajectory design in urban cache-enabled UAV networks. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 8842694. [CrossRef]
5. Wu, J.; Wang, H.; Li, N.; Yao, P.; Huang, Y.; Yang, H. Path planning for solar-powered UAV in urban environment. *Neurocomputing* **2018**, *275*, 2055–2065. [CrossRef]
6. Liu, Y.; Wang, H.; Su, Z.; Fan, J. Deep learning based trajectory optimization for UAV aerial refueling docking under bow wave. *Aerosp. Technol.* **2018**, *80*, 392–402. [CrossRef]
7. Guruji, A.K.; Agarwal, H.; Parsediya, D.K. Time-efficient A* algorithm for robot path planning. *Procedia Technol.* **2016**, *23*, 144–149. [CrossRef]
8. Luo, M.; Hou, X.; Yang, J. Surface optimal path planning using an extended Dijkstra algorithm. *IEEE Access* **2020**, *8*, 147827–147838. [CrossRef]
9. Luo, G.; Yu, J.; Mei, Y.; Zhang, S. UAV path planning in mixed-obstacle environment via artificial potential field method improved by additional control force. *Asian J. Control* **2015**, *17*, 1600–1610. [CrossRef]
10. Zhu, Q.; Yan, Y.; Xing, Z. Robot path planning based on artificial potential field approach with simulated annealing. In Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications, Jinan, China, 16–18 October 2006; IEEE: New York, NY, USA, 2006; Volume 2, pp. 622–627.
11. Lifen, L.; Ruoxin, S.; Shuandao, L.; Jiang, W. Path planning for UAVS based on improved artificial potential field method through changing the repulsive potential function. In Proceedings of the 2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC), Nanjing, China, 12–14 August 2016; IEEE: New York, NY, USA, 2016.

12. Waydo, S.; Murray, R.M. Vehicle motion planning using stream functions. In Proceedings of the 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422), Taipei, Taiwan, 14–19 September 2003; IEEE: New York, NY, USA, 2003; Volume 2, pp. 2484–2491.
13. Zhao, Y.; Zheng, Z.; Liu, Y. Survey on Computational-Intelligence-Based UAV Path Planning. *Knowl.-Based Syst.* **2018**, *158*, 54–64. [CrossRef]
14. Wang, G.G.; Chu, H.C.E.; Mirjalili, S. Three-dimensional path planning for UCAV using an improved bat algorithm. *Aerosp. Sci. Technol.* **2016**, *49*, 231–238. [CrossRef]
15. Song, P.C.; Pan, J.S.; Chu, S.C. A parallel compact cuckoo search algorithm for three-dimensional path planning. *Appl. Soft Comput.* **2020**, *94*, 106443. [CrossRef]
16. Du, N.; Zhou, Y.; Deng, W.; Luo, Q. Improved chimp optimization algorithm for three-dimensional path planning problem. *Multimed. Tools Appl.* **2022**, *81*, 27397–27422. [CrossRef]
17. Wang, H.; Lyu, W.; Yao, P.; Liang, X.; Liu, C. Three-dimensional path planning for unmanned aerial vehicle based on interfered fluid dynamical system. *Chin. J. Aeronaut.* **2015**, *28*, 229–239. [CrossRef]
18. Sewak, M. *Deep Reinforcement Learning*; Springer: Singapore, 2019.
19. Ben Amarat, S.; Zong, P. 3D path planning, routing algorithms and routing protocols for unmanned air vehicles: A review. *Aircr. Eng. Aerosp. Technol.* **2019**, *91*, 1245–1255. [CrossRef]
20. Yang, L.; Qi, J.; Song, D.; Xiao, J.; Han, J.; Xia, Y. Survey of robot 3D path planning algorithms. *J. Control. Sci. Eng.* **2016**, *2016*, 7426913. [CrossRef]
21. Chen, J.; Du, C.; Zhang, Y.; Han, P.; Wei, W. A clustering-based coverage path planning method for autonomous heterogeneous UAVs. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 25546–25556. [CrossRef]
22. Lamini, C.; Benhlima, S.; Elbekri, A. Genetic algorithm based approach for autonomous mobile robot path planning. *Procedia Comput. Sci.* **2018**, *127*, 180–189. [CrossRef]
23. Low, E.S.; Ong, P.; Cheah, K.C. Solving the optimal path planning of a mobile robot using improved Q-learning. *Robot. Auton. Syst.* **2019**, *115*, 143–161. [CrossRef]
24. Yang, Y.; Juntao, L.; Lingling, P. Multi-robot path planning based on a deep reinforcement learning DQN algorithm. *CAAI Trans. Intell. Technol.* **2020**, *5*, 177–183. [CrossRef]
25. Lin, G.; Zhu, L.; Li, J.; Zou, X.; Tang, Y. Collision-free path planning for a guava-harvesting robot based on recurrent deep reinforcement learning. *Comput. Electron. Agric.* **2021**, *188*, 106350. [CrossRef]
26. Chen, P.; Pei, J.; Lu, W.; Li, M. A deep reinforcement learning based method for real-time path planning and dynamic obstacle avoidance. *Neurocomputing* **2022**, *497*, 64–75. [CrossRef]
27. He, L.; Aouf, N.; Song, B. Explainable Deep Reinforcement Learning for UAV autonomous path planning. *Aerosp. Sci. Technol.* **2021**, *118*, 107052. [CrossRef]
28. Zhang, S.; Li, Y.; Dong, Q. Autonomous navigation of UAV in multi-obstacle environments based on a Deep Reinforcement Learning approach. *Appl. Soft Comput.* **2022**, *115*, 108194. [CrossRef]
29. Zhou, X.; Gao, F.; Fang, X.; Lan, Z. Improved bat algorithm for UAV path planning in three-dimensional space. *IEEE Access* **2021**, *9*, 20100–20116. [CrossRef]
30. Serrano, W. Deep Reinforcement Learning Algorithms in Intelligent Infrastructure. *Infrastructures* **2019**, *4*, 52. [CrossRef]
31. Guo, S.; Zhang, X.; Zheng, Y.; Du, Y. An autonomous path planning model for unmanned ships based on deep reinforcement learning. *Sensors* **2020**, *20*, 426. [CrossRef] [PubMed]
32. Fujimoto, S.; Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 1587–1596.
33. Papachristos, C.; Kamel, M.; Popović, M.; Khattak, S.; Bircher, A.; Oleynikova, H.; Dang, T.; Mascarich, F.; Alexis, K.; Siegwart, R. Autonomous Exploration and Inspection Path Planning for Aerial Robots Using the Robot Operating System. In *Robot Operating System (ROS)*; Springer: Cham, Switzerland, 2019; pp. 67–111.