UAV path planning based on the improved PPO algorithm

Chenyang Qi, Chengfu Wu, Lei Lei, Xiaolu Li, Peiyan Cong
School of Automation,
Northwestern Polytechnical University
Xi'an, China
e-mail: 2020262547@mail.nwpu.edu.cn chiefwu@nwpu.edu.cn
lei_lei@mail.nwpu.edu.cn lixiaolu@mail.nwpu.edu.cn 2252169660@qq.com

Abstract—**In this paper, we consider the problem of unmanned aerial vehicle (UAV) path planning. The traditional path planning algorithm has the problems of low efficiency and poor adaptability, so this paper uses the reinforcement learning algorithm to complete the path planning. The classic proximal policy optimization (PPO) algorithm has problems that the samples with large rewards in the experience replay buffer will seriously affect training, this situation causes the agent's exploration performance degradation and the algorithm has poor convergence in some path planning tasks. To solve these problems, this paper proposes a frequency decomposition-PPO algorithm (FD-PPO) based on the frequency decomposition and designs a heuristic reward function to solve the UAV path planning problem. The FD-PPO algorithm decomposes rewards into multi-dimensional frequency rewards, then calculate the frequency return to efficiently guide UAV to complete the path planning task. The simulation results show that the FD-PPO algorithm proposed in this paper can adapt to the complex environment, and has outstanding stability under the continuous state space and continuous action space. At the same time, the FD-PPO algorithm has better performance in path planning than the PPO algorithm.**

Keywords- UAV; path planning; proximal policy optimization; frequency decomposition

I. Introduction

UAV has been widely used in the military for dangerous and harsh tactical tasks such as intelligence acquisition, tactical reconnaissance, and suppression of enemy air defense systems [1]. UAV path planning is the basis for UAV to perform tasks and an important link in UAV task assignment. It refers to planning one or more paths with the least cost and achievable under the constraints of its own performance, obstacles and various threats [2]. With the deepening of research, the path planning algorithm has many mature achievements, which can be roughly divided into traditional planning algorithms, intelligent planning algorithms and reinforcement learning algorithms. Traditional planning algorithms mainly include Dijkstra algorithm [3], A* algorithm [4], artificial potential field method [5], etc. Intelligent planning algorithms mainly include genetic algorithm [6], particle swarm algorithm [7], etc. Reinforcement learning algorithms mainly include Q-learning algorithm [8], DQN algorithm [9], DDPG algorithm [10], and PPO algorithm used in this paper [11] et al.

Cheng et al. [12] improved the basic Dijkstra algorithm by calculating the residual error and constrained flight distance. Sudhakara et al. [13] proposed the enhanced artificial potential field (E-APF) method. This method solves the problem that the classical artificial potential field method cannot adapt to complex trajectory planning and is easy to fall into the local optimal solution. Chen et al. [14] proposed a global path planning method for robots in static environment based on genetic algorithm. Fernandes et al. [15] proposed the enhanced diversity particle swarm optimization (EDPSO) algorithm. The main characteristic of this algorithm is that it has peaks of diversity in its population, making it possible to escape from local minima effectively.

Reinforcement learning uses trial and error for exploration and does not require prior knowledge. It obtains rewards by instructing the agent to continuously interact with the environment [16]. So, reinforcement learning algorithms can effectively solve the problems of long running time, low solution efficiency and poor adaptability which exist in traditional path planning methods. The framework of reinforcement learning is shown in Fig. 1.

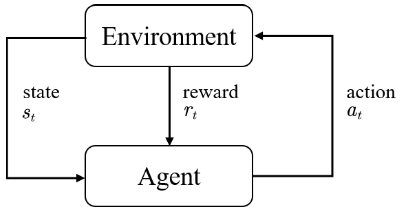

Fig. 1. Reinforcement learning framework

Researchers are increasingly interested in reinforcement learning. Zhou et al. [17] proposed a UAV path planning algorithm based on guided reinforcement Q-learning, which accelerated the convergence speed of the algorithm. Jia et al. [18] optimized the DDPG algorithm based on the potential field model in the artificial potential field method to realize path planning in complex environments. The PPO algorithm proposed by OpenAI [11] is improved on the Actor-Critic framework, inherits the step size selection mechanism of the trust region policy optimization algorithm, and draws on the policy-based estimation idea. But the PPO algorithm exists the problem of slow convergence and it falls into the dead zone easily. Li et al. [19] combined hierarchical reinforcement learning with the PPO algorithm that effectively jointly trains all levels of the hierarchical structure. Shen et al. [20] proposed an algorithm named proximal policy optimization based on self-directed action selection (SDAS-PPO). This algorithm adds an experience pool to guide action selection and improves the sample utilization rate, but there will be a problem of gradient explosion during the training process.

The overall framework of reinforcement learning in UAV path planning is shown in Fig. 2.

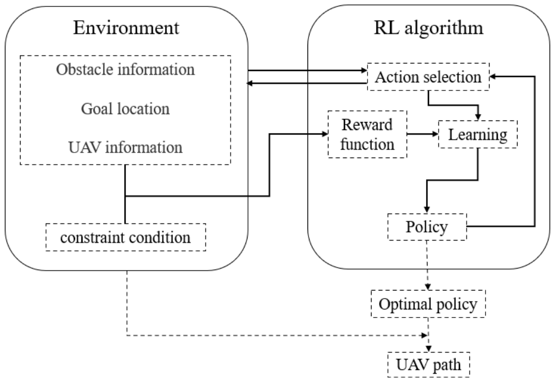

Fig. 2. UAV path planning framework

II. PROXIMAL POLICY OPTIMIZATION ALGORITHM

The PPO algorithm is a policy-based reinforcement learning algorithm, and it is an improvement to the policy gradient (PG) algorithm, which retains the advantage of excellent performance in continuous state space and continuous action space. The PG algorithm is updated by computing an estimate of the policy gradient and utilizing stochastic gradient ascent. The PG algorithm is the on-policy method, that is, the same policy is used for sampling and training [21]. But the PG algorithm is very sensitive to the step size, and it is difficult to choose a suitable step size. The definition of policy gradient is shown in (1).

$$\nabla \overline{R}_\theta = \hat{E}_t \left[\nabla_\theta \log \pi_\theta (a_t \mid s_t) R_t \right] \tag{1}$$

where π_θ is the random policy distribution and R_t is the total reward.

In order to make up for the shortcomings of the PG algorithm, on the one hand, the PPO algorithm introduces the off-policy when updating the parameters, it can ensure that a set of data obtained by sampling can be reused. On the other hand, the clipped surrogate objective [11] is used to ensure that there is little difference between the old and new policies.

The off-policy means that the sampling policy and the training policy are not the same policy. The method of importance sampling can ensure the update between different policies, that is, parameter θ in the sampling network can be used to update parameter θ' in the training network. The mathematical expression of importance sampling is as follows.

Suppose there is a function $f(x)$, we use data sampled from distribution q to estimate the expected value of function $f(x)$ with distribution p. Since the data is sampled from q, we use the equation shown in (2) to calculate the expected value.

$$E_{x\sim p}\left[f(x) \right] = \int f(x) \frac{p(x)}{q(x)} q(x) dx = E_{x\sim q}\left[f(x) \frac{p(x)}{q(x)} \right] \tag{2}$$

where $p(x)/q(x)$ is called the importance weight and can be used to correct the distribution.

So, the definition of policy gradient is shown in (3).

$$\nabla \overline{R}_\theta = \hat{E}_t \left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta'}(a_t \mid s_t)} \nabla_\theta \log \pi_\theta (a_t \mid s_t) A^{\theta'}(s_t, a_t) \right] \tag{3}$$

where (s_t, a_t) is sampled by the policy $\pi_{\theta'}$. $A^{\theta'}(s_t, a_t)$ is the advantage function.

The clipped surrogate objective refers to truncating the objective function to limit the gap between the two policies after updating, then achieve stable learning. The objective function is defined as (4).

$$\begin{aligned} J_{PPO}^{CLIP}(\theta) = \sum_{(s_t, a_t)} \min\Bigg(& \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta'}(a_t \mid s_t)} A^{\theta'}(s_t, a_t), \\ & clip\left(\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta'}(a_t \mid s_t)}, 1-\varepsilon, 1+\varepsilon \right) A^{\theta'}(s_t, a_t) \Bigg) \end{aligned} \tag{4}$$

where ε is a hyperparameter, and in this paper the value is 0.2.

The diagram of the restricted range of the objective function is shown in Fig. 3, The abscissa in the figure is $r_t(\theta) = \dfrac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta'}(a_t \mid s_t)}$.

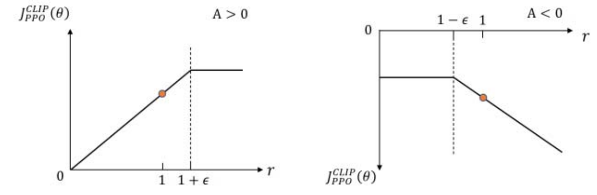

Fig. 3. restricted range of objective function

For the network structure with shared parameters between the policy and the value function, the error term of the value function and the entropy bonus term of the policy model are added to the objective function to encourage exploration. The objective function is shown in (5).

$$J_{PPO}(\theta) = E_t \left[J_{PPO}^{CLIP}(\theta) - c_1 (V_\theta(s) - V_{target})^2 + c_2 H(s, \pi_\theta) \right] \tag{5}$$

where c_1, c_2 are hyperparameters. $(V_\theta(s) - V_{target})^2$ is the mean square error of the value function. $H(s, \pi_\theta)$ is the entropy value of policy π_θ.

III. Frequency Decomposition-PPO Algorithm

In the UAV path planning, the UAV will obtain a larger reward only after reaching the target, and the reward obtained in the early exploration process is relatively small. Historical data with a large reward will affect the training of the PPO algorithm, causing the UAV to always perform actions with high reward when observing different states. For the above reasons, UAV will spend a lot of time exploring and learning, resulting in decreased exploration performance. And the PPO algorithm uses the method of clipped surrogate objective, which makes the algorithm has a slow convergence speed and is easy to fall into local optimum [22].To solve the above problems, the FD-PPO algorithm based on frequency decomposition is proposed, and this paper designs a heuristic reward function for path planning.

A. Frequency Decomposition

In this paper, frequency decomposition refers to the method of decomposing the reward into the sum of exponentially weighted components when calculating the returns. This method decomposes the reward into different frequencies, then calculates the return at each frequency and gives different weights to balance the return. Finally, this method can improve the training efficiency of the algorithm. The process of frequency decomposition is as follows.

Decompose reward into values with a base of 2, the length of the i-th segment frequency decomposition is 2^{i-1}. That is, the frequency value of the first segment is from 0 to 1, the frequency value of the second segment is from 1 to 3, the frequency value of the third segment is from 3 to 7, etc. The reward is decomposed into the frequency of each segment, and the filling ratio of each segment is the value of the frequency decomposition. Assuming that the reward is 9, the schematic diagram of its frequency decomposition is shown in Fig. 4. The frequency values of the first three segments are filled, a quarter of the fourth segment is filled, and the following are empty, so the frequency decomposition is (1, 1, 1, 0.25, 0, 0, …). Under the frequency decomposition method, the reward is decomposed into multi-dimensional frequency reward.

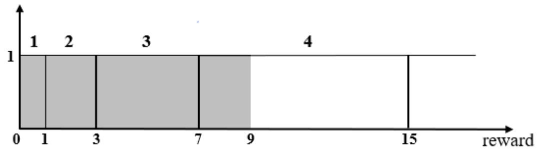

Fig. 4. diagram of frequency decomposition

Assuming that the reward is positive, the remaining amount after the reward value fills the $i-1$-th segment is:

$$r - \sum_{j=0}^{i-2} k^j = r - \frac{k^{i-1} - 1}{k - 1} \tag{6}$$

where r is the reward value. k is the base of the frequency decomposition.

Divide the remaining amount after filling the $i-1$-th segment by the length of the frequency of the i-th segment, and limit it between 0 and 1, to get the proportion of the i-th segment being filled. The proportion is also the value of the i-th element in the positive reward frequency decomposition. We also consider when the reward is negative:

$$r^i = sign(r) \times clamp((|r| - \frac{k^{i-1} - 1}{k - 1}) / k^{i-1}, 0, 1) \tag{7}$$

Equation (7) is to calculate the value of the i-th element in the frequency decomposition of the reward.

After decomposing the reward into multi-dimensional frequency reward, the frequency decomposition return needs to be calculated for each step in the episode. Calculating the return usually requires multiplying the reward for each step by a discount factor and summing it up. This method is modified in this paper to apply the method of frequency decomposition.

First, the multi-dimensional frequency reward is calculated according to (7), the values of all dimensions are multiplied by the discounts corresponding to each step. Then the values of the same dimensions for the subsequent steps are summed, and multiply the weight of each frequency segment. Finally add the values of each frequency segment to get the frequency decomposition return. The formula for calculating the frequency decomposition return in the episode is shown in (8):

$$G_{FD} = \sum_{j=1}^{M} \sum_{i=1}^{N} \gamma^{i-1} r_i^j \omega_j \tag{8}$$

where M represents the number of frequency segments. N represents the number of steps in an episode. γ is the discount factor. r_i^j represents the value of the j-th dimension in the frequency decomposition reward of the i-th step in the episode. ω_j is the weight of the j-th frequency segment.

The frequency decomposition value function is obtained by calculating the expectation of the frequency decomposition return:

$$V_{FD}(s) = E\left[G_{FD} \mid s_t = s\right] \tag{9}$$

The objective function of the PPO algorithm is shown in (5). In this paper, the proposed frequency decomposition value function is added to the objective function. The objective function of the FD-PPO algorithm is as follows:

$$J_{FD-PPO}(\theta) = E_t\left[J_{PPO}^{CLIP}(\theta) - c_1(V_\theta(s) - V_{FD}(s))^2 + c_2 H(s, \pi_\theta)\right] \tag{10}$$

Pseudocode for the FD-PPO algorithm is provided in Algorithm 1. It describes the training process of the FD-PPO algorithm in detail. First, initialize the network parameters and experience replay buffer, reset the environment state. After collecting enough samples according to the policy, calculate the frequency decomposition return G_{FD} and the advantage function $\hat{A}_t$. Then optimize the reward function and update the network parameters θ. For a set of sample data, it needs to be updated K times. Finally, copy the updated network parameters θ to θ_{old}.

Algorithm 1 FD-PPO algorithm

```
     Algorithm initialization. Initialize policy parameters $\theta$, initialize
     value function parameters $\phi$, initialize replay memory buffer $\mathcal{D}$
1:   for i = 0 to MaxEpisode do
2:       Initialize training environment, obtain the initial state $s_o$
3:       While True == Terminal Condition do
4:           Collect set of trajectories $\mathcal{D}$ by running policy $\pi_{\theta_{old}}$ in the
             environment
5:           Compute reward-to-go $\hat{R}_t$ and frequency decomposition
             return $G_{FD}$ in (12)
6:           Compute advantage estimates $\hat{A}_t$ based on the value
             function and frequency decomposition value function
             $V_{FD}(s)$ in (13)
7:           for k = 0 to K do
8:               Optimize objective function $J_{FD-PPO}(\theta)$ in (14)
9:               update the policy parameters $\theta$ use stochastic
                 gradient ascent
10:          end for
11:          $\theta_{old} \leftarrow \theta$
12:      end while
13:  end for
```

B. Heuristic Reward Function

The essence of the reward function in reinforcement learning is to convey the goal to the agent. Meanwhile, the reward function should motivate the agent to achieve the goal. Maximizing reward is the goal of the agent in the process of action selection. In the environment of UAV path planning, reinforcement learning algorithms are generally trained using discrete reward functions. At the beginning of training, since the UAV has little or no knowledge of the environment, it will randomly choose the action, and the goal can only be achieved after a long period. Inappropriate reward settings can also lead to situations where the agent is lazy. For example, in the UAV path planning, the UAV is easy to circle in place, accumulating reward and not exploring new areas when rewards are inappropriate.

To address the above problems, this paper designs a heuristic reward function that combines heuristic ideas into the reward function setting. It provides a better reward and punishment mechanism for the agent to achieve the purpose of improving the efficiency of the algorithm.

In UAV path planning, the setting of reward function is mainly related to four aspects: the distance between the UAV and the obstacle, the distance between the UAV and the target, the position of the UAV (to prevent exceeding the boundary), and the range of the UAV. Based on the above aspects, this paper adds heuristic ideas, and sets the reward function as follows.

r_1 represents the reward and punishment relationship between the UAV and the target and the obstacle.

$$r_1 = \begin{cases} 0, & d_1(t) < d_1(t-1) \\ -1, & d_1(t) \geq d_1(t-1) \\ 10, & d_1(t) < d_{goal} \\ -1, & d_2(t) < d_{obs} \end{cases} \tag{11}$$

where $d_1(t)$ represents the distance between the UAV and the target at the current moment, and it is the heuristic factor of the reward function. $d_1(t-1)$ represents the distance between the UAV and the target at the last moment. $d_2(t)$ represents the distance between the UAV and the nearest obstacle. $d_{goal}(t)$ represents the threshold from the target. d_{obs} represents the threshold of the distance between the UAV and the nearest obstacle.

r_2 represents the penalty when the UAV goes out of bounds.

$$r_2 = \begin{cases} 0, & X_{low} < x < X_{high}, Y_{low} < y < Y_{high} \\ -30, & otherwise \end{cases} \tag{12}$$

where X_{high} and X_{low} represent the upper and lower limits of the UAV in the horizontal direction. Y_{high} and Y_{low} represent the upper and lower limits of the UAV in the vertical direction.

r_3 represents the range limitation of UAV.

$$r_3 = \begin{cases} -L_{step}/2 + c, & d_1(t) < d_{goal} \\ 0, & otherwise \end{cases} \tag{13}$$

where L_{step} represents the number of decisions made by UAV and it is the heuristic factor of the reward function. c is a constant.

The number of decisions is proportional to the flight range of the UAV, so L_{step} is used to represent the range constraint of UAV.

The heuristic reward function is shown in (14).

$$R = r_1 + r_2 + r_3 + a / d_1(t) \tag{14}$$

where a is a constant, $d_1(t)$ and L_{step} are the heuristic factors of the reward function, which guide the UAV to reach the target faster and use a shorter range.

IV. PROBLEM DESCRIPTION AND MODELING

A. *UAV path planning*

UAV path planning problem is a basic problem in UAV application. In order to solve this problem conveniently, this paper establishes the UAV path planning model in a two-dimensional plane, as shown in Fig. 5.

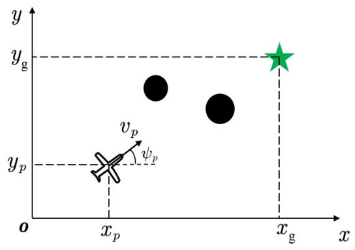

Fig. 5. Two-dimensional plane path planning model

In Fig. 5, the circle represents the position of the obstacle. The five-pointed star represents the target area. (x_p, y_p) represents the position of the UAV and (x_g, y_g) represents the position of the target. v_p and ψ_p represent the speed and the yaw of the UAV. The goal of the UAV is to avoid obstacles and find the optimal path that satisfies the constraints to reach the specified target.

In this paper, the UAV is simplified to the particle model, and regardless of physical characteristics. The kinematic equation of the UAV is shown in (15).

$$\begin{cases} \dot{x}_p = v_p \cos\psi_p \\ \dot{y}_p = v_p \sin\psi_p \\ \dot{\psi}_p = \omega_p \end{cases} \tag{15}$$

where ω_p represents the angular velocity of the UAV.

The fourth-order Runge-Kutta method is used to solve the kinematic equation of the UAV. And in UAV path planning, the enemy's air defense radar threat, mountain environment threat and missile threat are mainly considered. This paper simplifies threats as obstacles.

B. *State and Action Settings*

State space: The state of the UAV, the information of obstacles and the location of target will affect decision-making. This paper uses the joint state $s = [s_p, s_o, s_g]$ as the state for reinforcement learning. $s_p = [x_p, y_p, v_p, \psi_p]$ represents the state of the UAV, including the position, speed, and yaw of the UAV. $s_o = [s_{o1}, s_{o2}, \cdots, s_{on}]$ represents the state of obstacles in the environment, including the location information of obstacles. $s_g = [x_g, y_g, r]$ represents the state information of the target, including the position and radius of the target. It is assumed that the UAV carries the airborne GPS equipment and the airborne radar load equipment to obtain position, speed and obstacle information.

Action space: According to the particle model of the UAV, set the action of reinforcement learning to the linear velocity and angular velocity, that is, $a = [v_p, \omega_p]$ .In this paper, the linear velocity of the UAV is taken as a fixed value, and the angular velocity is a continuous value, which is closer to the real UAV motion model than discrete actions.

V. SIMULATIONS AND ANALYSIS

A. *Simulation Conditions and Setting Initial Values*

In order to verify the effectiveness and convergence of the FD-PPO algorithm in UAV path planning, this paper uses the python programming environment to experiment in different maps. At the same time, the simulation results of the PPO algorithm and the FD-PPO algorithm proposed in this paper are compared.

The training parameters set in the experiment are given as: clipping parameter $\varepsilon = 0.2$, $K = 80$, actor network learning rate $lr\_actor = 0.0003$, critic network learning rate lr_critic , discount factor $\gamma = 0.99$. The maximum number of training episodes is 3000, that is, $MaxEpisode = 3000$. If the number of steps in the episode exceeds 200, this episode will be judged as failure and the training will be restarted. The environmental parameters are shown in Table 1.

Table 1 UAV path planning simulation experiment parameters

Experimental parameter name	parameter value
Space dimension	2
UAV initial position x_p, y_p / m	[0,0]
UAV initial speed v_p / $(m \cdot s^{-1})$	15
UAV initial heading angle ψ_p	$[0, 2\pi]$
Angular velocity ω_p / $(rad \cdot s^{-1})$	[-1,1]
Boundary range x, y / m	[-300,800] × [-300,800]
Target area radius/ m	12

B. *Simulation results*

1) *Effectiveness Experiment*

In order to verify the effectiveness of the FD-PPO algorithm, this paper uses maps with different obstacle environments for simulation experiments. It can be seen in the Fig. 6 that the black areas are different types of obstacles. The green circle is the starting position of the UAV, the five-pointed star is the ending position of the UAV, the red dotted line is the path of the UAV, the red area is the target area.

In Fig. 6 and Fig. 8, different obstacles are set up, in which the circular obstacles represent enemy detection equipment, and the rectangular obstacles represent the obstacles of terrain. Fig. 6 and Fig. 8 are the path simulation diagrams after using the FD-PPO algorithm to train. It can be seen from the figure that UAV successfully avoids the obstacles and reaches the target without collision, also the path satisfies constraints of the UAV. Fig. 7 and Fig. 9 are the average reward curves in the training process under the two maps. The abscissa represents the number of training steps, and the ordinate represents the reward. After training, the reward curves begin to converge at about 150,000 steps in both cases, and the maximum value of the reward appears.

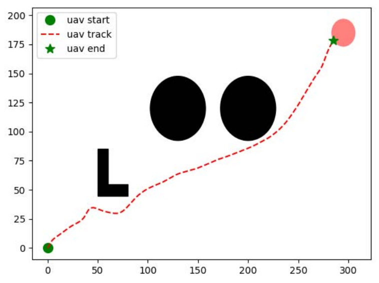

Fig. 6. Path planning trajectory

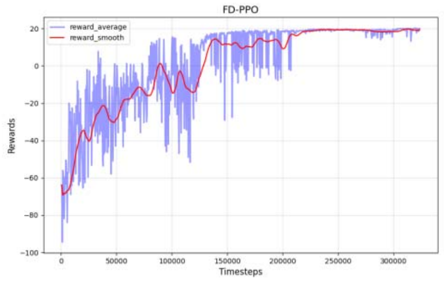

Fig. 7. Average reward curve

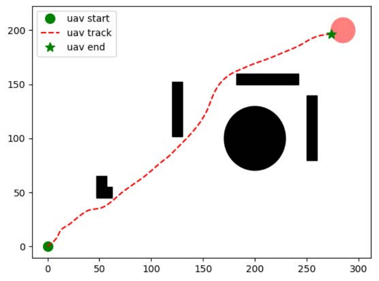

Fig. 8. Path planning trajectory

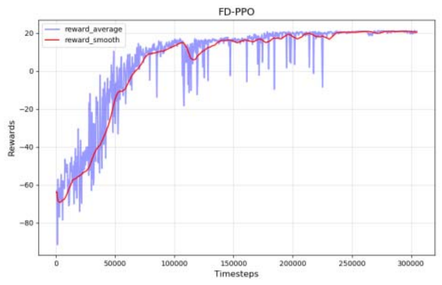

Fig. 9. Average reward curve

In order to verify the applicability of the algorithm in a variety of environments, a map with dense obstacles is set up in this paper. In this environment, the obstacles are relatively dense, and the UAV needs to avoid a large number of obstacles to complete the task. The flight trajectory of the UAV path planning is shown in Fig. 10. After a period of training, the UAV avoids dense obstacles and reaches the target. Fig. 11 is the average reward curve. At around 120,000 steps, the reward curve begins to converge, and the maximum value of the reward appears.

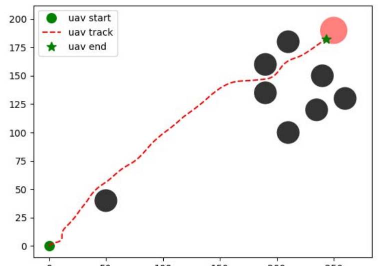

Fig. 10. Path planning trajectory

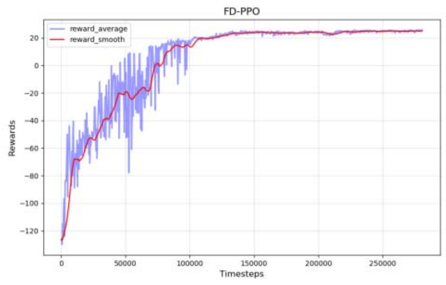

Fig. 11. Path planning trajectory

The above simulation results show that the FD-PPO algorithm can successfully plan a path that satisfies the constraints of the UAV under different obstacle environments, and has excellent convergence results.

2) *Comparative Experiment*

In order to compare the performance of the FD-PPO algorithm and the PPO algorithm, the two algorithms are simulated in the same map. Fig. 12 is the comparison result of the average rewards obtained by the two algorithms in the selected map, where the red curve is the simulation result using the FD-PPO algorithm, and the green curve is the simulation result using the PPO algorithm. It can be seen from the figure that the two algorithms have basically reached the convergence, but the convergence of the FD-PPO algorithm is significantly better than that of the PPO algorithm. The FD-PPO algorithm begins to converge at about 130,000 steps, the maximum value of the reward begins to appear, and the PPO algorithm begins to converge around 230,000 steps. After the algorithms converge, the reward of the FD-PPO algorithm is greater than that of the PPO algorithm, indicating that the path length planned by the FD-PPO algorithm is superior to that of the PPO algorithm. At the same time, the FD-PPO algorithm is more stable after the reward converges, indicating that the model trained by the FD-PPO algorithm has better robustness.

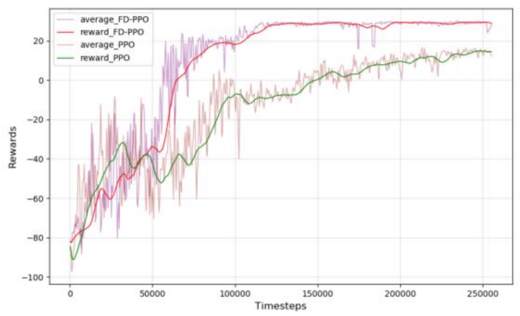

Fig. 12. Simulation results of the comparative experiment

The performance of the algorithm in the testing phase is also an important reference. As shown in Table 2, the two trained models are tested for 200 episodes, and the test results mainly include the average reward, path length, and success rate (reach the target and no collision is defined as success).

Table 2 Test results of two algorithms

Algorithm	Average reward	Path length (m)	Success rate (%)
FD-PPO	29.59	377.75	97.5
PPO	18.36	431.82	91.9

From Table 2, it can be concluded that the FD-PPO algorithm has a higher average reward, a shorter planned path length and a higher success rate than the PPO algorithm. That is to say that the FD-PPO algorithm has achieved excellent training results.

VI. CONCLUSION

In this paper, we proposed the FD-PPO algorithm based on the frequency decomposition and designed a heuristic reward function to solve the problem of UAV path planning. We consider the actual UAV motion model and select the continuous action space in the modeling process. The main conclusions are as follows:

In this paper, we set up different environments to verify the effectiveness of FD-PPO algorithm. The simulation results show that the UAV can excellently complete the task in various environments, and the convergence and stability of the algorithm are outstanding. In order to compare the effect of the FD-PPO algorithm and the PPO algorithm, we set up a comparative experiment. The simulation results show that the FD-PPO algorithm has faster convergence speed, higher reward and better stability. In the test environment, the FD-PPO algorithm outperforms the PPO algorithm in terms of average reward, path length and success rate.

In the actual environment, the obstacles will be more complex and there are dynamic obstacles. In the future, the research will be on the UAV path planning in more complex environments, while considering the environments with dynamic obstacles.

References

[1] Shen, L.C. Theories and Methods of Autonomous Cooperative Control of Multiple UAVs. National Defense Industry Press, Beijing 2013.

[2] Fan, J., Lei, T. A Survey of UAV Path Planning. Journal of Zhengzhou University (Engineering Science), 2021, 42(03): 39-46.

[3] Wang H, Yuan Y, Yuan Q. Application of Dijkstra algorithm in robot path-planning. IEEE, 2011.

[4] Akshay Kumar Guruji, Himansh Agarwal, D.K. Parsediya. Time-efficient A* Algorithm for Robot Path Planning[J]. Procedia Technology, 2016, 23.

[5] Zhang J Y, Zhao Z P, Liu D. A path planning method for mobile robot based on artificial potential field. Journal of Harbin Institute of Technology, 2006, 38(8): 4.

[6] Y. Volkan Pehlivanoglu, Perihan Pehlivanoglu, An enhanced genetic algorithm for path planning of autonomous UAV in target coverage problems, Applied Soft Computing, Volume 112,2021,107796, ISSN 1568-4946.

[7] Ss A, Yu P A, Ch B, et al. Efficient path planning for UAV formation via comprehensively improved particle swarm optimization[J]. ISA Transactions, 2020, 97: 415-430.

[8] Watkins C, Dayan P. Technical Note: Q-Learning[J]. Machine Learning, 1992, 8(3-4): 279-292.

[9] Osband I, Blundell C, Pritzel A, et al. Deep Exploration via Bootstrapped DQN. 2016.

[10] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., & Tassa, Y., et al. Continuous control with deep reinforcement learning. Computer ence, 2015.

[11] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. Proximal policy optimization algorithms. 2017.

[12] Cheng, N.Y., Liu, Z.Q., LI, Y.Q. Path Planning Algorithm of Dijkstra-Based Intelligent Aircraft under Multiple Constraints. Journal of Northwestern Polytechnical University, 2020, 38(06): 1284-1290.

[13] Priyanka Sudhakara, Velappa Ganapathy, B. Priyadharshini, Karthika Sundaran, Obstacle Avoidance and Navigation Planning of a Wheeled Mobile Robot using Amended Artificial Potential Field Method, Procedia Computer Science, Volume 133, 2018, Pages 998-1004, ISSN 1877-0509.

[14] Chen, H.H., Du, X., Gu, W.K. Genetic algorithm based global path planning in a static environment. Journal of Zhejiang University (Science Edition), 2005, 32(1): 49-53.

[15] P.B. Fernandes, R.C.L. Oliveira, J.V. Fonseca Neto, Trajectory planning of autonomous mobile robots applying a particle swarm optimization algorithm with peaks of diversity, Applied Soft Computing, Volume 116, 2022, 108108, ISSN 1568-4946.

[16] Sutton R S, Barto A G. Reinforcement Learning: An Introduction[J]. IEEE Transactions on Neural Networks, 1998, 9(5): 1054.

[17] Zhou, B., Guo, Y., Li, N., et al. Path planning of UAV using guided enhancement Q-learning algorithm. Acta Aeronautica et Astronautica Sinica, 2021, 42 (in Chinese).

[18] Jia, Q., Yang, M., Miao, Y., & Li, X. Path planning using deep reinforcement learning based on potential field in complex environment. Journal of Physics: Conference Series, 2021, 1748(2), 022016 (6pp).

[19] Li, A. C., Florensa, C., Clavera, I., & Abbeel, P. Sub-policy Adaptation for Hierarchical Reinforcement Learning. In: International Conference on Learning Representations. 2019.

[20] Shen, Y., Liu, Q. Proximal Policy Optimization Based on Self-directed Action Selection. Computer Science, 2021, 48(12): 297-303.

[21] Bie, T., Zhu, X.Q., Fu, Y., et al. Safety priority path planning method based on Safe-PPO algorithm. Journal of Beijing University of Aeronautics and Astronautics, 2021, 1-15, DOI:10.13700/j.bh.1001-5965.2021.0580.

[22] Pan, L., Cai, Q., Huang, L. Multi-Path Policy Optimization. In: International Joint Conference on Autonomous Agents and Multi-agent Systems. New Zealand. 2020, pp. 1001-1009.