ISSN 13303651 (Print), ISSN 18486339 (Online)
https://doi.org/10.17559/TV-20250211002362
Original scientific paper

Implementation of Unmanned Aerial Vehicle Flight Path Planning Software Based on Integral Compensation Reinforcement Learning Algorithm

Xianlong MA

Abstract: Due to the complex nature of UAVs, flight path planning is a very difficult task. This paper proposes a UAV trajectory planning algorithm that combines model-free reinforcement learning with an improved depth deterministic strategy gradient algorithm with integral compensation. The trajectory planning problem is modeled as a Markov decision process with different degrees of environmental information missing, and the actions given by the trained agent from the observed state are taken as prior knowledge through the integral compensation method. PPO algorithm is used to train agents off-line in the established flight environment simulator, and the curvature of the trajectory is guaranteed by improving the correlation of the agent's action in time. The experimental results show that this method can generate curvature smooth flight path, and has a high success rate in complex flight environments. It can be extended to different flight environments.

Keywords: adaptive search; depth deterministic strategy gradient; flight path planning; integral compensation; track cost; unmanned aerial vehicles

1 INTRODUCTION

Flight path planning of unmanned aerial vehicle is the most important part of mission planning, and it is also one of the key technologies of unmanned aerial vehicle cooperative combat and future networked combat. Reasonable planning can make UAVs ((Unmanned Aerial Vehicles)) evade threats effectively, improve survival probability and combat efficiency. Intelligent optimization algorithm is the most widely used method to solve the flight path planning problem, and the Ant Colony Optimization (ACO) algorithm based on grid model is one of the most important methods. Compared with the VORONOI diagram, this method can automatically search the minimum cost track in free space without setting navigation nodes or constructing VORONOI diagram, and has strong adaptive ability [1], and is suitable for different threat types.

In contrast, reinforcement Learning (RL) is in the repair of real time, excellent generalization performance and commonality of design process, making it better in the course planning of robots, drones and other fields. Probabilistic roadmaps [4] are used to segment multiple local target points on a large map [2, 3]. Then, a reinforcement learning agent trained by deep deterministic policy gradient [5] guides the robot to move towards the local target point, which solves the remote path planning problem of the robot in a complex environment. Taking the whole map image as the observation state of the reinforcement learning agent [6], the deep Q network algorithm adopted is superior to A* and D* [7] in the environment of static and dynamic obstacles. All the above methods are based on model free reinforcement learning, and all use the mode of "offline training + use of unmanned aerial vehicles", but the countermeasures for failure of reinforcement learning agents in the use stage of unmanned aerial vehicles are not discussed. Although the gradient optimization method used in the offline training stage can be applied to the training of reinforcement learning agents in the use stage of unmanned aerial vehicles as a countermeasure, it consumes too much computing resources and reduces the real-time performance.

The dynamic antiintegral Compensation (IC) mechanism [8] contains internal state variables and is complicated in design. The stability criterion of the nonconvex form can be converted into the stability criterion of the LMI form by using the projection theorem. The feasibility conditions are less limited and the system dynamic response is good. For the above antiintegral compensation method, when the system is open loop stable (for linear systems, the state matrix of the controlled object is Hurwitz's), the global asymptotic stability can be achieved. When the open loop system is unstable, only local asymptotic stability can be achieved [9].

In the use stage of UAV, this paper combines integral compensation with reinforcement learning algorithm, and proposes an integral compensation reinforcement learning algorithm planning method, which only uses integral compensation to optimize the planning strategy of reinforcement learning agent. At the same time, a simple and effective motion filter is designed to ensure the stationarity of orbit curvature. Experimental results show that performance can be improved by allowing reinforcement learning agents to obtain richer environmental information, and the success rate of integral-compensated reinforcement learning algorithms with satisfactory trajectory planning can be improved. Finally, in the special environment with dense threat distribution, dynamic threat quantity and U shaped threat distribution, the robustness of the integral compensation reinforcement learning algorithm is verified.

2 RELATED WORK

The flight environment of unmanned aerial vehicles (UAVs) is becoming increasingly complex. Due to their own advantages and broad development prospects, UAVs have attracted more and more attention from the military forces of various countries. Therefore, their flight environment (also known as the confrontation environment) has become more and more complex. In addition to conventional threats and obstacles, the dynamics and uncertainties of the flight environment also need to be considered. Moreover, due to their own performance constraints, UAVs often cannot obtain accurate and real time global situation information and cannot respond to

changes in the situation in a timely manner.

The path planning problem is essentially an optimization problem. In addition to the above mentioned modeling methods, it is proposed to use Mixed Integer Linear Programming (MILP) to describe the path planning problem [10]; it is proposed to use Genetic Programming to model the problem [11] and use relevant optimization strategies to solve the problem. However, such methods have a large amount of computation, and it is more difficult to find the optimal solution when considering the dynamic constraints of the task executor itself. In order to make the planned path smooth and flyable, the Pythagorean velocity graph path planning method [11] and the clothoid curve path planning method are proposed based on research conclusions. Aiming at the requirements of cooperative path planning for existing problems, a cooperative path planning strategy based on consistent coordination variables [12] and a cooperative path planning strategy based on game theory are proposed.

The cooperative task planning problem is a typical combinatorial optimization problem. References [13] have proved that it is an NP complete problem, so it is difficult to solve. In order to reduce the difficulty of solving, the cooperative task planning problem is usually simplified into two sub problems: cooperative task allocation and path planning, and their coupling is weakened or not considered. Hierarchical decoupling methods are used to solve them separately [14]. The common points of these solutions are: solving task allocation and path planning separately. When solving task allocation, only the temporal constraints of tasks themselves, the interaction relationships between tasks, etc. are considered, without considering the kinematic constraints of the objects themselves, environmental obstacles, and threat area constraints; path planning takes the results of task allocation as its input and searches for smooth paths that satisfy the constraints of UAVs themselves, obstacles, threat areas, and collision avoidance. The hierarchical decoupling method reduces the difficulty of solving the problem and can be applied to offline task planning problems with a determined task space situation. However, the actual task space situation is usually time varying and uncertain, and the calculation process of the hierarchical decoupling method is asynchronous. Due to the time consumption of the task allocation process, its calculation results do not respond to the current task space situation, so the resulting path planning results often cannot meet the actual task requirements and are likely to cause task failures. Therefore, the application range of the hierarchical decoupling solution method is limited and cannot meet the actual task requirements.

Based on this, Reference [15] has studied the integrated solution algorithm for the task planning problem. The Euclidean line segment set is used to represent the path planning results of UAVs [16], the equivalent of the cooperative task planning problem is completed by establishing a mixed integer linear programming model, and finally the problem is solved by an integrated solution method; the task planning model is established by using the graph theoretic equivalent method, and the cooperative task planning problem of the same UAVs is solved by a graph search algorithm [17]; a centralized integrated task planning solution method is used, the equivalent of the UAV cooperative task planning problem is completed by the graph theoretic method, the UAV flight path is represented by the Dubins path, and finally the problem is solved by a genetic algorithm; on the basis of Reference [18], the ammunition quantity constraints of UAVs are additionally considered, and the problem is solved by changing the gene coding method of the genetic algorithm. The above references all complete the modeling and solution of the cooperative task planning problem based on a centralized architecture, and the planning and calculation process is completely completed on the ground station or a certain UAV. However, the centralized architecture generally has the disadvantages of strong dependence on the communication link and poor reliability, and its practicality is poor.

Some scholars have introduced reinforcement learning related algorithms into UAV control. A smart flight control system is constructed using reinforcement learning for the inner loop attitude control of UAVs, and its accuracy and performance are better than traditional PID UAV path planning [19]. The principle of reinforcement learning is used for motion planning to achieve collision avoidance of multiple UAVs in uncertain environments and in the presence of noise. Deep reinforcement learning is used for the trajectory optimization of UAVs [20]. The deep deterministic policy gradient (RL) algorithm in reinforcement learning is used to achieve the autonomous landing of UAVs on mobile platforms [21]. The RL algorithm uses neural networks to approximate the value function and directly differentiates the value expectation without the need to select the optimal action, which is suitable for the continuous action state space situation of UAV path planning. However, the conventional RL algorithm is prone to static errors when performing position control in UAV path planning. Regarding the cause of this error, some scholars have pointed out that the action value function derived from the Bellman equation in reinforcement learning is biased and inconsistent [22], and it is proved that the maximum operation in the Bellman equation introduces bias into the action value estimation, resulting in over estimation [23]. It shows that the distribution of state and action value estimates in reinforcement learning is skewed, making the agent tend to choose a strategy with lower expected returns and a higher probability of being over estimated, thus limiting the performance of the algorithm [24]. In order to reduce the impact of value estimation errors in the algorithm and achieve accurate tracking of the position in UAV path planning, a local backup process is proposed to ensure global consistency [25], thereby reducing the impact of estimation bias. However, this method will lead to an increase in algorithm complexity and is not conducive to deployment in UAV path planning.

UAV path planning methods are currently the focus of path planning research. Combining the rapidly exploring random tree in the dynamic domain with linear quadratic Gaussian motion planning, the planned path is better than the rapidly exploring random tree [26]. The central gravity optimization algorithm is improved to make this method applicable to dynamic environments and can effectively avoid threats. Task allocation is proposed for UAV path planning. In a dynamic environment, the tasks to be executed are segmented, and during the flight process, the

path is continuously updated to find the optimal path [27]. The D* (Dynamic A Star Algorithm) is also a commonly used path planning method based on the evaluation of the path cost function [28]. It expands nodes in the direction of the target point, uses the cost function to evaluate, selects the optimal node, further expands towards the target, and gradually completes the path planning. The characteristics of UAV path planning methods are as follows: path planning is carried out synchronously during the flight of the UAV, and the UAV needs to respond in a timely manner due to changes in the environment. The real time performance of path planning is the key point of such methods. Therefore, the generated path is often a feasible solution rather than the optimal solution. Aiming at the problems that the UAV path planning method cannot obtain the optimal solution and the offline path planning method cannot update the plan in real time, it is expected to propose an algorithm that can ensure the optimal path planning while taking into account real time performance. Since the reinforcement learning algorithm framework can incorporate other eligible algorithms into the population space and make up for the deficiencies of traditional algorithms, the reinforcement learning algorithm framework is adopted to incorporate UAV path planning and offline path planning into the algorithm population space, connect the two algorithms with knowledge, and combine the advantages of UAV path planning and offline path planning to propose a faster and more effective path planning method.

3 TRAJECTORY PLANNING OF UNMANNED AERIAL VEHICLES BASED ON INTEGRAL COMPENSATION REINFORCEMENT LEARNING ALGORITHM

3.1 Path Planning Method

Hypersonic vehicles are extremely costly. When applying reinforcement learning to real-world hypersonic vehicles, sample efficiency is the first issue to be considered. Therefore, this paper establishes an environment simulator to mimic the real flight environment. Since this paper focuses on the trajectory-planning problem of hypersonic vehicles rather than the guidance problem, the vehicle in the virtual environment is modeled as a simple model that can immediately respond to given commands.

Reinforcement learning methods learn the mapping from states to actions through interaction with the environment and can solve discrete-time Markov decision process problems. From the perspectives of partial observability and full observability, this paper models the trajectory planning problem as a partially observable Markov decision process (POMDP) and a Markov decision process (MDP).

The trajectory planning in this paper is divided into two stages: 1) In the offline training stage, a reinforcement learning (RL) agent that is not dependent on a fixed environment is trained as the baseline strategy for trajectory planning; 2) In the online planning stage, the Reinforcement Learning with Integral Compensation (RLIC) uses the environment simulator to predict future states for planning. Then, the strategy that outperforms the baseline strategy is selected as the execution strategy; otherwise, the baseline strategy will be used. In the first stage, an environment simulator is constructed to simulate the real flight environment, where there are vehicles, threats, and targets. Each time the environment simulator is reset, a random initial position is set for the vehicle, threats, and targets, which ensures that the planning strategy of the RL agent is not dependent on a fixed environment. The ultimate goal of this stage is to train an RL agent to control the virtual vehicle to successfully perform the penetration mission in the virtual environment. In the second stage, RL-IC plans the optimal strategy as the execution strategy, and the execution strategy generates an effective trajectory through interaction with the environment simulator. Fig. 1 illustrates this process. RL-IC first uses Integral Compensation (IC) to plan an integral compensation strategy that aims to maximize cumulative rewards over the next H time steps, starting from some initial state. Then, the strategy with the largest cumulative reward is selected as the optimal strategy between the baseline strategy and the integrated strategy. In the next round of planning, this optimal policy is used as the execution policy for interacting with the environment simulator. RL-IC plans steps every few times. The initial state of the program *Tp* time steps ahead of the actual state at the beginning of the current program, and both the initial state and subsequent states are predicted by the sub-simulator. Planning and executing operations are asynchronous, ensuring real-time performance of the RL-IC. At each sampling moment of the real vehicle system, the environmental simulator sends the attitude point of the virtual vehicle as the expected attitude point of the real vehicle. The motion filter ensures the smoothness of the desired attitude points in time. The environment simulator will update its dynamics regularly based on the real environment.

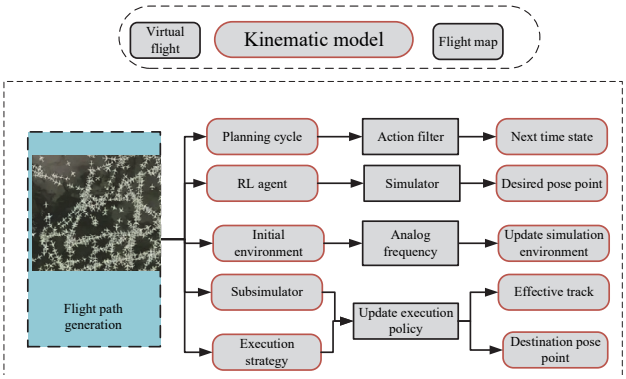

Figure 1 Flowchart of effective trajectory generation

3.2 Trajectory Planning Design of Unmanned Aerial Vehicles Based on the RL-IC Algorithm

In the trajectory planning of conventional unmanned aerial vehicles (UAVs) using reinforcement learning (RL), the position error is taken as the state. In the UAV trajectory planning based on the RL-IC algorithm, an integrator is introduced, and the errors at past time instants are accumulated to form the compensated state error, that is:

$$s_c^t = s_e^t + \lambda \sum_{i=1}^{t} s_e^i \tag{1}$$

Among them, s_c^t is the state error at time t, s_e^t is the compensated state error, and λ is a coefficient, usually with a relatively small value. In this way, the state input to the policy network includes both the current state error and the accumulated past errors. If there is a static error, the policy network will continuously output actions to reduce the static error until it becomes zero and the system reaches equilibrium.

Directly adding the state error after integral compensation to the original state space will increase the dimension of the state space, thereby increasing the complexity of the algorithm and the difficulty of exploration. Therefore, the original state error is replaced with the state error after integral compensation. This allows the state space to contain the accumulated error without increasing its dimension.

However, the introduction of the integral compensator increases the inertia of the system to some extent and prolongs the adjustment time of the UAV trajectory planning. Meanwhile, the defined integral-compensated state error includes the current error and all the accumulated past errors. As time and the number of steps in the control episode increase, the effect of the accumulated error becomes more and more significant. This may cause the control algorithm to overly consider the accumulated error of old data and reduce its attention to the current state error, thus degrading the performance of the algorithm.

To address this issue, a weighting factor λ is introduced to adjust the proportion of the past errors in the accumulated error. Therefore, the state error after integral compensation becomes:

$$s_c^t = s_e^t + \lambda \sum_{i=1}^{t} r^{t-i} s e^i \tag{2}$$

Among them, the addition of the coefficient gradually reduces the effect of the errors at previous time instants, ensuring that the algorithm focuses on the recent state errors. This enables the algorithm to strike a balance between paying attention to the current effects and the accumulated past errors.

The structure of the unmanned aerial vehicle (UAV) trajectory planning based on the RL-IC (Reinforcement Learning-Integrated Circuit) algorithm is shown in Fig. 2. The overall framework of the UAV trajectory planning adopts the policy-evaluation architecture. To meet the control requirements and ensure the training speed of the UAV trajectory planning, appropriate neural networks need to be selected to form the control policy and the evaluation function.

The policy network consists of a fully-connected neural network with two hidden layers. Each hidden layer contains 128 neurons, and the activation function of the hidden layers is the ReLU function. This policy network receives the compensated state error of the UAV and outputs four control signals. To ensure that the output control signals meet the requirements of the UAV, the output layer of this network uses the Tanh activation function.

The specific structure of the evaluation network is shown in Fig. 2. It also consists of a fully connected neural network with two hidden layers. Each hidden layer contains 128 neurons and uses the ReLU activation function. Inputs to the network include the current status of the drone and four control signals. Here, the drone status is entered directly as an input, and four control signals are entered in the first hidden layer. The output layer of the evaluation network uses a linear activation function to output an approximation of the q function, from which the policy gradient is derived for updating the policy network.

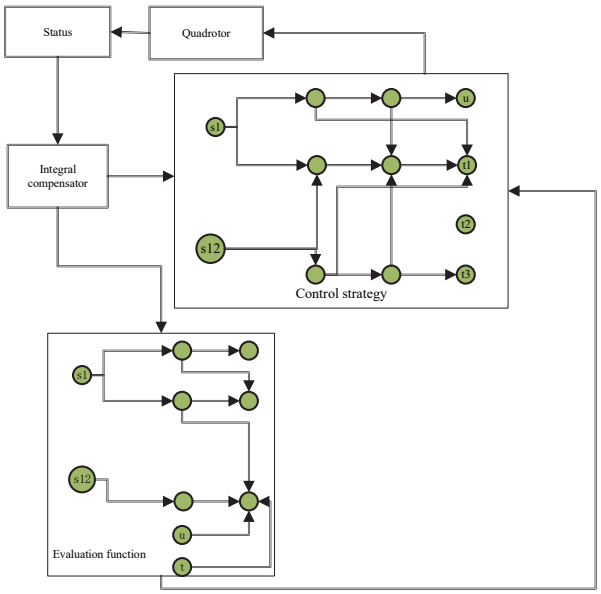

Figure 2 Structure of unmanned aerial vehicle trajectory planning based on RL-IC algorithm

3.3 Improved Reward Function

When solving the optimal path, a discrete reward function is usually used to evaluate the quality of actions. The specific application method is as follows: After receiving the actions of the mobile robot, the upper computer provides corresponding reward information according to the change of its state. Meanwhile, the mobile robot also selects the action with the maximum reward value according to the reward mechanism, so as to realize its learning and feedback on the environment.

In the initial stage of mobile robot training using the traditional Q-learning algorithm, it is unclear which action can obtain the maximum reward value. At this time, if the reward feedback is discrete, the mobile robot cannot make full use of the obtained information. Therefore, this paper designs a continuous heuristic reward function and defines the rewards in several different states, that is:

$$r = \begin{cases} -\infty, \text{ Obstacle} \\ 0, \text{ Other} \\ 1, \text{ End} \end{cases} \tag{3}$$

By calculating the Euclidean distance between the current coordinate point x and the end point coordinate y, a heuristic factor is introduced into the reward function. The calculation formula for the Euclidean distance is:

$$D = \sqrt{(x_{\text{now}} - x_{\text{end}})^2 + (y_{\text{now}} - y_{\text{end}})^2} \tag{4}$$

The calculation method of the reward function is:

$$R=\begin{cases} r+\eta(\frac{1}{D}m-\frac{1}{E}n), \text{Non-End} \\ 1, \text{End} \end{cases} \tag{5}$$

The heuristic reward function can provide continuous and immediate rewards to the robot during the learning process, enabling the robot to quickly determine the most valuable action in the action set under the current state. This ensures that the robot moves towards the end-point at each step, thereby improving the convergence efficiency of the algorithm.

First, set the initial value of the Q-matrix to 1. Given the positions of the starting point and the end-point, convert the two dimensional coordinates into a one-dimensional sequence. Since the starting point is random, it is necessary to compare the magnitudes of the serial numbers of the starting points. According to the relative position r between the starting point and the end-point, the value corresponding to each grid is increased or decreased accordingly u. The function for optimizing the initial process of the Q-matrix is expressed as:

$$Q_{\text{new}}(s_t,s_{t+1})=Q_{\text{old}}(s_t,s_{t+1})+\sqrt{(x_{\text{now}}-x_{\text{end}})^2}/D \tag{6}$$

The parameter update formula for the target network is:

$$\begin{aligned} u^{'}&=\lambda u+(1-\lambda)u^{'} \\ w^{'}&=\lambda w+(1-\lambda)w^{'} \end{aligned} \tag{7}$$

The parameter update formula for the action function is:

$$u_{i+1}=u_i+\lambda_u\nabla_u(D^u) \tag{8}$$

The training and learning process of the unmanned aerial vehicle (UAV) trajectory planning based on the RL-IC algorithm is summarized in Tab. 1.

Table 1 Optimization algorithm for UAV trajectory planning based on RL-IC

```
state_space = define_state_space()
action_space = define_action_space()
reward_function = define_reward_function()
integrator = create_integrator()
state_variables=perform_integral_compensation(integrator, state_space)
critic_network_params = random_initialization()
actor_network_params = random_initialization()
critic_network = create_network(critic_network_params)
actor_network = create_network(actor_network_params)
target_critic_network_params = critic_network_params
target_actor_network_params = actor_network_params
target_critic_network = create_network(target_critic_network_params)
target_actor_network = create_network(target_actor_network_params)
experience_replay_buffer = initialize_buffer()
for eachtraining_episode:
    N = initialize_noise_distribution()
    s = initialize_observation_state()
while episode_not_ended:
action = actor_network(s) + sample_noise(N)
    new_s, reward = apply_action(action)
    store_in_buffer(experience_replay_buffer, s, action, reward, new_s)
        s = new_s
    critic_network_params = update_critic_params(critic_network, experience_replay_buffer)
    critic_network=update_network(critic_network, critic_network_params)
    actor_network_params = update_actor_params(actor_network, experience_replay_buffer)
    actor_network=update_network(actor_network, actor_network_params)
    target_critic_network_params = soft_update(target_critic_network_params, critic_network_params)
    target_critic_network=update_network(target_critic_network, target_critic_network_params)
    target_actor_network_params = soft_update(target_actor_network_params, actor_network_params)
    target_actor_network=update_network(target_actor_network, target_actor_network_params)
```

The rewards for threat avoidance and target navigation are coupled with each other, which makes the design of the reward function difficult. In order to reduce the coupling between different rewards, this paper designs the target navigation reward from two aspects: when the connection between the aircraft and the target point intersects with the threat, the distance based reward is designed to guide the aircraft according to the change of the distance between the aircraft and the target point at adjacent moments. When there is no intersection, a heading based reward r is designed to guide the velocity vector of the vehicle towards the target point.

$$r_t^n=d_t^g-d_{t-1}^g \tag{9}$$

In addition, a generous reward is provided when the aircraft reaches the target point, and this reward is inversely proportional to the time taken to reach the target.

$$r_t^g=\begin{cases} 0, d_t^g>d_{\min}^g \\ 100*(1-\frac{T}{T_{\max}}), d_t^g\le d_{\min}^g \end{cases} \tag{10}$$

Finally, this paper takes into account the problem of minimizing the required overload of the trajectory. Since the action a of the RL agent is proportional to the required overload of the trajectory, a reward can be designed as follows:

$$r_t^a=-a^2 \tag{11}$$

3.4 RL-IC

Reinforcement learning itself is a black box optimization method. Usually, it is impossible to predict the harm caused by the agent's behaviour. Therefore, in order to ensure an effective remedy when the agent behaves abnormally, this paper combines RL and IC, and optimizes the strategy of the RL agent through IC in the practical application stage.

The parameterized strategy of IC is extremely sensitive to parameter changes. Especially when it is

necessary to plan a trajectory in a narrow flyable area, even a slight change in parameters can easily lead to the failure of the planning. Therefore, this paper uses the soft-update method to transition from the old Gaussian distribution to the new one (as shown), rather than directly replacing the old Gaussian distribution with the new one. The pseudo-code of RL-IC is provided in Tab. 2.

Table 2 Pseudo code of RL-IC

Input: Population size P, number of elite individuals E, planning length H, number of iterative optimizations K, IC strategy, initial mean, initial standard deviation, elite individual memory D, sub environment simulator, strategy of the RL agent, initial environment state

Set the initial state of the environment simulator as s.

Interact with the sub environment simulator H times usingthestrategy, and calculate the cumulative reward $r_{\text{sum}}^{RL} = \sum_{r=0...H} r\pi_\theta$.

$u = u_{\text{init}}, \delta = \delta_{\text{init}}$

For $k = 2$, …, K do

Sample P groups of IC strategy parameters from N.

Set the initial state of the sub environment simulator as s.

Interact with the sub-environment simulator H times using u, and calculate the cumulative reward $J(\theta_p^{CEM})$.

Select the top E elite individuals with the maximum cumulative reward.

Store the optimal elite individual in this round of optimization into D.

Calculate the new mean and standard deviation.

Select the elite individual with the highest cumulative reward in D as the optimal u.

In this method, although the path constructed by the RL agent can be guaranteed to be smooth, the curvature change on the path cannot be guaranteed to be smooth. In this paper, it is found that the curvature is not smooth due to the sudden change of the action in the adjacent moments of the RL agent, that is, the action has no correlation in time. In other words, this paper needs to ensure that the action instructions given by RL agents are time-dependent. In this paper, two different action filters, momentum filter and interpolation filter, are designed based on exponential weighted average. For convenience, this paper uses subscript t to represent the current decision times of the RL agent, and i to represent the simulation times of the current simulator.

Momentum filter: Assuming that the simulation number corresponding to the art time is i, then the action passed to the simulator at this time is a. In this paper, the action a given by the state s of the RL agent is used. As the expected value, the local average of the difference between a and a sim is calculated using the exponential weighted average, and then the smoothed action is calculated using this local average with a small action update rate:

$$v_i^d = \lambda v_{i-1}^d + (1-\lambda)a_t \tag{12}$$

Interpolation filter: Calculate the local average of action a given by the RL agent through the exponential weighted average, and use this local average as the smoothed action. Then, this paper uses Hermite interpolation to get the smooth transition function of the action at the adjacent decision time, and then calculates the actual action passed to the simulator. The transition function is obtained from the interpolation of constraints, which can effectively ensure the smoothness of the action.

$$a_t^{\text{smooth}} = \lambda a_{t-1}^{\text{smooth}} + (1-\lambda)a_t \tag{13}$$

5 SIMULATION VERIFICATION

In this paper, the PPO algorithm is used to train the RL agent, and the relevant hyperparameters are reported in Tab. 3. In addition, the soft actor-critic (SAC [22]) algorithm is additionally introduced for comparison. The selection of hyperparameters is based on the reference values provided in Ref. [22], with the additional use of prioritized experience replay [23], and a fine-tuning of the reward weight vector w. The trajectory planned by RL agent meets the constraints shown. Through the design of the reward function, the required overload and trajectory length (estimated flight time) of the trajectory are minimized as much as possible. In the training environment, 15 threats with a radius of 50 km are set, and the location of the threat is randomly generated each time the environment is reset.

Table 3 Hyperparameters in the training phase

Hyperparameter	Value
Number of vectorized environments	32
Gradient clipping threshold	5
The weight vector k of the reward	[1, 5, 1.5, 1, 1, 1]
Hyperparameter	[1, 5, 1.5, 1, 0.1, 1]

Fig. 3 reports the training results on three sets of random number seeds. The training results show that the modeling method proposed in this paper can achieve the same performance on different RL algorithms. PPO demonstrates more stable performance during the training process. The momentum filter has better performance than the interpolation filter. The MDP method has higher sample efficiency than the POMDP method, which, this paper believes, benefits from the network structure adopted in MDP and the introduction of global information.

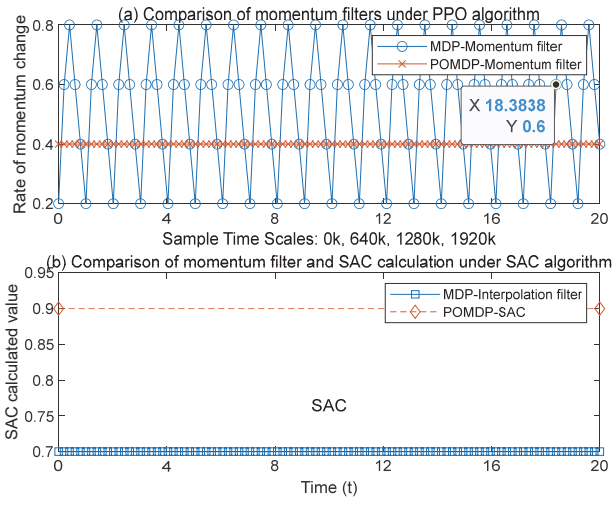

Figure 3 Training curves

Since PPO shows more stable performance during the training process, this paper will mainly use the agent trained by PPO for experimental verification. Fig. 4 shows

some successful cases and the change curves of the required overload of the trajectory. It can be seen from the figure that the action filters designed in this paper effectively ensure the smoothness of the required overload.

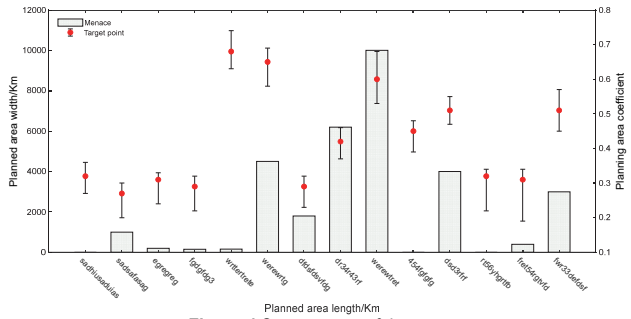

Figure 4 Some successful cases

In this paper, the penetration success rate is used as the evaluation index. All the trained RL agents are evaluated for 500 episodes in the training environment using the strategy without exploration noise, while RL-IC is evaluated for 100 episodes. The hyperparameters used by RL-IC are shown in Tab. 4. Their respective penetration success rates are shown in Fig. 5. It can be seen from the figure that the penetration success rate of the RL agents trained in MDP is significantly higher than that in POMDP. Moreover, after combining with RL-IC, the performance of each RL agent is greatly improved, and they all achieve an almost 100% success rate. In summary, the RL agents trained in MDP can provide a reliable baseline strategy, which can not only be used for planning but also serve as a backup strategy in case of emergencies. RL-IC makes up for the deficiencies of the RL agents and further improves the penetration success rate.

Table 4 Hyperparameters of RL-IC

Hyperparameter	Value
Population size	100
Elite number	10
Planned length	250
Number of iterative optimization	10
Soft replacement rate	0.25
Initial mean	[0…0]
Initial standard deviation	[0.5…0.5]

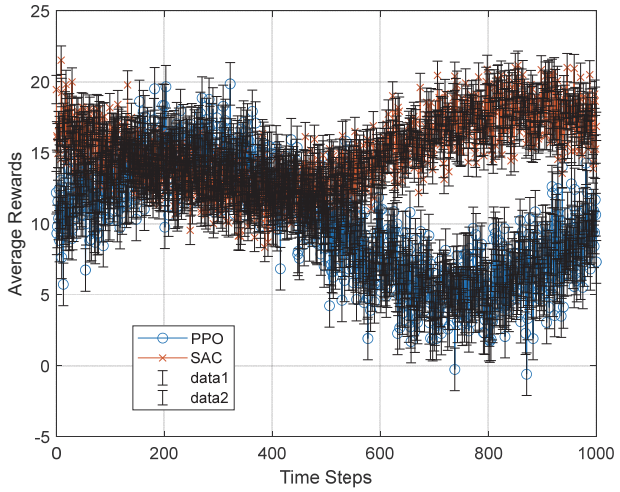

Figure 5 Penetration success rate

In the trajectory planning of unmanned aerial vehicles (UAVs) based on conventional RL algorithms, there are obvious static errors in position control. The position errors in the three directions of x, y, z are 0.5 m, 0.7 m, and 0.6 m respectively. However, the trajectory planning of UAVs based on the two RL-IC-based algorithms can stably control the UAV trajectory to the desired position, with a position error of 0. This indicates that the introduction of integral compensation can effectively eliminate static errors. Fig. 6 shows that the control signals output by the two RL-IC-based UAV trajectory planning algorithms are significantly smoother than those of the conventional RL-based UAV trajectory planning. This indicates that the introduction of integral compensation is beneficial to enhancing the stability of UAV trajectory planning.

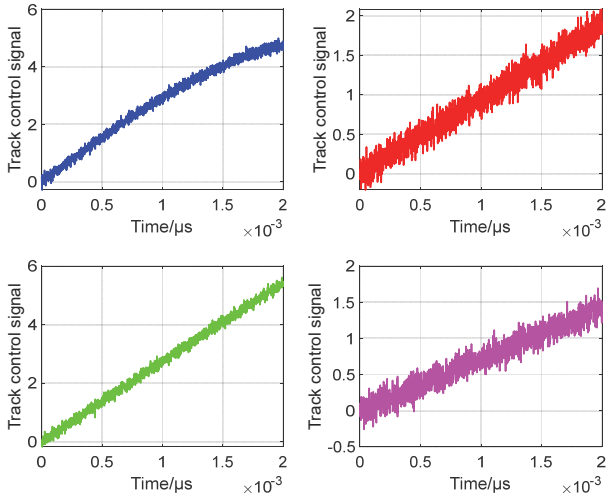

Figure 6 UAV trajectory planning signals

This setting is used to verify the performance of the algorithm when the reward function is rough. The control effects and control signals of UAV trajectory planning are shown in Fig. 7 and Fig. 8 respectively. In Fig. 7, the solid line represents the UAV trajectory planning based on the conventional RL algorithm, which has a poor control effect. The position and attitude response curves do not converge, indicating that in the case of a rough reward function, the UAV trajectory planning based on the conventional RL algorithm fails to learn a stable control strategy. The UAV trajectory planning based on the RL-IC ($r = 1$) algorithm is represented by the dashed line in the figure. Its control effect is better than that of the conventional RL-based UAV trajectory planning, and there is no oscillation in attitude control, indicating that the addition of cumulative error enhances the stability of the algorithm. However, the convergence speed of the RL-IC ($r = 1$) UAV trajectory planning is too slow, and it fails to control the position to the desired position within the 50-second simulation time. The reason for this problem is that the proportion of cumulative error is too large, causing the UAV trajectory planning to over-focus on the cumulative error, which leads to a decrease in the control rate and control performance of the UAV trajectory planning. The RL-IC ($r = 0.8$) UAV trajectory planning, represented by the dotted line in Fig. 5, can stably control the UAV to the desired position in a short time. This UAV trajectory planning uses the parameter to attenuate the past errors,

ensuring the balance between the current error and the past cumulative error, enabling the UAV trajectory planning to still learn a control strategy for stably controlling the UAV's attitude and position, while reducing the adjustment time. In addition, compared with the response curves in Fig. 3, when the reward function deteriorates, the control effect of the UAV trajectory planning based on the conventional RL algorithm becomes significantly worse, and it even fails to converge stably. In contrast, the two RL-IC-based UAV trajectory planning methods are less affected. Especially for the RL-IC UAV trajectory planning, the response curves under the two reward-function settings remain basically unchanged, indicating that the introduction of integral compensation can reduce the requirements of the learning algorithm for the setting of the reward function and improve the performance of the learning algorithm.

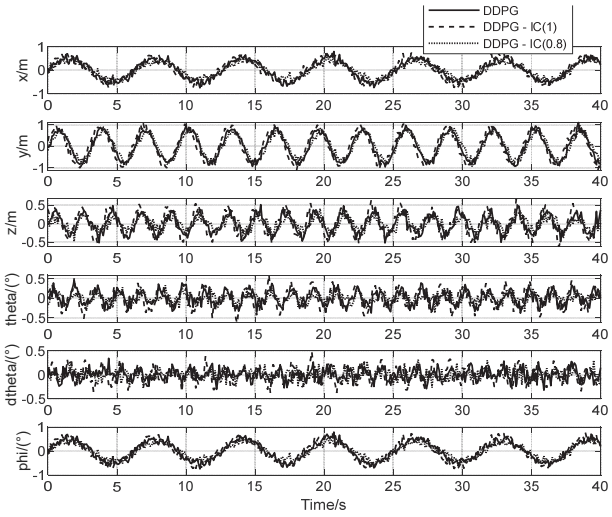

Figure 7 Position and attitude response curves under the action of UAV trajectory planning

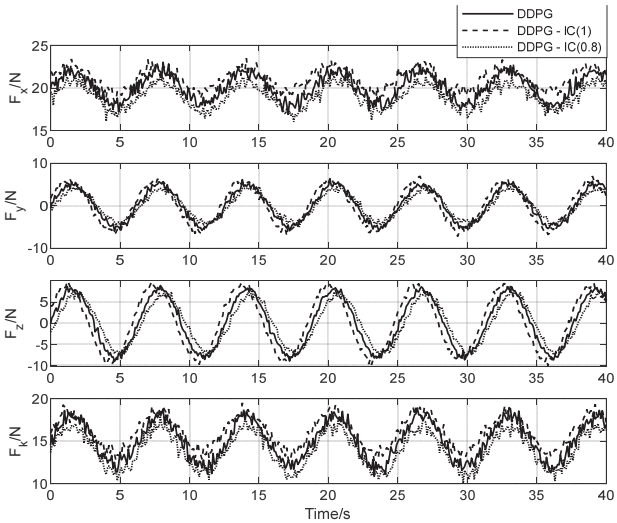

Figure 8 UAV trajectory planning signals

In the flight formation environment, four role networks (similar to four single-UAV environments) are used. Additionally, the formation state and formation reward are considered, and the RL-IC algorithm is adopted. This algorithm performs best in a single-UAV environment. The goal of this formation is to move from the starting point to the target point while maintaining a certain distance between UAVs. A total of ten networks are trained, including the A2C network for each UAV, as well as the target network and prediction network of RND. Fig. 9 shows the three-dimensional visualization of the learning process of the UAV formation (where the blue circles represent the air-defense networks, the black solid lines represent the movement paths of the UAVs, the red solid lines represent the movement paths of the missiles, the starting point is marked with a red five-pointed star, and the target point is marked with a green circle).

Fig. 9a shows the random maneuver 1 of the UAV formation during the learning process (with a decrease in external reward, the UAVs may move forward in a curved path or dodge when encountering missiles). Fig. 9b shows the random maneuver 2 of the UAV formation during the learning process (with a sharp decrease in external reward, the UAVs may dodge when encountering missiles). Fig. 9c shows the random maneuver 3 of the UAV formation during the learning process (with a sharp decrease in external reward, the UAVs immediately leave the air-defense network). Fig. 9d shows the first method for the trained UAV formation to fly to the destination (one UAV flies straight to the target, and the other three dodge the missiles). Fig. 9e shows the second method for the trained UAV formation to fly to the destination (all UAVs dodge the missiles). Fig. 9f shows the third method for the trained UAV formation to fly to the destination (one UAV flies to the target in a curved path, and the other three fly straight to dodge the missiles). Figs. 9g-9i show the best control strategies 1 (RLIC), 2 (RLIC), and 3 (RLIC) learned by the UAV formation to reach the destination respectively. As can be seen from Fig. 9, similar to the behaviour in a single-UAV environment, each UAV in the UAV formation initially flies randomly and then gradually starts to move towards the target point.

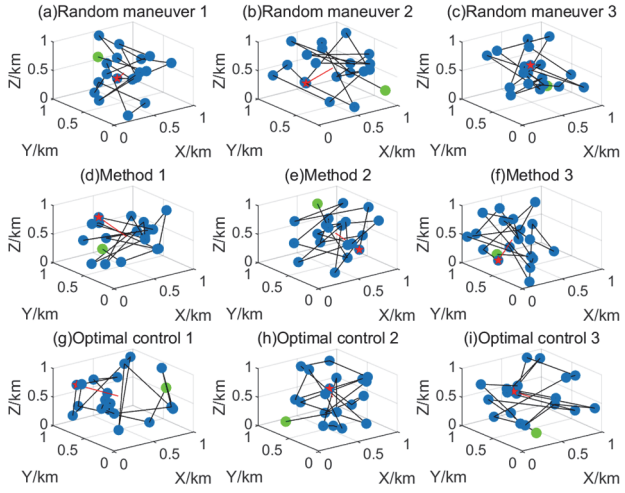

Figure 9 Three-dimensional visualization of the learning process of UAV formation in a 3D environment

The convergence processes of the traditional Q-learning algorithm and the improved Q-learning algorithm in a simple environment are shown in Fig. 10.

As can be seen from Fig. 10, first of all, during the iteration process of the traditional Q-learning algorithm in searching for the shortest path, the curve has a relatively large fluctuation amplitude in the initial learning stage.

This indicates that the algorithm has a low ability to utilize new knowledge and requires multiple learning attempts to converge. Secondly, due to the lack of a suitable reward function, the action selection is rather blind, which prolongs the exploration time. The improved RL algorithm can quickly converge and find the shortest path in the initial learning stage because it has obtained prior knowledge. Thirdly, thanks to the continuous reward function, the robot can immediately judge the quality of action states and thus select the optimal action. Therefore, the path length has relatively small fluctuations in several learning attempts before convergence, and the convergence stability is relatively high.

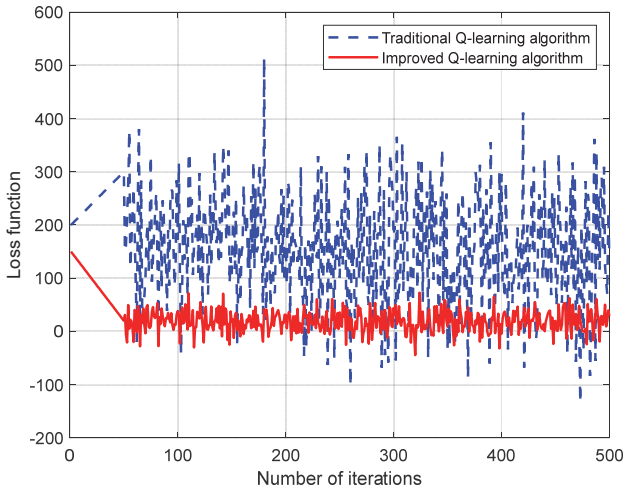

Figure 10 Convergence processes of the two algorithms in a simple environment

6 CONCLUSION

This paper considers the stationarity of minimum required overload, minimum flight time and required overload, and models the trajectory planning problem as partially observable Markov decision process and Markov decision process. In addition, exponential weighted average is introduced to design action filters to enhance the temporal correlation of RL agents' actions. This method guarantees the smoothness of the desired trajectory overload with little additional computational cost. In the experiment, MDP method shows the advantage of using global information for planning. The proposed integral compensation algorithm not only effectively avoids local optimality in trajectory planning, but also achieves satisfactory success rate. It addresses the shortcomings of previous RL-based trajectory planning methods, such as the tendency to fall into local optimality when planning fails and the lack of alternative solutions. Finally, the RL-IC method in this paper exhibits satisfactory generalization performance in various special environments. When applied to the autonomous control of unmanned aerial vehicles (UAVs), ordinary RL algorithms often result in static position errors. To tackle this issue, this paper introduces the concept of integral compensation and proposes the RL-IC algorithm. This algorithm can effectively resolve the problem of static position-tracking errors without increasing algorithmic complexity, thereby further expanding the application scope of RL algorithms in UAVs. Simulation results indicate that, when the model of the controlled object is unknown, the UAV trajectory planning controller based on the RL-IC algorithm can autonomously learn to stably control the UAV's trajectory to the desired position. Compared with ordinary RL algorithms, the position control error of the UAV trajectory planning controller based on the RL-IC algorithm is reduced to zero, and the stability and robustness of the control system are enhanced. The addition of integral compensation effectively eliminates static position errors, improves the stability of the algorithm, and reduces the requirements of the learning algorithm for the design of the reward function. Moreover, the introduction of the attenuation factor balances the proportion of the cumulative error and the current state error, enabling the algorithm to reduce the adjustment time while maintaining performance, thus enhancing the overall performance of the algorithm.

7 REFERENCES

[1] Sun, T., Sun, W., & Sun, C. (2024). Multi-UAV formation path planning based on compensation look-ahead algorithm. *Drones*, *8*(6), 251-276. https://doi.org/10.3390/drones8060251

[2] Bai, K., Wu, J., &Wu, H. (2024). High-precision time synchronization algorithm for unmanned aerial vehicle ad hoc networks based on bidirectional pseudo-range measurements. *Ad Hoc Networks*, *152*, 1.1-1.14. https://doi.org/10.1016/j.adhoc.2023.103326

[3] Chen, S., Xin, M., & Kong, R. S. (2024). Error compensation method of GNSS/INS integrated navigation system based on AT-LSTM during GNSS outages. *IEEE Sensors Journal*, *24*(12), 20188-20199. https://doi.org/10.1109/JSEN.2024.3395009

[4] Zhang, H., Wang, Y., & Shan, S. (2024). An enhanced GNSS/INS navigation compensation method using LSTM-FPN for bridging GNSS outages. *Measurement Science and Technology*, *36*(1), 16339-16352. https://doi.org/10.1088/1361-6501/ad9cab

[5] Cheng, J., Gao, Y., &Wang, H. (2025). Vision-assisted GNSS/INS high precision position in method based on adaptive maximum correntropy criterion in urban traffic environment. *Measurement*, *245*, 667-684. https://doi.org/10.1016/j.measurement.2025.116667

[6] Zhao, H., Liu, F., &Chen, W. (2024). A method for assisting GNSS/INS integrated navigation system during GNSS outage based on CNN-GRU and factor graph. *Applied Sciences*, *14*(18), Article 8131. https://doi.org/10.3390/app14188131

[7] Tang, Y., Jiang, J., &Liu, J. (2022). A GRU and AKF-based hybrid algorithm for improving INS/GNSS navigation accuracy during GNSS outage. *Remote Sensing*, *14*, 752-773. https://doi.org/10.3390/rs14030752

[8] Chen, L., Liu, Z., &Fang, J. (2022). A novel hybrid observation prediction methodology for bridging GNSS outages in INS/GNSS systems. *Journal of Navigation*, *75*(5), 1206-1225. https://doi.org/10.1017/S037346332200025X

[9] Dian, S. (2024). Charging optimization with an improved dynamic programming for electro-gasoline hybrid powered compound-wing unmanned aerial vehicle. *Energies*, *18*(1), 1-18. https://doi.org/10.3390/en18010030

[10] Cao, Y., Long, T., & Sun, J. (2023). Distributed task allocation algorithm for multiple unmanned aerial vehicle based on information retransmission and package loss compensation. *Acta Armamentario*, *44*(9), 2697-2708. https://doi.org/10.12382/bgxb.2022.1180

[11] Zhang, Z., Jiang, J., & Xu, H. (2024). Distributed dynamic task allocation for unmanned aerial vehicle swarm systems:

A networked evolutionary game-theoretic approach. *Chinese Journal of Aeronautics*, *37*(6), 182-204. https://doi.org/10.1016/j.cja.2023.12.027

[12] Yan, M., Yuan, H., & Xu, J. (2021). Task allocation and route planning of multiple UAV sin a marine environment based on an improved particle swarm optimization algorithm. *EURASIP Journal on Advances in Signal Processing*, *2021*(1), 94-103. https://doi.org/10.1186/s13634-021-00804-9

[13] Han, D., Jiang, H., & Wang, Y. Y. Q. (2024). Collaborative task allocation and optimization solution for unmanned aerial vehicles in search and rescue. *Drones*, *8*(4), 138-153. https://doi.org/10.3390/drones8040138

[14] Li, J., Jia, Z., & Liu, T. (2022). A landing trajectory tracking controller for fixed-wing UAV based on iterative learning control. *Proceedings of IOP Publishing*, *2*, 444-452. https://doi.org/10.1007/978-981-16-7423-5_44

[15] Do, H. T., Nicolau, F., & Prodan, S. I. (2022). Tracking control for a flat system under disturbances: A fixed-wing UAV example. *IFAC-Papers on Line*, *55*(16), 406-411. https://doi.org/10.1007/s10846-024-02099-y

[16] Zhao, D., Xia, L., &Dang, H. (2022). Design and control fair supply system for PEMFCUAV based on dynamic decoupling strategy. *Energy Conversion and Management*, *253*, Article 115159. https://doi.org/10.1016/j.enconman.2021.115159

[17] Sun, H., Sun, Q., &Wu, W. (2020). Altitude control for flexible wing unmanned aerial vehicle based on active disturbance rejection control and feedforward compensation. *International Journal of Robust and Nonlinear Control*, *30*(6), 4758-4773. https://doi.org/10.1002/rnc.4758

[18] Soares, C. G. (2024). Unmanned surface vessel-unmanned aerial vehicle cooperative path following based on a predictive line of sight guidance law. *Journal of Marine Science and Engineering*, *12*(10), 1818-1834. https://doi.org/10.3390/jmse12101818

[19] Zhang, X., Li, H., &Zhu, G. (2024). Finite-time adaptive quantized control for quadrotor aerial vehicle with full states constraints and validation on Q Drone experimental platform. *Drones*, *8*(6), 264-287. https://doi.org/10.3390/drones8060264

[20] Nemra, A. (2024). UAV trajectory tracking using proportional-integral-derivative-type-2 fuzzy logic controller with genetic algorithm parameter tuning. *Sensors*, *24*(20), 6678-6697. https://doi.org/10.3390/s24206678

[21] Cao, P., Lu, Y., & Lu, C. (2022). Light-weight unmanned aerial vehicle wireless power transfer system based on hollow copper coated aluminum tubes. *Progress in Electromagnetics Research Letters*, *107*, 1603-1622. https://doi.org/10.2528/pierl22081603

[22] Jiao, R., Chou, W., & Rong, Y. (2022). Anti-disturbance attitude control for quadrotor unmanned aerial vehicle manipulator via fuzzy adaptive sigmoid generalized super-twisting sliding mode observer. *Journal of Vibration and Control*, *28*(11-12), 1251-1266. https://doi.org/10.1177/1077546321989495

[23] Imanberdiyev, N., Sood, S., & Kircali, D. (2022). Design, development and experimental validation of a lightweight dual-arm aerial manipulator with a COG balancing mechanism. *Mechatronics*, *82*, 102719-102742. https://doi.org/10.1016/j.mechatronics.2021.102719

[24] Zhang, B., Sun, X., & Liu, S. (2020). Tracking control of multiple unmanned aerial vehicles incorporating disturbance observer and model predictive approach. *Transactions of the Institute of Measurement and Control*, *42*(5), 951-964. https://doi.org/10.1177/0142331219879858

[25] Wu, H., Pei, X., & Li, J. (2020). An improved magnetometer calibration and compensation method based on Levenberg-Marquardt algorithm for multi-rotor unmanned aerial vehicle. *Measurement and Control*, *53*(3-4), 276-286. https://doi.org/10.1177/0020294019890627

[26] Wang, B., Shen, Y., & Li, Y. G. Z. (2023). An adaptive sliding mode fault-tolerant control of a quadrotor unmanned aerial vehicle with actuator faults and model uncertainties. *International Journal of Robust and Nonlinear Control*, *33*(17), 10182-10198. https://doi.org/10.1002/rnc.6631

[27] Maire, P. L., Bertrand, L., & Munschy, M. (2020). Aerial magnetic mapping with an unmanned aerial vehicle and a fluxgate magnetometer: A new method for rapid mapping and upscaling from the field to regional scale. *Geophysical Prospecting*, *68*(7), 2307-2319. https://doi.org/10.1111/1365-2478.12991

[28] Li, B., Ban, H., & Gong, W. (2020). Extended state observer-based finite-time dynamic surface control for trajectory tracking of a quadrotor unmanned aerial vehicle. *Transactions of the Institute of Measurement and Control*, *42*(15), 2956-2974. https://doi.org/10.1177/0142331220935710

Contact information:

Xianlong MA
School of Aerospace, Northwestern Polytechnical University,
Xi'an, Shaanxi province 710072, China
E-mail: maxianlong@nwpu.edu.cn