Hindawi
International Journal of Aerospace Engineering
Volume 2023, Article ID 6654130, 15 pages
https://doi.org/10.1155/2023/6654130

Research Article

Representation Enhancement-Based Proximal Policy Optimization for UAV Path Planning and Obstacle Avoidance

Xiangxiang Huang,[1] **Wei Wang**,[1] **Zhaokang Ji**,[1] **and Bin Cheng**[2]

[1]*School of Computer and Information, Anhui Polytechnic University, Wuhu, China*
[2]*Chery Automobile, Wuhu, China*

Correspondence should be addressed to Wei Wang; familywei@mail.ahpu.edu.cn

Received 22 June 2023; Revised 21 September 2023; Accepted 9 October 2023; Published 8 November 2023

Academic Editor: Ke Feng

Path planning and obstacle avoidance are pivotal for intelligent unmanned aerial vehicle (UAV) systems in various domains, such as postdisaster rescue, target detection, and wildlife conservation. Currently, reinforcement learning (RL) has become increasingly popular in UAV decision-making. However, the RL approaches confront the challenges of partial observation and large state space when searching for random targets through continuous actions. This paper proposes a representation enhancement-based proximal policy optimization (RE-PPO) framework to address these issues. The representation enhancement (RE) module consists of observation memory improvement (OMI) and dynamic relative position-attitude reshaping (DRPAR). OMI reduces collision under partially observable conditions by separately extracting perception features and state features through an embedding network and feeding the extracted features to a gated recurrent unit (GRU) to enhance observation memory. DRPAR compresses the state space when modeling continuous actions by transforming movement trajectories of different episodes from an absolute coordinate system into different local coordinate systems to utilize similarity. In addition, three stepwise reward functions are formulated to avoid sparsity and facilitate model convergence. We evaluate the proposed method in three 3D scenarios to demonstrate its effectiveness. Compared to other methods, our method achieves a faster convergence during training and demonstrates a higher success rate and a lower rate of timeout and collision during inference. Our method can significantly enhance the autonomy and intelligence of UAV systems under partially observable conditions and provide a reasonable solution for UAV decision-making under uncertainties.

1. Introduction

Unmanned aerial vehicles (UAVs) have gained widespread popularity in various fields, such as aerial photography, plant protection, and military surveillance, due to high agility, low cost, and versatility [1–3]. In some emerging areas such as digital twin and intelligent manufacturing, UAVs can play an important role in degradation assessment [4], fault diagnosis [5], and health management [6]. Therefore, the ability for path planning and obstacle avoidance (PPOA) is paramount for intelligent UAVs [7]. Researchers have devoted significant effort to developing decision-making methods in recent years, including traditional mathematical approaches and machine learning approaches. In this paper, we focus on the machine learning approach, especially deep reinforcement learning (DRL), to enable UAVs to perform PPOA in complex and dynamic environments with random targets and continuous actions.

Traditional mathematical approaches, such as the Dijkstra algorithm [8], the A-star algorithm [9], and the particle swarm optimization (PSO) [10], require precise modeling of environments [11] and substantial prior knowledge [12] to solve path-planning problems. For instance, Fadzli et al. [8] improved the Dijkstra algorithm by introducing a junction degree of difficulty function to generate the shortest path indoors. Cai et al. [9] used the A-star algorithm to control UAVs to track known targets. H. Chen and P. Chen [10] combined the divide-and-conquer strategy with the A-star

algorithm into PSO to generate paths by dividing an entire path into segments. In the above methods, starting points and endpoints are predefined, and initial information, such as floor plans, terrains, and danger zones, is known. Thus, these methods are not suitable for the problem addressed in this paper, which is searching for random targets from random starting points under partially observable conditions.

In contrast to mathematical approaches, machine learning approaches, particularly reinforcement learning (RL), have advantages in creating intelligent UAVs. RL addresses the PPOA problem by maximizing rewards during an agent's interaction with environments. For instance, Hung and Givigi [13] proposed a Q-learning approach to coordinate a group of UAVs to fly together in a 2D scene, where the UAVs have discrete actions, constant altitude, and velocity. Yijing et al. [14] designed an adaptive and random exploration (ARE) framework consisting of an action module, a learning module, and a trap-escape module to adjust UAVs' paths, but the action space remained discrete. Similarly, Yan and Xiang [15] utilized the Euclidean distance to a target as the initial value of the q-function and integrated the ϵ-greedy algorithm with the Boltzmann strategy to select a discrete action in 2D space. Therefore, there is an urgent need to overcome the constraint of discrete actions in tabular scenarios and equip UAVs with continuous actions to perform complex tasks in 3D space.

To address the above challenges, an increasing number of researchers have turned to deep learning-based methods [16], especially deep reinforcement learning (DRL), to overcome the limitations of table-based RL methods [17]. DRL has achieved significant breakthroughs in various domains, including video games [18], power systems [19], financial trading [20], and automated assembly systems [21]. By using neural networks to approximate value functions, DRL can effectively handle complex path-planning tasks. For instance, Raja et al. [22] utilized deep Q-learning to optimize the flight parameters of roll, pitch, and yaw for a group of UAVs while minimizing the individual distance traveled by each UAV. Li et al. [23] employed the double deep q-network (double-DQN) to address the coverage path-planning problem by balancing exploitation and exploration through the ϵ-greedy policy. Roghair et al. [24] extended the dueling double-DQN (D3QN) to enhance exploration for obstacle avoidance in 3D environments. Despite the effectiveness of these methods in dealing with complex tasks such as swarm coordinated flight, area traversal coverage, and 3D obstacle avoidance, they still fall short in modeling continuous actions. To this end, Xu et al. [25] proposed a continuous model for the action space with a multiple experience pool and gradient truncation to improve convergence. Qi et al. [26] applied frequency decomposition (FD) during the proximal policy optimization (PPO) [27], which decomposed rewards into multidimensional frequencies and calculated the returns as the guidance of path-planning. Zhang et al. [28] combined the two-stream actor-critic network structure with the twin-delayed deep deterministic (TD3) policy to extract environmental features and achieve continuous controls. However, these methods require complete observations of environments, and their perceptions are limited under partially observable conditions. An intuitive and straightforward way to improve perceptual ability is the state-stacking approach [29], in which a sequence of states is concatenated to improve representation. However, this technique tends to expand the state space and increase training difficulty. Singla et al. [30] proposed a direct approach to environmental perception by equipping UAVs with monocular cameras to extract depth maps from RGB images. Similarly, Mansouri et al. [31] corrected the heading of UAVs toward the center of a mine tunnel by using a 2D LiDAR sensor. Notably, the above approaches require additional equipment to sense environments.

Improving the perceptual ability in a complex environment is crucial, but it is equally important to consider generalization ability and learning efficiency. However, the existing research [22, 23, 30, 32] on PPOA relied on fixed or prespecified targets, making it unsuitable for navigating to random locations. Furthermore, allowing UAVs to search for random targets confronts challenges such as large state space and low learning efficiency.

Based on the above literature review, we can draw the following research gaps. Firstly, most of the existing methods search for fixed targets with discrete actions, which limits the practicality and scalability of UAVs in complex and dynamic environments. Secondly, most of the existing methods assume complete observations of environments, which is unrealistic and impractical in the real world where UAVs often face partially observable conditions. Thirdly, most of the existing methods do not consider state space compression and observation memory enhancement, which is essential for improving learning efficiency and reducing collision rate. To solve the above problems, we propose a representation enhancement-based proximal policy optimization (RE-PPO) framework for autonomous navigation in obstacle-rich environments with random targets and continuous actions. The main contributions are as follows.

(i) We devise a representation enhancement (RE) module comprising two components: observation memory improvement (OMI) and dynamic relative position-attitude reshaping (DRPAR). OMI improves the perceptual ability and reduces the collision rate under partially observable conditions by separately extracting perception and state features through an embedding network and feeding the extracted features to a gated recurrent unit (GRU) to enhance the observation memory. DRPAR compresses the state space and improves the learning efficiency by transforming the movement trajectories from an absolute coordinate system to several local coordinate systems, which can capture the similarity among different episodes

(ii) We design three step-wise reward functions that avoid sparsity and facilitate model convergence by providing intermediate rewards based on collision, activation, and navigation. We also apply the PPO

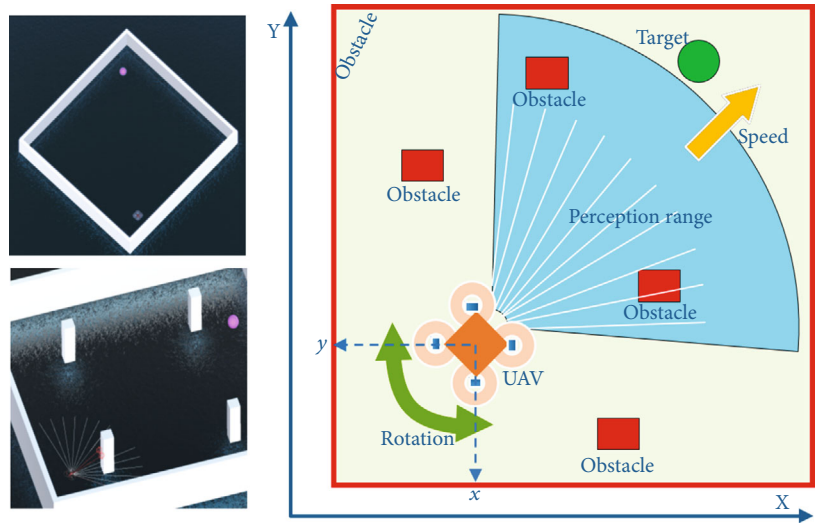


Figure 1: Schematic diagram of PPOA.

algorithm to learn an optimal policy for continuous actions, which enhances the practicality and scalability of our framework

(iii) We conduct extensive experiments in three 3D scenarios to evaluate the performance of our method. We compare our method with several baseline methods and demonstrate that our method achieves a faster convergence, a higher success rate, and a lower timeout and collision rate

The remainder of this paper is organized as follows. Section 2 describes the preliminaries of the proposed method. Section 3 presents the details. Section 4 shows the experimental results. Section 5 discusses the improvements and limitations. Finally, Section 6 presents conclusions.

2. Preliminary Work

2.1. Problem Formulation. The PPOA problem concerned in this study can be illustrated in Figure 1. In Figure 1, the right part is the 2D projection of the 3D scenario on the left. There are two kinds of obstacles in the 3D scenario, cuboid pillars and surrounding walls. In the 2D projection, the red rectangles represent the pillars whose number and location are uncertain, the four red sides represent the walls, the green circle denotes a random target, and the blue area indicates the perception range of the ray sensors through which the UAV can receive partial environment information. PPOA is aimed at making real-time decisions through incomplete sensed information to avoid obstacles and navigate to the target from a random starting point with continuous actions. This problem is challenging and practical, as it involves uncertainties and partial observations.

To address this problem, we propose a novel representation enhancement-based proximal policy optimization (RE-PPO) framework. Specifically, we formulate PPOA in an obstacle-rich area as a partially observable Markov decision process (POMDP). The observation vectors of POMDP consist of the state of the UAV and the incomplete sensed information from the ray sensors. The state of the UAV includes position, speed, and rotation, where a pair of speed and rotation forms an action. To achieve continuous actions, we apply PPO to model actions. We elaborate on the theories and definitions involved in our framework in the subsequent sections of this chapter.

2.2. POMDP. POMDP provides a principled mathematical framework for modeling and solving decision and control tasks under uncertainties [33]. POMDP contains the following components, S, A, T, R, O, Ω, and γ, where S represents a set of environmental states, A is a set of actions, T refers to a set of conditional transition probabilities between states, R is the reward function, O refers to a set of partial observations sensed by UAVs, Ω represents a set of conditional observation probabilities, and $\gamma \in [0, 1]$ is the discount factor.

For a given time t, the system is in a state $s_t \in S$, and the UAV captures an observation $o_t \in O$ and takes an action $a_t \in A$. A reward r_t is returned according to s_t and a_t. The taken action a_t causes the state s_t to transit to a new state s_{t+1} with a probability of $T(s_{t+1}|s_t, a_t)$, and the UAV will receive an observation o_{t+1} with a probability of $\Omega(o_{t+1}|s_{t+1}, a_t)$. The above process is repeated until an episode is over. The optimization goal for this process is to generate an action at each time that maximizes the total expected reward $R = \sum_{t=0}^{\infty} r_t \gamma$, where γ determines the weighting between immediate rewards and future rewards. When $\gamma = 0$, the UAV only cares about actions yielding the largest immediate rewards, and when $\gamma = 1$, the UAV focuses on maximizing the future rewards. Figure 2 shows the whole interaction process of POMDP.

2.3. Observation Space Definition. In this paper, the observation space consists of the sensed and the state information. We use the ray sensors to collect the sensed

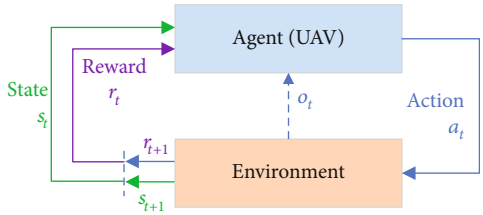


Figure 2: Interaction process between the UAV and the environment.

information as shown in Figure 1. The UAV sends 15 rays from different directions as illustrated in the following equation:

$$\mathrm{RAY} = [\mathrm{ray}_1, \mathrm{ray}_2, \mathrm{ray}_3, \cdots, \mathrm{ray}_n]^T. \tag{1}$$

In (1), each ray_i has a perceptual distance of 13 meters and returns the label type and distance in the corresponding direction as illustrated in the following equation:

$$\mathrm{ray}_i = [l_v, l_o, l_t, d], \tag{2}$$

where l_v refers to a void label without obstacles and targets in the corresponding direction, l_o means a ray detects an obstacle, l_t means a ray detects a target, and d represents the returned distance to the obstacles or the target. For void labels, the returned distance is set to zero. We use one-hot encoding to organize the sensed information for the three types of labels. Specifically, for each detected label, we represent the sensed information as a four-dimensional vector (1, 0, 0, 0) for voids, (0, 1, 0, d) for obstacles, and (0, 0, 1, d) for targets.

The position, speed, and rotation constitute the state information as illustrated in the following equation:

$$s_t^{\mathrm{uav}} = \begin{bmatrix} x_t \\ y_t \\ z_t \\ v_t \\ \alpha_t \\ \beta_t \\ \theta_t \end{bmatrix}, \tag{3}$$

where x, y, z is the UAV's real-time position, v is the real-time speed, and α, β, θ is the real-time rotation representing pitch, roll, and yaw, respectively. In practice, we fix the flight altitude z to simplify the problem complexity and obtain the following equation:

$$s_t^{\mathrm{uav}} = \begin{bmatrix} x_t \\ y_t \\ v_t \\ \theta_t \end{bmatrix}. \tag{4}$$

s_t^{uav} is updated between time intervals through the following equation:

$$\begin{bmatrix} x_{t+1} \\ y_{t+1} \\ v_{t+1} \\ \theta_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + v_t \cos\theta_t \\ y_t + v_t \sin\theta_t \\ \eta_t * C \\ \rho_t * \dfrac{\pi}{3} \end{bmatrix}, \tag{5}$$

where v_t and θ_t are the two components of an action at time t and $\eta_t \in [0, 1]$ and $\rho_t \in [-1, 1]$ are the two control parameters for v_t and θ_t, respectively. $\eta_t = 0$ means the UAV is hovering, and $\eta_t = 1$ means the UAV travels with the max speed $C = 2.8$ m/s. $\rho_t = -1$ means the UAV rotates 60 degrees to the left, and $\rho_t = 1$ means the UAV rotates 60 degrees to the right. Combining (1) with (4), the observation o_t at time t can be derived as shown in the following equation:

$$o_t = [\mathrm{ray}_1, \mathrm{ray}_2, \mathrm{ray}_3, \cdots, \mathrm{ray}_n, x, y, v, \theta]^T. \tag{6}$$

2.4. Proximal Policy Optimization. DRL can be separated into the value function-based and policy-based categories based on the way to maximize cumulative rewards. The value function-based methods cannot model continuous actions. Therefore, we choose the policy-based methods as our solution.

The policy-based methods are aimed at learning an agent's policy π. During interaction with the environment, the received reward can be written as the following equation:

$$R_\theta = \sum_\tau R(\tau)\pi_\theta(\tau), \tag{7}$$

where τ represents the trajectory generated in each episode. $R(\tau)$ is the cumulative reward. π_θ represents the policy adopted by the neural network parameter θ. In order to enhance the decision-making ability, the gradient ascent algorithm is used to optimize the policy as shown in the following equation:

$$\nabla R_\theta = \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n} R(\tau^n)\nabla \log \pi_\theta(a_t^n | s_t^n), \tag{8}$$

where T_n is the step number in an episode and N is the episode number. However, this approach requires a large number of episodes and suffers from slow learning.

The PPO algorithm uses an actor-critic architecture to accelerate policy optimization. The critic network $V_\phi(s_t)$ is used to evaluate the state s_t at time t as shown in the following equation:

$$V_\phi(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + .. + \gamma^n V_\phi(s_{t+n}). \tag{9}$$

In the context of this study, the UAV cannot observe the complete environmental states. Therefore, we use o_t as s_t.

Then, the loss function of the critic network is described as (10) in which the historical data are integrated with the gradient descent algorithm to improve evaluation accuracy.

$$\text{Loss}(\phi) = -\Sigma_{t=1}^{T}\left(\Sigma_{t'>t}\gamma^{t'-t}r_{t'} - V_{\phi}(s_t)\right)^2. \tag{10}$$

The actor network introduces the advantage equation $\widehat{A}_t$ into the objective function to improve training efficiency. As shown in (11), $\widehat{A}_t$ represents the advantage of the action relative to the expectation of the state s_t.

$$\widehat{A}_t = \Sigma_{t'>t}\gamma^{t'-t}r_{t'} - V_{\phi}(s_t). \tag{11}$$

Additionally, the actor network introduces the importance sampling, which improves the utilization of historical experiences and accelerates training speed. As shown in (12), the importance sampling $r_t(\theta)$ is a technique for computing the importance weighting between a sampling distribution and a target distribution, which calculates the probability ratio of the experience under the current policy and the old policy.

$$r_t(\theta) = \frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}. \tag{12}$$

By combining (11) with (12), we obtain the objective function of the actor network, as shown in the following equation:

$$J_{\text{PPO}}^{\text{clip}}(\theta) = \Sigma_{t=1}^{T}\ \min\left(r_t(\theta)\widehat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\widehat{A}_t\right), \tag{13}$$

where ϵ is the clip parameter and clip truncates the value of $r_t(\theta)$ within the range of $[1-\epsilon, 1+\epsilon]$ to avoid large gradient volatility and ensure training stability.

3. Proposed Method

In this section, we describe the details of our RE-PPO framework. The overall framework of RE-PPO is shown in Figure 3. The OMI module employs an embedding network to process the state information and the sensed information of o_t. Then, it enhances the observation memory through a GRU network to improve perception and reduce collisions under partially observable conditions. The DRPAR module reshapes the state s_t^{uav} of different episodes through a coordinate transformation, and the similarity of the reshaped states from different episodes can be used to compress the state space and improve training efficiency. The outputs s_t^{reshape} of RE will be passed to PPO to model continuous actions. During training, we formulate three step-wise reward functions to guide policy optimization.

3.1. Observation Memory Improvement. In PPOA tasks, UAVs are unable to observe the complete environmental information. Improving observation memory for environmental exploration can reduce collisions and improve search

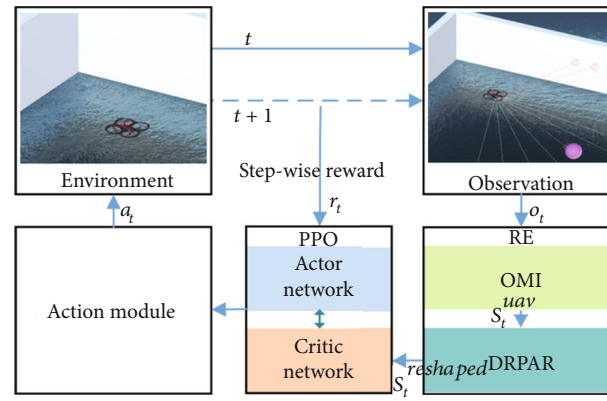


Figure 3: RE-PPO framework.

efficiency. Previous work [34] viewed the components of o_t as a whole and passed the whole to a deep neural network to extract features. However, direct processing for observations slows down training speed.

Instead, we process the state and the sensed information separately. As shown in (14), the state and the ray components are plugged into the embedding networks e to extract features. After the processing of the feedforward neural network Linear, we concatenate the two neuronal representations as n_t.

$$n_t = [\text{Linear}(e(s_t^{\text{uav}})), \text{Linear}(e(\text{RAY}))]. \tag{14}$$

The concatenated n_t is fed into a GRU network for memory enhancement. The GRU network uses a reset gate unit and an update gate unit to process the sequence data. The reset gate unit combines the current observation n_t with the previous memory information while discarding the candidate hidden state h_{t-1} to achieve oblivion. Equation (15) shows the process through the reset gate unit, where Linear is the linear transformation network, and σ is the sigmoid activation function, which constrains the results within the range of (-1, 1).

$$r_t = \sigma(\text{Linear}(n_t, h_{t-1})). \tag{15}$$

Equation (16) shows the process through the update gate unit. The update gate unit regulates the updating of candidate hidden states with the current input n_t and the previous hidden state h_{t-1}.

$$z_t = \sigma(\text{Linear}(n_t, h_{t-1})). \tag{16}$$

The candidate hidden state h_t' for the current time step is obtained by integrating r_t, h_{t-1}, and n_t, as shown in (17), where $\odot$ denotes the element-wise product, and the activation function tanh constrains the output of h_t' within the range of $(-1, 1)$. And the final hidden state h_t is given in (18).

$$h_t' = \tanh\left(\text{Linear}(n_t, h_{t-1} \odot r_t)\right), \tag{17}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h_t'. \tag{18}$$

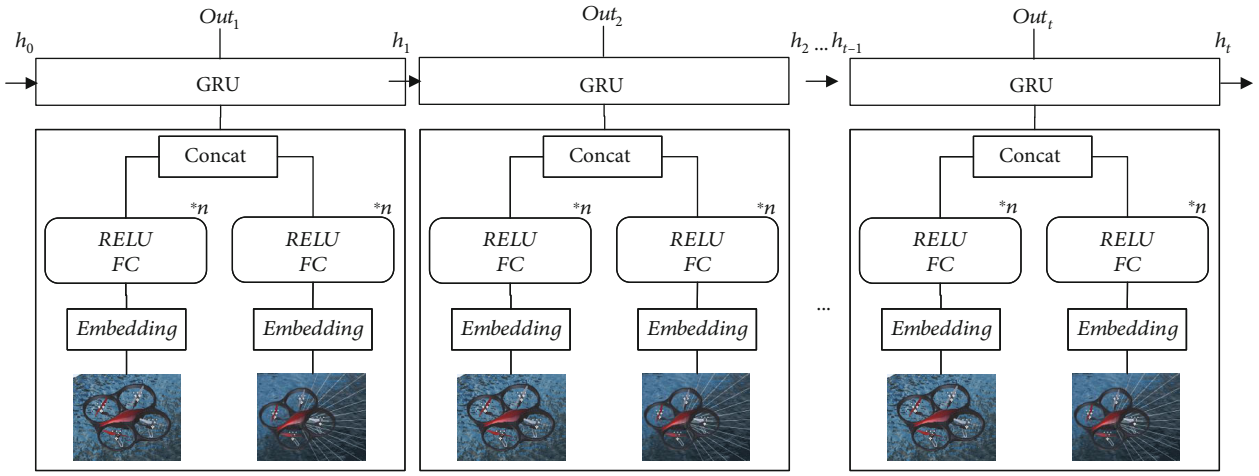


Figure 4: Whole process of OMI.

Through the processing of GRU, the decision trajectory τ consists of $\{h_0, a_0, r_0, h_1, a_1, r_1, h_2, \cdots\}$, where a_t denotes the action and r_t is the instantaneous reward. The whole process of OMI is shown in Figure 4.

3.2. Dynamic Relative Position and Attitude Reshaping. Similar trajectories exist among different episodes in PPOA tasks involving searching for random targets, as shown in Figure 5. Previous work [35] ignores the underlying relationships between episodes, resulting in a large state space. To compress the state space and facilitate policy convergence, we propose the DRPAR strategy to extract similar intrinsic features.

After specifying the UAV's state and the target during the initialization phase of each episode, instead of recording the real-time state, we just record the dynamic relative differences between the initial state and the real-time state, as shown in (19) and (20), respectively.

$$\Delta \mathrm{POS}_t = (\Delta x_t, \Delta y_t) = (x_t - x_0, y_t - y_0), \tag{19}$$

$$\Delta\theta_t = \theta_t - \theta_0. \tag{20}$$

In (19) and (20), $\Delta \mathrm{POS}_t$ represents the dynamic relative difference between the real-time position (x_t, y_t) and the starting point (x_0, y_0), and $\Delta\theta_t$ is the dynamic relative difference between the real-time rotation θ_t and the initial rotation θ_0. DRPAR transforms the trajectories of different episodes in an absolute coordinate system into several local coordinate systems. The trajectory similarity of different episodes can be extracted and utilized in the local coordinate systems. The state space is compressed by the similarity of the reshaped positions and attitudes; thus, the convergence speed is improved. After DRPAR, we use $\Delta \mathrm{POS}_t$ and $\Delta\theta_t$ to replace the corresponding components of s_t^{uav} and combine the replaced result with the sensed information to formulate the reshaped state s_t^{reshape}. The critic network of PPO takes s_t^{reshape} as an input to execute an evaluation, as shown in the following equation:

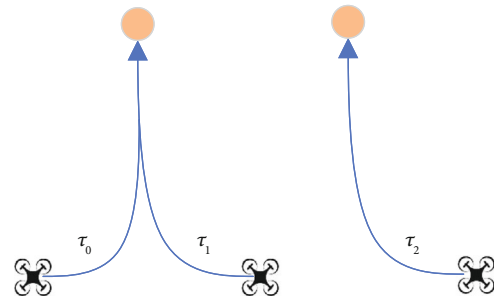


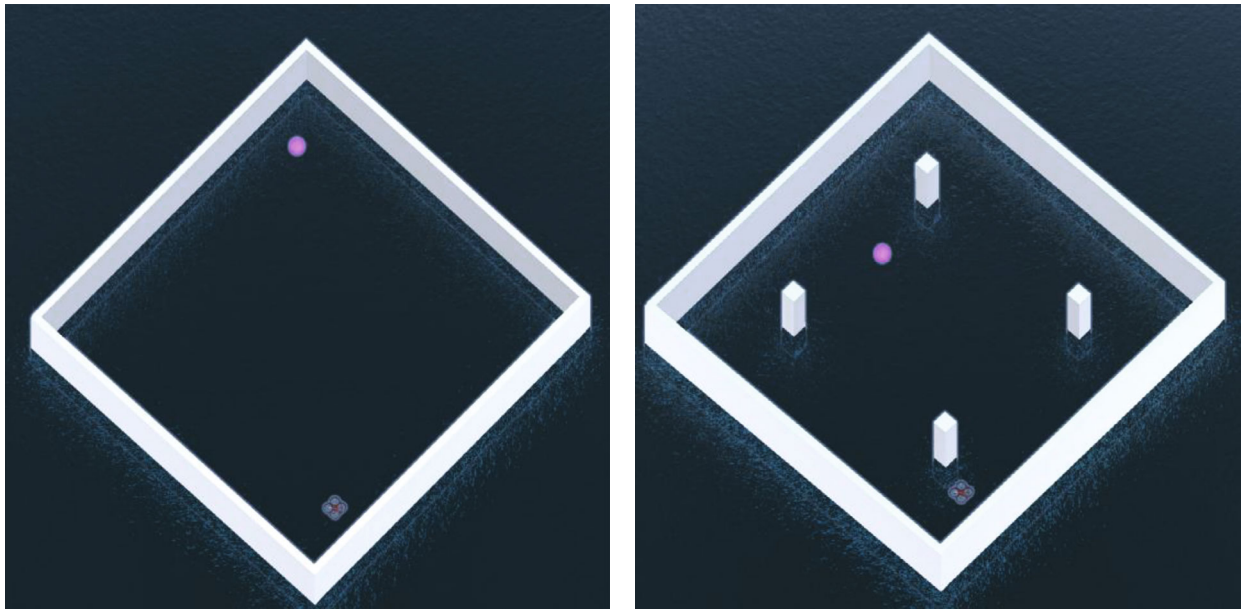
Figure 5: Different trajectories have similar intrinsic features. Trajectory τ_0 and trajectory τ_1 share the same segment. Trajectory τ_1 and trajectory τ_2 also have similar action sequences.

$$V_\phi\left(s_t^{\mathrm{reshape}}\right) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + .. + \gamma^n V_\phi\left(s_{t+n}^{\mathrm{reshape}}\right). \tag{21}$$

3.3. Reward Function Design. The design of reward functions is a crucial issue in DRL. We design three types of step-wise rewards: an obstacle avoidance reward, a per-step reward, and a navigation reward, to avoid the sparsity of episode-wise rewards and to facilitate model convergence.

To encourage the UAV to avoid obstacles during navigation, we design the obstacle avoidance reward based on the distance between the UAV and the obstacles, as illustrated in (22) where $\min(d_1, \cdots, d_n)$ denotes the closest distance from the UAV to the obstacles, and L is the specified threshold indicating the safe distance. When $\min(d_1, \cdots, d_n)$ is lower than L, a negative reward is returned. When $\min(d_1, \cdots, d_n)$ tends to zero, a collision occurs, and the maximum negative reward $-\lambda_1 L$ is returned. And when $\min(d_1, \cdots, d_n)$ is greater than or equal to L, the returned reward is zero.

$$r_{\mathrm{avoidance}} = \lambda_1 \min\left(\min\left(d_1, \cdots, d_n\right) - L, 0\right). \tag{22}$$



(a) Scenario A (b) Scenario B

Figure 6: Two 3D scenarios designed in Unity3D.

Furthermore, to encourage the UAV to move more actively, we design the per-step reward as shown in (23). The purpose of the per-step reward contains two aspects: to reduce the number of steps in one episode and to prevent the UAV from getting stuck in stagnation due to obstacle avoidance.

$$r_{\text{step}} = -\lambda_2. \tag{23}$$

PPOA is aimed at navigating to the target as efficiently as possible by providing a significant positive reward when the UAV reaches the target. During navigation, once the ray sensors detect the target, the UAV moves closer until the distance d between the UAV and the target is less than the specific value $d_{\min}$ in (24). At this point, λ_3 is awarded as the reward; otherwise, the reward is zero.

$$r_{\text{navigation}} = \begin{cases} 0, & \text{otherwise}, \\ \lambda_3, & d \le d_{\min}. \end{cases} \tag{24}$$

In summary, the reward r received by the UAV in each step is the combination of $r_{\text{avoidance}}$, r_{step}, and $r_{\text{navigation}}$, as shown in the following equation:

$$r = r_{\text{avoidance}} + r_{\text{step}} + r_{\text{navigation}}. \tag{25}$$

4. Experiments

4.1. Experimental Scene Design. We construct two virtual scenarios in Unity3D, a powerful cross-platform 3D engine, to verify the performance of RE-PPO. As shown in Figure 6, the UAV and the target (the pink ball) are randomly generated in the two scenarios. And their heights are fixed with the same value. In Scenario A, the white walls are the obstacles that delimit the search range. In Scenario B, in addition to the white walls, there are cuboid pillars placed in uncertain locations as another type of obstacle. The cuboid pillars have different shapes and sizes from the white walls, which adds to the complexity of PPOA. The UAV needs to avoid collisions with both types of obstacles when searching for the target. In addition, we used ML-Agents [36], a deep learning framework, to communicate data between our algorithms and the 3D scenarios.

4.2. Model Design and Parameter Selection. The neural network architecture employed in this study is illustrated in Table 1. The architecture comprises three sections: a representation network, an OMI network, and an actor-critic network. The representation network comprises a perception network and a state network, which are responsible for extracting observation features. Specifically, the perception network handles the sensed information from the ray sensors, while the state network is responsible for processing the state information of the UAV. Both networks are linear fully connected networks, each with two layers of 128 and 64 neurons, respectively. Following the feature extraction of the representation network, the outputs are concatenated to form a 128-dimensional feature representation. The second section is the OMI network based on the GRU architecture. This network comprises a single layer with 128 neurons to enhance the observation memory through selectively remembering and forgetting information and updating the memory representation. The third section is the actor-critic network that receives the reshaped states from DRPAR to model continuous actions. The actor network and the critic network are both linear fully connected networks consisting of two layers of 64 neurons each. They are used to fit the policy $\pi_\theta(s_t^{\text{reshape}})$ and the state value function $V_\phi(s_t^{\text{reshape}})$.

Table 1: Network architecture.

Network	Neuron number	Type
Perception	(128, 64)	Linear
State	(128, 64)	Linear
OMI	128	GRU
Critic	(64, 64)	Linear
Actor	(64, 64)	Linear

The optimization parameters of PPO are presented in Table 2. The discount factor λ in the generalized advantage estimation (GAE) is set to 0.97, and the reward discount factor is set to 0.9. The two values, being close to 1.0, are chosen to emphasize the importance of long-term rewards. The clip parameter ϵ is set to 0.2 to effectively control the magnitude of policy adjustment. And the N-step is set to 3 when estimating advantages in GAE. During each sampling, both the actor network and the critic network undergo iterations 10 times. Both networks employ a learning rate of 1.0e-4, and the Adam optimizer is utilized in the optimization process.

Table 3 shows the parameters of our reward functions. L is set to 1.0 to indicate the safe distance from the UAV to an obstacle. λ_1 is the collision penalty factor set to 1.0 to return the maximum negative reward when a collision happens. λ_2 is the per-step penalty factor set to 0.001 to return the small negative reward to activate the UAV. λ_3 is the navigation reward factor set to 1.0 to return the large positive reward when the UAV successfully navigates to the target. And $d_{\min}$ is the threshold set to 0.1 to measure the proximity between the UAV and the target. If the distance between the UAV and the target is smaller than $d_{\min}$, it is approximately considered that the UAV has reached the target.

4.3. Result and Analysis. We conduct comprehensive experiments to evaluate the performance of related methods. In terms of training, we present a comparative analysis of the trends of different methods concerning the per-episode cumulative reward and the per-episode step length. In addition, we comparatively analyze the statistical results of our reward functions concerning success rate. In terms of inference, we comparatively analyze the statistical results of different methods concerning success, timeout, and collision rates. And finally, we present the PPOA process of RE-PPO in the two 3D scenarios.

We design three end-of-episode conditions in our experiments. Firstly, we set the maximum step limit for each episode to 1000 steps. If the UAV exceeds this limit, an episode ends immediately. Secondly, the end-of-episode condition is triggered if the UAV collides with an obstacle. Lastly, the episode ends immediately if the UAV successfully navigates to the target. Once one of these end-of-episode conditions is met, two new random locations for the UAV and the target are generated, respectively, and a new episode begins.

4.3.1. Per-Episode Cumulative Reward. The per-episode cumulative reward during training is the core evaluation indicator for the merits of DRL. The comparative trends of the per-episode cumulative reward of the four methods are shown in Figure 7. In Figure 7, the horizontal coordinate represents the training step, the vertical coordinate represents the per-episode cumulative reward, the lines with different colors denote the average performances in three experiments, and the colored regions depict the standard deviation of the four methods. In the following content, RE-PPO represents the proposed method combining RE with PPO, OMI-PPO removes DRPAR from RE and combines OMI with PPO, and DRPAR-PPO removes OMI from RE and combines DRPAR with PPO.

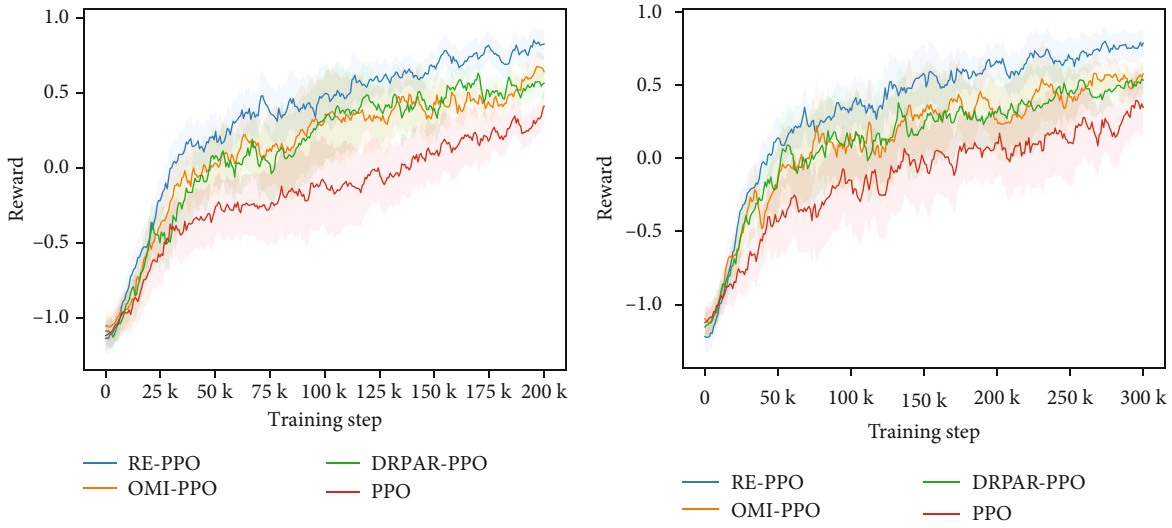
Table 2: Optimization parameters of PPO.

Parameter	Value
λ	0.97
ϵ	0.2
N-step	3
Actor iteration	10
Critic iteration	10
Actor learning rate	1.0e-4
Critic learning rate	1.0e-4
Optimizer	Adam
Reward discount factor	0.9

Table 3: Reward function parameters.

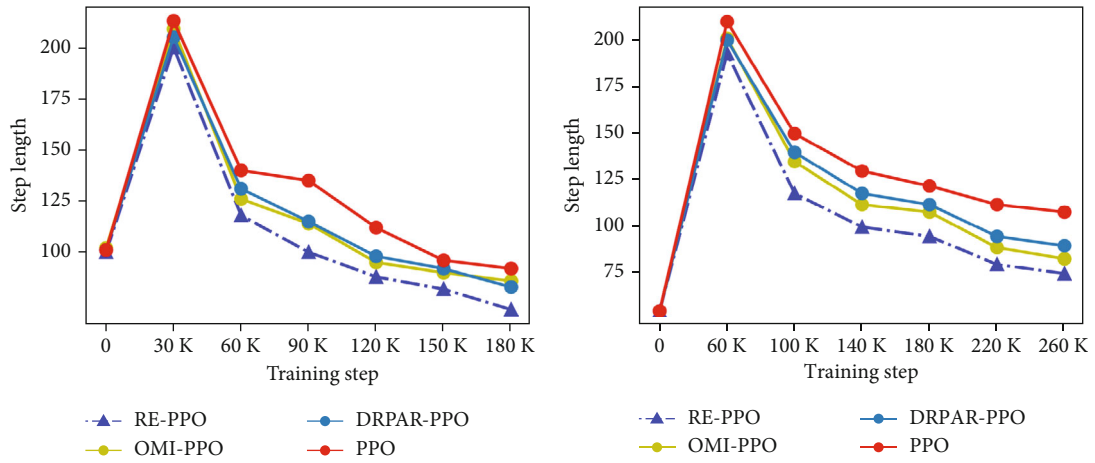
Parameter	Value
L	1.0
λ_1	1.0
λ_2	0.001
λ_3	1.0
$d_{\min}$	0.1

From Figure 7, we can see that the rewards of the four methods show an overall increasing trend within a limited number of training steps. At the beginning stage of training (0K-25K steps in Scenario A and 0K-40K steps in Scenario B), the performances of the four methods have little difference, and all methods show a significant growth trend. The reason is that all methods have a great potential to improve their decision-making ability through limited experience in the early stages of training. As training proceeds, the decision-making ability of the four methods becomes increasingly distinct due to their different capabilities in extracting and utilizing latent knowledge. Compared with RE-PPO, the performance of OMI-PPO and DRPAR-PPO is weaker. The reason is that OMI-PPO or DRPAR-PPO only considers a single enhancement module. OMI enhances observation memory through selectively remembering and forgetting information to improve decision-making ability. DRPAR extracts similarity between episodes by coordinate transformation to compress state space. Therefore, the per-episode cumulative reward of OMI-PPO and DRPAR-PPO is higher than that of PPO, proving the validity of OMI and DRPAR.



(a) Trends in Scenario A (b) Trends in Scenario B

Figure 7: Per-episode cumulative reward trends in Scenarios A and B.



(a) Trends in Scenario A (b) Trends in Scenario B

Figure 8: Per-episode step length trends in Scenarios A and B.

Due to the fact that Scenario B has additional obstacles, the four methods require more training to reach the same level as in Scenario A. Notably, the advantage of OMI-PPO over DRPAR-PPO is more significant in Scenario B than in Scenario A. The reason is that Scenario B has a more complex environment, requiring the UAV to have a stronger observation memory capability. In contrast, the environment in Scenario A is relatively simple; therefore, the performance difference between OMI-PPO and DRPAR-PPO is less pronounced. After training for a certain number of steps (100k steps in Scenario A and 150k steps in Scenario B), the advantage of RE-PPO becomes increasingly significant.

In most episodes, the reward of RE-PPO is higher than that of the other three methods and eventually converges to the highest value, approximately 0.75, in both scenarios. The standard deviation of RE-PPO is also smaller than the other three methods. Furthermore, at the end phase of training (185k-200k steps in Scenario A and 260k-300k steps in Scenario B), RE-PPO shows less trend fluctuation, indicating a more pronounced and faster convergence.

4.3.2. Per-Episode Step Length. The per-episode step length offers another perspective for describing convergence, stability, and efficiency during training. Figure 8 presents the comparative trends of the four methods. The step length is inferred by using the trained models at different stages. From Figure 8, it is observed that all methods show an increasing trend followed by a decreasing trend. At the initial stage of training, the UAV lacks decision-making ability and is highly susceptible to collision with obstacles, leading to episode termination. In Scenario A, all methods terminate their episodes within approximately 100 steps, while in Scenario B, the presence of additional obstacles causes all

Table 4: Success rate statistics in Scenario A.

Training step	ASN	AS	AN	SN
50K	31%	17%	12%	2%
100K	55%	28%	22%	5%
150K	72%	35%	28%	3%
200K	82%	44%	31%	6%

Table 5: Success rate statistics in Scenario B.

Training step	ASN	AS	AN	SN
50K	25%	13%	6%	1%
100K	39%	19%	11%	2%
150K	56%	24%	16%	2%
200K	66%	29%	19%	3%
250K	72%	33%	23%	2%
300K	79%	39%	26%	2%

methods to terminate their episodes within approximately 40 steps. With continued training, the UAV gradually improves its ability on obstacle avoidance; thus, the step length increases. After training for 30k steps in Scenario A and 60k steps in Scenario B, the step length decreases, indicating that the UAV has learned more experience to reach the target. As experience is gained, the success rate grows, requiring fewer steps in one episode to reach the target. Throughout the overall trend, the step length of RE-PPO is less than that of the other three methods, indicating its effectiveness in PPOA.

4.3.3. Reward Function Evaluation. To prove effectiveness, we count the success rates for our reward functions. Specifically, based on RE-PPO, we utilize the trained models of different reward functions at different stages to count the success rates, each model being run 100 times. In Tables 4 and 5, ASN denotes the combination of $r_{\text{avoidance}}$, r_{step}, and $r_{\text{navigation}}$; AS denotes the combination of $r_{\text{avoidance}}$ and r_{step}; AN represents the combination of $r_{\text{avoidance}}$ and $r_{\text{navigation}}$; and SN represents the combination of r_{step} and $r_{\text{navigation}}$.

From Tables 4 and 5, except for SN in Scenario B, the success rates of the three reward functions keep increasing as training steps grow. Since ASN simultaneously considers the rewards from obstacle avoidance, per-step, and navigation, its success rate keeps the highest level. When trained for 200k steps in Scenario A, its success rate reaches 83% and when trained for 300k steps in Scenario B, its success rate reaches 80%. Moreover, ASN exhibits the most rapid increase in success rate compared to the other reward functions.

Since AS excludes the navigation reward, there are no positive rewards to motivate the UAV to reach the target. Thus, compared to ASN, the success rate of AS is lower. When trained for 200k steps in Scenario A, the success rate is 44%, and when trained for 300k steps in Scenario B, the success rate is 39%.

Table 6: Inference statistics in Scenario A.

Completion rate	RE-PPO	OMI-PPO	DRPAR-PPO	PPO
SR	82%	66%	65%	43%
TR	15%	29%	30%	47%
CR	3%	5%	5%	10%

Table 7: Inference statistics in Scenario B.

Completion rate	RE-PPO	OMI-PPO	DRPAR-PPO	PPO
SR	79%	64%	60%	40%
TR	17%	31%	32%	47%
CR	4%	5%	8%	13%

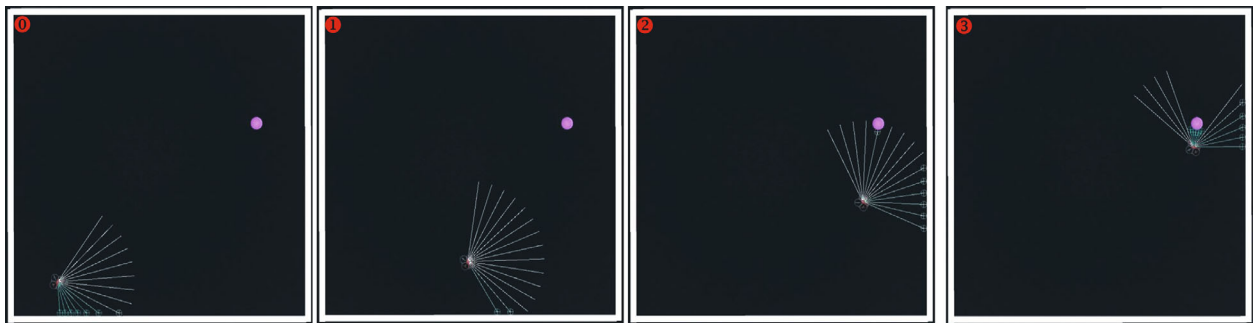
Due to the exclusion of the per-step reward, the enthusiasm of the UAV for movement is reduced in AN, leading to more focus on obstacle avoidance, which in turn causes the UAV to get stuck in being stationary and fail to reach the target within the specified steps. Thus, compared to ASN and AS, the success rate of AN is lower. When trained for 200k steps in Scenario A, the success rate is 31%, and when trained for 300k steps in Scenario B, the success rate is 26%.

Since SN excludes the obstacle avoidance reward, the UAV cannot receive negative feedback when collisions occur, causing frequent collisions and failures. Thus, SN has the lowest success rate compared to the other reward functions. When trained for 200k steps in Scenario A, the success rate is only 6%, and when trained for 300k steps in Scenario B, the success rate is only 2%. Interestingly, the success rate of SN in Scenario B decreases when the training steps increase from 200k to 250k. This suggests that the UAV cannot improve its ability to reach the target without the obstacle avoidance reward.

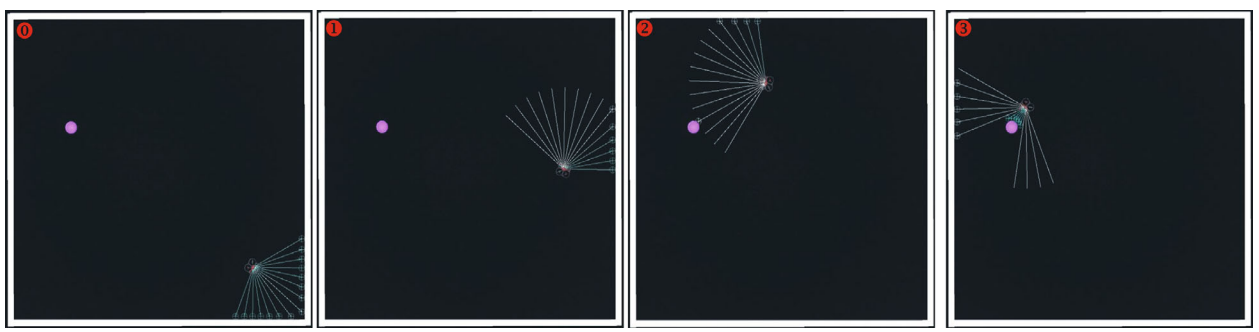
4.3.4. Inference Evaluation and Presentation. We use the trained models of the four methods to evaluate the completion rate during inference. The completion rate contains three aspects: success rate (SR), timeout rate (TR), and collision rate (CR). Specifically, each of the trained models is inferred 100 times. In each inference, if the step number exceeds the maximum step limit, 1000, it is considered a timeout.

Tables 6 and 7 show that the completion rate of RE-PPO is the highest. Due to the additional obstacles in Scenario B, the completion rates of the different methods decrease to some degree compared to those in Scenario A. The difference in statistical results between OMI-PPO and DRPAR-PPO in the two scenarios reveals a potential insight that OMI has better decision-making ability in complex scenarios than DRPAR. The reason is that OMI focuses on strengthening the observation memory, while DRPAR focuses on compressing the state space. PPO presents the worst performance due to the lack of additional enhancement modules.

Figures 9 and 10 show the PPOA process of RE-PPO in the two 3D scenarios. Each figure presents two episodes, and each episode captures four frames. In Figure 9(a), the initial

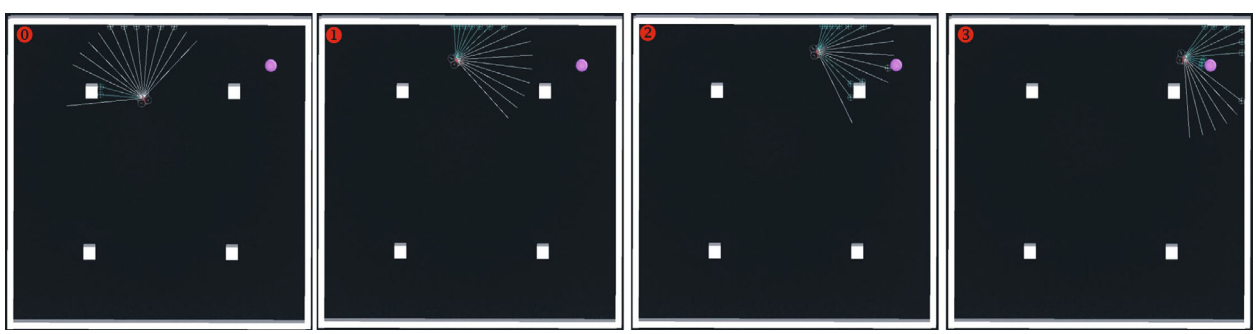


(a) PPOA process in Episode 1

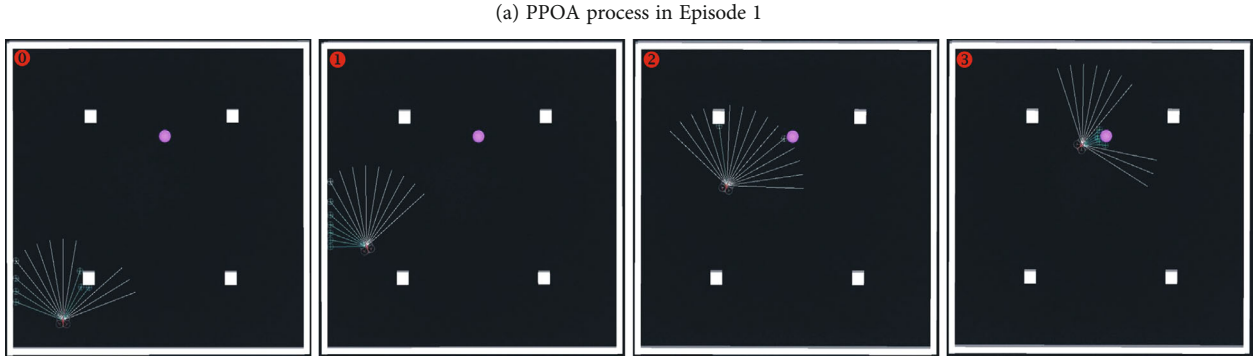


(b) PPOA process in Episode 2

Figure 9: PPOA process of RE-PPO in Scenario A.



(a) PPOA process in Episode 1



(b) PPOA process in Episode 2

Figure 10: PPOA process of RE-PPO in Scenario B.

orientation of the UAV deviates slightly from the direction toward the target. The UAV starts to move near the corner of the walls and adjusts its orientation through the perception of the ray sensor. When the target is sensed, the UAV gradually gets close to the target. To demonstrate the robustness of RE-PPO, in Figure 9(b), we set the UAV's initial orientation opposite to the target's direction. In this extreme case, the UAV turns to the left to avoid a collision with the

corner and moves to the target intelligently. In Figure 10(a), the target is positioned behind the pillar, while the UAV is near another. The UAV successfully identifies the two pillars and navigates toward the target. To further evaluate the effectiveness of RE-PPO, we increase the task difficulty in Figure 10(b). Specifically, we increase the distance between the UAV and the target and situate the UAV behind the pillar that occludes the UAV's perception of the target. Despite these challenges, the UAV can still find a reasonable trajectory to reach the target. From the overall process Of PPOA, it is observed that, during obstacle avoidance, the UAV adjusts its orientation to the direction indicated by more free rays and moves forward in this direction to avoid collisions. After detecting the target, the UAV adjusts its motion direction to the direction pointed by the rays that have sensed the target. These behaviors demonstrate the superior decision-making ability of RE-PPO, making it a promising approach for UAV control in complex environments.

4.4. Performance Evaluation in Real Scenarios. To demonstrate the effectiveness of RE-PPO, we evaluate its performance in a complex 3D city model, as shown in Figure 11. Compared with the 2D scenarios in previous work [13, 15], the city model is more realistic in presenting the process of PPOA.

Figure 12 shows the detailed process in a local area. We present our PPOA in two episodes. In both episodes, the initial state of the UAV and the target point are randomly generated, and the relative position between the UAV and the target point is opposite. There are obstacles, such as the buildings and the trees, around the UAV and the target point. Despite these interference factors, the UAV successfully navigates to the target point in both episodes. The results demonstrate the effectiveness and adaptability of RE-PPO.

5. Discussion

Although we have provided a rational and intuitive analysis of the experimental results in the previous section, two aspects still require further elaboration. Firstly, since the per-episode cumulative reward is a fundamental metric for evaluating the performance of DRL, we need to conduct an in-depth discussion on the instability of the per-episode cumulative reward for the four methods. Secondly, there are some limitations to our proposed approach, and these limitations will be the focus of future work.

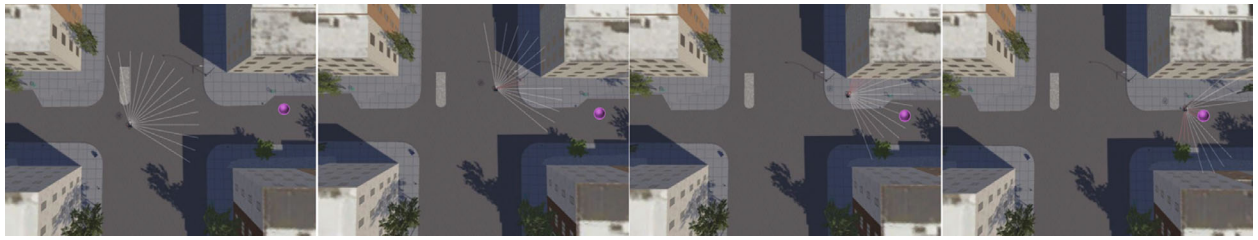
In DRL, instability is a common phenomenon. During training, an agent may encounter unfamiliar or known situations that require adjusting or reusing the policy. At this time, performance may decrease or improve. Based on the actual situations of our experiments, we analyze the reasons resulting in trend fluctuation, trend intersection, and trend approximation, as shown in Figure 13. In the figure, the rectangles indicate trend fluctuation, the circles indicate the intersection between trends, and the triangles indicate that the per-episode cumulative reward of other methods approximates that of RE-PPO.

Figure 11: A 3D city model.

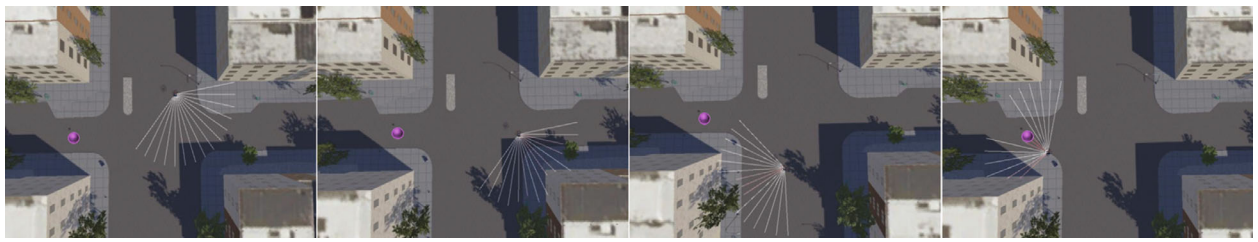
There are two reasons for the fluctuation. Firstly, the UAV's initial state and the target are random in each episode. Therefore, the cumulative rewards obtained by the UAV in each episode are different, which directly leads to the fluctuation. Secondly, the PPO-based methods clip the gradient when optimizing the objective function, which causes unstable gradient backpropagation and leads to indirect fluctuation.

The intersection mainly occurs between OMI and DARPAR; the reasons for this can be attributed to three aspects. Firstly, OMI introduces a GRU network to enhance the observation memory, and the network parameters of GRU require more experience to optimize decision-making, which introduces some training perturbations. Secondly, DARPAR only compresses the state space and thus cannot effectively make decisions for new trajectories. Thirdly, the randomness of the UAV's initial state and the target in each episode leads to uncertainties in a generated trajectory. The UAV receives higher rewards for similar trajectories generated before, while for new trajectories, the UAV receives lower rewards. Moreover, the frequency of the intersection is higher in Scenario B. The reason is that Scene B is more complex than Scene A, requiring more training steps with a higher occurrence probability of similar or new trajectories.

In some cases, the performance of OMI-PPO and DARPAR-PPO can approximate that of RE-PPO. The reason is that RE-PPO encounters extremely random challenges, such as the UAV's initial position being close to obstacles while the target is far from the UAV. Therefore, the UAV needs more steps to explore the environment, resulting in a decrease in the cumulative rewards and narrowing the gap with the comparative methods. The approximation mainly occurs in the early and middle stages of training, when the decision-making ability of related methods is still improving and is greatly influenced by random factors. As training progresses, the decision-making ability of RE-PPO gradually stabilizes and surpasses that of the other methods.

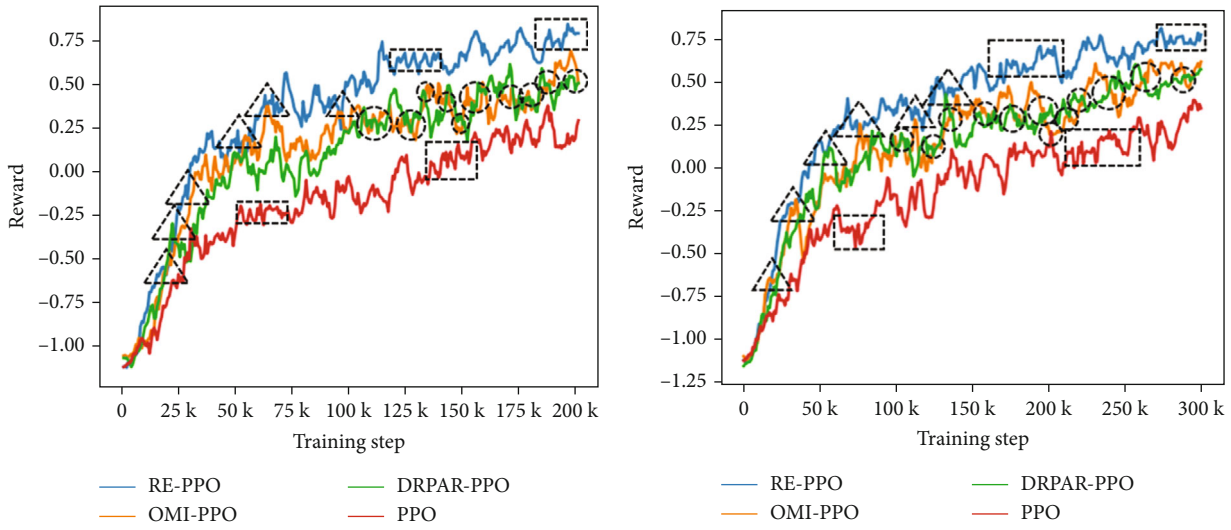


(a) PPOA process in Episode 1



(b) PPOA process in Episode 2

Figure 12: PPOA process of in the city model.



(a) Instability in Scenario A

(b) Instability in Scenario B

Figure 13: Instability of per-episode cumulative reward.

The proposed method still has some limitations. Firstly, OMI only separately extracts features for the perception and the state information and directly concatenates the extracted features. However, the direct concatenation ignores the weight between the perception and the state information. Therefore, an attention mechanism can be introduced in future work to weigh the fusion for the extracted features. Secondly, the experimental design in this paper has certain constraints, mainly manifested in the UAV's fixed altitude and the ray sensor's singularity, making it difficult to use in reality. Therefore, future work will increase the complexity of the experimental scenarios, free up the UAV's altitude, and introduce multiple heterogeneous sensors to adapt to more complex environments. Thirdly, in the design of the navigation reward $r_{\text{navigation}}$, a positive reward is only given to the UAV when it is close to the target, resulting in a lag in rewards and causing the UAV to perform additional steps. Therefore, future work will optimize the design of $r_{\text{navigation}}$ by providing a progressive reward to the UAV based on the relative position change between the UAV and the target.

6. Conclusion

In this study, we propose a RE-PPO framework to address the challenges of partial observation and large state space when searching for random targets through continuous actions. The RE module consists of OMI and DRPAR. We designed three 3D virtual scenarios to demonstrate the effectiveness of RE. The experimental results show that RE-PPO

achieves a faster convergence, a higher success rate, and a lower rate of timeout and collision. The experimental results also reveal an interesting conclusion that the performance difference between OMI and DRPAR in a simple environment is insignificant, while in a complex environment, OMI works better than DRPAR.

Future work mainly focuses on improving the applicability in more complex and uncertain environments. We will explore more effective methods for observation memory improvement and dynamic relative position-attitude reshaping to enhance perception ability and state space compression effect. We will also try to use other reinforcement learning algorithms instead of PPO to compare the advantages and disadvantages of different algorithms in this task. Moreover, we will deploy RE-PPO on real UAVs and conduct practical applications in various domains, such as express logistics, environmental monitoring, and maritime search and rescue.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Authors' Contributions

Xiangxiang Huang proposed the main idea and wrote the manuscript, Wei Wang deployed the experimental environment and implemented experiments, Zhaokang Ji analyzed the results and revised the manuscript, and Bin Cheng provided the hardware for experiments. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

The authors would like to express their sincere gratitude to Professor Tao Liu for her help during the research process. Additionally, this research was jointly supported by the Anhui Province Natural Science Foundation under Grant 2208085QD106, the Foundation for Talented Scholars of Anhui Polytechnic University under Grants 2021YQQ013 and 2022YQQ094, the Industry Collaborative Innovation-fund of Anhui Polytechnic University and Jiujiang District under Grant 2021cyxtb4, and the Shanghai Key Laboratory of Intelligent Manufacturing and Robotics under Grant ZK2203.

References

[1] S. Wen, Q. Zhang, X. Yin, Y. Lan, J. Zhang, and Y. Ge, "Design of plant protection UAV variable spray system based on neural networks," *Sensors*, vol. 19, no. 5, p. 1112, 2019.

[2] Y. Wang, Y. Yue, M. Shan, L. He, and D. Wang, "Formation reconstruction and trajectory replanning for multi-UAV patrol," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 2, pp. 719–729, 2021.

[3] S. Jiang, Q. Li, W. Jiang, and W. Chen, "Parallel structure from motion for UAV images via weighted connected dominating set," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–13, 2022.

[4] K. Feng, J. Ji, Y. Zhang, Q. Ni, Z. Liu, and M. Beer, "Digital twin-driven intelligent assessment of gear surface degradation," *Mechanical Systems and Signal Processing*, vol. 186, article 109896, 2023.

[5] K. Feng, Y. Xu, Y. Wang et al., "Digital twin enabled domain adversarial graph networks for bearing fault diagnosis," *IEEE Transactions on Industrial Cyber-Physical Systems*, vol. 1, pp. 113–122, 2023.

[6] K. Feng, J. Ji, Q. Ni, Y. Li, W. Mao, and L. Liu, "A novel vibration-based prognostic scheme for gear health management in surface wear progression of the intelligent manufacturing system," *Wear*, vol. 522, article 204697, 2023.

[7] S. Jiang, W. Jiang, and L. Wang, "Unmanned aerial vehicle-based photogrammetric 3d mapping: a survey of techniques, applications, and challenges," *IEEE Geoscience and Remote Sensing Magazine*, vol. 10, no. 2, pp. 135–171, 2022.

[8] S. A. Fadzli, S. I. Abdulkadir, M. Makhtar, and A. A. Jamal, "Robotic indoor path planning using Dijkstra's algorithm with multi-layer dictionaries," in *2015 2nd International Conference on Information Science and Security (ICISS)*, Seoul, Korea, 2015.

[9] Y. Cai, Q. Xi, X. Xing, H. Gui, and Q. Liu, "Path planning for UAV tracking target based on improved A-star algorithm," in *2019 1st International Conference on Industrial Artificial Intelligence (IAI)*, Shenyang, China, 2019.

[10] H. Chen and P. Chen, "A novel three-dimensional path planning method for fixed-wing UAV using improved particle swarm optimization algorithm," *International Journal of Aerospace Engineering*, vol. 2021, Article ID 7667173, 19 pages, 2021.

[11] H.-y. Zhang, W.-m. Lin, and A.-x. Chen, "Path planning for the mobile robot: a review," *Symmetry*, vol. 10, no. 10, p. 450, 2018.

[12] J. Gao, W. Ye, J. Guo, and Z. Li, "Deep reinforcement learning for indoor mobile robot path planning," *Sensors*, vol. 20, no. 19, p. 5493, 2020.

[13] S.-M. Hung and S. N. Givigi, "A q-learning approach to flocking with UAVs in a stochastic environment," *IEEE Transactions on Cybernetics*, vol. 47, no. 1, pp. 186–197, 2017.

[14] Z. Yijing, Z. Zheng, Z. Xiaoyi, and L. Yang, "Q learning algorithm based UAV path learning and obstacle avoidence approach," in *2017 36th Chinese Control Conference (CCC)*, Dalian, China, 2017.

[15] C. Yan and X. Xiang, "A path planning algorithm for UAV based on improved q-learning," in *2018 2nd International Conference on Robotics and Automation Sciences (ICRAS)*, Wuhan, China, 2018.

[16] Y. Xu, K. Feng, X. Yan et al., "Cross-modal fusion convolutional neural networks with online soft label training strategy for mechanical fault diagnosis," *IEEE Transactions on Industrial Informatics*, pp. 1–12, 2023.

[17] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[18] M. Jaderberg, W. M. Czarnecki, I. Dunning et al., "Human-level performance in 3d multiplayer games with population-

based reinforcement learning," *Science*, vol. 364, no. 6443, pp. 859–865, 2019.

[19] Z. Zhang, D. Zhang, and R. C. Qiu, "Deep reinforcement learning for power system applications: an overview," *CSEE Journal of Power and Energy Systems*, vol. 6, no. 1, pp. 213–225, 2019.

[20] T. L. Meng and M. Khushi, "Reinforcement learning in financial markets," *Data*, vol. 4, no. 3, p. 110, 2019.

[21] J. Li, D. Pang, Y. Zheng, X. Guan, and X. Le, "A flexible manufacturing assembly system with deep reinforcement learning," *Control Engineering Practice*, vol. 118, p. 104957, 2022.

[22] G. Raja, S. Anbalagan, V. S. Narayanan, S. Jayaram, and A. Ganapathisubramaniyan, "Inter-UAV collision avoidance using deep-q-learning in flocking environment," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 1089–1095, New York, NY, USA, 2019.

[23] S. Li, X. Chen, M. Zhang, Q. Jin, Y. Guo, and S. Xing, "A UAV coverage path planning algorithm based on double deep q-network," *Journal of Physics: Conference Series*, vol. 2216, article 012017, 2022.

[24] J. Roghair, A. Niaraki, K. Ko, and A. Jannesari, "A vision based deep reinforcement learning algorithm for UAV obstacle avoidance," *Intelligent Systems and Applications: Proceedings of the 2021 Intelligent Systems Conference (IntelliSys) Volume 1*, , pp. 115–128, Springer, 2022.

[25] G. Xu, W. Jiang, Z. Wang, and Y. Wang, "Autonomous obstacle avoidance and target tracking of UAV based on deep reinforcement learning," *Journal of Intelligent & Robotic Systems*, vol. 104, no. 4, p. 60, 2022.

[26] C. Qi, C. Wu, L. Lei, X. Li, and P. Cong, "UAV path planning based on the improved PPO algorithm," in *2022 Asia Conference on Advanced Robotics, Automation, and Control Engineering (ARACE)*, pp. 193–199, Qingdao, China, 2022.

[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, https://arxiv.org/abs/1707.06347.

[28] S. Zhang, Y. Li, and Q. Dong, "Autonomous navigation of UAV in multi-obstacle environments based on a deep reinforcement learning approach," *Applied Soft Computing*, vol. 115, article 108194, 2022.

[29] A. S. Tasbas, S. O. Sahin, and N. K. Üre, "Reinforcement learning based self-play and state stacking techniques for noisy air combat environment," in *AIAA SCITECH 2023 Forum*, National Harbor, MD, 2023.

[30] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 1, pp. 107–118, 2021.

[31] S. S. Mansouri, C. Kanellakis, D. Kominiak, and G. N. Lakopoulos, "Deploying MAVs for autonomous navigation in dark underground mine environments," *Robotics and Autonomous Systems*, vol. 126, article 103472, 2020.

[32] T. Wang, R. Qin, Y. Chen, H. Snoussi, and C. Choi, "A reinforcement learning approach for UAV target searching and tracking," *Multimedia Tools and Applications*, vol. 78, no. 4, pp. 4347–4364, 2019.

[33] M. Lauri, D. Hsu, and J. Pajarinen, "Partially observable Markov decision processes in robotics: a survey," *IEEE Transactions on Robotics*, vol. 39, no. 1, 2023.

[34] C. Wang, J. Wang, J. Wang, and X. Zhang, "Deep-reinforcement-learning-based autonomous UAV navigation with sparse rewards," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6180–6190, 2020.

[35] C. Wang, J. Wang, X. Zhang, and X. Zhang, "Autonomous navigation of UAV in large-scale unknown complex environment with deep reinforcement learning," in *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 858–862, Montreal, QC, Canada, 2017.

[36] A. Juliani, V.-P. Berges, E. Teng et al., "Unity: a general platform for intelligent agents," 2018, https://arxiv.org/abs/1809.02627.