AN UAV MANEUVERING DECISION-MAKING ALGORITHM BASED ON DEEP TRANSFER REINFORCEMENT LEARNING

Ke LI[1], Kun ZHANG[1,*], Haining LIU[2], Yang LI[2] & Qian WANG[3]

[1]School of Electronics and Information, Northwestern Polytechnical University, Xi'an, China
[2]Shenyang Aircraft Design and Research Institute, Shenyang, China
[3]National Elite Institute of Engineering, Northwestern Polytechnical University, Xi'an, China
[*]Corresponding author

Abstract

The UAV has been used to assist and replace people to execute dirty, boring, and difficult missions due to low cost, high mobility, and unmanned feature. One of the important technical issues form various engineering fields is finding an optimal path from start point to end point and designing a controller to manipulate it following the path in an interactive environment. A large amount of research has been devoted to improving the autonomous decision-making ability of UAV in an interactive environment, where finding the optimal maneuvering decision-making policy becomes one of the key issues for improving the autonomy of UAV during executing missions. In this paper, we propose an UAV maneuvering decision-making algorithm based on deep transfer reinforcement learning for autonomous air-delivery in an interactive environment containing some threatens. Firstly, we refine the long-distance guidance task and the near-end aiming task from air-delivery mission, and design the UAV maneuvering decision-making model based on MDPs. Specifically, in order to improve the learning efficiency of policy, we design a novel reward function based on reward shaping under the guidance of expert experience. Finally, we propose the UAV maneuvering decision-making algorithm based on Twin Delayed Deep Deterministic Policy Gradient (TD3). The algorithm we proposed could accelerate the convergence of the decision-making policy and increase the stability of the policy during the later stage of training process. The effectiveness of the proposed algorithm is illustrated by the curves of training parameters and extensive experimental results for testing the trained policy.

Keywords: unmanned aerial vehicle (UAV), maneuvering decision-making, autonomous air-delivery, deep transfer reinforcement learning, expert experience

1. Introduction

With the rapid development of UAV avionics, the UAV has been used to assist and replace people to execute dirty, boring, and difficult missions due to low cost, high mobility, and unmanned feature[1]. Therefore, the UAV is widely used to surveillance, search, tracking, delivery, and other missions which could be finished step by step[2]. Thus, how to improve autonomy of UAV while performing some tasks without risking human lives has become the research focus. For instance, some people use UAV to carry out delivery of relief supplies, extinguishing, and so on[2]. Consequently, it has become one of the key issues for engineering applications to improve the autonomous flight capability of UAV. Now, the UAV is mainly used to execute some tasks which could be done automatically instead of finished manually, such as long-distance air-delivery[3]. During the process of performing air-delivery mission, UAV flies towards required placement area and need to avoid threatens such as mountain, buildings, no-fly zone and so on. And UAV should adjust its nose so as to make payload fall on the center of target area after UAV dropping. As shown in Figure 1, the diagram describes the process of performing long-distance guidance task and near-end aiming task involved in long-distance air-delivery mission. In this paper, UAV avoids threatens appearing within mission area while flying towards required placement area in the long-distance guidance task. When UAV enters

the adjustment area whose radius is 2x dropping range of payload, UAV continues executing near-end aiming task and need to adjust its nose to make the impact position of payload as close as possible to the centre of required placement area.

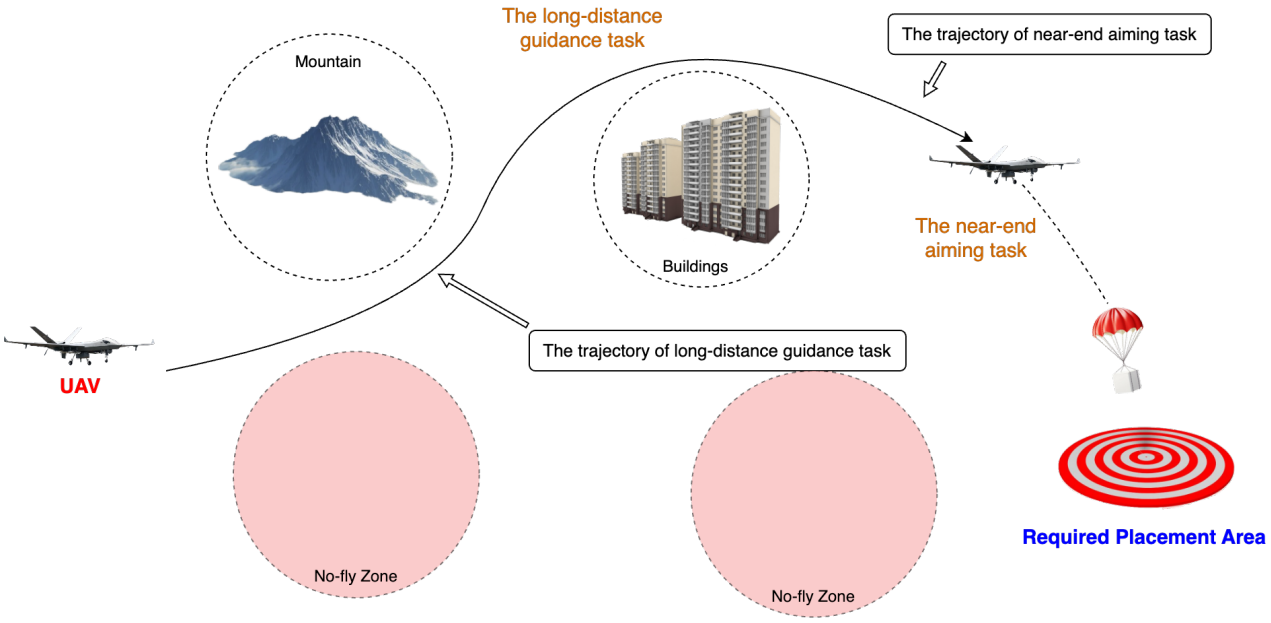


Figure 1 – The diagram of UAV autonomous guidance and aiming problem consist of the long-distance guidance task and the near-end aiming task.

According to the description of tasks mentioned above, one of the important technical issues from these tasks is finding an optimal path from start point to end point and designing a controller to manipulate it following the path [4]. At present, the optimal path should be found by some path planning algorithms, such as Visibility Graph [5, 6], randomly sampling search algorithms including Rapidly-exploring Random Tree [7], Probabilistic Roadmap [8], heuristic algorithms including A-Star [9], Sparse A-Star [10], and D-star [11], genetic algorithms [12], and so on. Then, a controller could be designed to operate UAV following the path planned by algorithm, where various trajectory tracking algorithms [13, 14] are proposed. However, there are some disadvantages in the solution mentioned above. For example, the data of terrain and obstacles is so difficult to obtain that our capability of environment modelling is limited, because the optimal path relies on lots of priori knowledge about the environment. Moreover, when the environment becomes dynamic and there are some moving obstacles, the scheme designed above is not flexible enough to alter their control strategies immediately. A replan of paths has to be scheduled to adapt to the changes in the environment. Furthermore, because conventional path planning algorithms need much more time to calculate optimal path, it's difficult to be applied to solving real-time problems. Therefore, it is necessary to design an end-to-end algorithm which could be used to operate UAV to fly autonomously in a dynamic environment without path planning and trajectory tracking.

A research highlight is inspired by AlphaGo developed by Google based on deep reinforcement learning (DRL), which could play Atari games using a kind of end-to-end decision-making algorithm, called Deep Q Network (DQN) [15]. The performance of this algorithm reached human level after an extensive training, and it attracts lots of researchers from various fields to study the applications of DRL in all kinds of engineering problems. Meanwhile, the Deep Deterministic Policy Gradient (DDPG) [16] was proposed to solve the dimension explosion caused by the continuity of action space and state space. And the experience replay method sampling from experience buffer is constructed in these algorithms and allows agent to remember and learn from historical data. Moreover, the Prioritized Experience Replay (PER) [17] was proposed to improve the efficiency of learning from experiences. It uses the potential value of historical data to increase the convergence rate of policy network, because a priority model of each sample is designed to evaluate the profit of samples for training of policy network at current step.

In addition, it's important for training policy to design the reward function of problem. Traditional formulation of reward function comes from original model of problem [15], such as CartPole, Pendulum and other games from Atari, but there're not suitable reward founction for a new problem. Thus, how to construct an appropriate reward function is not the key issue for many popular work, and lots of researchers mainly focus on the improvements of algorithms instead of modelling. However, it's also essential for solving problems from specific research fields to define modified model when we attempt to apply DRL for a new problem. Traditionally, some researchers consider that the reward function involved in problem model is designed by factual experience so that the trained policy extremely relied on the capability of designer. Although some work published found effective policy recently, we could find that different formulation of reward function brought different training effect and trained policy.
In the present work, we aim to tackle the challenges mentioned above and focus on the UAV maneuvering decision-making for the long-distance guidance task and the near-end aiming task. The main work are summarized as follows:

- We refine the long-distance guidance task and the near-end aiming task and the UAV maneuvering decision-making model based on MDPs is built. Particularly, we design a dynamic state space to adapt to different stage including flying towards target, avoiding threaten and aiming. Among the components of model, we devote our air-to-ground drop theory to designing the UAV maneuvering decision-making model.
- We propose the UAV maneuvering decision-making algorithm for tasks we presented above based on TD3 [18] with a novel reward function to train a policy for generating efficient control signal to manipulate UAV in an interactive environment. Specially, we designed a novel reward function based on reward shaping under the guidance of expert experience. By contrast, the reward model we defined could improve the training efficiency.
- We design and construct lots of experiments to validate the availability of our proposed algorithm and model. Simulation results show that the algorithm we proposed could improve the autonomy of UAV during the long-distance guidance task and the near-end aiming task. And the novel reward function modified by expert experience could improve the training efficiency of algorithm.

2. Methodology

The UAV has been used to help people finish some dangerous and repetitive missions, such as crop protection, wildlife surveillance, traffic monitoring, electric power inspection, search, and rescue operations. A need for more advanced and simple UAV autonomous flight solution has emerged. As mentioned above, traditional solution of real time obstacle avoidance for manipulators and UAVs is that algorithm plans an optimal path and then UAV follows path by trajectory tracking method. In this paper, we redefined the process of UAV autonomous flight and constructed the UAV Maneuvering Decision-Making Model for Air-delivery mission based on MDPs. On the other hand, we proposed a novel UAV Maneuvering Decision-Making algorithm based on Deep Transfer Reinforcement Learning (DTRL).
As shown in Figure 2, we constructed the UAV Maneuvering Decision-Making Model for Air-Delivery consisting of long-distance guidance task and the near-end aiming task based on MDPs firstly. Among this model, we designed action space, state space and basic reward of each task which were used to demonstrate the characteristics of UAV autonomous flight during air-delivery. Moreover, we designed and realized the UAV maneuvering decision-making algorithm including UAV maneuvering decision-making policy based on neural network, and policy network was optimized according to data sampled from historical experiences by PER. Meanwhile, we constructed the shaping reward of each task to increase the convergence rate of policy network.

2.1 The UAV Maneuvering Decision-making Model for Air-delivery Mission Based on MDPs

As mentioned above, the air-delivery mission was decomposed into long-distance guidance task and the near-end aiming task. We would present the definitions of long-distance guidance task and the

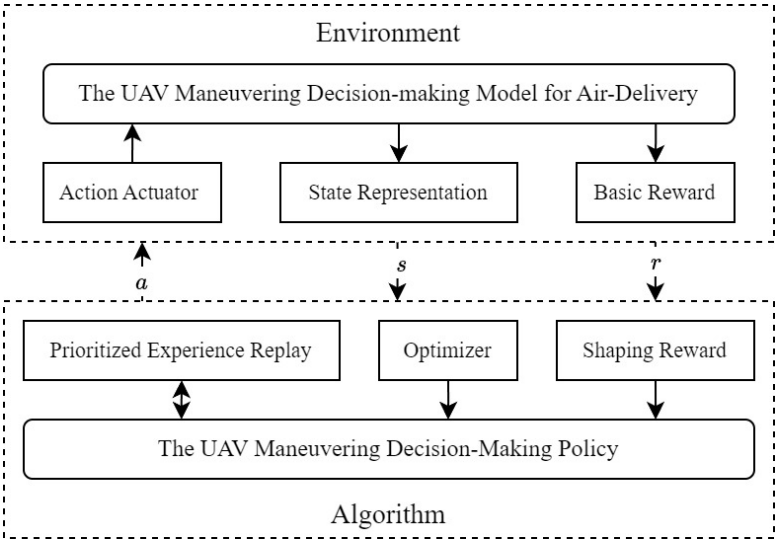


Figure 2 – The verification structure of UAV Maneuvering Decision-making algorithm showing relationship between UAV Maneuvering Decision-making Model and UAV Maneuvering Decision-making Algorithm.

near-end aiming task and design the state space, action space and reward function of each task based on MDPs in this section.

2.1.1 The Definitions of Long-distance Guidance Task and Near-end Aiming Task

In this paper, we defined that UAV should fly from start point to a specific point and finish aiming at required target area while avoiding threatens appearing within the mission area in order to execute air-delivery mission. Furthermore, we refined Long-distance Guidance Task and Near-end Aiming Task, while we described the definitions of Long-distance Guidance Task and Near-end Aiming Task in detail.

(1) Long-distance Guidance Task

Within the air-delivery mission, UAV needed to avoid threatens appearing in mission area during the preliminary stage of flying from start point to a specific point. UAV could perceive surrounding threatens by some airborne sensors. Meanwhile, UAV should fly towards required placement area containing target point when it considers avoiding threatens.

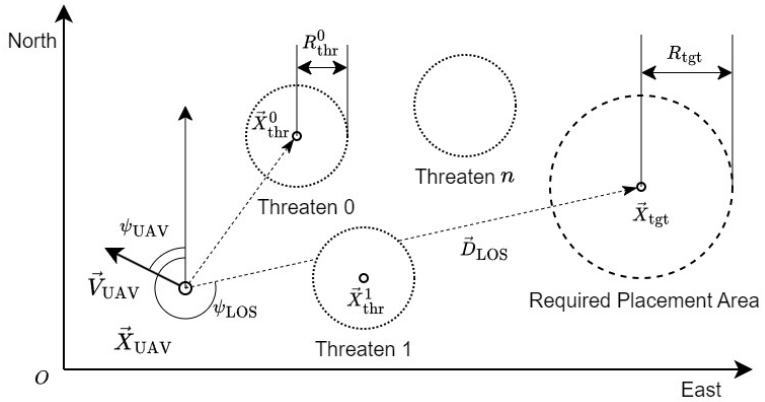


Figure 3 – The vector diagram of relationship among UAV, threatens and required placement area involved in the long-distance guidance task.

As shown in Figure 3, UAV started from a random point, and flying to required placement area was its purpose. In this paper, the flight state of UAV could be defined by position $\vec{X}_{\text{UAV}}$ and velocity $\vec{V}_{\text{UAV}}$. The

surrounding i-th threaten could be described by posion $\vec{X}^i_{\mathtt{thr}}$ and influence radius $R^i_{\mathtt{thr}}$. Threatens were usually no-fly zones, peaks, buildings and other areas which UAV isn't allowed to enter. And required placement area could be defined by position $\vec{X}_{\mathtt{tgt}}$ and effective radius $R_{\mathtt{tgt}}$.

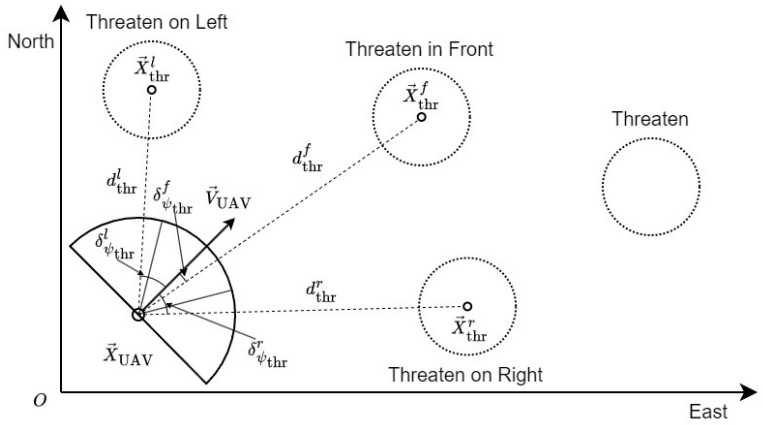


Figure 4 – The perception diagram of UAV observing threatens existing in the surroundings.

Moreover, UAV could observe threaten on left, threaten in front and threaten on the right by airborne sensors, such as radar, electro-optical pod, SAR and so on. As shown in Figure 4, when threatens entered into the perception range of sensors, UAV could observe that threaten and obtain the position of it. But the number of threatens information is up to 3, it means that the closest threaten can be observed separately on left, on right and in front. We defined a simplified criterion which was used to determine whether UAV could observe threaten $\vec{X}^i_{\mathtt{thr}}$ as

$$g_{\mathtt{sensor}}(\vec{X}_{\mathtt{UAV}},\vec{X}^i_{\mathtt{thr}}) = \begin{cases} 90^\circ \le \delta_{\psi_{\mathtt{thr}}} < 30 \wedge d_{\mathtt{thr}} \le d^{\max}_{\mathtt{sensor}} & \mathtt{Threaten\ on\ Left} \\ 30^\circ \le \delta_{\psi_{\mathtt{thr}}} < -30 \wedge d_{\mathtt{thr}} \le d^{\max}_{\mathtt{sensor}} & \mathtt{Threaten\ in\ Front} \\ -30^\circ \le \delta_{\psi_{\mathtt{thr}}} < -90 \wedge d_{\mathtt{thr}} \le d^{\max}_{\mathtt{sensor}} & \mathtt{Threaten\ on\ Right} \end{cases} \tag{1}$$

where $\delta_{\psi_{\mathtt{thr}}}$ was the relative azimuth between threaten and UAV's observation axis. $d_{\mathtt{thr}}$ indicated the distance between threaten and UAV and $d^{\max}_{\mathtt{sensor}}$ represented the maximum observation distance of airborne sensors. Next, we could obtain three sets $W^{\mathtt{left}}_{\mathtt{thr}}$, $W^{\mathtt{front}}_{\mathtt{thr}}$, $W^{\mathtt{right}}_{\mathtt{thr}}$ including threatens on left, in front, on right separately. Then, we got the closest threaten to the UAV on left, in front and on right in terms of $d_{\mathtt{thr}}$ and $g_{\mathtt{sensor}}$. For example, we could obtain the closest threaten on left by

$$g^{\mathtt{left}}_{\mathtt{sensor}} = \underset{i \in W^{\mathtt{left}}_{\mathtt{thr}}}{\arg\min}\, d^i_{\mathtt{thr}} \tag{2}$$

where $d^i_{\mathtt{thr}}$ indicated the distance between UAV and i-th threaten which belonged to $W^{\mathtt{left}}_{\mathtt{thr}}$.
In addition, we defined the termination condition of long-distance guidance task as

$$g_1(s) = \begin{cases} \|\vec{X}_{\mathtt{UAV}} - \vec{X}^i_{\mathtt{thr}}\| \le R^i_{\mathtt{thr}}, i = 1,2,\cdots,N_{\mathtt{thr}} & \mathtt{Failed\ termination} \\ \|\vec{X}_{\mathtt{UAV}} - \vec{X}_{\mathtt{tgt}}\| \le R_{\mathtt{tgt}} & \mathtt{Successful\ termination} \end{cases} \tag{3}$$

where $N_{\mathtt{thr}}$ indicated the number of threatens. Equation 3 showed that if UAV flies into the cutoff area of any threaten, this task will be failed. Otherwise, UAV finished task successfully when UAV entered the effective area of target point.

(2) Near-end Aiming Task

If UAV entered the effective area of target point, that meant UAV flyied into required placement area, UAV would continue to execute aiming task. Different from guidance task, UAV needed to adjust its attitude for eliminating impact error of payload by updating the maneuvering value of UAV.
As shown in Figure 5, we calculated the falling trajectory $\vec{A}_{\mathtt{drop}}$ of payload for estimating the impact position of payload. Comparing with target position $\vec{X}_{\mathtt{tgt}}$, we assessed the impact error $\vec{\omega}_{\mathtt{drop}}$ of payload, and corrected the control input of UAV to elimating the impact error of payload. When UAV

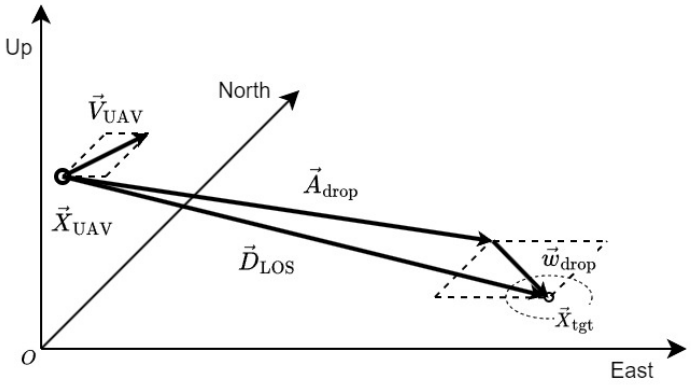


Figure 5 – The vector diagram of relationship between UAV and required placement area involved in the near-end aiming task.

was located in $\vec{X}_{\mathrm{UAV}}$, we calculated $\vec{A}_{\mathrm{drop}}$ by 3 degree of freedom kinematic model. Then, we could obtain the impact position of payload by

$$\vec{X}_{\mathtt{payload}} = \vec{X}_{\mathrm{UAV}} + \vec{A}_{\mathtt{drop}} \tag{4}$$

where $\vec{X}_{\mathtt{payload}}$ indicated the payload's landing position. Next, we could calculate the impact error by

$$\vec{\omega}_{\mathtt{drop}} = \vec{X}_{\mathtt{tgt}} - \vec{X}_{\mathtt{payload}} \tag{5}$$

Moreover, we defined the termination condition of aiming task as

$$g_2(s) = \begin{cases} \|\vec{X}_{\mathrm{UAV}} - \vec{X}_{\mathtt{tgt}}\| > e^{\mathtt{max}}_{\mathtt{aiming}} & \mathtt{Failed\ termination} \\ \|\vec{\omega}_{\mathtt{drop}}\| \leq R^{\mathtt{fine}}_{\mathtt{impact}} & \mathtt{Successful\ termination} \end{cases} \tag{6}$$

where $e^{\mathtt{max}}_{\mathtt{aiming}}$ indicated the maximum distance of aiming for UAV, that meant the aiming task ended failed when the distance between UAV and target exceeded $e^{\mathtt{max}}_{\mathtt{aiming}}$. $R^{\mathtt{fine}}_{\mathtt{impact}}$ represented the allowable maximum deviation for UAV dropping, that meant UAV finished aiming task successfully when $\|\vec{\omega}_{\mathtt{drop}}\|$ was less than it.

2.1.2 Markov Decision Processes

During the process of performing air-delivery mission, UAV maneuvering decision-making could be regarded as a sequential decision processes. Moreover, the pilot of UAV usually considered current information from environment while selecting optimal control input. Thus, we could consider that this decision processes was Markovian and use MDPs to model the UAV maneuvering decision-making model for air-delivery.

The MDPs could be described by a tuple

$$\{T, S, A(s), P(\cdot,|s,a), R(s,a)\} \tag{7}$$

where T represented the decision episode, S represented the state space, $A(s)$ represented the action space, and the transition probability $P(\cdot,|s,a)$ represented the probability distribution of the environment at the next moment when the action $a \in A(s)$ was executed in the environment in the state $s \in S$. The reward function $R(s,a)$ represented the benefit that agent gets when $a \in A(s)$ was taken in the state $s \in S$. Then, we could make a complete mathematical description of the sequence decision problem based on MDPs.

As shown in Figure 6, MDPs could be described as follows: when the state of environment was initialized by s_0, the agent chose the action $a_0 = \pi(s_0)$, where $\pi(s)$ represented the policy of agent, and the state of environment would be updated to s_1 according to $P(s_1,|s_0,a_0)$. Meanwhile, the environment also returned the reward signal $r_0 = R(s_0,a_0)$ to agent. This process mentioned above would end until the state of environment became termination state.

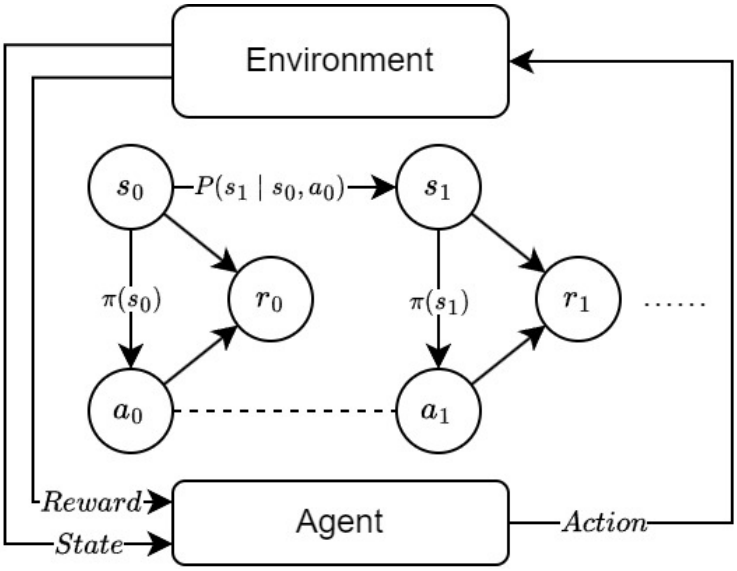


Figure 6 – The traditional structure of finite Markov decision processes.

During the process of interactions between environment and agent, a sequence of rewards $(r_0, r_1, \cdots)$ was produced. Among this process, agent was stimulated by external rewards, and the policy of agent would converge when the expectation of future rewards about policy was maximum. Therefore, the utility function (in the state $s \in S$, the expected rewards obtained by adopting the policy $\pi(s)$) is $v(s, \pi)$. When current policy was optimal, the 8 should be satisfied.

$$v(s) = \sup v(s, \pi), s \in S \tag{8}$$

Based on the characteristics of the UAV maneuvering decision-making problem for air-delivery, we used the infinite stage discount model as the utility function, as shown in 9.

$$v(s, \pi) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\mu}^{s} [R(s_t, a_t)], s \in S \tag{9}$$

In the above formula, $\gamma \in [0, 1]$ was the future reward discount factor, and $\mathbb{E}$ represented mathematical expectation. Then, the optimal policy under the discount model could be obtained by solving 8.

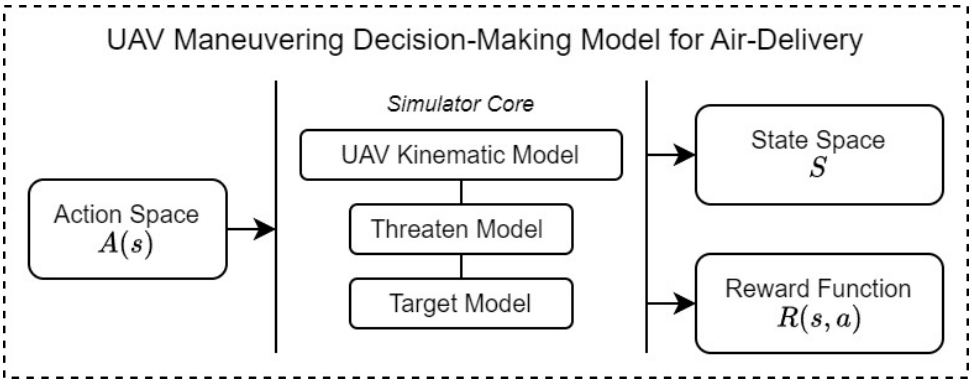


Figure 7 – The simulation structure of UAV maneuvering decision-making model for air-delivery.

In the following, we would construct the UAV Maneuvering Decision-making Model for Air-delivery based on the definitions of long-distance guidance task and near-end aiming task. As shown in Figure 7, we designed and realized the simulator core including UAV kinematic model, threaten model and target model, which would generate the samples for optimizing policy. In the environment implemented based on this structure, agent manipulated UAV to execute the long-distance guidance task and the near-end aiming task. On this basis, we would design state space S, action space $A(s)$, reward function $R(s, a)$ of each task.

2.1.3 The State Space, Action Space and Reward Function of Air-delivery

We designed the state space, action space and reward function of each tasks involved in air-delivery mission.

(1) The State Space of Long-distance Guidance Task

Considering the definition of long-distance guidance task, we defined the state space as

$$S_{\texttt{guidance}} = \{v_{\texttt{UAV}}, h_{\texttt{UAV}}, N_z^{\texttt{UAV}}, d_{\texttt{los}}^{\texttt{tgt}}, \phi_{\texttt{los}}^{\texttt{tgt}}, d_{\texttt{thr}}^{\texttt{front}}, \phi_{\texttt{thr}}^{\texttt{front}}, d_{\texttt{thr}}^{\texttt{left}}, \phi_{\texttt{thr}}^{\texttt{left}}, d_{\texttt{thr}}^{\texttt{right}}, \phi_{\texttt{thr}}^{\texttt{right}}\} \tag{10}$$

where $v_{\texttt{UAV}}$, $h_{\texttt{UAV}}$ and $N_z^{\texttt{UAV}}$ were UAV's speed, height and steering overload. $d_{\texttt{los}}^{\texttt{tgt}}$ and $\phi_{\texttt{los}}^{\texttt{tgt}}$ represented the distance and azimuth of required placement area relative to UAV. $d_{\texttt{thr}}^{\texttt{front}}$ and $\phi_{\texttt{thr}}^{\texttt{front}}$ indicated the distance and azimuth of threaten in front relative to UAV. $d_{\texttt{thr}}^{\texttt{left}}$ and $\phi_{\texttt{thr}}^{\texttt{left}}$ were the distance and azimuth of threaten on left relative to UAV. $d_{\texttt{thr}}^{\texttt{right}}$ and $\phi_{\texttt{thr}}^{\texttt{right}}$ were the distance and azimuth of threaten on right relative to UAV.

(2) The State Space of Near-end Aiming Task

Considering the definition of near-end aiming task, we defined the state space as

$$S_{\texttt{aiming}} = \{v_{\texttt{UAV}}, h_{\texttt{UAV}}, N_z^{\texttt{UAV}}, d_{\texttt{los}}^{\texttt{tgt}}, \phi_{\texttt{los}}^{\texttt{tgt}}, \theta_{\texttt{los}}^{\texttt{tgt}}, a_{\texttt{payload}}\} \tag{11}$$

where $\theta_{\texttt{los}}^{\texttt{tgt}}$ indicated the pitch of target relative to UAV. $a_{\texttt{payload}}$ represented the maximum horizontal range of payload in current situation.

(3) The Action Space of Each Task

Based on the simulator core of UAV maneuvering decision-making model, we could establish the action space of long-distance guidance task and near-end aiming task as

$$A(s) = \{N_z\} \tag{12}$$

where N_z indicated the steering overload of UAV.

(4) The Reward Function of Each Task

Moreover, in order to optimize UAV's policy, we defined the reward function as below, that considered the termination condition of each tasks.

$$R(s,a) = \begin{cases} 1.0, & \texttt{Successfully Termination} \\ -1.0, & \texttt{Failed Termination} \\ 0.0, & \texttt{Otherwise} \end{cases} \tag{13}$$

13 showed that if task terminated successfully, $R(s,a)$ would return 1.0. If task satisfied failed termination, $R(s,a)$ would return -1.0. Otherwise, $R(s,a)$ returned 0.0. Thereby, reward could encourage agent to find policy that maximizes the expectation of future rewards.

2.2 The UAV Maneuvering Decision-making Algorithm for Air-delivery based on DTRL

According to UAV maneuvering decision-making model for Air-delivery implemented above, we proposed the UAV maneuvering decision-making algorithm for air-delivery based on PER-TD3. Due to Markov property of long-distance guidance task and near-end aiming task, we designed the UAV maneuvering decision-making policy based on neural network and used TD3 [18] to optimize the policy network. Meanwhile, PER was used to generate training samples in order to speed up training process. Moreover, we introduced a novel shaping reward function under the guidance of expert experience so as to accelerate the convergence of policy. As shown in Figure 8, the algorithm we proposed was composed of PER, policy network, shaping reward function and training procedure.

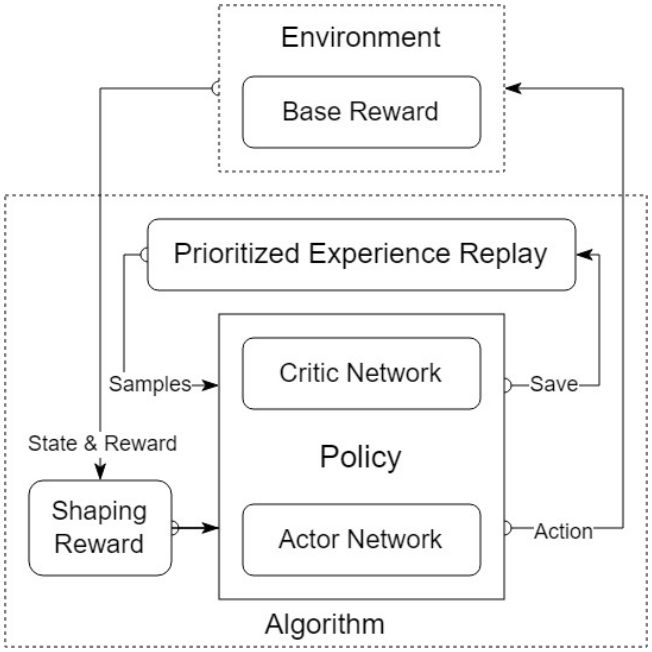


Figure 8 – The structure of UAV maneuvering decision-making algorthm for air-delivery.

2.2.1 The Framework of PER-TD3

PER-TD3 was a model-free, off-policy and DRL-based algorithm designed by Actor-Critic architecture. It was used to solved those problems having continuous state space and action space. Meanwhile, because TD3 didn't consider the diversity of data and doesn't fully utilize historical experience, policy converged in TD3 exhibit low convergence rate and poor stability. Therefore, PER was used to generate training data, which could improve the utilization of the potential value of historical data, thereby increasing convergence rate and enhancing the stability of trained policy.

As shown in Figure 9, in order to solve overestimation and suboptimal policy due to function approximation error, PER-TD3 was composed of clipped double q-learning (CDQ), delayed policy updating and target policy smoothing regularization.

Because of overestimation, Double Deep Q-Network used target network to reduce the influence for policy training. PER-TD3 used similar mechanism to eliminate overestimation, and proposed that two target critic networks Q_{θ_1}, $Q_{\theta_1'}$, Q_{θ_2} and , $Q_{\theta_2'}$ were used to restrict the upper bound of value estimation. Thus, the optimization target function of critic network was defined as

$$y = r + \gamma \min_{i=1,2} Q_{\theta_i'}(s', a) \tag{14}$$

Because of CDQ, the value estimation tended to low variance, which improved the robust of policy updating. Though CDQ could reduce overestimation bias, the basic problem, that was estimation variance, should be accomplished. Therefore, the policy network $\pi_\phi(s)$ was updated after critic networks were updated $N_{\texttt{delayed}}$ times, because the value error of critic network could be reduced by multiple training sessions. We could use 15 to update parameters of critic networks and actor networks.

$$\begin{cases} \theta_i' = \tau\theta_i + (1-\tau)\theta_i', i = 1,2 \\ \phi' = \tau\phi + (1-\tau)\phi' \end{cases} \tag{15}$$

$\tau \in (0,1)$ was a hyperparameter involved in the "Soft" updating. Moreover, due to deterministic policy, the peaks of value estimation was usually overfitted. Thus, target policy smoothing regularization was used to evaluate state-action pair, which was defined as

$$y = r + \gamma \min_{i=1,2} Q_{\theta_i'}(s', \widetilde{a}) \tag{16}$$

where $\widetilde{a}$ was exeploratory action attached minor noise, which was generated by

$$\widetilde{a} = \pi_{\phi'}(s) + \varepsilon, \varepsilon \sim clip(N(0, \widetilde{\sigma}), -c, c) \tag{17}$$

where $clip(x, x_{\min}, x_{\max})$ was used to limit x between $x_{\min}$ and $x_{\max}$. $N(0, \widetilde{\sigma})$ indicated normal distribution whose variance was $\widetilde{\sigma}^2$.

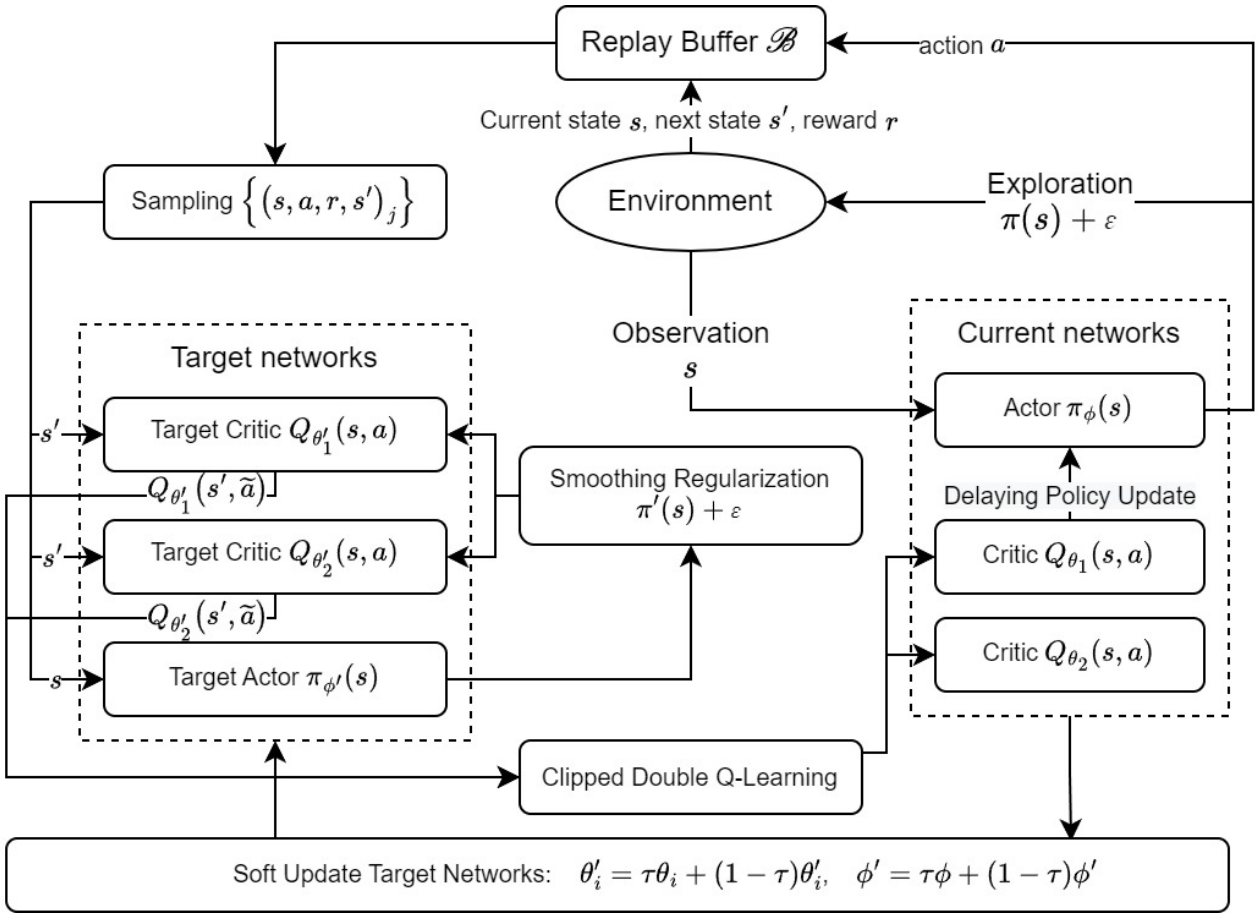


Figure 9 – The framework of PER-TD3.

2.2.2 Structure of UAV Maneuvering Decision-making Policy Network

As mentioned above, PER-TD3 consisted of two critic networks Q_{θ_1} and Q_{θ_2}, actor network π_ϕ and their target networks $Q_{\theta'_1}$, $Q_{\theta'_2}$ and $\pi_{\phi'}$. Therefore, we must design the structure of critic networks and actor network respectively.

(1) Structure of Actor Network

The actor network $\pi_\phi(s)$ was mainly used to output action in real-time according to state. The input of network was the environment state $s \in S$, and its output was action $a \in A(s)$. The structure of the designed actor network was shown in Figure 10.

Therefore, the input dimension of actor network was equal to the dimension of state space $\dim(S)$. The output dimension of actor network was equal to the dimension of action space $\dim(A(s))$.

(2) Structure of Critic Network

The critic network $Q_{\theta_i}(s,a)$ was used to evaluate the advantage of current action $a \in A(s)$ output by $\pi_\phi(s)$. The input of $Q_{\theta_i}(s,a)$ was the combination of state and action (s,a), and the output of $Q_{\theta_i}(s,a)$ was the state-action value function $Q(s,a)$. The structure of the designed critic network was shown in Figure 11.

Therefore, the input dimension of critic network was equal to the dimension of state space $\dim(S+A(s))$. The output dimension of actor network was equal to 1.

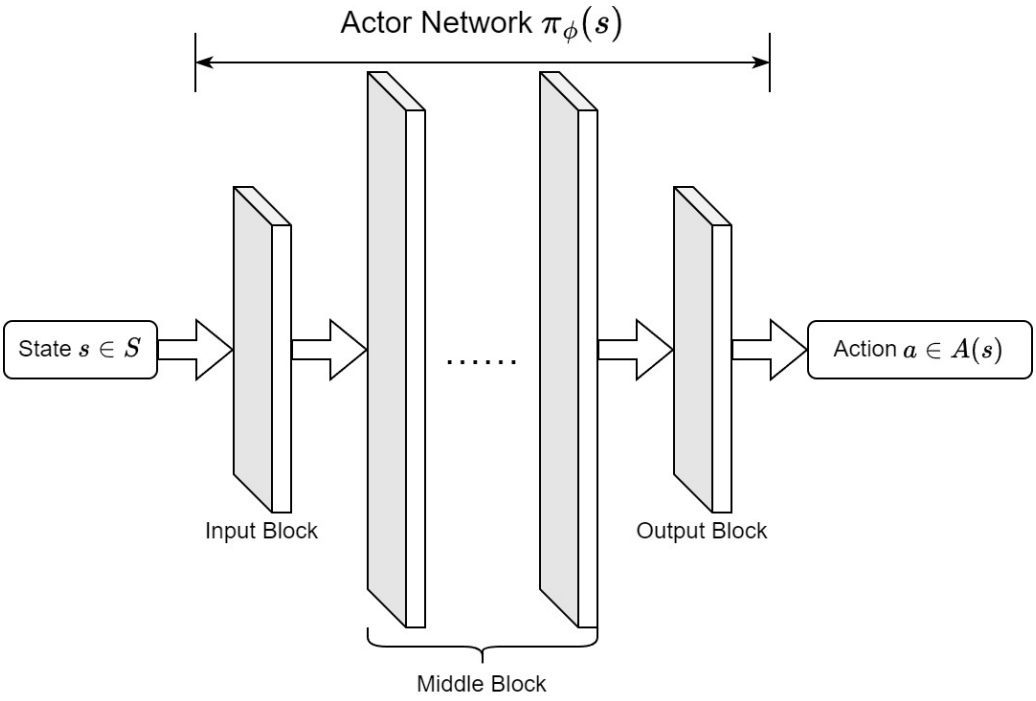


Figure 10 – The structure of actor network involved in PER-TD3.

In addition, before running these networks, the input value should be normalized for eliminating the influence of data's physical meaning. Moreover, the structure of target networks $Q_{\theta_i'}$ and $\pi_{\phi'}$ was similar to Q_{θ_i} and π_ϕ, and only the method of parameters updating was different.

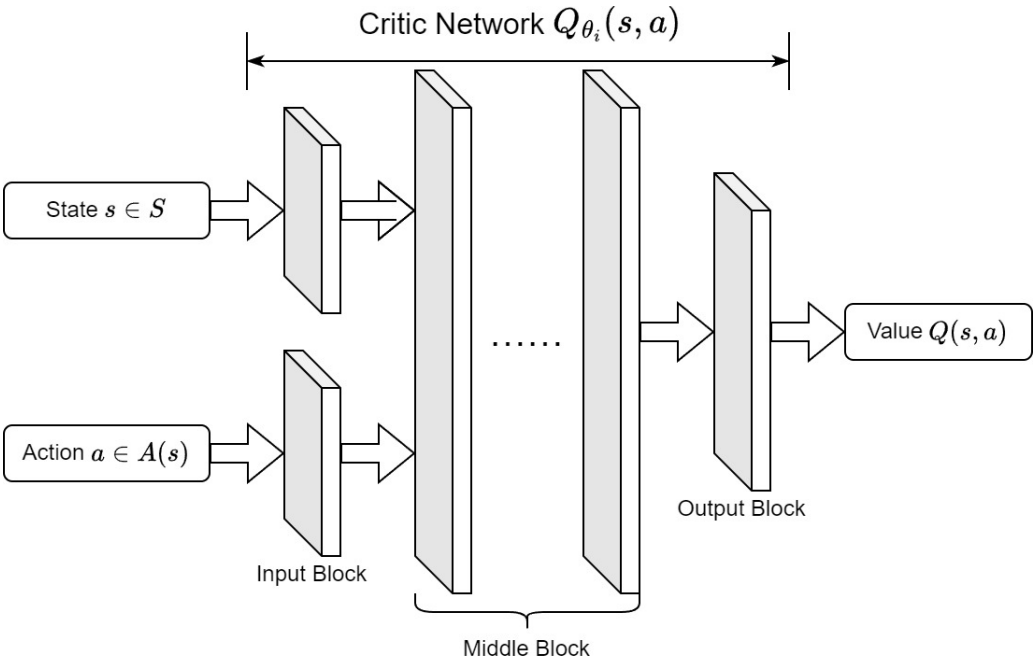


Figure 11 – The structure of critic network involved in PER-TD3.

2.2.3 Reward Shaping under The Guidance of Expert Experience

Although the algorithm we proposed could learn an optimal policy according to the reward function shown in 13, there was a serious challenge that could influence the convergence rate of policy, because the rewards environment returned were too sparse to learn useful knowledge such as those transitions whose reward was not zero.

Therefore, some researchers proposed a technique called reward shaping (RS) [19], which leveraged

the expert knowledge to reconstruct the reward model of the target domain to improve the agent's policy learning. More specifically, in addition to reward from environment, RS provided a shaping function $F : S \times S \times A \to \mathbb{R}$ to render auxiliary rewards, where $\mathbb{R}$ indicated the set of real number. Intuitively, RS would assign higher rewards to more beneficial transitions, which could guide the agent to find the optimal policy quickly. As a result, the agent would learn its policy by the newly shaped rewards $R' = R + F$, which meant that RS had altered the original reward with a different shaping reward function.

Along the line of RS, the potential-based RS (PBRS) [20] was one of the most classical approaches. PBRS gave algorithm an external signal for learning the optimal policy more quickly than before by adding a new reward shaping function $F(s,a,s')$, formed by the difference between two potential functions

$$F(s,a,s') = \gamma\Phi(s') - \Phi(s) \tag{18}$$

where $\Phi(s)$ came from the knowledge of expertise and evaluated the quality of a given state. Thereby, if we constructed a function over both the state and the action to form the potential function $\Phi(s,a)$, PBRS would be extended to a novel approach, called potential based state-action advice (PBA) [21], which could evaluate how beneficial an action is to take from state

$$F(s,a,s',a') = \gamma\Phi(s',a') - \Phi(s,a) \tag{19}$$

Therefore, we proposed a novel reward function of air-delivery mission based on PBRS and PBA, introducing domain knowledge and expert experience, as shown in Figure12.

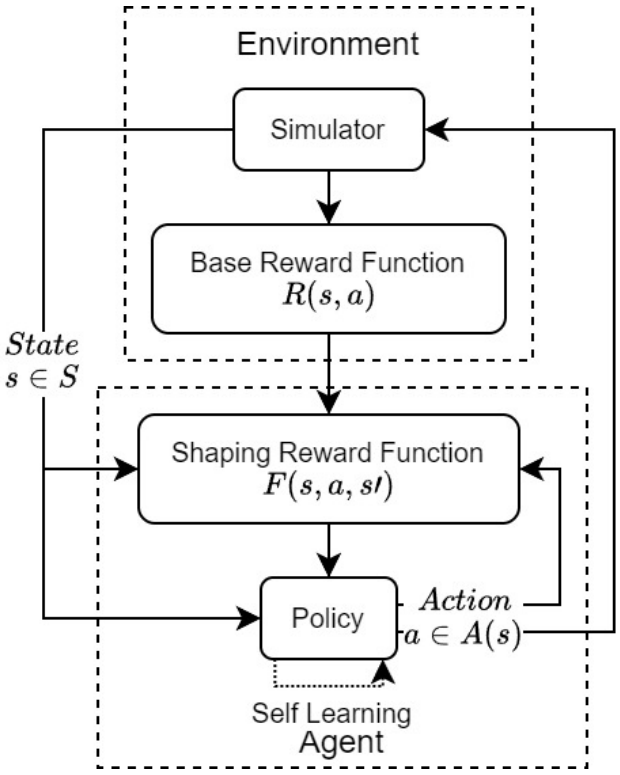


Figure 12 – The structure of reward shaping under the Guidance of Expert Experience.

(1) The Shaping Function of Long-distance Guidance Task

When UAV was performing long-distance guidance task, the distance and azimuth between UAV and required placement area, and the distance between observed threatens and UAV were the main influence factors. Therefore, we could construct the shaping function as

$$F_{\texttt{guidance}}(s,a,s') = \gamma\left[\Phi_{\texttt{d}}(s') + \Phi_{\phi}(s') + \sum_{i=0}^{2}\Phi^{i}_{\texttt{thr}}(s')\right] - \left[\Phi_{\texttt{d}}(s) + \Phi_{\phi}(s) + \sum_{i=0}^{2}\Phi^{i}_{\texttt{thr}}(s)\right] \tag{20}$$

where $\Phi_{\mathrm{d}}(s)$ represented the distance potential function, and $\Phi_{\phi}(s)$ represented the azimuth potential function. Considering the normalization of different influence factors, we designed $\Phi_{\mathrm{d}}(s)$ and $\Phi_{\phi}(s)$ as shown in 21 and 22.

$$\Phi_{\mathrm{d}}(s) = \frac{d_{\mathtt{los}}^{\mathtt{max}} - d_{\mathtt{los}}^{\mathtt{tgt}}}{d_{\mathtt{los}}^{\mathtt{max}} - d_{\mathtt{los}}^{\mathtt{min}}} \tag{21}$$

In 21, $d_{\mathtt{los}}^{\mathtt{max}}$ and $d_{\mathtt{los}}^{\mathtt{min}}$ indicated the maximum distance and the minimum between UAV and target respectively.

$$\Phi_{\phi}(s) = \frac{\pi - \phi_{\mathtt{los}}^{\mathtt{tgt}}}{\pi} \tag{22}$$

Moreover, $\Phi_{\mathtt{thr}}^{i}(s), i = 0,2,3$ represented threaten on left, in front and on right potential function, which was defined as

$$\Phi_{\mathtt{thr}}^{i}(s) = \begin{cases} 0, & \mathtt{If}\ \ d_{\mathtt{thr}}^{\cdot} > d_{\mathtt{sensor}}^{\mathtt{max}} \\ \dfrac{d_{\mathtt{sensor}}^{\mathtt{max}} - d_{\mathtt{thr}}^{\cdot}}{d_{\mathtt{sensor}}^{\mathtt{max}} - R_{\mathtt{thr}}^{\cdot}}, & \mathtt{If}\ \ d_{\mathtt{thr}}^{\cdot} > R_{\mathtt{thr}} \end{cases} \tag{23}$$

where $d_{\mathtt{thr}}^{\cdot}$ indicated the distance between UAV and threaten and $R_{\mathtt{thr}}$ represented the influence radius of threaten.

In order to accelerate the convergence of policy, we proposed a event-based burst reward under the guidance of expert experience based on PBA. In daily life, if we want to train dog to teach it some skills, we will give it some awards when it makes our desirable response. Gradually, when we want it to finish some work, we could give it some commands through changing information that it recives. Therefore, we could introduce expert experience to navigate agent to reach optimal state faster, as shown in 24.

$$F(s,a,s',a') = \gamma\Phi(s',a') - R^{\dagger} \tag{24}$$

$R^{\dagger}$ represented the advice reward coming from the expert experience, which could be defined by

$$R^{\dagger} = \begin{cases} 1.0, & \mathtt{If}\ \ d_{\mathtt{los}}^{\mathtt{t-1}} \geq d_{\mathtt{los}}^{\mathtt{t}} \\ 1.0, & \mathtt{If}\ \ N_{\mathtt{thr}}^{\mathtt{t-1}} \geq N_{\mathtt{thr}}^{\mathtt{t}} \\ 0.0, & \mathtt{Otherwise} \end{cases} \tag{25}$$

where $d_{\mathtt{los}}^{\mathtt{t-1}}$ and $d_{\mathtt{los}}^{\mathtt{t}}$ represented the distance between UAV and target at the last moment and present moment. $N_{\mathtt{thr}}^{\mathtt{t-1}}$ and $N_{\mathtt{thr}}^{\mathtt{t}}$ represented the number of observed threatens at the previous moment and the present moment. 25 meant that agent could receive positive motivation when UAV got close to target and avoided threatens successfully.

(2) The Shaping Function of Near-end Aiming Task

After UAV entered the near-end aiming task, the impact error and the azimuth between target and UAV were the main influence factors. Therefore, we can construct the shaping function as

$$F_{\mathtt{aiming}}(s,a,s') = \gamma\left[\Phi_{\mathtt{ie}}(s') + \Phi_{\phi}(s')\right] - \left[\Phi_{\mathtt{ie}}(s) + \Phi_{\phi}(s)\right] \tag{26}$$

where $\Phi_{\mathtt{ie}}(s)$ represented the impact error potential function, which was defined as

$$\Phi_{\mathrm{d}}(s) = \frac{d_{\mathtt{impact}}^{\mathtt{max}} - d_{\mathtt{impact}}}{d_{\mathtt{impact}}^{\mathtt{max}}} \tag{27}$$

In 27, $d_{\mathtt{impact}}^{\mathtt{max}}$ indicated the maximum impact error and $d_{\mathtt{impact}}$ indicated the impact error in current situation.

Similar to long-distance guidance task, we defined the advice reward of near-end aiming task as

$$R^{\dagger} = \begin{cases} 1.0, & \mathtt{If}\ \ d_{\mathtt{impact}}^{\mathtt{t-1}} \geq d_{\mathtt{impact}}^{\mathtt{t}} \\ 0.0, & \mathtt{Otherwise} \end{cases} \tag{28}$$

where $d_{\mathtt{impact}}^{\mathtt{t-1}}$ and $d_{\mathtt{impact}}^{\mathtt{t}}$ indicated the impact error of payload at the last moment and present moment. 28 meant that agent would receive award when the impact error of payload was reduced.

2.2.4 Training Procedure of UAV Maneuvering Decision-making Policy

As we mentioned above, we could design the training procedure of UAV maneuvering decision-making policy. In addition, we should construct experience replay method based on PER. The experience replay memory D was defined as

$$D = \{s, a, s', r\} \tag{29}$$

where s was current state, a was action selected by agent, s' was state at the next moment, and r was reward agent received. Traditionally, experience replay method selected training data from D uniformly, which meant the probability $P(i)$ of each sample to be select was equal. On the contrary, $P(i)$ in PER was not same, as defined in 30.

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha} \tag{30}$$

p_i was the priority of i-th sample in D, and α was a hyperparameter. When $\alpha = 0$, it was pure uniformly experience replay method. p_i was defined based on TD-Error, as shown in 31.

$$p_i = |\delta_i| + \varepsilon \tag{31}$$

δ_i was TD-error of i-th sample in D. In addition, a minimum ε was introduced to prevent p_i from being zero. δ_i was defined as

$$\delta_i = y_i - \min_{j=1,2} Q_{\theta_j}(s, a) \tag{32}$$

where y_i was the optimization target of critic, which was defined in 16. In order to reduce the distribution bias of transitions sampled by PER, importance sampling (IS) weights were used to correct the distribution bias of training data caused by PER. The IS weight ω_j could be calculated by

$$\omega_j = \left(\frac{1}{N} \cdot \frac{1}{P(j)}\right)^\beta \tag{33}$$

where N was the size of D. When $\beta = 1$, the distribution error of training set was fully compensated. When δ_j was calculated, the actual updating target was $\omega_j \cdot \delta_j$. Therefore, the gradient of critic network was

$$\Delta = \sum_j \omega_j \cdot \delta_j \cdot \nabla_{\theta_i} Q_{\theta_i}(s_j, a_j), i = 1, 2 \tag{34}$$

In order to ensure the stable convergence of $Q_{\theta_i}(s_j, a_j)$, ω_j was normalized by $\frac{\omega_j}{\max_i \omega_i}$. Thereby, the real IS weight ω_j could be defined as

$$\omega_j = \left(\frac{\min_i P(i)}{P(j)}\right)^\beta \tag{35}$$

Moreover, so as to improve the exploration ability of deterministic policy, we added random noise sampled from $N(0, \sigma)$ into the action output by $\pi_\phi(s)$, as shown in 36.

$$a_t = \pi_\phi(s_t) + N(0, \sigma) \tag{36}$$

To sum up, we could design the training procedure of UAV maneuvering decision-making algorithm for air-delivery, as shown in Algorithm 1.

Algorithm 1 The Training Procedure of UAV Maneuvering Decision-Making Algorithm for Air-delivery.

Input:

The hyperparameters of training networks: the size of minibatch k, networks' learning rate η;
The hyperparameters of updating policy: learning period K, memory capacity N, network parameters updating episode τ, actor updating episode $N_{\texttt{delayed}}$;
The hyperparameters of sampling: the availability exponent of PER α, IS exponent β;
The control parameters of simulation: maximum period M, maximum step per period T.

Output:

Critic network $Q_{\theta_i}(s,a), i=1,2$ and their target network $Q'_{\theta_i}(s,a), i=1,2$;
Actor network $\pi_\phi(s)$ and its target network $\pi'_\phi(s)$.

Initialize $Q_{\theta_i}(s,a), i=1,2$, $\pi_\phi(s)$ and their target $Q'_{\theta_i}(s,a), i=1,2$, $\pi'_\phi(s)$.
for $m=1$ to M **do**
  Reset environment and read the initial state s_0.
  Output a_0 according to equation (36).
  for $t=1$ to T **do**
    Observe current state s_t and reward r_t of environment and calculate current action a_t according to equation (36).
    Save current transition (s_t, a_t, s_{t+1}, r_t) into experiences memory D.
    if $t \bmod K \equiv 0$ **then**
      Reset the gradient $\Delta = 0$ of critic networks with IS.
      for $j=0$ to k **do**
        Sample traing data $j \sim P(j)$ according to equation (30)
        Calculate IS weight ω_j according to equation (35)
        Calculate TD-error δ_j of training data according to equation (34) and update its priority according to equation (31)
        Accumulate Δ.
      end for
      Update the parameters of $Q_{\theta_i}(s,a), i=1,2$ according to Δ with learning rate η.
      Update the parameters of target networks $Q'_{\theta_i}(s,a), i=1,2$ according to equation (15)
      if $t \bmod N_{\texttt{delayed}}$ **then**
        Update the parameters of $\pi_\phi(s)$ according to target function $\max\left[\min_{i=1,2} Q_{\theta_i}(s,a)\right]$.
        Update the parameters of target networks $\pi'_\phi(s)$ according to equation (15)
      end if
    end if
  end for
end for

3. Results and Analysis

Based on the model and algorithm we presented above, experiments were conducted to prove the rationality of the model and verify the availability of the algorithm. In the following, we will explain settings of the simulation experiments, details of the training process, results of Monte-Carlo (MC) test experiments, as well as their analysis.

3.1 The Settings of Simulation Experiments

In the simulation experiments, the mission area was restricted to $100\text{km} \times 100\text{km}$ airspace and height of UAV was bound to $[500\text{m}, 10000\text{m}]$. For each simulation experiment, the UAV's initial state and threatens' position was randomly generated. In order to make simulation data as close to real world as possible, we set simulation step to 0.5s because operator of UAV always manipulates it every $[0.5\text{s}, 1.0\text{s}]$.

According to the training procedure of algorithm, before we started to train, some hyperparameters should be assigned. Hyperparameters assignments of the algorithm are shown in Table 1. Moreover, we designed the structure of critic networks and actor networks in each tasks, which were shown in Table 2, Table 3, Table 4 and Table 5 respectively. In this paper, the networks were all designed to be

Table 1 – The parameters assignment of algorithm for air-delivery mission.

Parameter	Value	Meaning
K	100	policy's learning period
N	100000	historical buffer capacity
τ	0.01	soft updating parameter
$N_{\texttt{delayed}}$	10	delayed updating period
k	128	size of minibatch
η	0.001	networks' learning rate
α	0.5	availability exponent of PER
β_0	0.4	initial IS exponent
M	1000	maximum training episodes
T	5000	maximum steps per episode

Table 2 – The structure of critic network $Q_{\theta_i}(s,a), i = 1,2$ for the long-distance guidance task.

Layers	Layer Structure	
	Units	Activation Function
Input layer of state	128	ReLU
Input layer of action	128	ReLU
Hidden layer 1	256	ReLU
Hidden layer 2	256	ReLU
Hidden layer 3	256	ReLU
Hidden layer 4	256	ReLU
Hidden layer 5	256	ReLU
Hidden layer 6	256	ReLU
Hidden layer 7	256	ReLU
Hidden layer 8	256	ReLU
Output layer	1	-

Table 3 – The structure of actor network $\pi_\phi(s)$ for the long-distance guidance task.

Layers	Layer Structure	
	Units	Activation Function
Input layer	64	tanh
Hidden layer 1	256	tanh
Hidden layer 2	256	tanh
Hidden layer 3	256	tanh
Hidden layer 4	256	tanh
Hidden layer 5	256	tanh
Hidden layer 6	256	tanh
Hidden layer 7	256	tanh
Hidden layer 8	256	tanh
Output layer	1	tanh

dense network.

3.2 The Simulation Results and Analysis of Air-delivery Mission

After effective training, we collected some parameters generated during the process of training critic and actor. In Figure 13 and Figure 14, we plotted the loss curve of critic and actor, the successful rate curve in long-distance guidance task by PER-TD3 with base reward and advice reward. In Figure 15

Table 4 – The structure of critic network $Q_{\theta_i}(s,a), i = 1,2$ for the near-end aiming task.

Layers	Layer Structure	
	Units	Activation Function
Input layer of state	128	ReLU
Input layer of action	128	ReLU
Hidden layer 1	256	ReLU
Hidden layer 2	256	ReLU
Hidden layer 3	256	ReLU
Hidden layer 4	256	ReLU
Output layer	1	-

Table 5 – The structure of actor network $\pi_\phi(s)$ for the near-end aiming task.

Layers	Layer Structure	
	Units	Activation Function
Input layer	64	tanh
Hidden layer 1	256	tanh
Hidden layer 2	256	tanh
Hidden layer 3	256	tanh
Hidden layer 4	256	tanh
Output layer	1	tanh

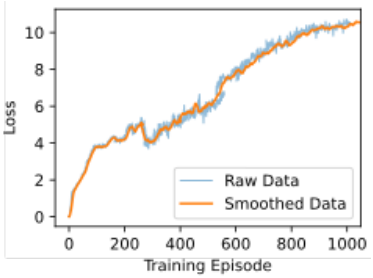
and Figure 16, we plotted the loss curve of critic and actor, the successful rate curve in near-end aiming task by PER-TD3 with base reward and advice reward.

In Figure 13a, Figure 13b, Figure 14a, Figure 14b, Figure 15a, Figure 15b, Figure 16a and Figure 16b, the loss of actor and critic involved in PER-TD3 with base reward and advice reward for long-distance guidance task and near-end aiming task were plotted. We could find that the curve of actor network's loss climbs gradually over time and is stable after enough training in long-distance guidance task. The curve of actor network's loss declines gradually over time and is stable after enough training in near-end aiming task. Meanwhile, the loss of critic network decreases gradually over time, and finally become stable to a small amount. Moreover, we could find that the loss of critic by PER-TD3 with advice reward is stabler than those by PER-TD3 with base reward. We think that those meaningful samples are used to improve policy by adding advice reward which could be regarded as special features.

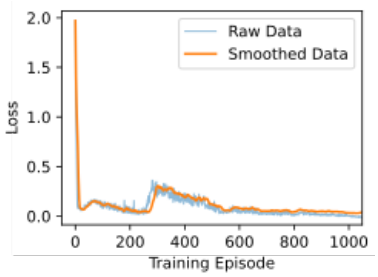
In addtion, Figure 13c, Figure 14c, Figure 15c and Figure 16c are the successful rate over time in long-distance guidance task and near-end aiming task by PER-TD3 with base reward and advice reward, respectively. We can find that the successful rate is approaching 100% after a period of time. The successful rate indicates the ratio of successful count to total count in the last 100 interactive periods. In an interactive period, UAV starts from random position and ends while satisfying termination conditions.

Furthermore, we performed a group of Monte-Carlo (MC) experiments to evaluate the quality of trained results of the algorithms. Some results from MC experiments were visualized, including the flight trajectory of UAV in air-delivery mission, involving long-distance guidance task and near-end aiming task, as shown in Figure 17 and Figure 18, which are the flight trajectory of UAV trained by algorithm with base reward and advice reward respectively. In these figures, the red solid line represents the flight trajectory of UAV, the red point and the green "x" indicate the start position and the end position of UAV respectively, the blue "x" indicates the target position, the red dashed line represents the trajectory of payload, the green dash-dot line represents the observed radius of threaten, the green dashed line represents the influence radius of threaten, and the green "x" represents the position of threaten.

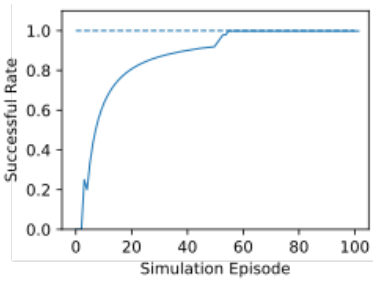
Among all the results of Figure 17 and Figure 18, we can see that UAV could avoid threatens au-



(a) The Loss Curve of Actor Network.

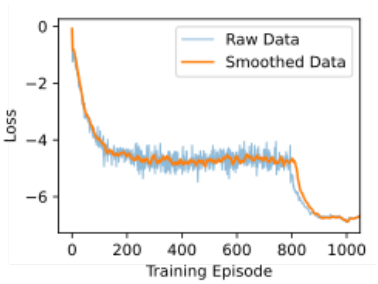


(b) The Loss Curve of Critic Network.

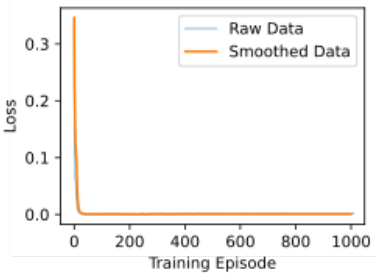


(c) The Successful Rate Curve.

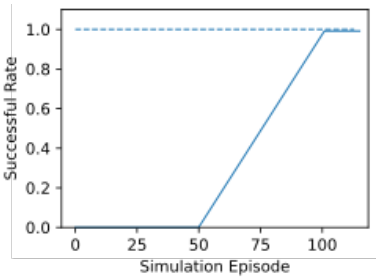
Figure 13 – The Evaluation Curve of Training Experiment for Long-distance Guidance Task by PER-TD3 with Base Reward.



(a) The Loss Curve of Actor Network.

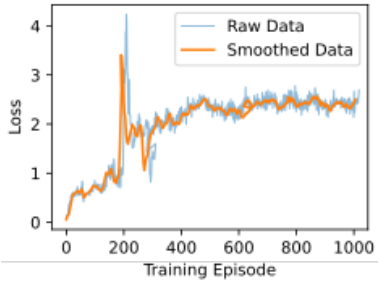


(b) The Loss Curve of Critic Network.

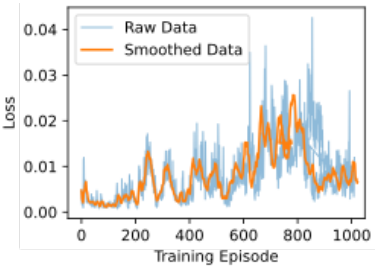


(c) The Successful Rate Curve.

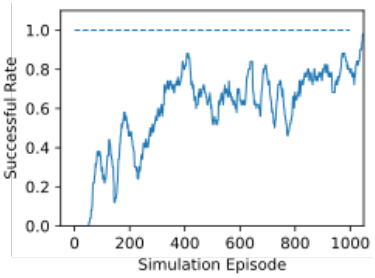
Figure 14 – The Evaluation Curve of Training Experiment for Long-distance Guidance Task by PER-TD3 with Advice Reward.



(a) The Loss Curve of Actor Network.

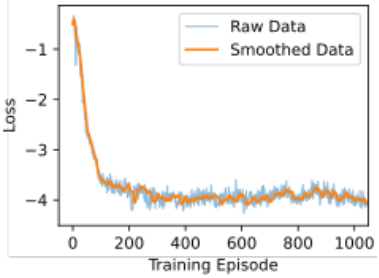


(b) The Loss Curve of Critic Network.

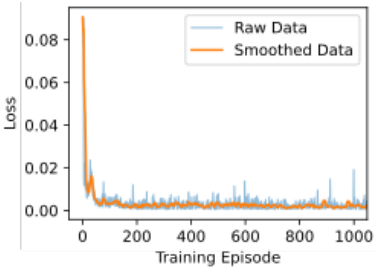


(c) The Successful Rate Curve.

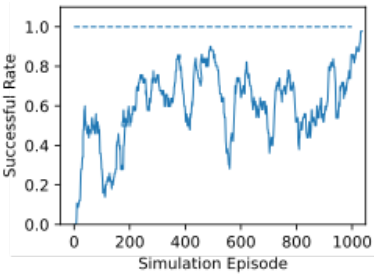
Figure 15 – The Evaluation Curve of Training Experiment for Near-End Aiming Task by PER-TD3 with Base Reward.



(a) The Loss Curve of Actor Network.

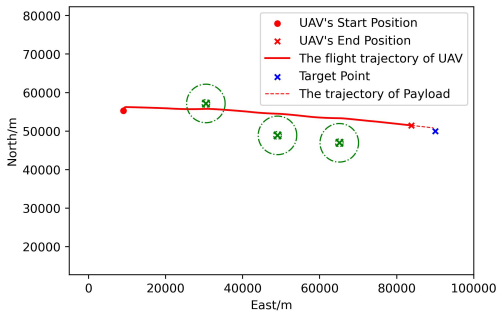


(b) The Loss Curve of Critic Network.

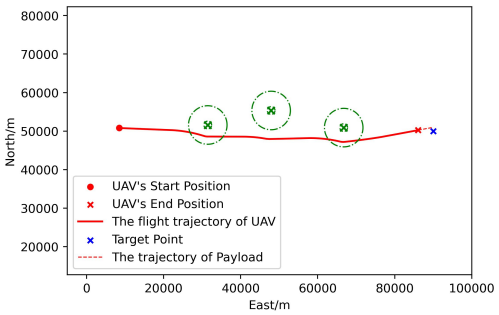


(c) The Successful Rate Curve.

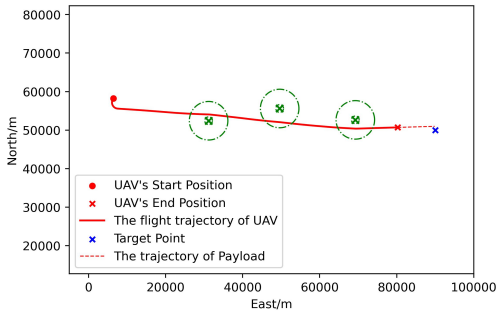
Figure 16 – The Evaluation Curve of Training Experiment for Near-End Aiming Task by PER-TD3 with Advice Reward.



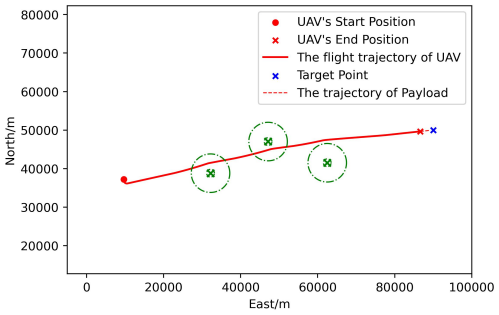
(a) NO.1 test experiment.



(b) NO.3 test experiment.

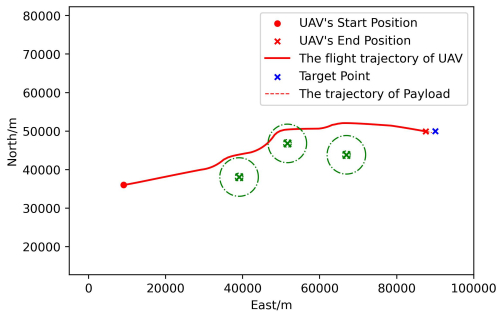


(c) NO.4 test experiment.

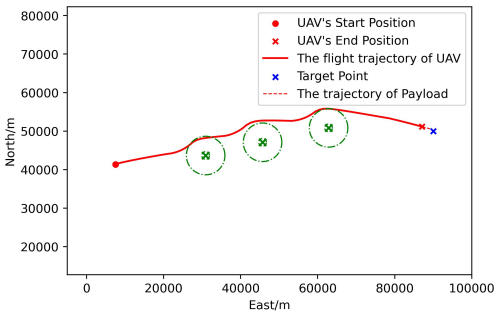


(d) NO.5 test experiment.

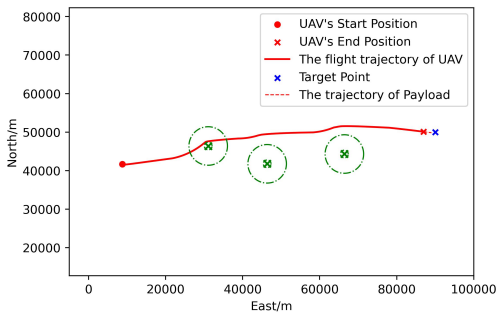
Figure 17 – The flight trajectory of UAV controled by agent trained by base reward while executing air-delivery mission.



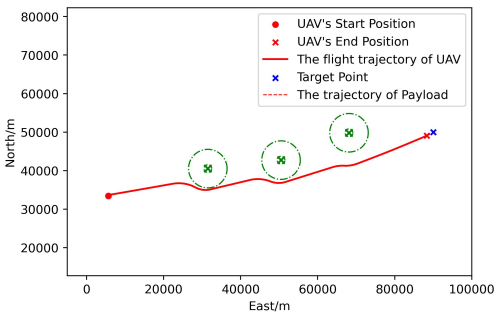
(a) NO.1 test experiment.



(b) NO.3 test experiment.



(c) NO.5 test experiment.



(d) NO.5 test experiment.

Figure 18 – The flight trajectory of UAV controled by agent trained by advice reward while executing air-delivery mission.

tonomously and fly towards target point from arbitrary position and random azimuth. And UAV could adjust its attitude automatically to aim required palcement area. These results show that the trained policy of each algorithm with base reward and advice reward in air-delivery mission has converged to near optimal performance.

4. Conclusions

In the present work, we refined and described the long-distance guidance task and the near-end aiming task from air-delivery mission. According to the definitions of tasks, we proposed the UAV maneuvering decision-making algorithm based on DTRL to execute air-delivery mission autonomously. Among this work, we designed and constructed the UAV maneuvering decision-making model based on MDPs, consisting of state space, action space and reward function of each task. Specifically, we introduce expert experience to modify reward function, where we construct shaping reward function based on PBRS and PBA to generate base reward and advice reward. This novel construction method of shaping function take domain knowledge and expert advice into account to improve the inference quality of trained policy network.

Furthermore, we designed extensive experiments to verify the performance of proposed algorithms and model. We used base reward and advice reward to train policies repectively and compared their loss curve and successful rate during the process of convergence. Among these simulation results, we can see that advice reward could help algorithm converge more stably than base reward. Moreover, we made lots of MC experiments. Simulation results demonstrate that the algorithm we proposed could obtain effective policy to execute the long-distance guidance task and the near-end aiming task.

In the future, we will consider the influence of information dimension loss, which means environment is partially observed. And we will extend the proposed algorithm to manipulate real UAVs in a 3D environment while performing special missions.

5. Funding

This work was funded by the Key Research and Development Program of Shaanxi (2022GXLH-02-09), the Fundamental Research Funds for the Central Universities (D5000230311, HYGJZN202301).

6. Contact Author Email Address

kunzhang@nwpu.edu.cn

7. Copyright Statement

References

[1] Menouar, H.; Guvenc, I.; Akkaya, K.; Uluagac, A.S.; Kadri, A.; Tuncer, A. UAV-enabled intelligent transportation systems for the smart city: Applications and challenges. *IEEE Communications Magazine* **2017**, *55*, 22–28.

[2] Mathisen, S.G.; Leira, F.S.; Helgesen, H.H.; Gryte, K.; Johansen, T.A. Autonomous ballistic airdrop of objects from a small fixed-wing unmanned aerial vehicle. *Autonomous Robots* **2020**, *44*, 859–875.

[3] Klinkmueller, K.; Wieck, A.; Holt, J.; Valentine, A.; Bluman, J.E.; Kopeikin, A.; Prosser, E. Airborne delivery of unmanned aerial vehicles via joint precision airdrop systems. AIAA Scitech 2019 Forum, 2019, p. 2285.

[4] Yang, L.; Qi, J.; Xiao, J.; Yong, X. A literature review of UAV 3D path planning. Proceeding of the 11th World Congress on Intelligent Control and Automation. IEEE, 2014, pp. 2376–2381.

[5] Huang, S.; Teo, R.S.H. Computationally efficient visibility graph-based generation of 3D shortest collision-free path among polyhedral obstacles for unmanned aerial vehicles. 2019 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, 2019, pp. 1218–1223.

[6] Cheng, X.; Zhou, D.; Zhang, R. New method for UAV online path planning. 2013 IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC 2013). IEEE, 2013, pp. 1-5.

[7] Sun, Q.; Li, M.; Wang, T.; Zhao, C. UAV path planning based on improved rapidly-exploring random tree. 2018 Chinese Control and Decision Conference (CCDC). IEEE, 2018, pp. 6420–6424.

[8] Yan, F.; Liu, Y.S.; Xiao, J.Z. Path planning in complex 3D environments using a probabilistic roadmap method. *International Journal of Automation and computing* **2013**, *10*, 525–533.

[9] Tseng, F.H.; Liang, T.T.; Lee, C.H.; Der Chou, L.; Chao, H.C. A star search algorithm for civil UAV path planning with 3G communication. 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing. IEEE, 2014, pp. 942–945.

[10] Meng, B.b.; Gao, X. UAV path planning based on bidirectional sparse A* search algorithm. 2010 International Conference on Intelligent Computation Technology and Automation. IEEE, 2010, Vol. 3, pp. 1106–1109.

[11] Zhang, Z.; Wu, J.; Dai, J.; He, C. A novel real-time penetration path planning algorithm for stealth UAV in 3D complex dynamic environment. *IEEE Access* **2020**, *8*, 122757–122771.

[12] Silva Arantes, J.d.; Silva Arantes, M.d.; Motta Toledo, C.F.; Junior, O.T.; Williams, B.C. Heuristic and genetic algorithm approaches for UAV path planning under critical situation. *International Journal on Artificial Intelligence Tools* **2017**, *26*, 1760008.

[13] Kaminer, I.; Pascoal, A.; Hallberg, E.; Silvestre, C. Trajectory tracking for autonomous vehicles: An integrated approach to guidance and control. *Journal of Guidance, Control, and Dynamics* **1998**, *21*, 29–38.

[14] Lee, H.; Kim, H.J. Trajectory tracking control of multirotors from modelling to experiments: A survey. *International Journal of Control, Automation and Systems* **2017**, *15*, 281–292.

[15] Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; others. Human-level control through deep reinforcement learning. *nature* **2015**, *518*, 529–533.

[16] Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971* **2015**.

[17] Schaul, T.; Quan, J.; Antonoglou, I.; Silver, D. Prioritized experience replay. *arXiv preprint arXiv:1511.05952* **2015**.

[18] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[19] Babak Badnava, Mona Esmaeili, Nasser Mozayani, and Payman Zarkesh-Ha. A new potential-based reward shaping for reinforcement learning agent. In *2023 IEEE 13th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 01–06. IEEE, 2023.

[20] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

[21] Eric Wiewiora, Garrison W Cottrell, and Charles Elkan. Principled methods for advising reinforcement learning agents. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 792–799, 2003.