Hindawi
International Journal of Aerospace Engineering
Volume 2022, Article ID 3139610, 14 pages
https://doi.org/10.1155/2022/3139610

Research Article

A UAV Pursuit-Evasion Strategy Based on DDPG and Imitation Learning

Xiaowei Fu,[1] **Jindong Zhu**,[1,2] **Zhaoying Wei**,[3] **Hui Wang**,[1] **and Sili Li**[1]

[1]*Northwestern Polytechnical University, Xi'an 710129, China*
[2]*AVIC Shenyang Aircraft Design& Research Institute, Shenyang 110035, China*
[3]*Xi'an Shiyou University, Xi'an 710065, China*

Correspondence should be addressed to Xiaowei Fu; fxw@nwpu.edu.cn

Received 4 November 2021; Accepted 19 April 2022; Published 9 May 2022

Academic Editor: Xingling Shao

The UAV pursuit-evasion strategy based on Deep Deterministic Policy Gradient (DDPG) algorithm is a current research hotspot. However, this algorithm has the defect of low efficiency in sample exploration. To solve this problem, this paper uses the imitation learning (IL) to improve the DDPG exploration strategy. A kind of quasiproportional guidance control law is designed to generate effective learning samples, which are used as the data of the initial experience pool of DDPG algorithm. The UAV pursuit-evasion strategy based on DDPG and imitation learning (IL-DDPG) is proposed, and the algorithm obtains the data from the experience pool for experience playback learning, which improves the exploration efficiency of the algorithm in the initial stage of training and avoids the problem of too many useless exploration in the training process. The simulation results show that the trained pursuit-UAV can flexibly adjust the flight speed and flight attitude to pursuit the evasion-UAV quickly. It also verifies that the improved DDPG algorithm is more effective than the basic DDPG algorithm to improve the training efficiency.

1. Introduction

At present, UAVs are more and more widely used, such as sensor networks [1], data security [2], smart network systems [3], intelligent transportation systems [4], automatic identification systems [5], target encirclement control [6], and pursuit-evasion confrontation [7]. The UAV pursuit-evasion confrontation is the game between two drones with conflicts of interest. The pursuit-UAV tries to capture the evasion-UAV through the pursuit maneuver strategy, and the evasion-UAV tries to escape by evasion maneuver strategy.

The methods on the UAV pursuit-evasion strategy include differential game method [8], expert system method [9], and influence diagram decision method [10]. However, the common problem of these methods is that it is more difficult to obtain analytical solutions. The DDPG algorithm is a policy-based reinforcement learning (RL) method which can use neural network (NN) for end-to-end learning. Research on the pursuit-evasion strategy based on the DDPG algorithm [11] is a current research hotspot.

Based on the DDPG algorithm, Zhang et al. [12] studied the cooperative pursuit of incoming targets by UAV swarm and designed a guided return function for specific pursuit tasks. Song et al. [13] designed a reward function considering the tracking error and trajectory stability for the landing trajectory tracking control problem of UAVs, then proposed a trajectory tracking control method based on DDPG algorithm. The trained result has higher accuracy than the traditional PID control method.

A problem of RL is that the efficiency of sample exploration is low, which makes learning and training inefficient. In the early training stage of reinforcement learning, a relatively large random noise is set for the exploration strategy to improve the exploration ability. But it will also produce a lot of inefficient samples (that is, useless action exploration), resulting in small rewards at the initial training stage. Therefore, how to improve the exploration ability, obtain efficient samples, and improve the utilization rate of samples is an urgent problem to be solved for RL training.

Expert experience and the mixed decision-making technology have been used to accelerate the training process of reinforcement learning. Wang [14] used reinforcement learning algorithm based on expert knowledge to solve the UAV path planning problem. The algorithm used multiple tasks with known environmental parameters to train the UAV and then transferred the trained result knowledge to the training of new tasks to accelerate the training process. Wu [15] studied the UAV reactive obstacle avoidance algorithm based on transfer learning and deep reinforcement learning, which makes the UAV quickly and efficiently respond to unfamiliar scenarios. In order to improve the motor skills of the manipulator and the learning ability of unmanned driving, Lu [16] and Zuo [17] integrated the experience of experts in their respective fields into reinforcement learning algorithm and designed the reinforcement learning algorithm under different tasks. Mu [18] studied the UAV cooperative formation maintenance and collision avoidance method based on the fusion of model knowledge and data training. The switching system based on the consensus theory and the multiagent cooperative collision avoidance method was learned in advance before training, which improves the training efficiency of the UAV formation control method.

Inspired by these works, the UAV pursuit-evasion strategy based on DDPG and imitation learning (IL-DDPG) is proposed, the algorithm can avoid excessive useless exploration and converge more quickly. The main contributions of this paper are as follows:

(1) A kind of quasiproportional guidance control law is designed for the instructor to realize effective pursuit. The control law can be used to generate effective learning samples for the pretraining of IL-DDPG algorithm

(2) The exploration strategy of the DDPG algorithm is improved. In the pretraining stage, the instructor maneuver samples generated by the quasiproportional guidance control law are used as the data of the initial experience pool. The algorithm obtains the data from the experience pool for experience playback learning, which improves the exploration efficiency of the algorithm in the initial stage of training and avoids the problem of too many useless exploration in the training process

The rest of this paper is organized as follows. In Section 2, the system model and problem statement are presented. UAV pursuit strategy based on DDPG is presented in Section 3. In Section 4, UAV pursuit strategy based on IL-DDPG is proposed. Then, Section 5 provides the experimental results. Conclusions are given in Section 6.

2. Problem Description and Modeling

2.1. The Pursuit-Evasion Problem of UAV. In the pursuit-evasion problem, the pursuit-UAV must chase and capture the evasion-UAV, and the evasion-UAV must escape and stay away from the pursuit-UAV.

For this problem, a zero-sum differential game model with control constraints is established. The geometric model of pursuit-evasion is shown in Figure 1.

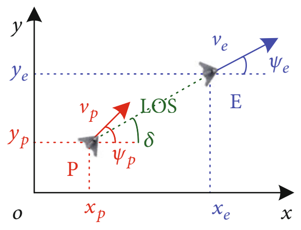


Figure 1: Geometric model of two-dimensional plane pursuit-evasion game.

In Figure 1, P represents the pursuit-UAV, E represents the evasion-UAV, v_p is the speed of the pursuit-UAV, v_e is the speed of the evasion-UAV, ψ_p is the heading angle of the pursuit-UAV, ψ_e is the heading angle of the evasion-UAV, and δ is the angle of the Line of Sight (LOS); LOS refers to the ray of the pursuit-UAV P pointing to the evasion-UAV E. The goal of the pursuit-UAV is to capture the target in the shortest time. The goal of the evasion-UAV is to stay away from the pursuit-UAV and to avoid being captured in the preset time or to maximize the delay time of being captured. The standard differential game is described as (1) and (2) [19].

$$\min T_c = f\left(v_p, \psi_p, v_e, \psi_e, L\right), \tag{1}$$

$$\max T_c = g\left(v_p, \psi_p, v_e, \psi_e, L\right), \tag{2}$$

where L is the distance between the two UAVs and T_c is the moment when the pursuit-UAV P captures the evasion-UAV E. Equation (1) is the objective function of the pursuit-UAV, and (2) is the objective function of the evasion-UAV.

2.2. The Kinematic Model of UAV. The motion state equations of the UAVs are defined as

$$\begin{cases} \dot{x}_i = v_i \cos \psi_i \\ \dot{y}_i = v_i \sin \psi_i \\ \dot{v}_i = a_i \\ \dot{\psi}_i = \omega_i \end{cases} \quad (i = p, e), \tag{3}$$

where ω_i represents the angular velocity of the UAVs and a_i represents the acceleration of the UAVs.

The motion control variables of the UAVs are

$$\begin{cases} -v_{p\,\max} \le v_p \le v_{p\,\max}, \\ -v_{e\,\max} \le v_e \le v_{e\,\max}, \\ -\omega_{p\,\max} \le \omega_p \le \omega_{p\,\max}, \\ -\omega_{e\,\max} \le \omega_e \le \omega_{e\,\max}, \end{cases} \tag{4}$$

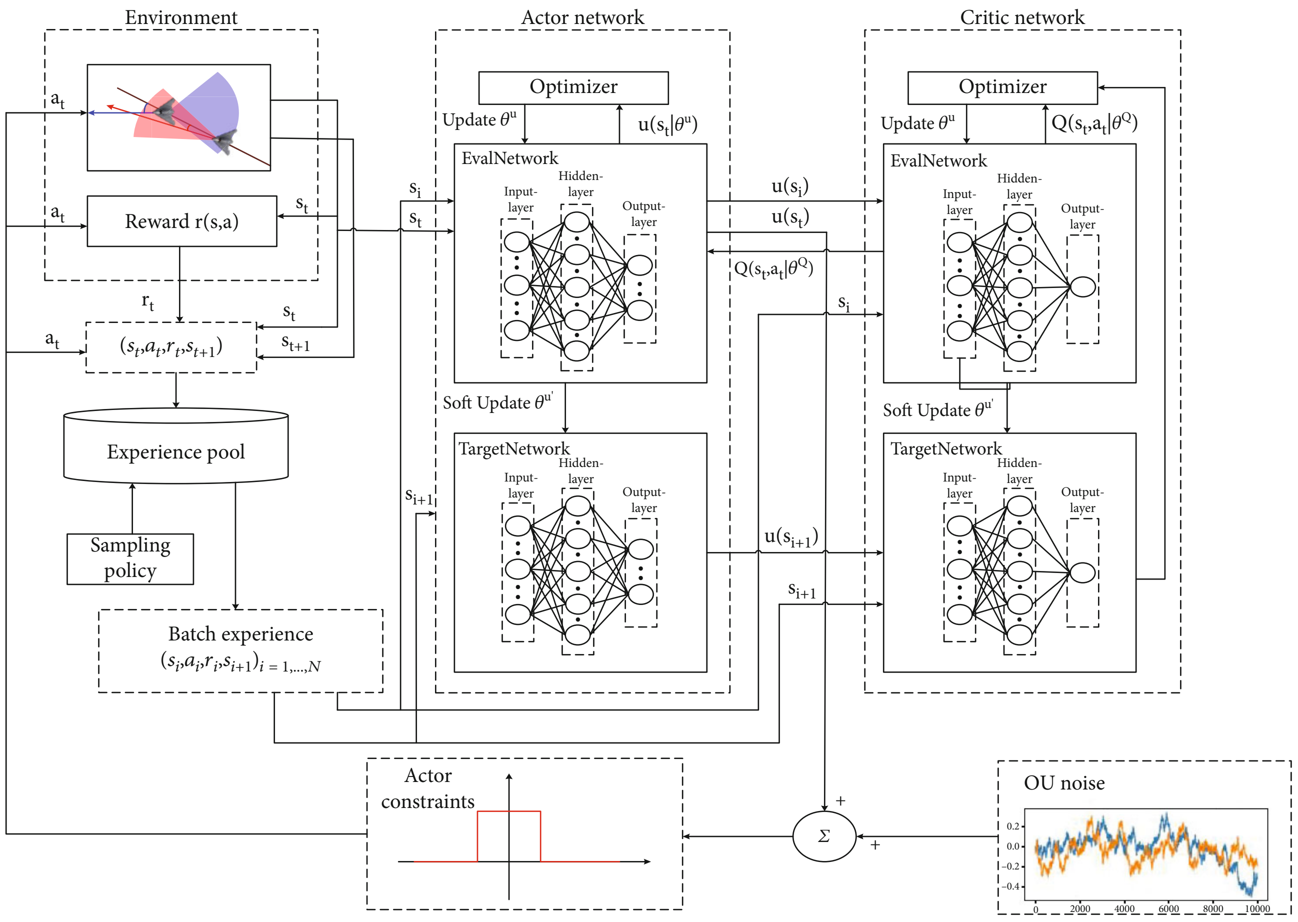


FIGURE 2: DDPG network structure.

where $v_{p\ \max}$ and $v_{e\ \max}$ are the maximum speed of the UAVs and $\omega_{p\ \max}$ and $\omega_{e\ \max}$ are the maximum angular velocity of the UAVs.

$$\begin{cases} \omega_{i\ \max}\Delta T = \Delta\psi_i \\ n_{i\ \max} g = \dfrac{v^2{}_i}{r_{i\ \min}} \\ r_i \sin\Delta\psi_i \approx \dfrac{v_i \Delta T}{2} \end{cases} \quad (i = p, e), \tag{5}$$

where ΔT is the simulation time step, r_i is the turning radius, $r_{i\ \min}$ is the minimum turning radius, $\Delta\psi_i$ is the maximum turning angle within ΔT, and $n_{i\ \max}$ is the maximum lateral overload. Therefore, the maximum angular velocity can be obtained.

$$\omega_{i\ \max} = \frac{\arcsin\left(\Delta T n_{i\ \max} g/(2v_i)\right)}{\Delta T} \quad (i = p, e). \tag{6}$$

The initial state of the UAVs is defined as

$$\begin{cases} x_i(t_0) = x_{i0} \\ y_i(t_0) = y_{i0} \\ v_i(t_0) = v_{i0} \\ \psi_i(t_0) = \psi_{i0} \end{cases} \quad (i = p, e). \tag{7}$$

If the distance between the evasion-UAV and the pursuit-UAV is within the capture range of the pursuit-UAV and does not increase, the capture is successful, as shown in (8), and the capture range can be the detection range or the attack range of the UAV.

$$\|d_t\| \le l_c, \tag{8}$$

where l_c is the capture range of the UAV and $\|d_t\|$ is the 2-norm of the two-dimensional vector (d_{xt}, d_{yt}) and it can be calculated by

$$\begin{cases} d_{xt} = |x_{pt} - x_{et}| \\ d_{yt} = |y_{pt} - y_{et}| \end{cases}, \quad (9)$$

where d_{xt} and d_{yt} represent the instantaneous relative distances of the pursuit-UAV and the evasion-UAV in the x-axis and the y-axis at time t, respectively.

3. UAV Pursuit Strategy Based on DDPG

3.1. MDP Model. The MDP model can be divided into MDP state space model, MDP action space model, MDP state transition function, and MDP reward function.

3.1.1. MDP State Space Model. The UAV is set to carry on-board GPS equipment and gyroscopes to obtain its own position and speed, namely, $\xi_p = [x_p, y_p, v_p, \psi_p]$. We also set the UAV to carry the on-board airborne radar to obtain detected target's position and speed, namely, $\xi_e = [x_e, y_e, v_e, \psi_e]$.

In order to increase the adaptability of the algorithm, the relative position is used to establish the state space model.

$$S = [\alpha_p, \alpha_e, \alpha_{pe}, d_{pe}, v_p, \Delta v_{pe}], \quad (10)$$

where α_p and α_e are the angles between the speed direction of the pursuit-UAV and evasion-UAV and the LOS, respectively. $\alpha_{pe} = \alpha_e - \alpha_p$ is the angle between the speed direction of the two UAVs, d_{pe} is the distance between the two UAVs, v_p is the speed of the pursuit-UAV, and Δv_{pe} refers to the speed difference between the two UAVs.

$$\Delta v_{pe} = v_p - v_e. \quad (11)$$

3.1.2. MDP Action Space. The control input of the UAV is a two-dimensional vector, namely, action space

$$A = [a_i, \omega_i](i = p, e), \quad (12)$$

where a_i is the acceleration of pursuit-UAV and evasion-UAV and $a_i = \dot{v}_i$, ω_i is the angular velocity of pursuit-UAV and evasion-UAV. Both v_i and ω_i satisfy the constraints (4).

(1) MDP state transition function

The state transition function is as shown in

$$\begin{pmatrix} x_i \\ y_i \\ v_i \\ \psi_i \end{pmatrix}_{t+1} = \begin{pmatrix} x_i \\ y_i \\ v_i \\ \psi_i \end{pmatrix}_t + \begin{pmatrix} v_i \cos \psi_i \\ v_i \sin \psi_i \\ a_i \\ \omega_i \end{pmatrix}_t \Delta T \quad (i = p, e). \quad (13)$$

3.1.3. MDP Reward Function. A combination of sparse reward and guided reward function is designed:

$$\begin{cases} R_{t1} = d_{t-1} - d_t, \\ R_t = kR_{t1} + R_{t2} + R_{t3}, \end{cases} \quad (14)$$

where R_t represents the total reward of the UAV. R_{t1} is a guided reward, d_t represents the distance between the pursuit-UAV and the evasion-UAV at time t, and d_{t-1} represents the distance at time $t-1$; k is proportionality; R_{t2} represents the sparse reward of the pursuit-UAV being too far away from the evasion-UAV; R_{t3} represents the sparse reward of the pursuit-UAV to complete the task.

R_{t1} is the variation of the relative distance between the pursuit-UAV and the evasion-UAV. When the relative distance becomes smaller, the pursuit-UAV gets a positive reward; when the relative distance becomes larger, the pursuit-UAV gets a negative reward.

$$R_{t2} = \begin{cases} -R_{\text{far}}, & \text{if } d_t > D_{\text{far}}, \\ 0, & \text{otherwise}, \end{cases} \quad (15)$$

where D_{far} represents the relative distance threshold and R_{far} is a large positive constant, which punishes the algorithm when the pursuit-UAV's action strategy is incorrect and the distance from the evasion-UAV is too far.

$$R_{t3} = \begin{cases} R_{\text{finish}}, & \text{if } d_t < l_c, \\ 0, & \text{otherwise}, \end{cases} \quad (16)$$

where R_{finish} is a large positive constant, which rewards the algorithm when the UAV completes the task.

3.2. DDPG Algorithm. The core of reinforcement learning is that the agent obtains rewards by interacting with the environment and adjusts the strategy according to the size of the rewards to realize the optimization of decision-making. Deep reinforcement learning (DRL) combines the approximate fitting of deep learning (DL) and the decision-making optimization of reinforcement learning (RL). The most representative DRL algorithm is the deep Q-learning (DQN) algorithm.

DQN uses two networks with the same structure but different parameters. One network generates the current Q value, and the other network generates the target Q value. Then, these two values are used to minimize the loss function, and the parameters of the current network are copied to the target network after a period of time. DQN uses experience replay to break the relevance of RL data and uses random sampling to extract data from the experience pool for training.

The Deep Deterministic Policy Gradient (DDPG) algorithm which was developed based on the core idea of DQN also uses the Actor-Critic dual network mechanism and combines the advantages of the value function and the strategy function method.

```
Training algorithm for UAV strategy based on DDPG
initial experience pool **D** with memory size **M**
initial the Eval networks of Actor network and Critic network:$\mu(s;\theta^u)$ and $Q(s,a|\theta^Q)$
**for***episode*=1 to *MaxEpisode***do**
  initialize OU-Noise $N(t)$
  initialize the state of pursuit-UAV and evasion-UAV in set range randomly,
obtain the initial state $s_0$of simulation environment
  **for***t*=1 to *MaxStep***do**
    select action$a_t = f_{clip}(u(s_t|\theta^u)+N_t)$ of pursuit-UAV where $f_{clip}$is the action constraint processing process $s_t$
    select maneuver strategy for evasion-UAV
    input the control signal into the UAV integrate to get the next state of UAV, and calculate the environment state $s_{t+1}$
    obtain the immediate reward $r_t$ from the environment
    store experience sample $[s_t, a_t, r_t, s_{t+1}]$ in **D**
    randomly sample form **D** to get a sample set of *BatchSize* $\{[s_t, a_t, r_t, s_{t+1}]\}$
    update the Eval network parameter $\theta^Q$ of the Critic
    update the Eval network parameter $\theta^u$ of the Actor
    update the Target network parameters $\theta^{Q'}$ and $\theta^{u'}$ of Critic network and Actor network by (20)
    if the episode end condition is satisfied, break
  end for
end for
```

Algorithm 1: The UAV pursuit strategy using DDPG.

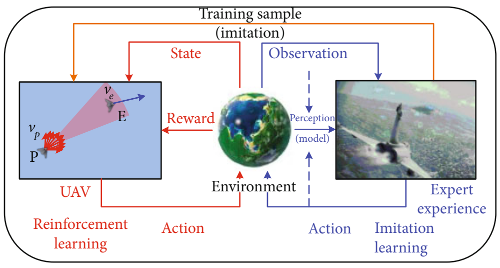


Figure 3: The algorithm framework based on IL-DDPG.

The DDPG algorithm has four subnetworks, and the network structure of the algorithm is shown in Figure 2.

Actor network and Critic network both have target network (TargetNet) and evaluation network (EvalNet), so DDPG has a total of 4 subnetworks.

The Actor selects action $\mu(s_t)$ according to the action probability provided by itself. The Critic_EvalNet evaluates the current state and the value $Q(s_t, \mu(s_t))$ of the action selected by the Actor, and the Critic_TargetNet evaluates the next state s_{t+1} and the value $Q'(s_{t+1}, \mu(s_{t+1}))$ of the action $\mu(s_{t+1})$ selected by the Actor_TargetNet for s_{t+1}. Then, the Actor will adjust the probability of the action according to Critic's evaluation of the action [20, 21]. (θ^Q, θ^u) and $(\theta^{Q'}, \theta^{u'})$ are the EvalNet and TargetNet parameters of the Critic and Actor, respectively. Actor and Critic use different functions to train and update the parameters.

Critic uses the mean square error loss function to update the parameters θ^Q of the Critic_EvalNet through the gradient of the neural network, as shown in

$$L\left(\theta^Q\right) = \frac{1}{K}\sum_i \left(Q\left(s_i, a_i\middle|\theta^Q\right) - y_i\right)^2, \tag{17}$$

where K is the sample size and is y_i defined as

$$y_i = r(r_i, a_i) + \gamma Q'\left(s_{i+1}, u'\left(s_{i+1}\middle|\theta^{u'}\right)\middle|\theta^{Q'}\right). \tag{18}$$

Actor uses the gradient of Equation (19) to update the parameter θ^u of the Actor_EvalNet.

$$\nabla_{\theta^u} J = \frac{1}{K}\sum_i \left(\nabla_a Q\left(s, a\middle|\theta^Q\right)\Big|_{s=s_i, a=u(s_i)} \nabla_{\theta^u} Q\left(s\middle|\theta^u\right)\Big|_{s=s_i}\right). \tag{19}$$

Like DQN, the EvalNet will train the network parameters in real time to update (θ^Q, θ^u), and the TargetNet parameters $(\theta^{Q'}, \theta^{u'})$ will follow the EvalNet through soft updates. The advantage of using soft update is to make algorithm training more stable and easy to guarantee convergence. The soft update is described as

$$\begin{cases} \theta^{u'} \longleftarrow \tau\theta^u + (1-\tau)\theta^{u'}, \\ \theta^{Q'} \longleftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}, \end{cases} \tag{20}$$

where τ is the inertial update rate.

A major innovation of DDPG is the use of motion noise. Adding a random noise to the action generated by the Actor turns the deterministic decision into a random process. It enhances the exploration of the algorithm. Commonly used random noises are Gaussian Noise and Ornstein-Uhlenbeck (OU) Noise.

OU Noise is also called OU process, which is a random process. It will explore a certain distance around the mean value in the positive or negative direction. This is conducive

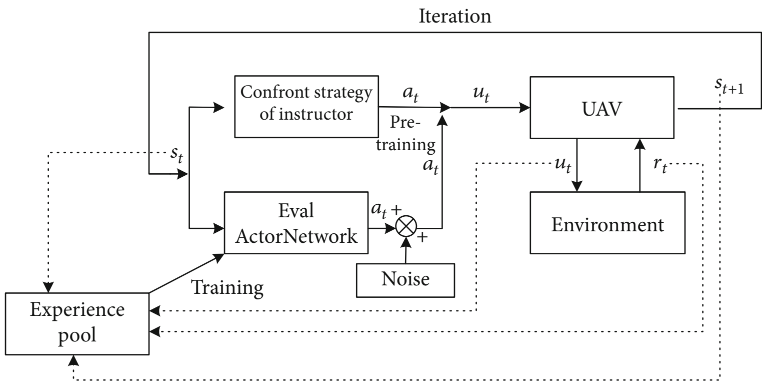


FIGURE 4: Offline training and exploration process of UAV.

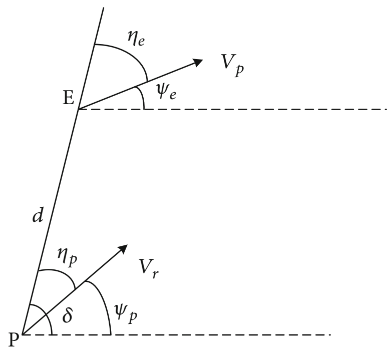


FIGURE 5: Relative movement of pure proportional method.

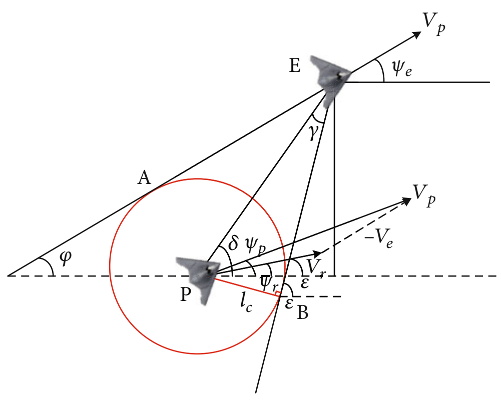


FIGURE 6: Diagram of quasiproportional guidance control law.

to exploring in one direction and can improve the efficiency of exploration and training for inertial systems.

The agent obtains the sample set (s_t, a_t, r_t, s_{t+1}) of the training network in the process of interacting with the environment and stores these samples in the experience pool. During training, the agent selects some minibatch samples according to the random sampling strategy to train the neural network parameters through experience replay.

3.3. Training Process of DDPG Offline Algorithm. In our experiment, the control period is set to the simulation step. It should be noted that the subscript t of states s_t and a_t represents the time step rather than the actual flight time, and the actual flight time is $T = t\Delta T$. Algorithm 1 shows the flow of training algorithm for UAV pursuit strategy based on DDPG.

TABLE 1: Table of training hyperparameter.

Training hyperparameter	Symbol	Value
Discounting factor	γ	0.9
Inertial update rate	τ	0.01
Memory size	M	30000
Size of batch experience	Batch size	64
Simulation time step	ΔT	0.1
Learning rate of Critic network	α_Q	0.002
Learning rate of Actor network	α_u	0.001
Number of episodes	MaxEpisode	4000
Number of steps in one episode	MaxStep	300

4. UAV Pursuit Strategy Based on IL-DDPG

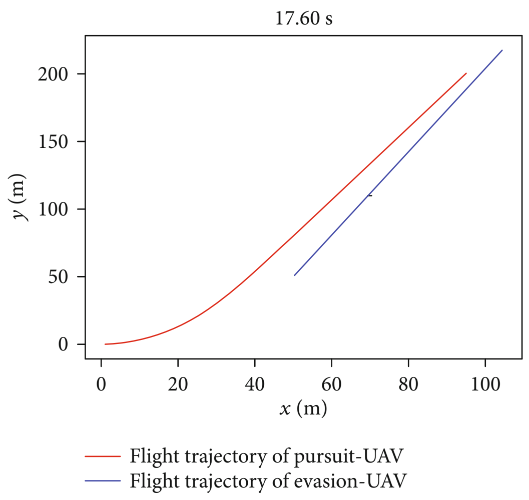
4.1. Imitation Learning. Model-free and model-based reinforcement learning methods both learn a strategy from scratch that maximizes the cumulative return. For complex tasks, the agent has a huge search space and cannot get meaningful rewards frequently in the initial stage, which leads to a slow convergence rate of reinforcement learning.

IL means that the agent uses the decision data provided by experts to learn the best strategy [22]. It can be used to solve problems that the reward cannot be given. We can integrate IL with RL to accelerate the process of strategy learning by providing effective samples through experts' demonstrations.

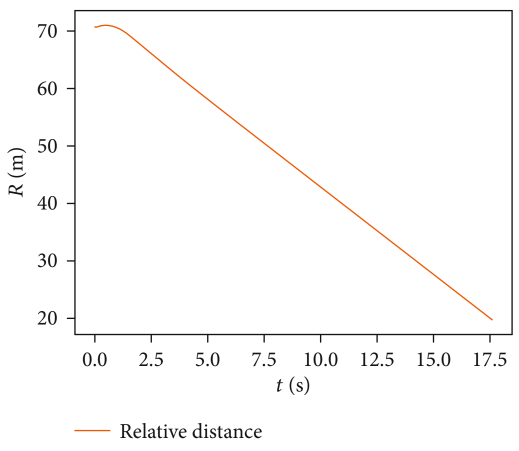
At present, scholars have successfully verified the feasibility of this method. A deep Q-learning from demonstrations (DQfD) algorithm is proposed [23], which combines the TD updates with the supervised classification of the instructor's actions, and the demonstrations are used to pretrain the Q network in the DQN, and at the same time, the demonstrations are put into the experience pool, and these expert data are used to accelerate the learning process on a

Table 2: Table of experiment parameters.

Experiment parameter	Symbol	Value
Space dimension	—	2
Pursuit-UAV initial position	$\left(x_p, y_p\right)$	$[0, 50] \times [0, 50]$ m
Pursuit-UAV initial velocity	v_p	10 m/s
Pursuit-UAV initial heading	ψ_p	$[0, 2\pi]$
Pursuit-UAV velocity range	$\left[v_{p\ \min}, v_{p\ \max}\right]$	$[9, 13]$ m/s
Pursuit-UAV angular velocity range	$\omega_{p\ \max}$	3 rad/s
Evasion-UAV initial position	(x_e, y_e)	$[0, 50] \times [0, 50]$ m
Evasion-UAV initial velocity	v_e	10 m/s
Evasion-UAV initial heading	ψ_e	$[0, 2\pi]$
Evasion-UAV velocity range	$[v_{e\ \min}, v_{e\ \max}]$	$[9, 13]$ m/s
Evasion-UAV angular velocity range	$\omega_{e\ \max}$	3 rad/s
Distance threshold	D_{far}	600 m
Capture radius	l_c	20 m



(a) Full flight trajectories of two UAVs



(b) Relative distance

Figure 7: The evasion-UAV with linear motion.

large scale. The DQfD is proved that it has better initial performance than DQN.

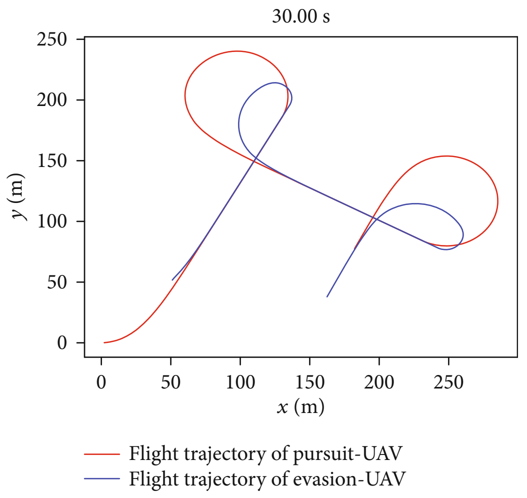
On this basis, the DDPG algorithm is combined with the demonstrations in a similar way to construct the DDPGfD algorithm [24].

4.2. IL-DDPG Algorithm. DDPG based on imitation learning algorithm (IL-DDPG) is designed to solve the maneuver decision-making problem of the UAV pursuit-evasion. The design of this algorithm mainly includes two aspects; one is the algorithm framework, and the other is the maneuver strategy of the instructor.

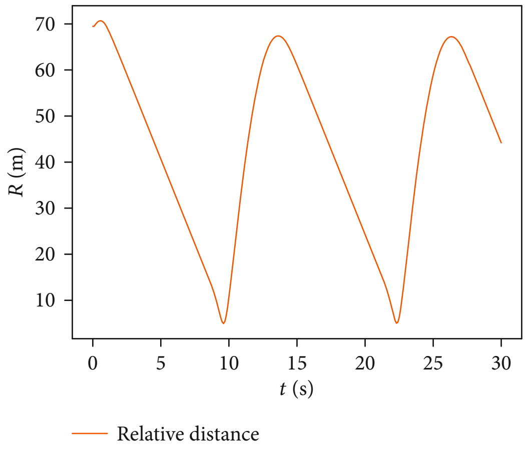
4.2.1. Framework of IL-DDPG Algorithm. Figure 3 shows the algorithm framework of IL-DDPG. In this framework, the instructor's strategy is used to generate amounts of experience and store them in the experience pool in the initial stage. And these experiences are used to train the network by RL.

Figure 4 shows the process of UAV offline training and exploration. Before starting any interaction with the environment, IL-DDPG initially only trains the demonstrations, which is the pretraining process. A value function that satisfies the Bellman equation is used to imitate the instructor so that it can be updated with TD_error once the UAV starts interacting. The subsequent learning and training of IL-DDPG are consistent with the DDPG algorithm.

4.2.2. Instructor Confront Strategy. The main improvement of our algorithm is the design of DDPG initial exploration



(a) Full flight trajectories of two UAVs



(b) Relative distance

Figure 8: The evasion-UAV with the classic escape strategy.

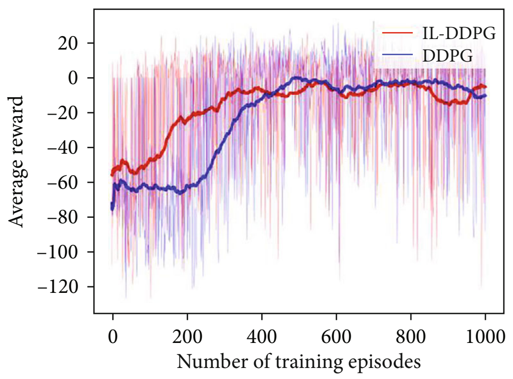


Figure 9: Average rewards of IL-DDPG and DDPG algorithm (the thin line of the background is the real-time curve of the total reward per episode, and the thick line is the average reward curve per episode).

strategy, that is, instructor confront strategy. Proportional guidance is one of the missile guidance methods, and it is also often used in the interception of maneuvering targets. Therefore, the proportional guidance method can be used as our instruction strategy.

The pure proportional navigation method [25, 26] is that during the guidance process, the rotational angular velocity of the controlled object's velocity vector is proportional to the rotational angular velocity of LOS, and the core equation of the guidance is shown as

$$\varepsilon = \frac{d\psi}{dt} - K\frac{d\delta}{dt} = 0, \tag{21}$$

where K is the scale factor and its range is $(1, \infty)$, and $\varepsilon = 0$ is the ideal control relation equation describing the guidance method. Figure 5 shows the relative movement relationship of pure proportional method.

The disadvantage of pure proportional guidance is that the normal overload required to hit the target is directly related to the target speed at the hit point and the UAV's attack mode, and it leads to difficulty in selecting the value of$K$. We can use the generalized proportional guidance method to improve the characteristics of proportional guidance.

The normal overload is selected according to the rotation angular velocity of the LOS, namely, $n = K|\dot{d}|\dot{\delta}$. The normal overload when the UAV hits the target is

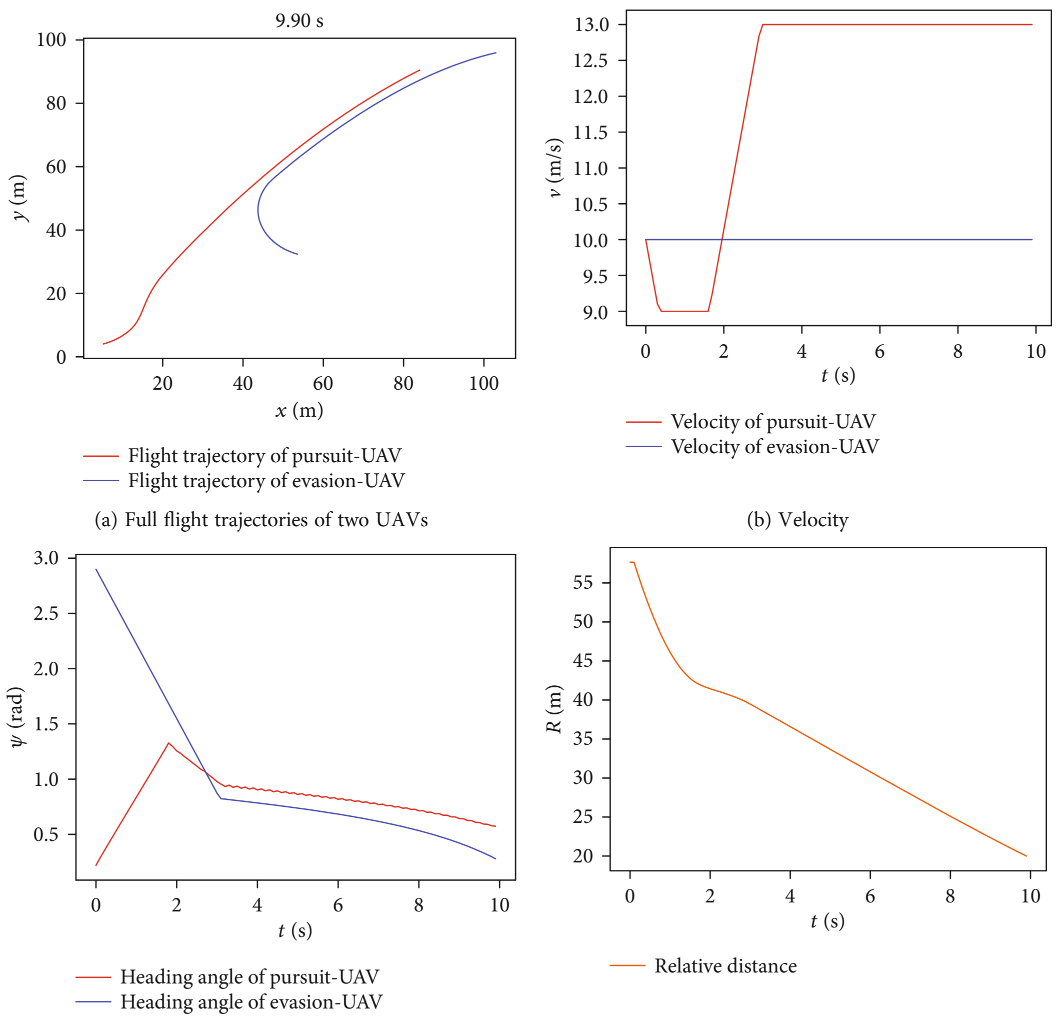
$$n_k = \frac{1}{g}\frac{\left(\dot{v}_p \sin\eta_p - \dot{v}_e \sin\eta_e + v_e\dot{\psi}_e \cos\eta_e\right)}{\cos\eta_p - (2/Kg)}. \tag{22}$$

It can be seen that the required overload at the hit point has nothing to do with the UAV speed and attack direction.

Considering the characteristics of the UAV's capture range, a quasiproportional guidance control law [27] is designed as shown in Figure 6. Compared with pure proportional guidance, it fully considers the difference between UAV guidance and missile guidance.

In Figure 6, the red circle is the effective capture range of the pursuit-UAV, and l_c is the capture radius of the UAV. v_r is the relative speed of the pursuit-UAV and the evasion-UAV, and $\overrightarrow{v_r} = \overrightarrow{v_p} - \overrightarrow{v_e}$. ψ_r is the angle of the v_r. EA and EB are the two guiding boundary lines; ε and φ are the angles of these lines. δ is the angle of LOS, and γ is the angle between LOS and the two boundary lines.

The quasiproportional guidance law guides the pursuit-UAV P to make the evasion-UAV E fall into the capture range of P. For this purpose, the relative velocity vector v_r and its angle φ_r are controlled. In the guidance process, approaching the target along EA or EB depends on the difference between ψ_r and the boundary line. If $|\psi_r - \varphi|$ is



(a) Full flight trajectories of two UAVs

(b) Velocity

(c) Heading angle

(d) Relative distance

FIGURE 10: Flight trajectories and test results using DDPG.

smaller, EA will be selected as the guiding boundary; if $|\psi_r - \varepsilon|$ is smaller, EB will be selected.

If *EB* is chosen as the guidance boundary line, the quasi-proportional guidance instruction can be

$$n = K v_r \dot{\varepsilon}, \tag{23}$$

and we can get

$$\dot{\varepsilon} = -\frac{v_r \sin \psi_r}{d \cos \delta} + \frac{\dot{d}}{d}(\tan \delta + \tan \gamma) + \frac{l_c}{d \cos \gamma}, \tag{24}$$

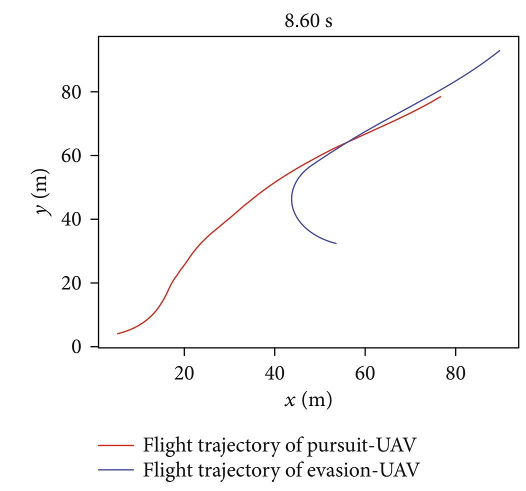
where d is the distance between pursuit-UAV P and evasion-UAV E.

The state transition equation of maneuver decision control is shown as

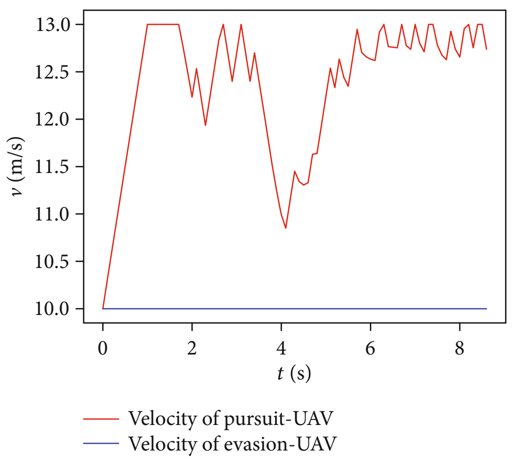
$$\begin{pmatrix} x_i \\ y_i \\ v_i \\ \psi_i \end{pmatrix}_{t+1} = \begin{pmatrix} x_i \\ y_i \\ v_i \\ \psi_i \end{pmatrix}_t + \begin{pmatrix} v_{it} \cos \psi_{it} \\ v_{it} \sin \psi_{it} \\ -n_{it} \sin (\psi_{rt} - \psi_{it}) \\ -n_{it} \cos (\psi_{rt} - \psi_{it}) \end{pmatrix} \Delta T \quad (i = p). \tag{25}$$

5. Simulation Experiments

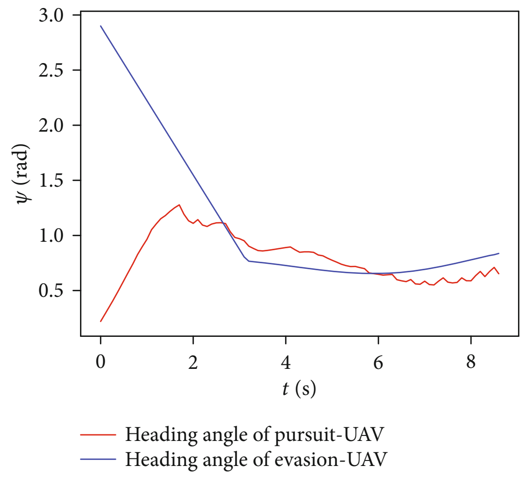
5.1. Experimental Settings. The simulation system is constructed based on Python, using PyCharm Community 2020.2 and Anaconda3 platforms. The deep learning environment adopts TensorFlow 1.14.0. The computer is configured



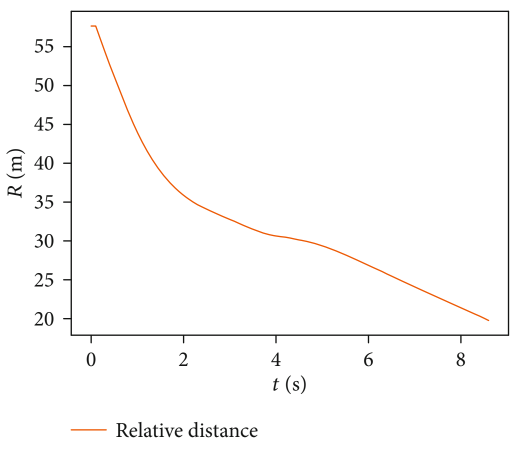
(a) Full flight trajectories of two UAVs



(b) Velocity

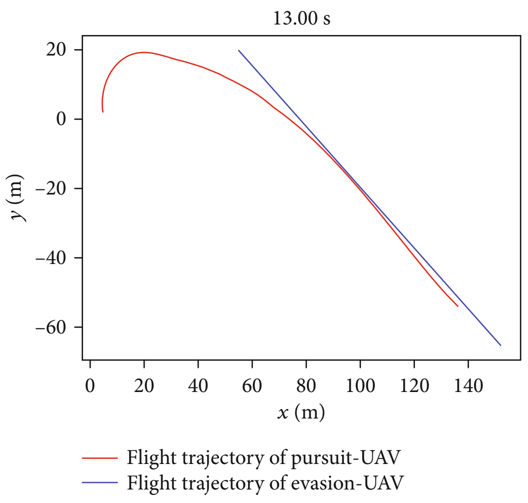


(c) Heading angle

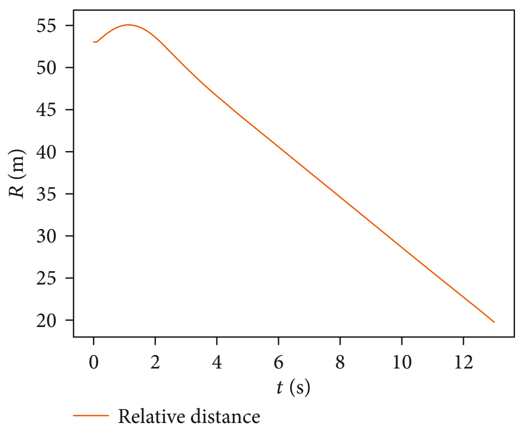


(d) Relative distance

FIGURE 11: Flight trajectories and test results using IL-DDPG.

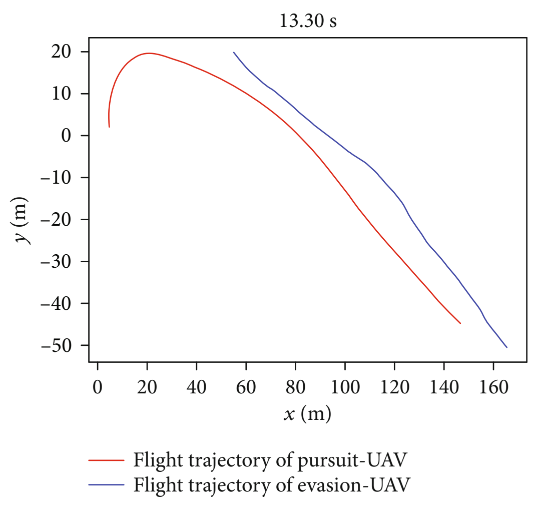


(a) Full flight trajectories of two UAVs

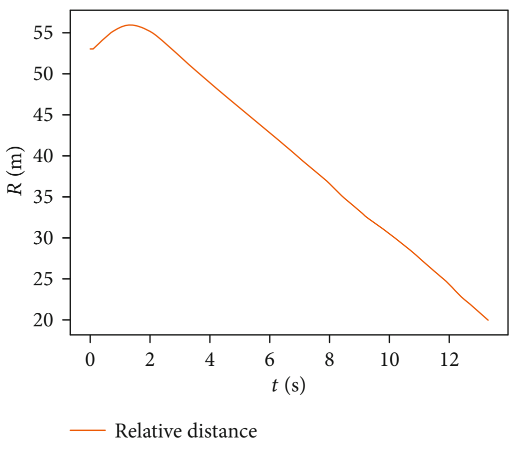


(b) Relative distance

FIGURE 12: Trajectories of facing evasion-UAV with linear motion.

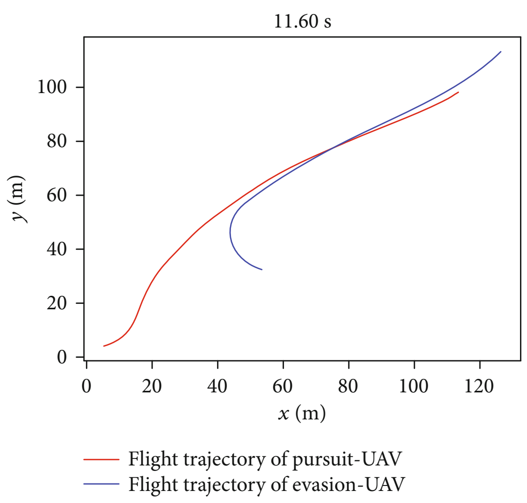


(a) Full flight trajectories of two UAVs

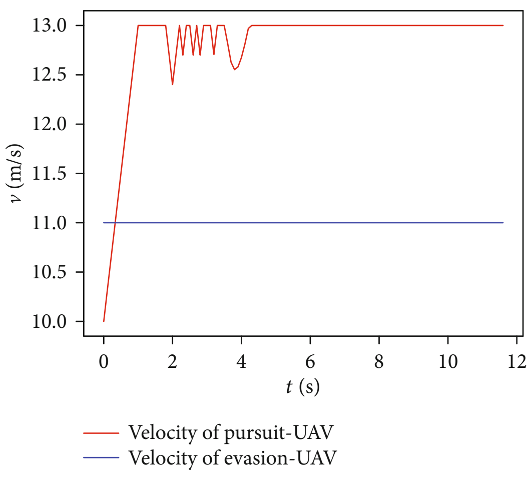


(b) Relative distance

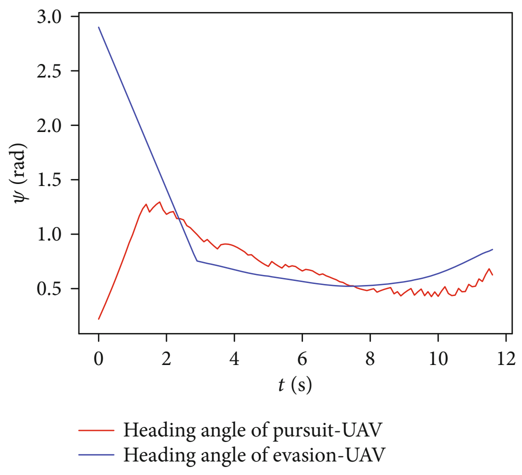
FIGURE 13: Trajectories of facing evasion-UAV with a random-strategy motion.



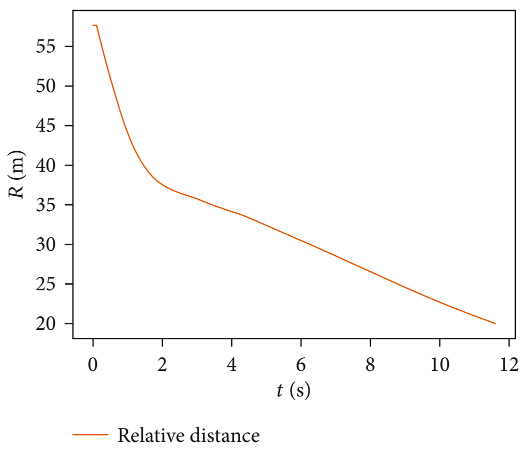
(a) Full flight trajectories of two UAVs



(b) Velocity



(c) Heading angle



(d) Relative distance

FIGURE 14: Flight trajectories and test results using IL-DDPG.

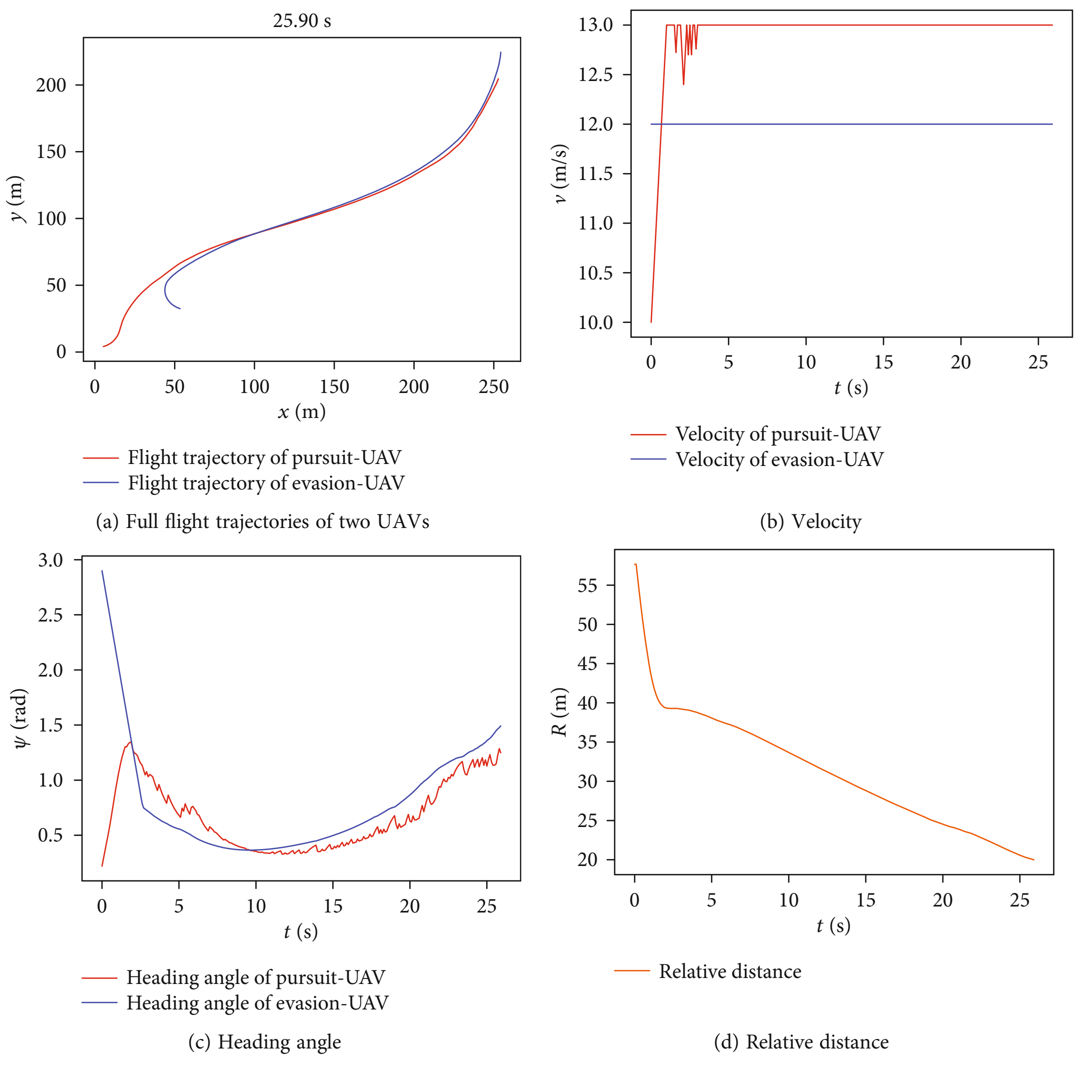


Figure 15: Flight trajectories and test results using IL-DDPG.

as Inter i7-9700F@3.00 GHz CPU, GTX 1660Ti GPU, 16 GB RAM.

The training parameters of algorithms in the experiment are shown in Table 1.

The experiment parameters of the UAV pursuit-evasion game simulation environment are shown in Table 2.

The evasion-UAV adopts the classic escape strategy [28], namely,

$$\begin{cases} a_e = 0, \\ \omega_e = \begin{cases} -w_{e\max}, & \xi_e < -w_{e\max}, \\ \xi_e, & -w_{e\max} \le \xi_e \le w_{e\max}, \\ w_{e\max}, & \xi_e > w_{e\max}, \end{cases} \end{cases} \tag{26}$$

where calculation of ξ_e is shown in the following:

$$\xi_e = \frac{\arctan\left(\left(y_e - y_p\right)/\left(x_e - x_p\right)\right) - \psi_e}{\Delta T}. \tag{27}$$

All networks are multilayer feedforward neural network with a single hidden layer. The number of neurons in each layer of Target-Actor network and Eval-Actor network is [6, 128, 2]. Their hidden layer uses Relu (x) as activation function, and their output layer uses Tanh (x) as activation function. The input of Critic network are the MDP state and generated actions by the Actor network, so the number of neurons in each layer of Target-Critic network and Eval-Critic network is [8, 128, 1]. Their activation functions are the same as those of Actor network. The training applies *ADAM Optimizer* as optimizer.

5.2. Simulation Results

5.2.1. Instructor Confront Strategy. The pursuit-UAV only uses the designed quasiproportional guidance strategy. The evasion-UAV adopts two different strategies: uniform linear motion and the classic escape strategy. The speed of the evasion-UAV is set for 10 m/s.

As shown in Figure 7, the evasion-UAV escapes in a straight line simply, and the pursuit-UAV successfully captures the evasion-UAV after adjusting the speed direction. But in Figure 8, the evasion-UAV escapes successfully. It is because the pursuit-UAV that uses the quasiproportional guidance method as the pursuit guidance law needs time to adjust its heading, which creates an opportunity for the evasion-UAV to escape within the predetermined maximum time.

Although the quasiproportional guidance law cannot let the pursuit-UAV capture the evasion-UAV when the evasion-UAV uses the classic escape strategy, it can guide the pursuit-UAV to explore good initial experience as an instructor strategy.

5.2.2. Comparison. Average reward is used to verify the convergence and effectiveness of the proposed algorithm, and it is defined as the average value of reward in latest 100 episodes.

With the same training parameters and the same experiment parameters, the average rewards of the trained results obtained by the IL-DDPG and DDPG algorithms are shown in Figure 9.

As shown as Figure 9, the IL-DDPG algorithm converges faster than the DDPG algorithm, and it is more stable than the DDPG algorithm.

In order to compare the trained results of the two algorithms under the same initial conditions, the results are used to simulate the UAVs pursuit-evasion process. The simulation results are shown in Figures 10 and 11.

It can be seen from Figures 10 and 11 that the trained results obtained by the IL-DDPG algorithm achieves a shorter capture time.

Furthermore, as shown in Figures 12 and 13, the pursuit-UAV using the IL-DDPG algorithm can adjust its speed and heading in time to capture the evasion-UAV, no matter whether the evasion-UAV adopts uniform linear motion strategy or random motion strategy.

These experiments prove that the UAV pursuit strategy based on the IL-DDPG algorithm has a good generalization, and the trained UAV can successfully complete the pursuit task in the pursuit-evasion game.

Figures 14 and 15 increase the velocity of evasion-UAV to 11 m/s and 12 m/s, respectively. It can be seen that pursuit-UAV can capture the evasion-UAV within a given time, which verifies the Imitation of the IL-DDPG algorithm.

6. Conclusion

The training algorithm of the UAV pursuit strategy based on the IL-DDPG algorithm introduces a quasiproportional guidance control law as the instructor strategy to improve the exploration efficiency in the early stage of DDPG training and avoids the problems of excessive useless exploration. Simulation results show the effectiveness and generalization of this algorithm.

For the future work in this paper, we should study how to effectively combine the imitation learning and the multi-experience pool technology to accelerate the training of the algorithm.

Data Availability

The numerical data used to support the findings of this study is included within the article.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgments

This research was funded in part by the Aviation Science Foundation of China under Grant 2020Z023053001 and Natural Science Foundation of Shaanxi Province (No. 2020JM-537).

References

[1] M. Wu, L. Tan, and N. Xiong, "A structure fidelity approach for big data collection in wireless sensor networks," *Sensors*, vol. 15, no. 1, pp. 248–273, 2015.

[2] P. Yang and J. Ren, "Data security and privacy protection for cloud storage: a survey," *IEEE Access*, vol. 8, pp. 131723–131740, 2020.

[3] S. Huang, A. Liu, S. Zhang, T. Wang, and N. N. Xiong, "BD-VTE: a novel baseline data based verifiable trust evaluation scheme for smart network systems," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 3, 2021.

[4] K. Gao, F. Han, P. Dong, N. Xiong, and R. du, "Connected vehicle as a mobile sensor for real time queue length at signalized intersections," *Sensors*, vol. 19, no. 9, p. 2059, 2019.

[5] H. Li, J. Liu, K. Wu, Z. Yang, R. W. Liu, and N. Xiong, "Spatio-temporal vessel trajectory clustering based on data mapping and density," *IEEE Access*, vol. 6, pp. 58939–58954, 2018.

[6] X. Yue, X. Shao, and W. Zhang, "Elliptical encircling of quadrotors for a dynamic target subject to aperiodic signals updating," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–14, 2021.

[7] Q. N. Meng, *Differential game with constraints and its application in air combat*, Shenyang Aerospace University, 2018.

[8] J. Xie, *Differential game theory for multi UAV pursuit maneuver technology based on collaborative research*, Harbin Institute of Technology, 2015.

[9] H. H. Chin, "Knowledge-based system of super maneuver selection for pilot aiding," *Journal of Aircraft*, vol. 26, no. 12, pp. 1111–1117, 1989.

[10] J. E. Matheson, *Using Influence Diagrams to Value Information and Control*, John Wiley& Sons, New York, 1988.

[11] L. Tan, Q. H. Gong, and H. X. Wang, "Pursuit-evasion game algorithm based on deep reinforcement learning. Aerospace," *Control*, vol. 36, no. 6, pp. 3–8, 2018.

[12] Y. Z. Zhang, J. L. Xu, K. J. Yao et al., "Pursuit missions for UAV swarms based on DDPG algorithm," *Acta Aeronautica et Astronautica Sinica*, vol. 41, no. 10, pp. 314–326, 2020.

[13] X. Y. Song, Y. X. Wang, Z. H. Cai, J. Zhao, X. L. Chen, and D. L. Song, "Landing trajectory tracking control of unmanned aerial vehicle by deep reinforcement learning," *Aeronautical Science & Technology*, vol. 31, no. 1, pp. 68–75, 2020, (In Chinese).

[14] G. F. Wang, *Research on reinforcement learning aided by expert knowledge and its application in path planning of UAV*, Zhejiang University, 2017.

[15] Y. H. Wu, *Research on reactive obstacle avoidance using deep reinforcement learning and transfer learning*, Huazhong University of Science & Technology, 2019.

[16] L. P. Lu, *Robot skill acquisition based on imitation learning and reinforcement learning*, Dalian University of Technology, 2019.

[17] S. X. Zuo, *Intelligent control of autonomous driving based on deep reinforcement learning*, Harbin Institute of Technology, 2018.

[18] Z. Z. Mu, *Research on collaborative path planning of multi robot formation*, University of Chinese Academy of Sciences, 2020.

[19] I. Liubarshchuk and I. Althöfer, "The problem of approach in differential-difference games," *International Journal of Game Theory*, vol. 45, no. 3, pp. 511–522, 2016.

[20] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the Brownian motion," *Revista Latinoamericana de Microbiología*, vol. 15, no. 1, p. 29, 1973.

[21] Z. Hu, K. Wan, X. Gao, Y. Zhai, and Q. Wang, "Deep reinforcement learning approach with multiple experience pools for UAV's autonomous motion planning in complex unknown environments," *Sensors*, vol. 2020, p. 20, 2020.

[22] J. Bagnell, "An invitation to imitation," Tech. Report, CMU-RI-TR-15-08, Robotics Institute, Carnegie Mellon University, 2015.

[23] T. Hester, M. Vecerik, O. Pietquin et al., "Deep Q-learning from demonstrations," in *The Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 3223–3230, New Orleans, USA, February 2018.

[24] M. Vecerik, T. Hester, J. Scholz et al., "Leveraging demonstrations for deep reinforcement learning of robotics problems with sparse rewards," 2017, https://arxiv.org/pdf/1707.08817.pdf.

[25] Y. F. Wang, Y. W. Fang, and X. B. Zhou, "The status quo of proportional navigation guidance law and its development," *Fire Control and Command Control*, vol. 10, pp. 8–12, 2007.

[26] Z. Q. Li and D. Y. Zhou, "A guidance law of variable structure adaptive research for target escape with acceleration," *Fire Control and Command Control*, vol. 34, no. 5, pp. 109-110+124, 2009.

[27] Z. Y. Guan, *Research on guidance technology and attitude control of the UAV*, Harbin Engineering University, 2015.

[28] H. L. Shen, T. Furukawa, G. Dissanayake, and H. F. Durrant-Whyte, "A time-optimal control strategy for pursuit-evasion games problems," in *Processing of the 2004 IEEE International Conference on Robotics and Automation*, pp. 3962–3967, New Orleans, LA, 2004.