DOI: 10.13009/EUCASS2019-419

An Autonomous Maneuvering Decision Algorithm of UAV Based on Deep Deterministic Policy Gradient under Endpoint Constraints

Ke Li, Haotian Shi*, Quan Zhao*, Peipei Liu*, Chen Niu** and Kun Zhang**
**Northwestern Polytechnical University*
Xi'an 710072, China and kunzhang029@gmail.com
***Xi'an JiaoTong University*
Xi'an 710061, China and kunzhang029@gmail.com

Abstract

It's well known to all that operating an UAV (Unmanned Aerial Vehicle) safely and efficiently under endpoint constraints can be challenging in an interactive environment. And how to autonomously generate the optimal decision policy of UAV maneuvering is one of the key issues of solving this matter. In this paper, we proposed an autonomous maneuvering decision algorithm based on deep reinforcement learning to train the UAV agent for generating efficient maneuvers under endpoint constraints in an interactive environment. Finally, according to the self-developed simulation examples, a desirable maneuvering decision policy is developed by the algorithm we proposed.

1. Introduction

With the development of the UAV Technology in recent years, the performance of UAV has been improved rapidly in all aspects [1]. Improving UAV autonomous flight capability, reducing human intervention and avoiding human errors are the research focus of UAV researchers in various countries. The traditional UAV's flight guidance usually controls the maneuvering strategy of UAV to make it fly along the corresponding flight trajectory in the mission area. Usually, the maneuvering decision of UAV is based on matrix game [2], dynamic programming [3], neural network [4], expert system [5], dynamic Bayesian network [6], influence diagram [7] and trajectory prediction [8]. Due to the breakthrough progress in electronic technology and the rapid development of artificial intelligence technology, various artificial intelligence algorithms have been continuously applied to the control field in recent years. In 2015, DeepMind (Google) published Deep Q-Learning (DQN) in Nature, which is the first model to successfully combine deep learning with reinforcement learning [9].

In this paper, to solve the problem of autonomous generation of maneuvering decision for UAV in autonomous flight, we proposed an autonomous maneuvering decision algorithm based on deep reinforcement learning to train the UAV agent for generating efficient maneuvers under endpoint constraints in an interactive environment. Particularly, two specific forms of action function and policy function were designed by deep neural networks and a training data set based on the experience buffer was constructed. Meanwhile, we planned a training framework based on the Deep Deterministic Policy Gradient. Additionally, a UAV maneuvering decision model was established by the MDPs (Markov Decision Processes [10]) to describe the UAV maneuvers policy. Especially, the continuous state space, the continuous action space, and some specific reward functions are designed. The state space mainly included the relative state with respect to the target state. The action space was formed by the UAV's control variables. And the reward functions were designed aiming at the features of problems.

Finally, to verify the algorithm that we proposed, some self-developed simulation examples were given, and the verification experiment included two kinds of simulation examples. The first example was to enable the UAV to reach a fixed target point in the two-dimensional plane (horizontal plane), that means the UAV must fly to a specified position from a random position in the horizontal plane. The second example was to enable the UAV to be adjusted to a deterministic attitude (azimuth), that means the UAV could adjust itself to adapt to a required attitude from any state.

Next, we will introduce the UAV maneuvering decision model based on the MDPs and Autonomous Maneuvering Decision Algorithm of UAV based on DDPG.

Copyright © 2019 by First Author and Second Author. Published by the EUCASS association with permission.

DOI: 10.13009/EUCASS2019-419

2. The UAV Maneuvering Decision Model based on MDPs

2.1 The Markov Decision Processes Theory

The Markov decision process is an important method to study the state space of discrete event dynamic systems and its mathematical basis is stochastic process theory. In a finite period, the process describes that a decision maker periodically or continuously observes a stochastic dynamic system with markovian and makes decisions sequentially. The Markov decision process theory can be described by a quintuple $\langle T, S, A(s), P(\cdot\,|s,a), R(s,a)\rangle$, where T represents the decision time, S represents the system state space, $A(s)$represents the system action space, and the transition probability $P(\cdot\,|s,a)$ represents the probability distribution of the system at the next moment when the system used the action $a \in A$ in the state $s \in S$. The reward function $R(s,a)$ represents the benefit that the decision maker gets when the action $a \in A$ is taken in the state $s \in S$. Based on the Markov decision process theory, we can make a complete mathematical description of the sequence decision problem.

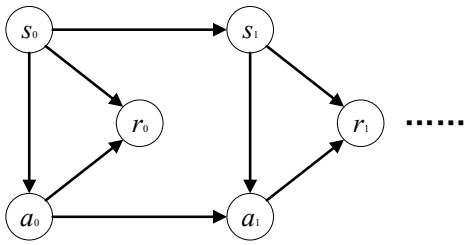

Figure 1: The structure of finit Markov decision processes.

As shown in Figure 1, the Markov decision process can be summarized as follows: the initial state s_0 of the system is that the decision maker chooses the action a_0 and executes it, the system moves to the system state s_1 according to a certain transition probability $P(\cdot\,|s_0,a_0)$, and so on. In this process, the decision maker earned rewards sequence $(r_0, r_1, \cdots)$. Among this process, decision makers are stimulated by external rewards, and the rewards are maximized by constantly updating the strategy. The strategy adopted by the decision maker is $a = \pi(s)$ and the utility function (in the state $s \in S$, the expected return obtained by adopting the strategy π) is $v(s,\pi)$. When current strategy is the optimal strategy, the formula (1) should be satisfied.

$$v(s) = \sup_{\pi} v(s,\pi), s \in S \tag{1}$$

Based on the characteristics of the autonomous maneuver control problem of UAV, we use infinite stage discount model as the utility function, as shown in formula (2).

$$v(s,\pi) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi}^{s}\left[R(s_t,a_t)\right], s \in S \tag{2}$$

In the above formula, $\gamma \in [0,1]$ is the future reward discount factor. According to the above formula, the optimal strategy under the discount model can be obtained.
In the following, based on the many characteristics of Markov decision process theory, including state space *S*, action space $A(s)$, transition probability $P(\cdot\,|s,a)$ and reward function $R(s,a)$, the autonomous flight control model of UAV is established.

2.2 The 3-DoF Flight Simulation Model of UAV

In the autonomous flight control model of the UAV, the 3-DoF flight simulation model of the UAV is used to achieve the motion simulation. According to the mathematical characteristics of the model, the transition probability of the system is $P(\cdot\,|s,a) = 1$. For example, the formula (3) is the 3-DoF flight simulation model of the UAV.

DOI: 10.13009/EUCASS2019-419

$$\begin{cases} \dfrac{dv}{dt} = \left(N_x - \sin\theta\right) g \\ \dfrac{d\theta}{dt} = \left(N_y \cos\gamma_c - \cos\theta\right) \dfrac{g}{v} \\ \dfrac{d\psi_c}{dt} = -\dfrac{N_y g \sin\gamma_c}{v\cos\theta} \\ \dfrac{dx}{dt} = v\cos\theta\cos\psi_c \\ \dfrac{dy}{dt} = v\sin\theta \\ \dfrac{dz}{dt} = -v\cos\theta\sin\psi_c \end{cases} \tag{3}$$

The tangential overload of the aircraft is N_x in the aircraft coordinate system. The normal overload in the aircraft coordinate system is N_y. The symbol v is the aircraft speed, the symbol θ is the track tilt angle of UAV, the symbol ψ_c is the track deflection angle of UAV, and the symbol γ_c is the speed tilt angle of UAV. The (x, y, z) is respectively the 3 directions coordinates of UAV in the geographic coordinate system. The symbol m is the mass of UAV and the symbol g represents the gravity acceleration. In the simulation process, we use the numerical analysis to solve the differential equations and the system's input and output are separately $\left(x, y, z, \psi_c, \theta, N_x, N_y, \gamma_c\right)$ and $(x, y, z, \psi_c, \theta)$.

2.3 The Flight State Space and Action Space of UAV

The autonomous maneuvering control of UAV under position constraint is designed to solve the problem that the UAV can fly to a target point autonomously from any position and arbitrary attitude in the horizontal plane. On the other hand, the autonomous maneuvering control under angle constraint is designed to solve the problem that the UAV is converted into a fixed attitude from a certain attitude in a horizontal plane through a certain maneuvering control strategy. For the above two problems, we can establish the corresponding geometric model, and their state space and action space are designed.

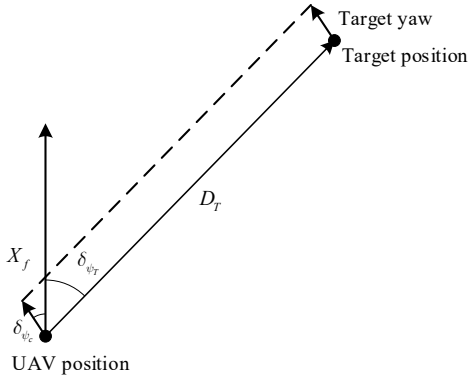

Figure 2 The geometric relationship between UAV and target state

According to the 3-DoF flight simulation model of UAV, the position of UAV is set to be $X_{UAV} = (x, y, z)$, and its reference coordinate system is the geographic coordinate system (the same below). The attitude of UAV is $(\psi_c, \theta, \gamma_c)$. Meanwhile, the target position is $X_{TGT} = (x, y, z)$, and the target entry yaw is ψ_T. Figure 2 shows the geometric relationship of the maneuvering decision problem that we proposed. The symbol X_f represents the longitudinal direction of the UAV. The symbol D_T is the distance between the UAV and the target position. The symbol δ_{ψ_T} is the orientation of the target position relative to the vertical axis of the UAV. The symbol δ_{ψ_c} represents the deviation between UAV and target attitude.

(1) The UAV flight action space

According to the formal of the control variable of the UAV's 3-DoF flight simulation model, a vector is used to describe the flight action of UAV, as shown in formula (4), which is the UAV's flight action space.

$$A = \left\{N_a\right\} \tag{4}$$

DOI: 10.13009/EUCASS2019-419

The symbol N_a is the control variable in the horizontal plane, and the relationship between the UAV's control amount and it is as shown in formula (5).

$$\begin{cases} N_x = 0.0 \\ N_y = |N_a| \\ \gamma_c = \begin{cases} 0, N_a \geq 0 \\ 180, N_a < 0 \end{cases} \end{cases} \tag{5}$$

(2) The UAV flight state space under position constraint
According to the UAV's position and the target position, the definition of the flight state space of UAV under position constraint in the horizontal plane can be obtained, as shown in formula (6).

$$S = \left\{ D_T, \delta_{\psi_T}, N_y \right\} \tag{6}$$

The symbol $D_T \in [0, D_T^{max}]$ is the distance between UAV and target position; the symbol $\delta_{\psi_T} \in [-\pi, \pi]$ is the azimuth of the target point relative to the longitudinal axis of UAV; the symbol $N_y \in \left[0, N_y^{max}\right]$ is the absolute value of the current decision action. As shown in formula (7), these are the calculation functions for each element of the state space.

$$\begin{cases} D_T = \left| X_{UAV} - X_{TGT} \right| \\ \delta_{\psi_T} = \arccos\left(\dfrac{\overrightarrow{X_f} \cdot \overrightarrow{D_T}}{\left|\overrightarrow{X_f}\right| \cdot \left|\overrightarrow{D_T}\right|} \right) \end{cases} \tag{7}$$

The calculation results of D_T and δ_{ψ_T} are absolute, and the symbols of them are determined according to the right-hand rule.
(3) The UAV flight state space under attitude constraint
According to the definition of the UAV's attitude and the target yaw, the UAV's flight state space under attitude constraint in the horizontal plane can be obtained, as shown in formula (8).

$$S = \left\{ \delta_{\psi_c} \right\} \tag{8}$$

The symbol $\delta_{\psi_c} \in [-\pi, \pi]$ is the deviation between the UAV's yaw and the target azimuth. As shown in formula (9), it's the calculation function of δ_{ψ_c}.

$$\delta_{\psi_c} = \psi_T - \psi_c \tag{9}$$

The calculation result of δ_{ψ_c} is absolute, and the symbol is determined according to the right-hand rule.

2.4 The Flight Assessment Function of UAV under Endpoint Constraints

In the Markov decision process, the reward function determines the normal direction of the future evolution of the system and also determines the intention of the decision maker. For the problems under position and attitude constraint, the corresponding reward functions are designed separately.
(1) The reward function under position constraint
As described in Section 1.2, the autonomous maneuvering control under position constraint is designed to solve the problem: in the horizontal plane, the UAV can fly from an arbitrary position to the target position. Accordingly, there is an end condition for the task, as shown in formula (10).

$$\lim_{t \to \infty} \left| X_{UAV} - X_{TGT} \right| \leq D_{\min} \tag{10}$$

DOI: 10.13009/EUCASS2019-419

D_{min} is the minimum distance for the UAV to complete the task. According to the definition of the above formula, the corresponding reward function can be obtained, as shown in formula (11).

$$R(s,a) = \omega_{D_T} \cdot \frac{D_T^{\max} - D_T}{D_T^{\max}} + \omega_{N_a} \cdot \frac{N_y^{\max} - N_y}{N_y^{\max}} \tag{11}$$

$\omega_{D_T} \in [0,1]$ is the distance benefit coefficient; $\omega_{N_a} \in [0,1]$ is the maneuvering benefit coefficient; D_T^{max} is the maximum distance between the UAV and the target position; and N_y^{max} is the maximum turning overload.

(2) The reward function under attitude constraint

For the autonomous maneuvering control problem of the UAV under attitude constraint, the end condition of the task is as shown in formula (12).

$$\lim_{t\to\infty} |\psi_c - \psi_T| \le \delta_\psi^{\min} \tag{12}$$

δ_ψ^{min} is the minimum deviation between the UAV and the target yaw. According to the above formula, the corresponding reward function can be obtained, as shown in formula (13).

$$R(s,a) = \frac{\pi - |\delta_{\psi_T}|}{\pi} \tag{13}$$

(3) Termination reward function

In addition to the above two reward functions for the final mission goals, we define the reward on success or failure of the task, as shown in the formula (14).

$$R(s,a) = \begin{cases} 1.0, & \text{Successful Ending} \\ -1.0, & \text{Failed Ending} \end{cases} \tag{14}$$

3. The UAV's Autonomous Maneuvering Decision Algorithm based on Deep Reinforcement Learning

3.1 The Deep Reinforcement Learning Method

With the development of computer technology and electronic technology, artificial intelligence technology has advanced by leaps and bounds in recent years. As a popular direction, deep reinforcement learning has attracted the attention of scholars in various fields [11]. Reinforcement learning is a method that solves the problem of maximizing reward or achieving specific goals through learning strategies in the process of interacting with the environment. Figure 3 shows the normal framework of reinforcement learning.

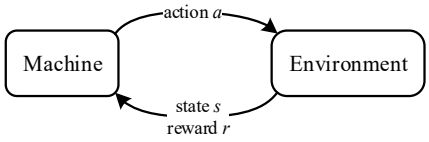

Figure 3: The normal structure of reinforcement learning

Deep reinforcement learning is a traditional reinforcement learning based algorithm. It uses deep learning to approximate the state-action value function and strategy function for solving the combined explosion problem in the case of large state space and action space. Figure 4 shows the structure of the Deep Deterministic Strategy Gradient (DDPG) method, which is a model-free, off-policy deep reinforcement learning method based on the Actor-Critic architecture and can solve the continuous control problem [12]. This algorithm consists of four parts: actor network $\mu(s;\theta^\mu)$, critic network $Q(s,a;\theta^Q)$, target actor network $\mu'(s;\theta^{\mu'})$ and target critic network $Q'(s,a;\theta^{Q'})$, and the replay buffer D. Compared with the traditional reinforcement learning algorithm, DDPG creates four new technologies:

DOI: 10.13009/EUCASS2019-419

experience replay mechanism, target network, batch normalization, additional noise. These methods solve the problems of deep reinforcement learning in practical applications. In the learning processes, the actor network makes the corresponding actions according to the environment state and appends additional noise, and then a tuple is constructed by the current state, action, reward and future state and saved into the experience replay buffer. Finally, sample randomly from replay buffer to obtain a training batch. The net parameters are optimized according to the optimal algorithms such as the gradient descent method and the target network parameters are smoothly updated.

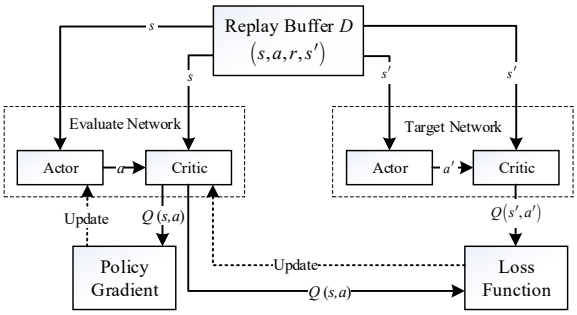

Figure 4: The structure of Deep Deterministic Policy Gradient

In general, before using the reinforcement learning method to solve the problem, it is necessary to establish a corresponding problem model based on Markov decision process theory. Therefore, based on the autonomous flight control model established in section 1, the deep deterministic policy gradient method is used to establish the autonomous maneuvering control algorithm of the UAV.

3.2 The UAV's Autonomous Maneuvering Control Function based on Neural Network

As mentioned earlier, DDPG is a kind of deep reinforcement learning algorithm based on the Actor-Critic framework. Figure 5 shows the normal structure of actor-critic deep reinforcement learning. In the training process, the dynamic evolution environment is used to generate the system state $s \in S$. The actor network gives actions $a \in A(s)$ according to the system state and executes it on the environment. In the processes, TD-error is used to optimize the critic network parameters. Similarly, the actor network parameters are optimized by utilizing $max\ Q\ (s,a)$.

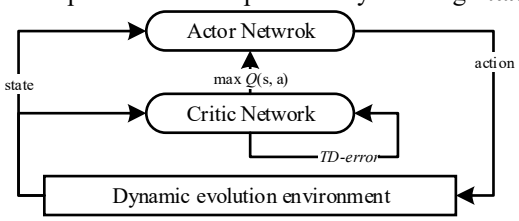

Figure 5: The normal structure of Actor-Critic deep reinforcement learning

According to the autonomous flight control model and the DDPG method described above, the neural network technology is used to design the two kinds of networks involved in the autonomous maneuvering control algorithm of UAV.

(1) Actor Network

The actor network $\mu(s;\theta^{\mu})$ mainly implements real-time decision under the current system state. The network input is the current system state $s \in S$, and the output is the action $a \in A(s)$ that the system should take. According to the UAV flight state space defined above, the number of network input nodes is $dim(S)$, and the number of output nodes is $dim(A)$. As shown in Figure 6, it is the normal structure of actor network.

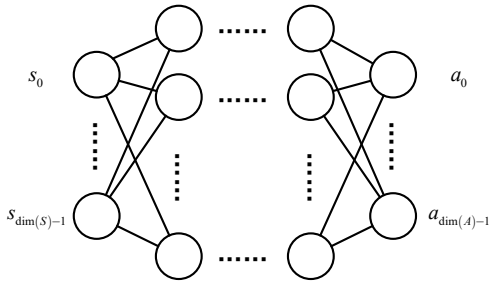

Figure 6: The normal structure of Actor network

DOI: 10.13009/EUCASS2019-419

(2) Critic Network

The critic network is used to evaluates the optimal degree of the current action $a \in A(s)$. The network input is $[s, a]$, and the network output is $Q(s, a)$. According to the flight state space and action space of UAV defined above, the number of network input nodes is $dim(S) + dim(A)$, and the number of network output nodes is 1. As shown in Figure 7, it is the normal structure of critic network.

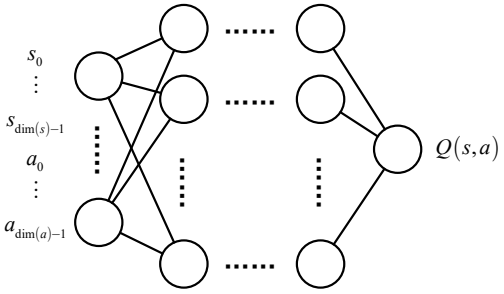

Figure 7: The normal structure of Critic network

When we call these networks, the state $s \in S$ and action $a \in A(s)$ need to be normalized according to the definition of the state space and action space described above. And then, they are input into the networks. In addition, for the target actor network $\mu'\left(s; \theta^{\mu'}\right)$ and target critic network $Q'\left(s, a; \theta^{Q'}\right)$, their structure are similar to $\mu(s; \theta^{\mu})$ and $Q(s, a; \theta^{Q})$. Only the parameters updating method is distinguished.

3.3 The UAV's Maneuvering Control Algorithm based on DDPG

As mentioned above, DDPG is a deep reinforcement learning method, which can solve the continuous decision problem well and has certain advantages in solving problems with large-scale state space and action space. The core of DDPG is the Actor-Critic structure, which has the advantages of two kinds of reinforcement learning methods including value based and policy gradient.

The critic network used in DDPG comes from the Q-Learning algorithm, which is an off-policy method. According to the definition of the utility function, that is the formula (2), a function describing the state-action can be obtained, as shown in the formula (15).

$$Q(s,a) = \mathbb{E}_{\pi}\left[v(s,\pi)\right] \tag{15}$$

Formula (15) is the state-action value function. Therefore, the optimal decision can be defined as the formula (16).

$$a_t = \arg\max_{a} Q(s_t, a) \tag{16}$$

The above equation indicates that the optimal decision is when the system state is $s_t \in S$. Therefore, the optimal strategy can be obtained by solving $Q(s, a)$. According to the formula (2) and the formula (15), it is easy to obtain an iterative formula of the Q-Learning method, as shown in the formula (17).

$$Q(s,a) = Q(s,a) + \alpha\left[r + \gamma \max_{a} Q(s',a) - Q(s,a)\right] \tag{17}$$

In formula (18), $s \in S$ is the current state of the system; $a \in A(s)$ is the current action; $r = R(s, a)$ is the current reward; and $s' \in S$ is the next state of the system. Based on the formula (17), the loss function of the network $Q(s, a; \theta^{Q})$ can be obtained, as shown in the formula (18).

$$L\left(\theta^{Q}\right) = \left[r + \gamma \max_{a'} Q'\left(s,a;\theta^{Q'}\right) - Q\left(s,a;\theta^{Q}\right)\right]^2 \tag{18}$$

According to the Q-Learning algorithm [13], the gradient of the loss function of the critic network can be obtained, as shown in formula (19).

DOI: 10.13009/EUCASS2019-419

$$\nabla_{\theta^Q} L\left(\theta^Q\right) = \mathbb{E}_{s,a,r,s'}\left[\left(r + \gamma \max_{a'} Q'\left(s,a;\theta^{Q'}\right) - Q\left(s,a;\theta^Q\right)\right)\nabla_{\theta^Q} Q\left(s,a;\theta^Q\right)\right] \tag{19}$$

Meanwhile, the actor network in DDPG is derived from the Policy Gradient algorithm, which is a policy-based reinforcement learning method. Its advantage over the value function is that the solution result is the optimal strategy. In reinforcement learning, the strategy is divided into two types: deterministic strategy and nondeterministic strategy. The mathematical definition is shown in formula (20).

$$\begin{cases} a \sim \pi(s), \text{Nondeterministic} \\ a = \mu(s), \text{Deterministic} \end{cases} \tag{20}$$

According to the DPG theorem [14], the gradient formula of the optimization objective function of the actor network can be obtained, as shown in formula (21).

$$\nabla_{\theta^\mu}\left[v(s,\mu)\right] = \mathbb{E}_{s,a,r,s'}\left[\nabla_a Q\left(s,a;\theta^Q\right)\nabla_{\theta^\mu}\mu\left(s;\theta^\mu\right)\right] \tag{21}$$

In addition, DDPG also defines an experience replay buffer *D* for storing historical data. At the same time, the data of buffer is used to train the actor network and the critic network. As shown in equation (22) is the definition of the elements in replay buffer *D*.

$$D = \left\{\left[s,a,r,s'\right]\right\} \tag{22}$$

$s \in S$ represents the current state of the system; $a \in A(s)$ is the current action; $r = R(s,a)$ is the current reward; and $s' \in S$ is the next state of the system.
In order to eliminate the problem of network divergence caused by poor data independence, based on the successful experience of DQN algorithm, DDPG follows the concept of target network, that is target actor network $\mu'\left(s;\theta^{\mu'}\right)$ and target critic network $Q'\left(s,a;\theta^{Q'}\right)$. At the same time, referring to the supervising learning, the parameters of the target networks are updated in a smooth update method, as shown in formula (23).

$$\begin{cases} \theta^{Q'} = \tau\theta^Q + (1-\tau)\theta^{Q'} \\ \theta^{\mu'} = \tau\theta^\mu + (1-\tau)\theta^{\mu'} \end{cases} \tag{23}$$

In equation (23), $\tau \in (0,1)$ is the hyperparameters updating coefficient.
Since the deterministic strategy is poorly exploratory during the training process, the DDPG uses a method of superimposing noise on the output of the actor network to solve the problem, as shown in formula (24).

$$a_t = \mu\left(s_t;\theta^\mu\right) + \mathcal{N}(t) \tag{24}$$

In formula (24), $\mathcal{N}(t)$ is the Ornstein–Uhlenbeck process [15].
According to the above all, the training processes of the UAV autonomous maneuvering control algorithm based on DDPG is shown in Table 1.

Table 1: The training processes of the UAV autonomous maneuvering control algorithm.

The UAV autonomous maneuvering control algorithm
Initialize the experience replay buffer *D*
Initialize the actor network $\mu(s;\theta^\mu)$, the target actor network $\mu'\left(s;\theta^{\mu'}\right)$, the critic network $Q(s,a;\theta^Q)$ and the target critic network $Q'\left(s,a;\theta^{Q'}\right)$
For *episode* = 1 to M:
Initialize the Ornstein – Uhlenbeck process $\mathcal{N}(t)$, and get the initial state s_0 of the UAV flight simulation

DOI: 10.13009/EUCASS2019-419

environment

For t = 1 to T:

According to $a_t = \mu(s_t;\theta^\mu) + \mathcal{N}(t)$ generate the action

Execute the action a_t and get the reward r_t

Meanwhile, get the next state s_{t+1} of the UAV flight simulation environment

And save the current information (s_t, a_t, r_t, s_{t+1}) into buffer D

Randomly sampling from D to get a group of samples $\{[s, a, r, s']\}$

Update parameters θ^Q and θ^μ according to the formula (18) and (21) separately

Update parameters $\theta^{Q'}$ and $\theta^{\mu'}$ according to the formula (23)

End For

End For

After training according to the process in Table 1, the corresponding optimal actor network can be obtained. During the process of testing, the network output can be directly used as the decision result, as shown in formula (25).

$$a = \mu\left(s;\theta^\mu\right), s \in S \tag{25}$$

4. Simulation & Analysis

According to the autonomous flight control model of UAV and the autonomous maneuvering control algorithm of UAV, two simulation experiments are designed, which are the autonomous flight simulation of UAV under position and attitude constraint in the horizontal plane. After finishing the corresponding training, we prove the effectiveness of the autonomous maneuvering control algorithm we proposed by analysing the training data and test results.

In this paper, the size of the simulation experiment spatial space is set to 50km×50km, and the height is 5km. According to the definition of the autonomous flight training environment and the autonomous maneuvering control algorithm of the UAV, the number of episodes is $M = 1000$ in this training, and the maximum number of decision times is $T = 500$ in a single episode. In addition, the decision cycle of the training environment is 0.1s.

4.1the Autonomous Flight Simulation of UAV Under Position Constraint

As shown in Figure 8, four testing results are shown after training. The first row is the flight trajectory of the UAV, and the second row is the corresponding reword change. The red solid point is the starting position; the green solid point is the end position; the dotted circle near the end position is the maximum range of the end point; the red curve is the flight trajectory of the UAV. The successful condition of the task under position constraint for the UAV is entering the endpoint ring successfully.

It can be seen that the UAV can fly from any starting position to the end position using a certain maneuvering policy. With the development of simulation, the reward grows gradually. These cases prove that the autonomous maneuvering control algorithm can solve the autonomous flight problem of UAV under position constraint in the horizontal plane.

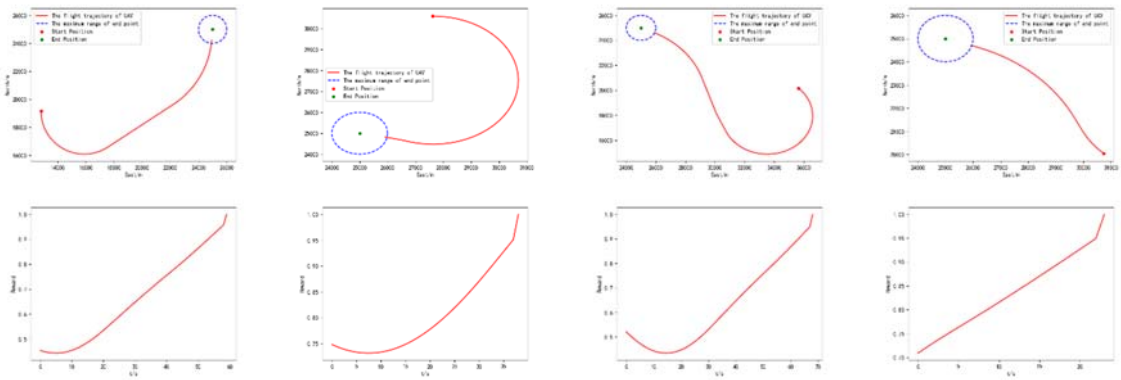

Figure 8: The testing results of autonomous flight simulation of UAV under position constraint

DOI: 10.13009/EUCASS2019-419

4.2The Autonomous Flight Simulation of UAV Under Attitude Constraint

As shown in Figure 9, four testing results are showed. The first row is the flight trajectory of the UAV, and the second row is about the change of the UAV's azimuth and the corresponding reward changing. The red solid point is start position; the green dotted line is the line of UAV's azimuth at the end of mission; the blue dotted line is the line of target azimuth; the red solid line is the flight trajectory of the UAV. In the azimuth change diagram, the red solid line is the change of the UAV's azimuth, and the green dotted line represents the target azimuth. The numbers in the legend represent the UAV's end azimuth angle and the target azimuth angle. The successful condition of the task under attitude constraint, the deviation between the UAV's azimuth and the target azimuth line is not greater than 3°.
It can be seen from the flight trajectory diagram that the UAV can change from any azimuth to a given target azimuth by using a certain maneuvering policy. For the corresponding azimuth change diagram, the UAV azimuth is always changing to the target azimuth, and the reward increases steadily. The test results show that the autonomous maneuvering control algorithm of the UAV can solve the autonomous flight problems of the UAV under attitude constraint in the horizontal plane.

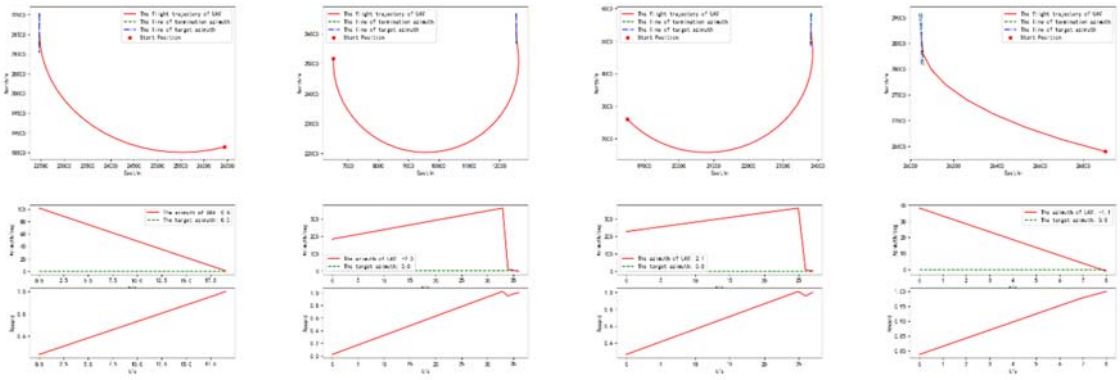

Figure 9: The testing results of autonomous flight simulation of UAV under attitude constraint

5. Conclusion

In this paper, we discuss the autonomous flight control problem of UAV under position constraint and attitude constraint. According to Markov decision process theory, we establish the autonomous flight simulation environment of UAV. Aiming to the problems under position constraint and attitude constraint, the corresponding UAV flight state space, action space and reward function are designed respectively. Finally, the UAV autonomous maneuvering control algorithm is designed based on the DDPG method, and the corresponding simulation training is finished. The test results prove that the UAV autonomous maneuvering control algorithm can solve the problems of autonomous flight control of UAV under position constraint and attitude constraint.

6. Acknowledgment

This work was supported by the Key Laboratory Project Foundation (6142504190105), the Innovative Talents Promotion Plan in Shaanxi Province (2017KJXX-15), the Science and Technology on Avionics Integration Laboratory and Aeronautical Science Foundation (20155153034).

References

[1] YIN Xinfan, ZHANG Guichuan, PENG Xianmin, LI Lei, and TIAN Bin. 2018. Intelligent development and application of military UAV technology. *Defense technology review.*

[2] Austin, Fred, et al. Game theory for automated maneuvering during air-to-air combat. *Journal of Guidance, Control, and Dynamics* 13.6(1990):1143-1149.

[3] Ferrari, S. , Anderson, M. , Fierro, R. , & Lu, W. . (2012). Cooperative navigation for heterogeneous autonomous vehicles via approximate dynamic programming. *Decision & Control & European Control Conference*. IEEE.

[4] Wong, Y. C. , & Sundareshan, M. K. . (1998). Data fusion and tracking of complex target maneuvers with a simplex-trained neural network-based architecture. *IEEE World Congress on IEEE International Joint Conference on Neural Networks*. IEEE.

DOI: 10.13009/EUCASS2019-419

[5] Liangqing, F. , Nan, J. , & Gendu, Z. . (1997). Distributed electricity maneuver expert system and its decision support system. *COMPUTER ENGINEERING*.
[6] Li, J. , Li, X. , Jiang, B. , & Zhu, Q. . (0). A Maneuver-Prediction Method Based on Dynamic Bayesian Network in Highway Scenarios. *The 30th China Control and Decision Conference.*
[7] Zhengmin, L. , Liang, A. , Changsheng, J. , & Qingxian, W. U. . (2010). Application of multistage influence diagram in maneuver decision-making of ucav cooperative combat. *Electronics Optics & Control, 17*(10), 10-13.
[8] Houenou, A. , Bonnifait, P. , Cherfaoui, V. , & Yao, W. . (2013). Vehicle trajectory prediction based on motion model and maneuver recognition. *IEEE/RSJ International Conference on Intelligent Robots & Systems*. IEEE.
[9] (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529-533.
[10] Nunes, L. G. N. , Solon Venâncio de Carvalho, & Rita de Cássia Meneses Rodrigues. (2009). Markov decision process applied to the control of hospital elective admissions. *Artificial intelligence in medicine, 47*(2), 159-171.
[11] Sutton, R. , & Barto, A. . (1998). Reinforcement learning: an introduction (adaptive computation and machine learning). *IEEE Transactions on Neural Networks, 9*(5), 1054.
[12] Lillicrap, T. P. , Hunt, J. J. , Pritzel, A. , Heess, N. , Erez, T. , & Tassa, Y. , et al. (2015). Continuous control with deep reinforcement learning. *Computer Science*.
[13] Watkins, C. J. C. H. . (1992). Technical note: q-learning. *Machine Learning, 8*.
[14] Silver, D. , Lever, G. , Heess, N. , Degris, T. , Wierstra, D. , & Riedmiller, M. . (2014). Deterministic policy gradient algorithms. *International Conference on International Conference on Machine Learning*. JMLR.org.
[15] Schoebel, R. , & Zhu, J. . (1998). Stochastic volatility with an ornstein-uhlenbeck process: an extension. *European Finance Review, volume 3*(1), 23-46.
[16] Wang, P. , Chan, C. Y. , & Li, H. . (2018). Automated driving maneuvers under interactive environment based on deep reinforcement learning.
[17] Lecun, Y. , Bengio, Y. , & Hinton, G. . (2015). Deep learning. *Nature, 521*(7553), 436.
[18] Ioffe, S. , & Szegedy, C. . (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift. *International Conference on International Conference on Machine Learning*. JMLR.org.