2020 International Conference on Unmanned Aircraft Systems (ICUAS)
Athens, Greece. September 1-4, 2020

A UAV Autonomous Maneuver Decision-Making Algorithm for Route Guidance

Kun Zhang, Ke Li, Jianliang He, Haotian Shi, Yongting Wang, and Chen Niu

***Abstract*— In order to improve the autonomy of UAV for route guidance, an UAV autonomous maneuver decision-making algorithm is proposed. The optimal UAV maneuver policy is developed in an interactive environment. Firstly, the UAV maneuver decision-making model based on MDPs was established, where the flight state space, the flight action space and the flight assessment function are designed. Then, we proposed the algorithm based on Double Deep Q-Learning with Prioritized Experience Replay. Finally, according to the simulation results, the efficiency and autonomy of algorithm we proposed is proved.**

***Keywords*— UAV, maneuver decision-making, markov decision processes, deep reinforcement learning**

I. INTRODUCTION

The unmanned aerial vehicle (UAV) is applied to all walks of life because of high mobility, great flying height and low cost. With the development of the electronic and UAV techniques in recent years, the performance of UAV has been improved rapidly in all aspects[1]. How to improve the autonomous capability of UAV flight and avoid human errors[2] becomes the research focus of researchers in various countries.

When the UAV flies from a specific state to a target point, the flight path should be planned in advanced. Then, the operator of UAV manipulates the UAV into flying to target point according to the original route. At present, some guidance and control methods could replace the operator of UAV. Usually, the traditional methods including matrix game[3], influence diagram[4], dynamic Bayesian network[5], approximate dynamic programming[6]-[7], expert system[8], and evolution algorithms[9] have been applied for solving this problem. However, matrix game and influence diagram methods need to establish a clear and complete problem model, and the process of modeling is complicated; dynamic Bayesian network needs to understand the problem fully, and has low adaptability in the face of unknown situations; approximate dynamic planning requires the complete state transition probability for the problem; the expert system requires researchers to construct a perfect rule base; evolution methods have low efficiency when it's used to solve the problems online. Because of the breakthrough progress in electronic technology and the rapid development of artificial intelligence technology, various artificial intelligence algorithms have been gradually applied to the decision-making field in recent years. Mnih V, et al proposed Deep Q-Learning Network (DQN)[10]-[11] that improved the algorithm's capacity which solved the decision-making problem with continuous state space. Then, Van Hasselt H, et al used Double Q-Learning[12] to replace the optimization target of DQN for overcoming overestimate action values under certain conditions, and thus, the Double Deep Q-Learning Network (DDQN)[13] is proposed. In order to utilize the diversity of historical data fully, Schaul T, et al introduced prioritized replay buffer into DDQN[14]-[15] ,that is PER-DDQN.

In this paper, we proposed a UAV Autonomous Maneuver Decision-Making algorithm for route guidance based on PER-DDQN to train the UAV for generating efficient maneuver under endpoint constraint in an interactive environment. Particularly, a specific form of critic function was designed by deep neural network and experience replay memory for storing historical data and generating training set based on prioritized replay buffer was constructed. Meanwhile, we designed a training framework based on DDQN. Additionally, the UAV maneuver decision-making model was established by the Markov Decision Processes (MDPs)[16]-[17] to describe the process of UAV maneuver decision-making. Especially, the flight state space, the flight action space, and the reward functions are designed. Finally, to verify the performance of the algorithm that we proposed, some simulation experiments were given. The simulation 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.

*Resrach supported by the Key Laboratory Project Foundation (6142504190105), Natural Science Basic Research Program of Shaanxi (2020JM-147), China Scholarship Council Foundation (201806295012), the Top International University Visiting Program for Outstanding Young Scholars of Northwestern Polytechnical University, the Seed Foundation of Innovation and Creation for Graduate Students in Northwestern Polytechnical University (ZZ2019021), the Innovative Talents Promotion Plan in Shaanxi Province (2017KJXX-15), the Science and Technology on Avionics Integration Laboratory and Aeronautical Science Foundation (20155153034).

Kun Zhang is with the Northwestern Polytechnical University, Shaanxi, 710072, P. R. China and Science and Technology on Electro-Optic Control Laboratory, Henan, 471009, P.R. China (e-mail: kunzhang@nwpu.edu.cn).

Ke Li is with the Northwestern Polytechnical University, Shaanxi, 710072, P. R. China (keli_nwpu@mail.nwpu.edu.cn)

Jianliang He is with the Science and Technology on Electro-Optic Control Laboratory, Henan, 471009, P.R. China (corresponding author, phone: 037963323363; e-mail: hejianlian_457@163.com).

Haotian Shi is with the Northwestern Polytechnical University, Shaanxi, 710072, P. R. China (e-mail: htshi@mail.nwpu.edu.cn).

Yongting Wang is with the Science and Technology on Electro-Optic Control Laboratory, Henan, 471009, P.R. China (e-mail: yongtingwang_457@163.com).

Chen Niu is with the Xi'an JiaoTong University, Shaanxi, 710061, P. R. China (e-mail: niuchen.xjtu@gmail.com).

II. THE UAV MANEUVER DECISION-MAKING MODEL BASED ON MDPS

A. The Markov Decision Processes Theory

The Markov decision process is an important theory to study the sequential decision-making and its mathematical basis is a stochastic process theory. It's used to describe the discrete dynamic environment and simulate a decision maker who observes a markovian environment periodically or continuously and makes decisions sequentially according to a specific policy.

In general, the Markov decision process theory[16]-[17] can be defined by a quintuple $\{T, S, A(s), P(\cdot \mid s,a), R(s,a)\}$, where T represents the decision time, S represents the system state space, $A(s)$ represents the system action space, the transition probability $P(\cdot \mid S, A)$ represents the probability distribution of the system state at next moment when the system used the action $a \in A(s)$ in the state $s \in S$, and the reward function $R(s,a)$ represents the benefit that the decision maker gets when the action $a \in A(s)$ is applied into the system with the state $s \in S$. Based on the Markov decision process theory, we can make a complete mathematical definition of the sequence decision problem.

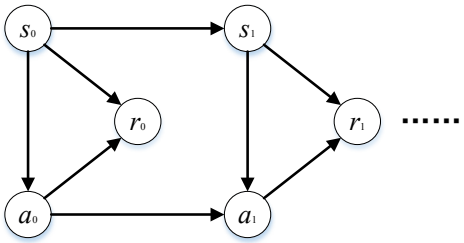


Figure 1. The structure of finite Markov decision processes.

As shown in Fig. 1. , the Markov decision process can be summarized as follows: the decision system maker who observes the initial state s_0 of system chooses the action a_0 and executes it according to a specific policy. Then, the system moves to the system state s_1 according to a certain transition probability $P(\cdot \mid s_0, a_0)$, and decision maker keeps repeating this above process until system state satisfies the termination condition. In this process, the decision maker earned rewards sequence $(r_0, r_1, \cdots)$. Among this process, the decision maker is stimulated by external rewards, and the reward received periodically is maximized by constantly updating the policy. The policy adopted by the decision maker is $\pi(s)$ and the utility function (at the state $s \in S$, the expected return obtained by adopting the strategy π) is $v(s,\pi)$. When current policy is optimal, it should meet the (1).

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

Based on the characteristics of the autonomous maneuver decision-making problem of UAV, we use infinite stage discount model as the utility function, as shown in (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 equation, $\gamma \in [0,1]$ is the future reward discount factor. According to the above equation, the optimal policy under the discount model can be obtained.

In the following content, according to many characteristics of Markov decision process theory, including state space S, action space $A(s)$, transition probability $P(\cdot \mid s,a)$ and reward function $R(s,a)$, the maneuver decision-making model of UAV is established.

B. The Flight Simulation Model of UAV

In the maneuver decision-making model of UAV, the flight simulation model of the UAV is used to achieve the motion simulation. According to the mathematical characteristics of model, the transition probability of the system is $P(\cdot \mid s,a) = 1$. The (3) is the flight simulation model of the UAV.

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

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 speed tilt angle of UAV is γ_c. v, θ and ψ_c indicate the aircraft speed, the track tilt angle of UAV and the track deflection angle of UAV respectively. The (x,y,z) represents 3 directions coordinates of UAV in the geographic coordinate system. Moreover, m, g and dt indicate the mass of UAV, the gravity acceleration and the simulation step-size. Thus, the state vector of UAV is (x,y,z,v,ψ_c,θ), and the control variables are (N_x, N_y, γ_c). During the simulation process, we use the numerical analysis to solve the differential equations.

C. The Flight State Space and Action Space of UAV

The autonomous maneuver decision-making 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.

According to the 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 attitude of UAV is described by Euler angles (ψ_c,θ,γ_c). Meanwhile, the target position is $X_{TGT}=(x,y,z)$. Fig. 2. shows the geometric relationship between UAV and target point involved in the maneuver decision-making problem that we proposed. The hollow dot located at the bottom left of figure represents the initial position X_{UAV} of UAV. Meanwhile, the solid dot in the top right of figure indicates the target point X_{TGT}. N and E are North and East directions respectively, and X_f represents the longitudinal direction of the UAV. D_T and ψ_T indicate the distance and azimuth between the UAV and the target position respectively and both symbols define the position of target point relative to UAV. Moreover, ψ_c represents the azimuth of UAV.

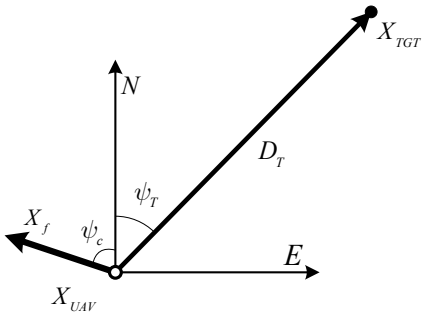


Figure 2. The diagram of geometric relationship between UAV and target point

1) The flight state space of UAV

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 (4).

$$S=\{D_T,\psi_T,\psi_c\} \tag{4}$$

In the above equation, $D_T\in[0,D_T^{\max}]$ and $\psi_T\in[-\pi,\pi]$ indicate the distance between UAV and target position and the azimuth of the target point relative to the UAV respectively. Moreover ψ_c represents the azimuth of UAV.

$$\begin{cases} D_T=|X_{UAV}-X_{TGT}| \\ \psi_T=\arccos\left(\dfrac{\overrightarrow{i_N}\cdot\overrightarrow{D_T}}{|\overrightarrow{D_T}|}\right) \end{cases} \tag{5}$$

As shown in (5), the calculation functions of D_T and ψ_T are described.

2) The flight action space of UAV

According to the formal of the control variable of the UAV's flight simulation model, a vector is used to describe the flight action of UAV, as shown in (6).

$$A(s)=\{a_0,a_1,\cdots,a_n\} \tag{6}$$

In the above equation, a_i represents a maneuver action. Based on the basic manipulation maneuver library[18], five maneuvers are designed. As shown in TABLE I. , a_i indicates level flight, turning right, turning left, turning right slightly and turning left slightly. Among the table, the a_i is defined by modifying the control variables of UAV including N_x, N_y and γ_c that are proposed in (3) and the units of these control variables are gravitational acceleration g and degree deg respectively.

TABLE I. THE DEFINITION OF BASIC MANIPULATION MANEUVER LIBRARY

Action Num	Control Variables			Action Name
	N_x	N_y	γ_c	
a_0	0.0	0.0	0.0	level flight
a_1	0.0	$N_y^{\max}$	0.0	turning right
a_2	0.0	$N_y^{\max}$	180.0	turning left
a_3	0.0	$0.5\times N_y^{\max}$	0.0	turning right slightly
a_4	0.0	$0.5\times N_y^{\max}$	180.0	turning left slightly

D. The Flight Assessment Function of UAV under Endpoint Constraints

In the Markov decision process, the reward function reflects the intention of decision maker and thus, determines the normal direction of algorithm's future evolution. Therefore, according to the relationship between UAV and target point, the reward function under position constraint is constructed. Meanwhile, in order to enhance the influence of final result after a series of decisions, we add the termination reward function into model.

1) The reward function under position constraint

As described in the above content, the autonomous maneuver decision-making 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 (7).

$$\left|D_T^k\right| \le D_{\min} \tag{7}$$

D_T^k indicates the k-th D_T . $D_{\min}$ represents the minimum distance for the UAV to complete the task. According to the definition of the above equation, we proposed the corresponding reward function after lots of pratice, as shown in (8).

$$R(s,a) = \left(\frac{\left|D_T^k\right| - \left|D_T^{k+1}\right|}{v_{UAV}^{\max} \cdot T_s} + 1 \right) \Big/ 2 \tag{8}$$

$\left|D_T^k\right|$ and $\left|D_T^{k+1}\right|$ indicates the k-th and the (k+1)-th D_T respectively. T_s indicates the step size of simulation. $v_{UAV}^{\max}$ represents the maximum velocity of UAV. As shown in (8), when the distance between UAV and target point shrinks, the decision maker can obtain a positive reward and this value is proportional to the reduction of distance between UAV and target point.

2) The termination reward function

In addition to the above reward function under position constraint, we define the reward on success or failure of the task, as shown in (9).

$$R(s,a) = \begin{cases} 1.0 & \text{Successful Ending} \\ 0.0 & \text{Failed Ending} \end{cases} \tag{9}$$

When the simulation period terminates, the algorithm will receive 1.0 if UAV reaches the target point successfully, and otherwise, it will receive 0.0. The failed termination is possibly By using this function, the final result will be utilized fully.

III. The UAV Autonomous Maneuver Decision-Making Algorithm based on PER-DDQN

A. The Double Deep Q-Learning with Prioritized Experience Replay Buffer

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. Reinforcement learning is a method that solves the problem that is maximizing reward or achieving specific goals through learning policy by interacting with the environment. Fig. 3. shows the normal structure of reinforcement learning.

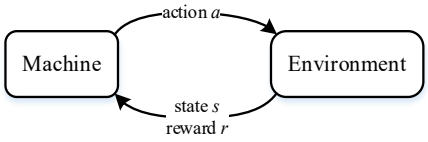


Figure 3. The normal structure of reinforcement learning

Deep reinforcement learning is a traditional reinforcement learning algorithm[19] combined with deep learning[20]. It uses deep neural network to approximate the state-action value function and policy function for solving the case of large state space. As shown in Fig. 4. , the double deep q-learning network with prioritized experience replay buffer (PER-DDQN[15]) consists of Prioritized Experience Memory mechanism, the training rule of Double Q-Learning and Critic Network. This method is a model-free off-policy deep reinforcement learning algorithm which uses prioritized mechanism to construct experience memory, and optimizes the state-action function approximator by double Q-learning. Meanwhile, the priority of each transition in the experience memory will be updated after the transition is sampled. In addition, in order to eliminate the bias induced by prioritized sampling due to the change of train set's distribution, the importance-sampling weights is used into the process of parameters optimization.

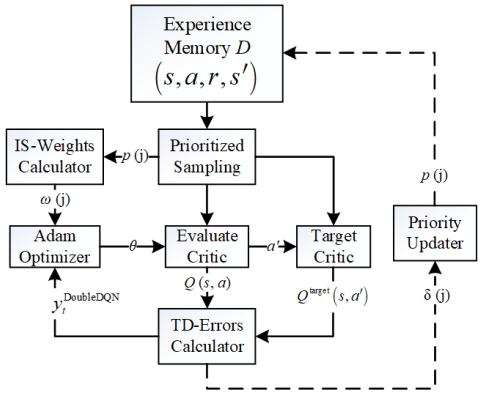


Figure 4. The structure of PER-DDQN

Therefore, we can establish the UAV autonomous maneuver Decision-making algorithm according to the characters of PER-DDQN. In general, before we use PER-DDQN, the problem's model based on MDPs should be constructed. In the content above, the UAV maneuvering decision-making model has been finished. Next, the UAV autonomous maneuver Decision-making algorithm will be designed.

B. The UAV Autonomous Maneuver Decision-making Algorithm based on PER-DDQN

As shown in Fig. 4. , the algorithm we proposed should contain three parts: the details of UAV autonomous maneuver decision-making function will be stated; the experience replay memory will be constructed based on prioritized experience replay buffer; the training process of UAV maneuver decision-making algorithm will be established.

1) The UAV Autonomous Maneuver Decision-making Function based on Neural Network

As mentioned earlier, the UAV autonomous maneuver decision-making algorithm is based on deep reinforcement learning theory. As shown in Fig. 5. , the critic network that is state-action function is used to solve the optimal action. Meanwhile, the TD-error[21] is adopted for training critic network. During the training process, the critic network

receives the state $s \in S$ generated by environment and outputs the optimal action $\arg\max_a Q(s,a)$.

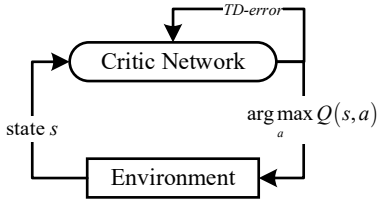


Figure 5. The structure of value-based reinforcement learning method

Therefore, the state-action function $Q(s,a;\theta)$ that is constructed by deep neural network and parameterized by vector θ is used to realize the UAV autonomous maneuver decision-making function. The critic network $Q(s,a;\theta)$ mainly implements real-time decision-making according to the state $s \in S$ of environment. As shown in (10), according to the current state $s_t \in S$ of environment, $Q(s,a;\theta)$ gives the optimal action $a_t \in A(s)$.

$$a_t = \arg\max_a Q(s_t, a; \theta) \tag{10}$$

Fig. 6. shows the structure of critic network. The input of $Q(s,a;\theta)$ is the state of environment, and the number of network input is the state dimension. In addition, the network output is the corresponding value $Q(s,a;\theta)$ of each action under current state, and the number of network output is the action dimension.

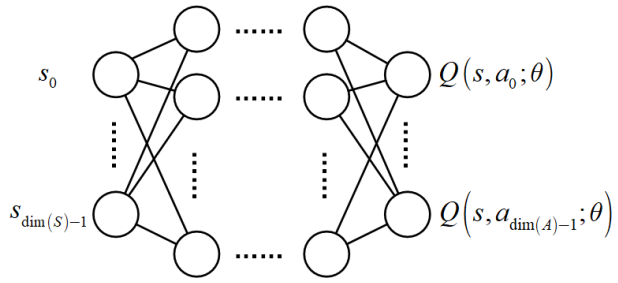


Figure 6. The structure of critic network

Moreover, in order to eliminate network divergence caused by the poor data independence, the target network $Q(s,a;\theta^-)$ is introduced into algorithm based on double Q-learning. Meanwhile, the structure of target network $Q(s,a;\theta^-)$ is same as $Q(s,a;\theta)$ and the parameters vector θ^- of target network is updated from time to time.

2) The Experience Replay Memory based on Prioritized Experience Replay Buffer

The experience replay memory is used to store historical data for the purpose of generating training set. As shown in (11), the elements of replay buffer memory *D* are defined.

$$D = \{s, a, r, s'\} \tag{11}$$

D indicates experience replay memory, and *s a r* and s' represent the current state of the system, the optimal action generated by $Q(s,a)$, the reward returned by environment and the next state of environment respectively. But uniform sampling can't utilize the historical data fully because the valuable transition at the end may appear at the beginning. Thus, the prioritized experience replay is used to sampling. The probability of being sampled for each transition is defined in (12).

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha} \tag{12}$$

$P(i)$ indicates the probability of being sampled for *i*-th transition. p_i and α represent the prioritization of transition and how much prioritization is used respectively. The proportional prioritization p_i is calculated by (13).

$$p_i = |\delta_i| + \varepsilon \tag{13}$$

As shown in (13), ε is a small positive constant that prevents the edge-case of transitions not being revisited once their error is zero. δ_i indicates the TD-error of *i*-th transition and it's calculated by (14), according to double Q-learning[12].

$$\delta_i = r_i + \gamma_i Q\left(s_i', \arg\max_a Q(s_i', a; \theta); \theta^-\right) - Q(s_i, a_i; \theta) \tag{14}$$

The transition (s_i, a_i, r_i, s_i') is sampled from memory *D*. Moreover, because prioritized sampling changes the distribution of the training set, we use importance-sampling (IS) weights to correct the bias.

$$\omega_i = \left(\frac{1}{N} \cdot \frac{1}{P(i)}\right)^\beta \tag{15}$$

The ω_i fully compensates for the non-uniform probabilities $P(i)$ if $\beta = 1$. The *N* indicates the capacity of experience memory *D*. These weights can be folded into algorithm update by using $\omega_i \delta_i$ instead of δ_i. For the stability of training, the weights should be normalized by $\omega_i / \max_j \omega_j$ so that they only scale the update downwards. Thus, the (15) could be simplified, as shown in (16).

$$\omega_i = \left(\frac{\min_j P(j)}{P(i)}\right)^\beta \tag{16}$$

Because the small bias could be ignored at the beginning, a updating schedule could be defined that linearly anneal β from an initial value β_0 to 1.

3) The Training Process of UAV Autonomous Maneuver Decision-making Algorithm

As mentioned above, we can solve the problem by optimizing the (2). The value-based reinforcement learning method can be used to solve the MDPs based problem. According to current lectures, we can use the double Q-learning and DQN to construct the training process of UAV autonomous maneuver decision-making algorithm. According to the definition of utility function, a function describing the state-action can be obtained, as shown in the (17).

$$Q(s,a) = \mathbb{E}_{\pi}\left[v(s,\pi)\right] \tag{17}$$

Therefore, the optimal action can be obtained by (18)

$$a_t = \arg\max_a Q(s_t, a) \tag{18}$$

The a_t indicates the optimal action when the environment gave the state $s_t \in S$. Thus, the optimal policy can be obtained by optimizing the $Q(s,a)$. Therefore, according to the definition of utility function, it's easy to gain an iterative equation of the double Q-Learning method, as shown in (19).

$$\begin{aligned} &Q(s,a;\theta) += \\ &\alpha^s\left[r + \gamma Q\left(s, \arg\max_a Q(s,a;\theta); \theta^-\right) - Q(s,a;\theta)\right] \end{aligned} \tag{19}$$

The $Q(s,a;\theta)$ indicates the state-action function approximated by deep learning as defined in Fig. 6. . The $Q(s,a;\theta^-)$ represents the target network that is same as the $Q(s,a;\theta)$. The $\gamma \in (0,1)$ is the future reward decay factor. Next, the training target of $Q(s,a;\theta)$ could be defined in (20).

$$y_t^{DoubleDQN} = r_t + \gamma Q\left(s_t', \arg\max_a Q(s_t', a; \theta); \theta^-\right) \tag{20}$$

As shown in (20), s_t' and r_t indicate the state of environment and the reward returned by environment when current state of environment is s_t and the action a_t is enabled in the environment. Then, based on the content above, the parameters-change of network can be obtained as (21).

$$\Theta = \Theta + \omega_j \cdot \delta_j \cdot \nabla_\theta Q(s_j, a_j; \theta), j = 1, 2, \cdots, k \tag{21}$$

The Θ indicates the parameter vector of network $Q(s,a;\theta)$. Before the network is training, the training set including k transitions is constructed by sampling from experience memory D. Then, the parameters-change is accumulated by (21), and the TD-error δ_j can be calculated by (22).

$$\delta_j = r_j + \gamma Q\left(s_j', \arg\max_a Q(s_j', a; \theta); \theta^-\right) - Q(s_j, a_j) \tag{22}$$

Finally, the optimal policy that is state-action function $Q(s,a;\theta)$ can be obtained by using Adam optimizer[22] according to $\omega_j \delta_j$. According to the content above, the training process of UAV autonomous maneuver decision-making algorithm is shown in TABLE II. .

TABLE II. THE TRAINING PROCESS OF UAV AUTONOMOUS MANEUVER DECISION-MAKING ALGORITHM

The UAV Autonomous Maneuver Decision-Making Algorithm	
1:	Input: minibatch *k*, step-size η, training period *K*, memory capacity *N*, exponents α and β, target network replace period *C*, maximum step *T*, maximum period *M*.
2:	Initialize experience memory *D*, the network $Q(s,a;\theta)$ and the target network $Q(s,a;\theta^-)$.
3:	**for *m* = 1 to *M*:**
4:	Observe s_0 and choose the optimal maneuver action a_0 according to (10)
5:	**for *t* = 1 to *T*:**
6:	Observe s_t and choose the optimal maneuver action a_t according to (10)
7:	**if t mod *K* ≡ 0:**
8:	**for *j* = 1 to *k*:**
9:	Sample transition $j \sim P(j)$ according to (12).
10:	Compute importance-sampling weight ω_j according to (16).
11:	Compute TD-error δ_j according to (22).
12:	Update the priority of transition $p_j = \lvert\delta_j\rvert$.
13:	Accumulate parameters-change Θ according to (21).
13:	**end for**
14:	Update weights θ by using Adam through learning rate η according to the target Θ.
15:	Copy parameters into the target network $Q(s,a;\theta^-)$ every *C* steps.
16:	**end if**
17:	**end for**
18:	Reset environment.
19:	**end for**

IV. EXPERIMENTS

A. The Implementation Structure of Experiments

Based on the content mentioned above, we can construct some experiments to verify the effectiveness of the algorithm we proposed.

Fig. 7. represents the implementation structure of simulation. The environment module indicates the realization of UAV maneuver decision-making model which gives the flight state *s* after the maneuver action *a* is executed, and calculated the reward *r*. Among the module, the maneuver action actuator realizes the optimal action generated by algorithm; the flight state generator calculates the flight state according to flight simulation; the flight assessment function computes the reward according to the information of flight simulation. On the other hand, the algorithm module

represents the implementation of UAV autonomous maneuver decision-making algorithm, which generates the optimal action according to the state of environment and is optimized gradually in order to maximize the expectation of reward. Inside the module, prioritized experience memory stores the historical data and samples the training set from memory.

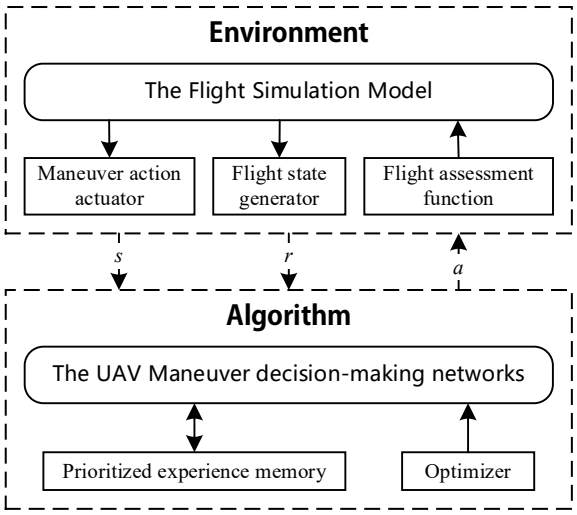


Figure 7. The Implementation Structure of Simulation

B. The Preprocessing of Input Data and The Network Architecture

Because each element of the flight state has different scales, the different dimension data should be normalized to uniform format. Thus, in addition to the flight state generated by the UAV maneuver decision-making model, the flight state should be preprocessing before the data is applied into the algorithm.

As shown in TABLE III. , these are the ranges of flight state elements. Thus, D_T should be divided by $D_T^{\max}$ which can be computed according to $v_{UAV}^{\max} \cdot T_s$; ψ_T and ψ_c are divided by 2π respectively. Finally, the input of the UAV autonomous maneuver decision-making network is the new tensor combined by converted flight state.

TABLE III. THE ELEMENTS DEFINITION OF FLIGHT STATE

ID	State element	Range
0	D_T	$[0, D_T^{\max}]$
1	ψ_T	$(0, 2\pi]$
2	ψ_c	$(0, 2\pi]$

According to the content of UAV autonomous maneuver decision-making network, the details of network can be defined in TABLE IV. . The input layer consists of 3 units which is same as the dimension of UAV flight state space. Moreover, the hidden layers are all fully-connected liner layers and consist of 20, 40, 40, 40 rectifier units respectively according to previous experience. The output layer is also fully-connected liner layer with 5 units which indicate the different Q values of different actions.

TABLE IV. THE DETAILS OF UAV AUTONOMOUS MANEUVER DECISION-MAKING NETWORK

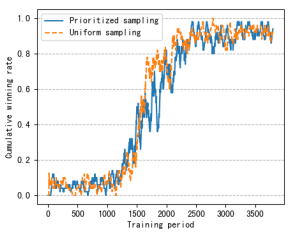
Layers	Layer structure	
	Units num	*Activation func*
Input layer	3	-
Hidden layer 1	20	ReLU
Hidden layer 2	40	ReLU
Hidden layer 3	40	ReLU
Hidden layer 4	40	ReLU
Output layer	5	-

C. Simulation Results & Analysis

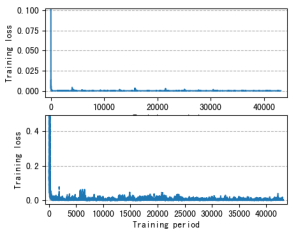
1) Parameters Settings

The UAV autonomous maneuver decision-making algorithm is a deep reinforcement-based method which uses prioritized experience memory to generate training set. In contrast to uniform sampling, the prioritized sampling could lead to exponential speed-ups. Moreover, the convergence of UAV autonomous maneuver decision-making function network is more stable than the same algorithm with uniform experience memory.

As shown in Fig. 8. , they are the comparison of training process between PER-DDQN and UER-DDQN (Double Deep Q-Learning Network with Uniform Experience Replay), where (a) and (b) indicate the comparison of cumulative winning rate and the training loss respectively. In (a), the y-axis cumulative winning rate represents the proportion of success count in the last 100 simulation and the x-axis training period is the episode of simulation. In (b), the top represents the training loss of PER-DDQN, and the bottom indicates the training loss of UER-DDQN. The y-axis training loss is the difference between network output and training target. The x-axis training period is different from that in (a) because there may be many training episodes during a simulation episode, and thus it represents the training episode of critic network. Though the difference of cumulative winning rate between PER-DDQN and UER-DDQN is small, the convergence process of PER-DDQN is stabler than the UER-DDQN's by analyzing (b).



(a) The comparison of cumulative winning rate between PER-DDQN and UER-DDQN



(b) The comparison of training loss between PER-DDQN and UER-DDQN

Figure 8. The comparison of training process between PER-DDQN and UER-DDQN

As shown in Fig. 9. , it's the comparison of cumulative winning rate of UAV autonomous maneuver decision-making algorithm under different beta0 (β_0). All the training processes of algorithms show that these converge to the expected direction, there is cyclical volatility during the training processes. But there is the least cyclical volatility during the training process among these algorithms when the algorithm is under the condition $\beta_0 = 0.4$.

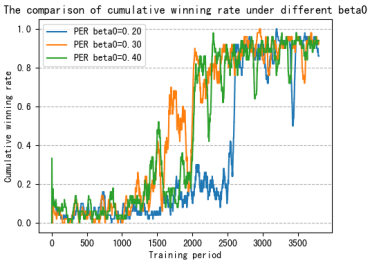


Figure 9. The comparison of cumulative winning rate generated by PER-DDQN under the different beta0

Thus, the parameter settings of the algorithm are shown in the TABLE V. . The capacity of experience memory is set to 10000, and the α which indicates how much prioritization is used is 0.6. Meanwhile, the initial value β_0 of the extent that compensates for the non-uniform probabilities, and the increment of β is 0.0001.

TABLE V. THE IMPORTANT HYPERPARAMETERS OF ALGORITHM.

Hyperparameter	Value
Capacity of memory	10000
α	0.6
β_0	0.4
β_{inc}	0.0001

In addition, as shown in TABLE VI. , these are some parameters of simulation. The UAV is limited in the area which is about $50km \times 50km$, in which the initial state of UAV is generated randomly at the beginning of every simulation period. The decision cycle is $1.0s$ in the simulation, in which the algorithm could make 1000 decisions at most. In order to test the training result, we run 5000 simulation and analyze the simulation results.

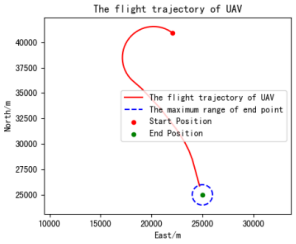
TABLE VI. THE PARAMETERS OF SIMULATION.

Parameter	Content
Flight area	$50km \times 50km$
Simulation step	$1.0s$
Training steps Maximum	1000
Training episodes num	3000

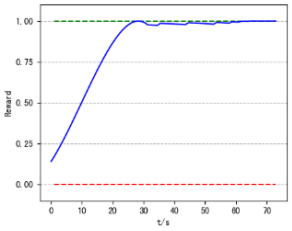
2) Results Analysis

According to the simulation results, we run 5000 simulations, among which the number of successful results is 4564, and the proportion of successful results is about 91.28%. Moreover, the average time of decision for training results is $0.438ms$ during the testing process.

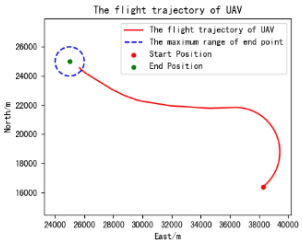
As shown in Fig. 10. , these are four representative test results selected from 5000 simulation results. The initial positions of UAV among these results distributed uniformly near the target point. And the UAV starts with the different azimuth and position. Thus, these four results could reflect the performance of the training result throughout the flight state space.



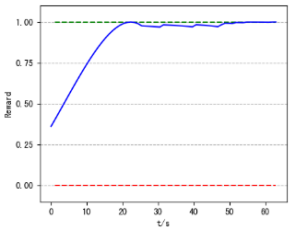
(a1) The flight trajectory diagram of UAV in simulation 1



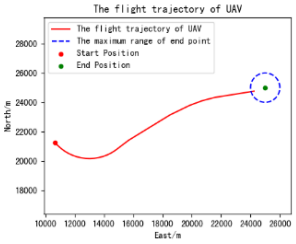
(a2) The trend diagram of reward in simulation 1



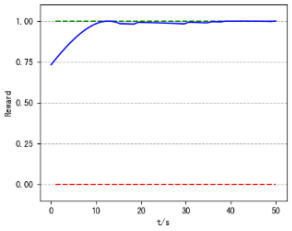
(b1) The flight trajectory diagram of UAV in simulation 2



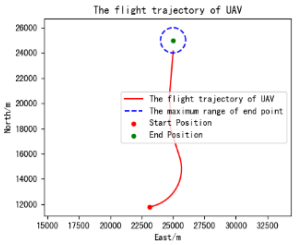
(b2) The trend diagram of reward in simulation 2



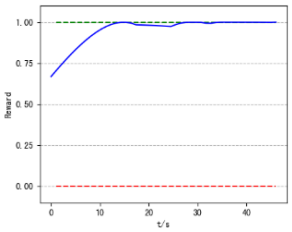
(c1) The flight trajectory diagram of UAV in simulation 3



(c2) The trend diagram of reward in simulation 3



(d1) The flight trajectory diagram of UAV in simulation 4



(d2) The trend diagram of reward in simulation 4

Figure 10. The simulation results of UAV autonomous maneuver decision-making algorithm for route guidance under position constraint

In Fig. 10. , (a1) ~ (d1) show the flight trajectory diagrams of UAV, and (a2) ~ (d2) indicate the trend diagrams of reward. In the flight trajectory diagrams, the red solid point is start position; the green solid point surrounded by blue dotted circle is end position; the red solid line is the flight trajectory of UAV; the horizontal and vertical axes indicate East and North directions respectively. In the trend diagrams, the blue

solid line represents the changing curve of reward; horizontal line $y=1.0$ is the successful reward; horizontal line $y=-1.0$ is the failed reward; the horizontal and vertical axes indicate simulation time that is equal to actual travel and reward respectively. When the UAV enters the range of blue dotted circle, the mission is finished successfully.

(a) The result of simulation 1: The distance from start position to end position is about 15*km*, and the initial azimuth of UAV away from the target point is more than 100°. The azimuth of UAV is adjusted towards target point by the Left Turn maneuver. And then, the UAV couldn't change the control variables until the UAV reach the target point. Correspondingly, the reward increases gradually until achieving 1.0 at the beginning of flight. Then, the reward is kept near 1.0 and the UAV receives 1.0 finally.

(b) The result of simulation 2: The UAV starts the position whose distance is about 16.4*km* from the target point, and the initial azimuth deviated from the target of UAV is about 90°. The UAV turns left until it's towards target, and then it maintains level flight until reaching target. On the other hand, the reward increases gradually, and keeps near 1.0. Finally, the UAV receives 1.0.

(c) The result of simulation 3: The distance between initial position of UAV and target is about 14*km*, and the UAV locates to the west of target. The UAV turn left slightly towards target. When the UAV is close to the line between the initial position and target, the UAV turn right slightly until it's towards target. Finally, the UAV reaches target successfully. In addition, the trend of reward also reflects the process of flight. It can't increase gradually until near 1.0, and the UAV receives 1.0 finally.

(d) The result of simulation 4: Similar to simulation 3, the UAV starts from 13*km* away from target. The UAV turn left slightly until it's towards target, and it reaches target finally. Correspondingly, the reward received by UAV increases steadily, and is kept near 1.0. At the end of flight, the reward is 1.0.

V. Conclusion

In this paper, we proposed the UAV autonomous maneuver decision-making algorithm aiming at the route guidance. Firstly, we established the UAV maneuver decision-making model based on MDPs, and constructed an environment used to generate training data. Thus, the flight state space and flight action space of UAV are designed, and the flight assessment function under position constraint is also designed for generating the reward function. Secondly, according to the PER-DDQN, the UAV autonomous maneuver decision-making algorithm is proposed. Finally, it's proved that the algorithm we proposed could solve the problem we designed above by simulation results. Thereby, the method we proposed could improve the autonomy of UAV efficiently, and it's one of the key issues for improving the intelligence of UAV.

References

[1] Xinfan Y, Guichuan Z, Xianmin P, et al. "Intelligent development and application of military UAV technology," *Defense Technology Review*, vol. 39, no. 5, Oct. 2018.

[2] Huang C Q. "Research on key technology of future air combat process intelligentization," *Aero Weaponry*, vol. 26, no. 1, pp. 11-19, Feb. 2019.

[3] Austin F, Carbone G, Hinz H, et al. "Game theory for automated maneuvering during air-to-air combat," *Journal of Guidance Control & Dynamics*, vol.13, no. 6, pp. 1143-1149, Jan. 1990.

[4] Q. Pan et al., "Maneuver decision for cooperative close-range air combat based on state predicted influence diagram," *IEEE International Conference on Information and Automation*, Macau, 2017, pp. 726-731.

[5] C. Lu, Z. Zhou, H. Liu and H. Yang, "Situation assessment of far-distance attack air combat based on Mixed Dynamic Bayesian Networks," *37th Chinese Control Conference*, Wuhan, 2018, pp. 4569-4574.

[6] Mcgrew J S, How J P, Williams B, et al. "Air-combat strategy using approximate dynamic programming," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 5, pp. 1641-1654, May 2012.

[7] Jun Fang, Limin Zhang, Wei Fang and Tao Xu, "Approximate dynamic programming for CGF air combat maneuvering decision," *2nd IEEE International Conference on Computer and Communications*, Chengdu, 2016, pp. 1386-1390.

[8] R. R. Mitchell, "Embedding a tactics expert system into air combat simulation software," *Proceedings of the IEEE National Aerospace and Electronics Conference*, Dayton, OH, USA, 1989, vol. 3, pp. 1027-1033.

[9] Mulgund, Sandeep, K. Harper, and G. Zacharias. "Large-Scale Air Combat Tactics Optimization Using Genetic Algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 1, pp. 140-142, Jan. 2001.

[10] Mnih V, Kavukcuoglu K, Silver D, et al. "Playing atari with deep reinforcement learning," *arXiv preprint*, arXiv:1312.5602, 2013.

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

[12] Hasselt H V. "Double Q-learning," *Advances in neural information processing systems*, Vancouver, Canada, 2010, pp. 2613-2621.

[13] Van Hasselt H, Guez A, Silver D. "Deep reinforcement learning with double q-learning," *30th AAAI conference on artificial intelligence*, Phoenix, Arizona, USA, 2016.

[14] Andre D, Friedman N, Parr R. "Generalized prioritized sweeping," *Advances in Neural Information Processing Systems*, Madison, Wisconsin, USA, 1998, pp. 1001-1007.

[15] Schaul T, Quan J, Antonoglou I, et al. "Prioritized experience replay," *International Conference on Learning Representations*, San Juan, Puerto Rico, 2016.

[16] Song H, Liu C C, Lawarree J, et al. "Optimal electricity supply bidding by Markov decision process," *IEEE Trans. on Power Systems*, vol. 15, no. 2, pp. 618-624, May 2000.

[17] Yang Q M, Zhang J D, Shi G Q. "Modeling of UAV path planning based on IMM under POMDP framework," *Journal of Systems Engineering and Electronics*, vol.30, no. 3, pp. 454-554, June 2019.

[18] T. Sun, S. Tsai, Y. Lee, S. Yang, S. Ting, "The Study on Intelligent Advanced Fighter Air Combat Decision Support System," *IEEE International Conference on Reuse and Integration*, Waikoloa Village, HI, USA, 2006, pp. 39-44.

[19] Sutton R S, Barto A G. *Reinforcement learning: An introduction*. MIT press, 2018, pp. 1-4.

[20] LeCun Y, Bengio Y, Hinton G. "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, May 2015.

[21] Tesauro G. "Temporal difference learning and TD-Gammon," *Communications of the ACM*, vol. 38, no. 3, pp. 58-68, 1995.

[22] Kingma D P, Ba J. "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations*, San Diego, CA, USA, 2015.