Journal of Systems Engineering and Electronics
Vol. 31, No. 4, August 2020, pp.734 – 742

Deep reinforcement learning and its application in autonomous fitting optimization for attack areas of UCAVs

LI Yue[1,2], QIU Xiaohui[2,*], LIU Xiaodong[3], and XIA Qunli[1]

1. School of Aerospace Engineering, Beijing Institute of Technology, Beijing 100081, China;
2. Science and Technology on Electro-Optic Control Laboratory, Luoyang 471000, China;
3. Beijing Aerospace Automatic Control Research Institute, Beijing 100854, China

Abstract: The ever-changing battlefield environment requires the use of robust and adaptive technologies integrated into a reliable platform. Unmanned combat aerial vehicles (UCAVs) aim to integrate such advanced technologies while increasing the tactical capabilities of combat aircraft. As a research object, common UCAV uses the neural network fitting strategy to obtain values of attack areas. However, this simple strategy cannot cope with complex environmental changes and autonomously optimize decision-making problems. To solve the problem, this paper proposes a new deep deterministic policy gradient (DDPG) strategy based on deep reinforcement learning for the attack area fitting of UCAVs in the future battlefield. Simulation results show that the autonomy and environmental adaptability of UCAVs in the future battlefield will be improved based on the new DDPG algorithm and the training process converges quickly. We can obtain the optimal values of attack areas in real time during the whole flight with the well-trained deep network.

Keywords: attack area, neural network, deep deterministic policy gradient (DDPG), unmanned combat aerial vehicle (UCAV).

DOI: 10.23919/JSEE.2020.000048

1. Introduction

In the future battlefield environment, preserving air superiority is very important and that unmanned combat aerial vehicles (UCAVs) joining the air combat will realize real zero casualty for pilots [1 – 4]. UCAVs are widely used in the future of air combat. Combined with advanced technologies, future UCAVs will be able to operate in highly contested airspace in pre-emptive and reactive roles such as suppression and destruction of enemy air defences, penetrating surveillance, and strike of high-value targets [5,6]. Furthermore, future capabilities are likely to extend on the operational speeds and maneuverability requirements to include supersonic regimes [7,8]. In the environment of high altitude and high speed, improving the accuracy of UCAVs' attack area fitting and autonomous decision-making ability to adapt to environmental changes become the key to the future development of UCAVs [9 – 12].

The value of the attack area is a key parameter that affects the operational performance of UCAVs [13]. The value of the attack area refers to the range where the missile can hit and damage the target with a certain probability under certain environmental parameters, including the height, velocity of the missile and the target, the ballistic inclination angle, the entry angle and the off-axis angle. The range includes the farthest distance and the nearest distance, which are called the far and near bounds in this paper. In the air combat environment, the attack area value of UCAV is obtained by the neural network acquired by learning a lot of data off-line. UCAVs today usually use the back propagation (BP) neural network fitting strategy to obtain the high accuracy value of the attack area [14,15]. However, in the future of air combat the environment will change rapidly. The adaptability of neural network and the accuracy of the fitting values decrease in the complex battlefield environment for the reason of fixed network structure and training data. Because the data is obtained from a fixed environment, the neural network obtained from off-line learning is only applicable to a fixed environment [16]. When the attack area is fitted in an environment different from the one from which the data is obtained, the fitting accuracy will inevitably decrease. The range of the real attack area will also change, and the error between the value obtained by neural network and the real value will change too. How to improve the strain capacity on neural network fitting for the attack area of UCAV becomes the key to enhancing the future air combat capacity. However, it is impossible for the aircraft to judge the air com-

Manuscript received November 12, 2019.
*Corresponding author.
This work was supported by the Key Laboratory of Defense Science and Technology Foundation of Luoyang Electro-optical Equipment Research Institute (6142504200108).

bat environment and update the fitting values of the attack area autonomously by the common neural network strategy [17 – 19].

The simple BP neural network strategy cannot cope with complex environmental changes and autonomously optimize decision-making problems. To solve these problems, the following measures can be taken. First, collect large amounts of simulation data in the attack area and obtain the network that can be used online by means of deep learning. Second, change the air combat environment, get the true value of the attack area in the new environment, and use the true value to correct the simulation value of the online network. Third, make the UCAV independently correct the attack area value by means of reinforcement learning. As an unmanned combat unit, UCAV needs to independently identify the environment in future complex air combat and use the correct optimization method to solve the problem of neural network inadaptability. This process involves data fitting and autonomous decision making. Deep learning (DL) is widely used in UCAV data processing [20 – 23]. Reinforcement learning (RL) is widely used in UCAV autonomous decision making [24 – 26].

The strategy of deep RL is introduced to solve the problem that how UCAVs make an autonomous decision on adaptation of network when the air combat environment changes. In this paper, we solve the problem of real-time autonomous fitting optimization for UCAVs' attack area in future battlefield environment by the deep deterministic policy gradient (DDPG) algorithm, which is a kind of deep RL algorithm. According to different continuous environment models, we can design the DDPG algorithm framework based on the python + tensorflow platform. The whole paper is divided into five section. The first section summarizes the role of UCAV and artificial intelligence technology in the UCAV field. The second section introduces the technological progress of DL, RL and deep Q network (DQN). In Section 3, BP neural network, DQN and DDPG algorithms are modeled and illustrated. In Section 4, simulation parameters are set and several algorithms mentioned in Section 3 are simulated and verified. Section 5 gives the conclusion about the simulation.

2. DL, RL, DQN and DDPG

DL is well known for its superhuman proficiency in air combat purposes where it is necessary to employ advanced computer intelligence that learns from data and makes intelligent decisions aimed at increasing profitability and sustainability. In future air combat environment, precise attack area value fitting of UCAVs still needs neural networks. Collecting and processing data is still important for aircraft. Because attack area value fitting needs large amounts of data and complicated network structure, the traditional monolayer neural network is difficult to deal with the complex data structure. Using the deep neural network is the key to solving this problem. The machine learning algorithm based on deep networks is called DL algorithm. Currently, many fields, such as image and voice recognition and information recognition, belong to the field of DL.

RL is another kind of artificial intelligence. It is RL that UCAVs need to improve the autonomous decision-making ability. RL is a branch of machine learning, where the machines gradually learn control behaviors via self-exploration of the environment [27]. RL employs an actor that iteratively interacts with the environment and modifies its control actions to maximize rewards received from the environment. The main advantage of the RL algorithms is that it learns to optimize control policies by exploration of the environment independent of the linearity or multivariability of the system [28]. The learned policy is obtained from numeric data of the reaction system, and it does not require parameter-tuning or real-time optimization, which allows RL easily adaptable to different control tasks once the framework is established. RL refers to a class of learning algorithms where a scalar evaluation of the performance of the network is available from the interaction of the network with the environment. RL aims at maximizing the expected evaluation by adjusting the parameters of the network.

In order to solve the problem of UCAV autonomous optimization of the far boundary of the attack area, we need to use both DL and RL. The DQN algorithm realizes the combination of RL and DL for the first time and makes remarkable achievements in practical application. In particular, the DQN algorithm introduces the technology of the objective function, the objective network and the experience playback in a pioneering way, which lays a solid foundation for the further development of deep RL. However, the DQN algorithm also has some limitations in practical application. For example, the DQN algorithm cannot deal with the continuous motion control problem, which greatly limits the application range of the DQN algorithm. Considering the shortcomings of the DQN algorithm, researchers have successively proposed more powerful deep RL algorithms, such as the DDPG algorithm for dealing with continuous motion control problems.

Lillicrap further proposed the DDPG algorithm that combines the DL strategy with the deterministic strategy gradient algorithm in 2016 [29]. This new algorithm has many advantages. For example, the experience playback mechanism is introduced to solve the problem of data correlation in this algorithm. The algorithm uses dual network structure, so that the learning process is more stable and the convergence is faster.

In order to solve autonomous decision-making and attack area fitting strategy problems of UCAVs under the condition of complex environment changes in the future of air combat, we adopt the DDPG algorithm that can deal with the problem of efficient continuous movement. In order to build the DDPG algorithm model for programming, we define the mathematical concepts involved in the autonomous decision-making problem of UCAVs. Since the principles of calculating the far and near boundaries of the attack area are the same, we take the far boundary fitting as an example. Choose UCAV as the agent, we define 10 typical air combat environments, which are recorded as $E_1, \ldots, E_{10}$.

The fitting values of the attack area in different air combat environments are recorded as $l(s)_1, \ldots, l(s)_{10}$.

The range is from 0 m to 200 m. The real values of the attack area obtained by program simulation are recorded as $L(s)_1, \ldots, L(s)_{10}$.

To decrease the error between fitting values and real values, the agent chooses the suitable coefficient autonomously and optimizes fitting values calculated by neural networks. We define these coefficients as actions which can be recorded as $a_1, \ldots, a_{10}$.

The range is from 0.8 to 1.2. The cost function can be recorded as

$$C = (l(s) - L(s))^2. \tag{1}$$

In the DDPG algorithm, the actor network and the critic network are updated at the same time. The actor network is used to update strategies, which refer to different coefficients adopted by UCAV. The critic network is used to provide gradient information and approximate the state action value function, which refers to the cost function.

The expected reward function can be expressed as

$$r = -C = -(l(s) - L(s))^2. \tag{2}$$

The target function of the DDPG algorithm is defined as the discount expectation of the cumulative reward:

$$J(\mu) = \mathrm{E}_\mu[r_0 + \gamma r_1 + \gamma^2 r_2 + \cdots + \gamma^n r_n] \tag{3}$$

where $J(\mu)$ is the cumulative reward, μ is the independent variable of the actor network, $r_0, \ldots, r_n$ are expected rewards, γ is the expected reward coefficient, and E_μ expresses mathematical expectation.

The action that maximizes the value of the target function is defined as the optimal action:

$$\mu^* = \arg\max_\mu J(\mu). \tag{4}$$

The process of training the value network is to find the optimal parameters in the value network to minimize the loss function:

$$L = \frac{1}{n}\sum_i (y_i - Q(s_i, a_i; \theta^Q))^2 \tag{5}$$

where L is the loss function, y is the real value function and $Q(s_i, a_i; \theta^Q)$ is the optimal value function. θ^Q expresses the independent variable in the critic network.

In conclusion, the goal of the DDPG algorithm is to maximize the value of the target function and minimize the value of the loss function.

3. Algorithm framework and parameter setting

Firstly, UCAVs should be able to fit the values of the attack area. Through the program simulation of the flight process in the fixed environment, a large amount of data close to the real situation is obtained in the off-line state. These data can reflect the flight characteristics of UCAVs in the fixed environment. Secondly, we can obtain the far and near bounds of the attack area under different initial conditions by dichotomy or golden section. Thirdly, we process the data into learnable data, which can be divided into two parts, the input part and the output part. After that, we design the neural network structure conforming to the rules of input and output, and put the data into the neural network for learning. Finally, after repeated operations, the neural network tends to be stable and convergent, the neural network used for real-time fitting of the attack area can be obtained.

Fig. 1 shows the designed neural network structure. The neural network includes the input layer, two hidden layers and the output layer.

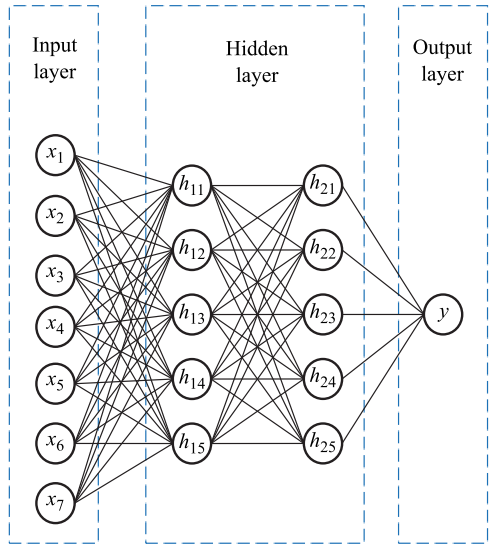


Fig. 1 Designed neural network structure

The input layer has seven input parameters, which correspond to the height, velocity of the missile and the tar-

get, the ballistic inclination angle, the entry angle and the off-axis angle respectively. Due to a large amount of data and the complex input parameters, in order to improve the learning effect of the BP neural network, it is necessary to design more than two hidden layers and appropriately design a number of neurons in the hidden layer. The output layer represents the far boundary of the attack area for a particular environment parameter. In this paper, we use the $7 \times 5 \times 5 \times 1$ neural network structure to process the data.

Fig. 2 shows the network structure of attack area fitting. Input-factors of the neural network include the height, speed, elevation angle, off-axis angle of UCAVs, and height, speed and entry angle of the targets. Output-factors are far boundary values and near boundary values. By simulating the real trajectory of the UCAV, we can obtain a large number of accurate far and near boundary values with different input-factors. We record the data and use them in factors. We record the data and use them to train the neural network. When air environment changes, error between fitting values and real values may increase. To decrease the error, UCAVs should adopt the correction coefficient for different air combat environments.

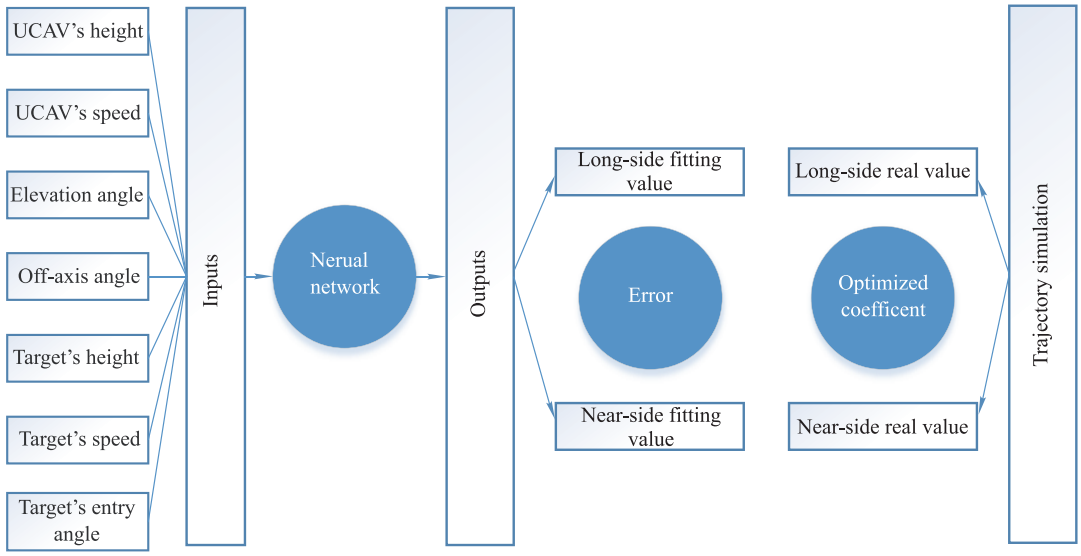


Fig. 2 Attack area fitting network structure

In the common aircraft, the pilot can artificially modify the value of the attack area obtained by neural network fitting through sensing the change of the environment. The specific range of modification is determined by off-line experimental test and experience. The neural network used only in fixed environments can be used in a variety of different environments by means of numerical correction. The environmental perception and the decision-making by pilots become the key to improving environmental adaptation. As an unmanned combat unit, UCAV cannot make decisions by pilots. We need to train UCAV in off-line states in order to improve the autonomous decision-making ability of UCAVs. In combination with environment tags and modified strategy, we further process the data used for attack area fitting and put the processed data into the two networks which are needed for the DDPG algorithm.

Fig. 3 shows the structure of the DDPG algorithm. Specific algorithm steps are as follows:

Step 1 Randomly initialize the weights of the critic network and the actor network: θ^Q and θ^μ.

Step 2 Initialize the target critic network and the target actor network: Q' and μ', and the weight parameter of the network is

$$\begin{cases} \theta^{Q'} \leftarrow \theta^Q \\ \theta^{\mu'} \leftarrow \theta^\mu \end{cases}. \tag{6}$$

Step 3 Initialize the experience playback pool R.

Step 4 Start a new round of RL and randomly initialize the process to search actions.

Step 5 Get the initial state value s_0.

Step 6 Start a new time step of learning.

Step 7 Calculate the action of the current time step according to the current strategy with noise N_t:

$$a_t = \mu(s_t; \theta^\mu) + N_t. \tag{7}$$

Step 8 Perform actions, record rewards and new status a_t, r_t and s_{t+1}.

Step 9 Store conversion experience data in the experience pool (s_t, a_t, r_t, s_{t+1}).

Step 10 Randomly sample a small batch of converted experience samples (s_i, a_i, r_i, s_{i+1}) from the experience pool, and set

$$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}; \theta^{\mu'}); \theta^{Q'}). \tag{8}$$

Step 11 Minimize the loss function and update the critic network:

$$L=\frac{1}{n}\sum_{i}(y_i-Q(s_i,a_i;\theta^Q))^2. \tag{9}$$

Step 12 Update the actor network with the gradient strategy algorithm:

$$\nabla_{\theta^\mu}J\approx\frac{1}{n}\sum_{i}\nabla_a Q(s,a;\theta^Q)|_{s=s_i,a=\mu(s_i)}\nabla_{\theta^\mu}\mu(s;\theta^\mu)|_{s_i}. \tag{10}$$

Step 13 Update the target network:

$$\begin{cases}\theta^{Q'}\leftarrow\tau\theta^Q+(1-\tau)\theta^{Q'}\\ \theta^{\mu'}\leftarrow\tau\theta^\mu+(1-\tau)\theta^{\mu'}\end{cases} \tag{11}$$

where τ is the weight parameter.

Step 14 Repeat Steps 6 – 13.

Step 15 When the round of learning is over, repeat Steps 4 – 14.

The block diagram in Fig. 3 is the framework of the DDPG algorithm. The DDPG algorithm is a kind of RL algorithm combined with DL. The computational fitting process of the algorithm belongs to the category of DL. Its frame optimization process belongs to the category of RL. RL begins with a process that does not update the framework parameters. During this process, the empirical data pool will be filled. After that, the agent (UCAV) will select the action according to the experience pool, and the environment will change with the action and feedback the reward value.

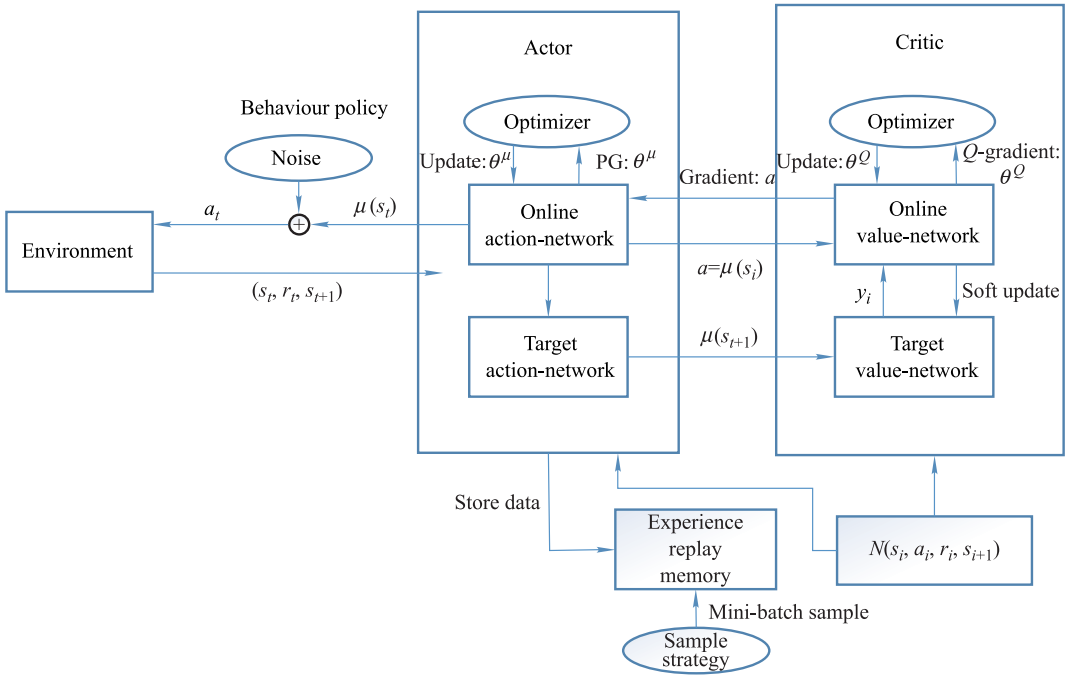


Fig. 3 DDPG algorithm structure

The actions mentioned in the algorithm refer to the optimization of the far-bound value of the attack area obtained by DL. The optimization method is to multiply the far-bound value of the attack area by a coefficient ranging from 0.8 to 1.2. The environment mentioned in the algorithm refers to the new far-bound value after UCAV selects the action (optimizing the far-bound value of the attack area). The feedback value mentioned in the algorithm refers to the error between the new far-bound value and the real far-bound value.

4. Simulation validation

First, we train the commonly used BP neural network. There are seven input parameters in total, and several special values are selected within a certain range. The selection is shown in Table 1.

According to the numerical selection scheme in Table 1, we can obtain a total of $7\times3\times5\times7\times3\times13\times9=257\,985$ combinations of environmental parameters. Select 250 000 parameter combinations randomly, and the corresponding far boundary values of the attack area could be obtained by simulation. These data are the training set and the rest of the combinations with their far boundary values are the test set.

Training parameters include iteration times, learning rate, accuracy target, and we set them as 100, 0.1, 0.004. By training the BP neural network, we get a network that can be used online to fit the far boundary value of the attack area.

Table 1 Parameter values selection

Parameter	Range	Value
Height of missile/m	0 – 18 000	0, 3 000, 6 000, 9 000, 12 000, 15 000, 18 000
Height of target/m	0 – 18 000	0, 3 000, 6 000, 9 000, 12 000, 15 000, 18 000
Velocity of missile	Ma=0 – 1	0.7, 0.8, 0.9
Velocity of target	Ma=0 – 1	0.60, 0.64, 0.68
Ballistic inclination angle/(°)	– 30 – +45	– 30, – 15, 0, 15, 30, 45
Entry angle/(°)	– 180 – +180	– 180, – 150, – 120, – 90, – 60, – 30, 0, 30, 60, 90, 120, 150, 180
Off-axis angle/(°)	– 60 – +60	– 60, – 45, – 30, – 15, 0, 15, 30, 45, 60

Consider the influence of natural environment parameters (temperature, altitude, etc.) on the above online network. If the flight environment of the aircraft is the same as the network training environment. The high precision far boundary value can be obtained by the BP neural network. If the flight environment of the aircraft is different from the network training environment, e.g., temperature, altitude and other parameters. Then the far boundary value of the attack area obtained by the BP network will occur with error. UCAVs can correct this error by the DDPG algorithm.

We first build the deep RL network framework according to the DDPG algorithm with the Python and tensorflow. The algorithm contains two networks: the critic network and the actor network. The critic network contains two hidden layers, each containing 400 neurons. The critic network uses the rectified linear unit (ReLU) activation function. The actor network also contains two hidden layers, each containing 400 neurons. The actor network uses the $\tanh$ activation function. Then, we record real values of UCAV's attack area by simulation in 10 kinds of typical air combat environments and store data into the memory experience pool. Then, we put the value of the attack area calculated by the trained neural network in a fixed environment into the algorithm. These data are initial values that wait to be modified. Select the appropriate policy to modify these initial values. Build the actor network and the critic network, and then set learning steps. Each round of study includes 200 steps, so that the agent can make decisions autonomously 200 times in one round of study.

Record the feedback used to update the value network and the strategy network. There are 2 000 learning rounds, and the first 150 rounds are used to supplement the experience memory pool. When the pool is filled, we begin to train and update the network, until the desired effect is achieved.

The specific parameters of deep RL are shown in Table 2, and the detail information of 10 kinds of environmental states are shown in Table 3.

When the step in which the error between real values and fitting values is less than 1 000 m, we call the step as a useful step. In each round of learning, we should record the result that how many steps the round needs for 50 useful steps in total per round. We can get the simulation result shown in Fig. 4 below after 2 000 rounds of learning. In the first 400 rounds of learning, results are stable within 200 times. After these 400 rounds of learning, the result begins to decline and quickly stabilizes within 80 times.

Table 2 Simulation parameters

Algorithm parameter	Value in program
Rounds of learning	2 000
Steps of learning	200
Learning rate of actor-network	0.001
Learning rate of critic-network	0.001
Reward discount	0.9
Memory pool capacity	30 000
Size of batch	32

Table 3 Information of environmental states

Environmental state	Atmospheric pressure/Pa	Atmospheric temperature/K	Atmospheric density/ (kg/m^3)	Wind speed/ (m/s)
1	26 500	216.65	0.088 9	0
2	26 500	216.65	0.413 5	0
3	26 500	223.252 1	0.088 9	0
4	26 500	223.252 1	0.413 5	0
5	5 529.1	216.65	0.088 9	0
6	5 529.1	216.65	0.413 5	0
7	5 529.1	223.252 1	0.088 9	0
8	5 529.1	223.252 1	0.413 5	0
9	26 500	216.65	0.088 9	3
10	5 529.1	223.252 1	0.413 5	3

We train UCAVs so that they will get the ability of decision-making that what policy should be adopted in different environments. In each round of learning, UCAVs correct values of the attack area are obtained by the neural network gradually through the learned ability. The faster UCAVs reach the goal that accumulates to 50 useful steps per round, the stronger the ability that UCAVs learn. Fig. 4 shows that with the help of the DDPG algorithm, after about 400 rounds of learning, UCAVs' correction ability is greatly improved. Trained UCAVs can reach the predetermined goal under about 80 steps of operation. Considering that the single step time is very short, the training effect

can meet needs of the engineering. After about 400 rounds of learning, UCAVs' correction ability is stable at about 80 steps, which indicates that UCAVs' learning ability is stable. Trained agents have been preliminarily equipped with artificial intelligence.

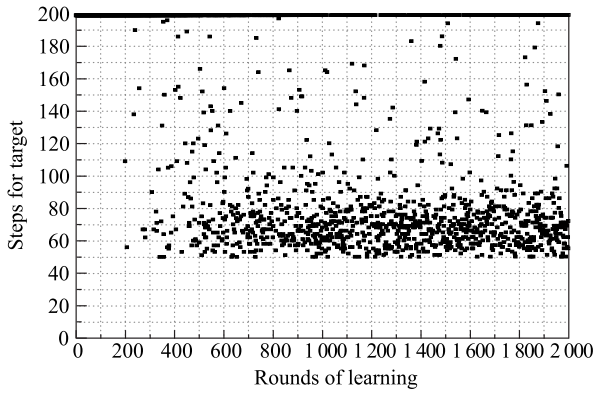


Fig. 4 Steps for targets in 2 000 rounds of learning

Fig. 5 reveals how the environmental states change with time. Ten typical air combat environments are arranged in the time sequence to simulate a complete change process of air combat environments. In Fig. 6, we compare the far boundary obtained by the common BP neural network with the far boundary obtained by the new DDPG algorithm and the DQN algorithm. In Fig. 7, the errors of the far boundary are compared. Fig. 6 and Fig. 7 show that the BP neural network algorithm cannot adapt to the changes in the environment. DQN and DDPG algorithms can guide UCAV to modify the attack area value. DQN can only deal with discrete actions. In the same environment, the correction of DQN is not as accurate as that of DDPG. The simulation results in the figure also demonstrate that the error of the far boundary value of the attack area corrected by the DDPG algorithm is smaller.

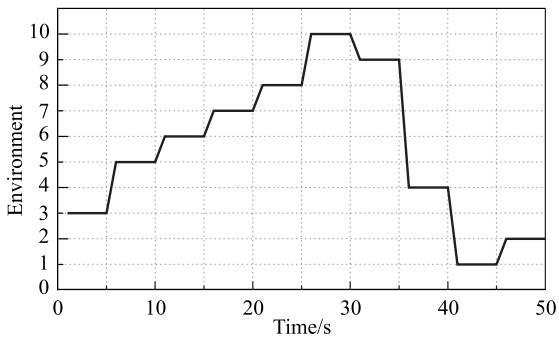


Fig. 5 Environmental states changing with time

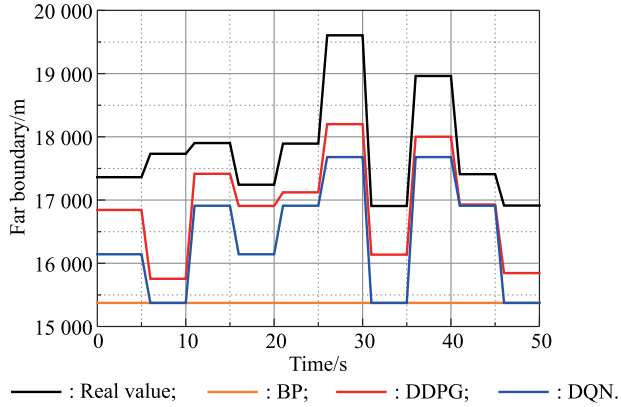


Fig. 6 Far boundary obtained by algorithms

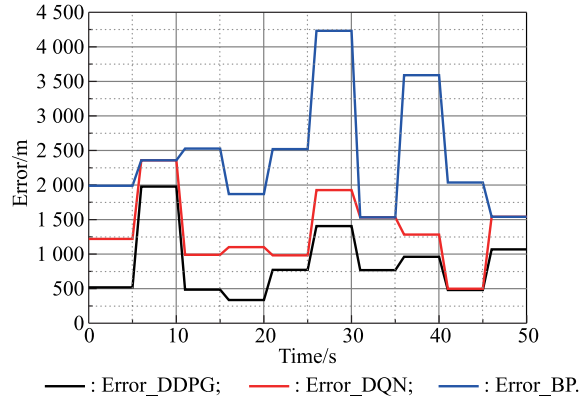


Fig. 7 Far boundary errors changing with time

In Fig. 8, we compare the near boundary obtained by the common BP neural network with the near boundary obtained by the new DDPG algorithm and the DQN algorithm. In Fig. 9, the errors of the near boundary are compared. Fig. 8 and Fig. 9 indicate that the BP neural network algorithm can not adapt to the changes in the environment for fitting the near boundary. In the same environment, the correction of DQN is not as accurate as that of DDPG. The simulation results in the figure also indicate that the error of the near boundary value of the attack area corrected by the DDPG algorithm is smaller.

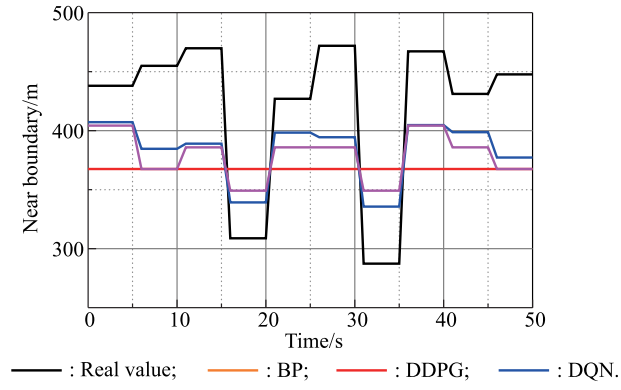


Fig. 8 Near boundary obtained by algorithms

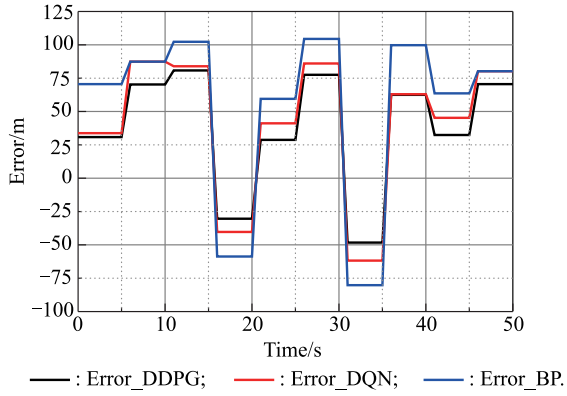


Fig. 9 Near boundary errors changing with time

These simulation results show that the DDPG algorithm used for autonomous decision-making of UCAV in the field of attack area calculation in complex air combat environments is more effective than the DQN algorithm and the BP neural network.

5. Conclusions

The DDPG algorithm proposed in this paper can well meet the requirement that UCAV can autonomously optimize the boundary value of the attack area in the complex air combat environment. This algorithm can guide an agent (UCAV) to acquire the ability of autonomous decision-making through offline learning. It reflects the extensive application of artificial intelligence technology in the field of unmanned aerial vehicles. The DDPG algorithm has the autonomous learning ability that the traditional BP neural network algorithm does not have. Compared to other traditional deep RL algorithms, such as DQN, the DDPG algorithm can choose continuous actions to solve continuous problems. This is very necessary to solve the problem of rapidly changing air combat environment. In this paper, the DDPG algorithm is applied to the optimization of the far boundary of the UCAV attack area. Through simulation and comparison, we can get the conclusion that the DDPG algorithm is effective for solving related problems. At the same time, the DDPG application framework proposed in this paper can provide reference and help to other artificial intelligence problems in the field of unmanned aerial vehicles.

References

[1] SIBLEY A K, JAIN T N, BUTLER M, et al. Remote scene size-up using an unmanned aerial vehicle in a simulated mass casualty incident. Prehospital Emergency Care, 2019, 23(3): 332 – 339.

[2] JAIN T, SIBLEY A, STRYHN H, et al. Comparison of unmanned aerial vehicle technology-assisted triage versus standard practice in triaging casualties by paramedic students in a mass-casualty incident scenario. Prehospital and Disaster Medicine, 2018, 33(4): 335 – 380.

[3] HANDFORD C, REEVES F, PARKER P. Prospective use of unmanned aerial vehicles for military medical evacuation in future conflicts. Journal of the Royal Army Medical Corps, 2018, 164(4): 293 – 296.

[4] RICHARD M, DANIEL S, VITALI V, et al.A third-party casualty risk model for unmanned aircraft system operations. Reliability Engineering and System Safety, 2014, 124: 105 – 116.

[5] MATTHEW M. The drone debate: a primer on the U.S. use of unmanned aircraft outside of conventional battlefields. Joint Force Quarterly, 2019, 92: 83 – 84.

[6] SEPULVEDA E, SMITH H.Technology challenges of stealth unmanned combat aerial vehicles. The Aeronautical Journal, 2017, 121(1243): 1261 – 1295.

[7] SUN F Y, LI Y J, DU Y, et al. A study on the high stability control for the integrated aero-propulsion system under supersonic state. Aerospace Science and Technology, 2018, 76: 350 – 360.

[8] YOSHIYASU H. Composing supersonic materials. Flight International, 2015, 188(5510): 43.

[9] YAZAN M, AMRO N, ALAA D, et al. Agent-based simulation of unmanned aerial vehicles in civilian applications: a systematic literature review and research directions. Future Generation Computer Systems, 2019, 100: 344 – 364.

[10] WEI X Q, YANG J Y, FAN X R. Distributed guidance law design for multi-UAV multi-direction attack based on reducing surrounding area. Aerospace Science and Technology, 2020, 99: 105571.

[11] KERRY N, ROBERT R, PAVEL F. Goal model analysis of autonomy requirements for unmanned aircraft systems. Requirements Engineering, 2018, 23(4): 509 – 555.

[12] PENG K M, LIN F, CHEN B M. Online schedule for autonomy of multiple unmanned aerial vehicles. Science China Information Sciences, 2017, 60(7): 217 – 229.

[13] HUI Y L, NAN Y, CHEN S D, et al. Dynamic attack zone of air-to-air missile after being launched in random wind field. Chinese Journal of Aeronautics, 2015, 28(5): 1519 – 1528.

[14] ASHARUL I K, YASEEN A. Unmanned aerial vehicle in the machine learning environment. Procedia Computer Science, 2019, 160: 46 – 53.

[15] LUCAS P O, MAURO S A, MARCATO J, et al. A convolutional neural network approach for counting and geolocating citrus-trees in UAV multispectral imagery. ISPRS Journal of Photogrammetry and Remote Sensing, 2020, 160: 97 – 106.

[16] LI Y W, CAO K. Establishment and application of intelligent city building information model based on BP neural network model. Computer Communications, 2020, 153: 382 – 389.

[17] WU G X, MIAO Y M, ZHANG Y, et al. Energy efficient for UAV-enabled mobile edge computing networks: intelligent task prediction and offloading. Computer Communications, 2020, 150: 556 – 562.

[18] QIU H X, DUAN H B. A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles. Information Sciences, 2020, 509: 515 – 529.

[19] LIU Q, LI M, YANG J, et al. Joint power and time allocation in energy harvesting of UAV operating system. Computer Communications, 2020, 150: 811 – 817.

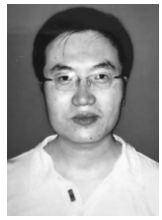
[20] FALLATI L, POLIDORI A, SALVATORE C, et al. Anthropogenic marine debris assessment with unmanned aerial vehicle imagery and deep learning: a case study along the beaches of the Republic of Maldives. Science of the Total Environment, 2019, 693: 133581.

[21] NEUPANE B, HORANONT T, HUNG N D. Deep learning based banana plant detection and counting using high-resolution red-green-blue images collected from unmanned aerial vehicle. PLoS One, 2019, 14(10): e0223906.

[22] JIAO Z Y, JIA G Z, CAI Y J. A new approach to oil spill detection that combines deep learning with unmanned aerial vehicles. Computers and Industrial Engineering, 2019, 135: 1300 – 1311.
[23] ZHAO X, YUAN Y T, SONG M D, et al. Use of unmanned aerial vehicle imagery and deep learning UNet to extract rice lodging. Sensors, 2019, 19(18): 3859.
[24] QU C Z, GAI W D, ZHONG M Y, et al. A novel reinforcement learning based grey wolf optimizer algorithm for unmanned aerial vehicles path planning. Applied Soft Computing Journal, 2020, 89: 106099.
[25] ZHAO X Y, ZONG Q, TIAN B L, et al. Fast task allocation for heterogeneous unmanned aerial vehicles through reinforcement learning. Aerospace Science and Technology, 2019, 92: 588 – 594.
[26] YANG J, YOU X H, WU G X, et al. Application of reinforcement learning in UAV cluster task scheduling. Future Generation Computer Systems, 2019, 95: 140 – 148.
[27] MA Y, ZHU W B, MICHAEL G B, et al. Continuous control of a polymerization system with deep reinforcement learning. Journal of Process Control, 2019, 75: 40 – 47.
[28] MA Z W, WANG C, NIU Y F, et al. A saliency-based reinforcement learning approach for a UAV to avoid flying obstacles. Robotics and Autonomous Systems, 2018, 100: 108 – 118.
[29] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning. Computer Science, 2015, 8(6): 187 – 200.

Biographies

LI Yue was born in 1995. He received his B.E. degree from Beijing Institute of Technology in 2016. He is currently a doctoral student in School of Aerospace Engineering, Beijing Institute of Technology. His main research interests include flight vehicle design, guidance and control.
E-mail: liyue627167955@163.com



QIU Xiaohui was born in 1975. He received his B.E. degree in industrial automation from Xi'an Jiaotong University in 1997, and M.E. degree in electronic information engineering from Northwestern Polytechnical University in 2008. He is a senior engineer in Science and Technology on Electro-Optic Control Laboratory. He is currently a doctoral student in School of Aerospace Engineering, Beijing Institute of Technology. His research interest is system simulation.
E-mail: qiuxh759@163.com

LIU Xiaodong was born in 1987. He received his B.E. degree from Qingdao University in 2008, and Ph.D. degree in automation control science and engineering from Beihang University in 2013. From 2013 to 2015, he was a post-doctoral researcher with Beijing Aerospace Automatic Control Institute, Beijing, China. Currently, he is a senior engineer of the Beijing Aerospace Automatic Control Institute. His research interests include adaptive control, integrated guidance and control for aircraft.
E-mail: k.start@163.com

XIA Qunli was born in 1971. He received his B.E. degree in launcher and design, M.E. degree in flight mechanics and Ph.D. degree in aircraft design from Beijing Institute of Technology in 1993, 1996 and 1999, respectively. He is currently an adjunct professor with Beijing Institute of Technology. His research interests are control and guidance technology.
E-mail: 1010@bit.edu.cn