Hindawi
International Journal of Aerospace Engineering
Volume 2022, Article ID 1544670, 13 pages
https://doi.org/10.1155/2022/1544670

Research Article

An Exoatmospheric Homing Guidance Law Based on Deep Q Network

Jin Tang[1,2] **Zhihui Bai**[3] **Yangang Liang**[1,2] **Fan Zheng**[4] **and Kebo Li**[1,2]

[1]*College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, China*
[2]*Hunan Key Laboratory of Intelligent Planning and Simulation for Aerospace Mission, Changsha 410073, China*
[3]*The 31102 Troops, Nanjing 210000, China*
[4]*The 41st Institute of Forth Academy of Aerospace Science and Technology Corporation, Xi'an 710000, China*

Correspondence should be addressed to Yangang Liang; liangyg@nudt.edu.cn

Received 15 June 2022; Accepted 4 August 2022; Published 22 August 2022

Academic Editor: Chuang Liu

A homing guidance law for exoatmospheric interceptor based on the Deep Q Network (DQN) algorithm is proposed in this paper. Aiming at the exoatmospheric interception problem, the guidance agent is built with the help of the deep reinforcement learning theory, and the action command is given according to the measurement information of the exoatmospheric interceptor for the accurate interception of the target. The homing guidance problem is first transformed into a Markov decision process, and a three-dimensional (3D) interception scenario is established. Then, the reward function considering the line-of-sight (LOS) rate and the final zero-effort-miss (ZEM) is designed, and the homing guidance problem is transferred to the reinforcement learning framework. After that, DQN is utilized to solve the exoatmospheric interception problem, and the guidance agent is obtained through a large amount of training. Finally, the guidance performance of DQN homing guidance law is verified by numerical simulation examples and compared with the classical true proportional navigation (TPN) guidance law. The results show that the guidance performance of the homing guidance law is better than that of TPN.

1. Introduction

The homing guidance law is a key factor to determine whether the missile can intercept the target or not. The current battlefield environment is becoming increasingly complex, which brings more challenges to the traditional guidance strategies. The design of intelligent guidance law has become a hot research issue [1].

Proportional navigation (PN) is one of the most widely used classical guidance laws [2], mainly including pure proportional navigation (PPN) [3] with reference to the interceptor velocity and true proportional navigation (TPN) [4, 5] with reference to the line of sight (LOS). PN has the advantages of simple structure, easy implementation, and good robustness. Therefore, it has been widely utilized in practical interception and guidance projects. Many improved forms of PN were proposed later. For example, based on the Lyapunov-like approach and inequality analysis method, the capture region of the realistic true proportional navigation (RTPN) guidance law against the arbitrarily maneuvering target for exoatmospheric homing guidance is thoroughly analyzed [6]. And later the capture region of 3D RTPN is further given, considering the full nonlinearity of the relative kinematics between the interceptor and target [7]. Actually, PN is with terrific guidance performance for nonmaneuvering and weakly maneuvering targets. However, when the target has a large-maneuvering acceleration, the guidance performance of PN will degrade significantly, because the commanded acceleration may exceed the overload saturation of the interceptor, which may cause the divergence of LOS rate and a large miss distance [8].

Optimal guidance (OG) is also widely utilized for solving the homing guidance problem [9]. OG law is derived based on optimal control theories with optimal guidance performance indexes. However, OG laws usually need accurate estimations of time-to-go; otherwise, the guidance performance will be greatly reduced. Meanwhile, OG law is not originally advanced for solving the problem of maneuvering

target interception. If the target is with large maneuverability, OG law may need to directly introduce the target acceleration or its estimation in the guidance command. However, the target acceleration estimation is very hard to obtain, and usually with large time lags, hence the guidance performance of OG laws against the highly maneuvering target cannot be theoretically guaranteed. Later, many scholars proposed various guidance laws based on differential geometry [10], sliding mode control [11, 12], differential game theory, etc. However, these derived guidance laws are usually suffering from the problems of complexity in form, requiring too much measurement information, or involving too many guidance parameters, resulting in problems such as poor robustness and difficulty in practical application.

With the rapid development of artificial intelligence (AI) technology, AI algorithms represented by reinforcement learning (RL) [13] are increasingly applied to the field of homing guidance. For example, based on reinforcement learning theory, the paper [14] proposes a new guidance law with better performance than PN, which considers the dynamic characteristics of the missile and the noise delay of the sensor and actuator. In addition, reference [15] studies PN with a variable proportional coefficient using the Q-learning algorithm, which discretizes the state space and action space of the homing guidance process. However, the traditional RL method is only applicable to discrete and low-dimensional state space and action space. It is difficult to deal with the continuous and high-dimensional environment in the actual interception process. The deep reinforcement learning (DRL) theory based on the combination of deep learning (DL) and RL can effectively solve the problem of spatial dimension explosion [16, 17], so it has great advantages in the field of homing guidance. The deep deterministic policy gradient (DDPG) is a typical DRL algorithm. A missile terminal guidance law based on DDPG is designed in [18]; then, the time and field of view (FOV) constraints are considered to generate the reward function. The numerical simulation results show that this guidance law is with stronger robustness rather than PN.

The targets of exoatmospheric interception [19, 20] are mainly spacecraft, such as spacecraft in elliptical orbits [21] or flexible spacecraft [22]. For this problem, Liang et al. [23] propose a model-based guidance law using DRL algorithm. The dynamic model of the guidance system is predicted and integrated into the path integral control process. The model-based reinforcement learning theory and the deep neural network (DNN) are used in this process. Another novel guidance law based on DDPG for maneuvering target interception is proposed in [24]. The reward function is designed based on miss distance, and the guidance performance is verified through numerical simulation. At present, many guidance laws for the exoatmospheric interception problem generally need the measurement information about the relative position, velocity, or target acceleration [25, 26]. The measurement variables are too many, and large measurement errors are usually involved, which degrades the performance of the traditional guidance laws. Some scholars introduce the RL method into the angle-only homing guidance problem for intercepting maneuvering targets. The proposed guidance laws only need the input consisting of the line-of-sight (LOS) angle and angular rate [27, 28]. Gaudet et al. [28] propose a novel guidance law based on reinforcement metalearning to solve the problem of terminal guidance for exoatmospheric interception. The guidance law does not need to estimate the relative range. It stabilizes seeker line-of-sight angles and their rate of change directly to commanded thrust for the missile's divert thrusters. The guidance model is trained and optimized using the proximal policy optimization (PPO) and meta-learning theory and then simulated and compared with the augmented zero effort miss (augmented ZEM) guidance law. It shows that the guidance performance of this RL-based guidance method is better.

There are limitations in the general guidance methods, such as unstable guidance performance, too many required measurements, and guidance parameters. Meanwhile, the traditional reinforcement learning methods are difficult to deal with the continuous and high-dimensional environment. Therefore, according to the nonmodel reinforcement learning theory, we propose a homing guidance law based on Deep Q Network (DQN) [29] in this paper. The homing guidance law only needs the measurement of LOS angle and angular rate and does not need to estimate the target maneuvering acceleration in advance. Referring to the principle of PN, the homing guidance law proposed in this paper mainly solves the interception problem by reducing the relative velocity which is perpendicular to line-of-sight between the interceptor and the target. A reward function is designed considering the LOS rate and the final ZEM. Then, we build the DQN agent in the Tensorflow framework and train the agent in the 3D homing guidance environment. The simulation results show that the homing guidance law proposed in the paper has strong adaptability to the environment and higher guidance accuracy compared with the classical TPN.

The paper is organized as follows. Section 2 mainly introduces RL algorithm used in this paper. Then, we model the homing guidance of RL in Section 3, including the model of relative motion, the model of measurement, and the Markov decision process (MDP) of homing guidance. In Section 4, we show the homing guidance law proposed in this paper and describe how to train and optimize DQN agent and adjust the hyperparameters of algorithm. Section 5 gives the training performance of the agent and the analysis of simulation results. Finally, the conclusions are presented in Section 6.

2. Reinforcement Learning Overview

"Trial and Error" is the central mechanism of RL method. In the process [30], the agent interacts with the environment to obtain the current state and immediate rewards and then executes actions to get feedback and generate the next state, which can update the policy and value function. The goal of the RL algorithm is to get a policy through a large number of samples, that is, the policy function π. The input is state information of the environment at the current time, and the output is the action executed by the agent. This process can be expressed as the following equation:

$$a = \pi(s), \tag{1}$$

where s is the current environment state and a is the executed action that comes from the state space and action space.

According to whether the action space is discrete or continuous, the RL algorithm can be divided into methods based on value function and policy gradient [31]. Classical methods based on value function include Q-learning and state action reward state action (SARSA) algorithm [32, 33]. Methods based on policy gradient include policy gradient algorithm and actor critic (AC) [34] which combines value function and policy gradient. Because the commanded acceleration of the exoatmospheric interceptor in the guidance problem is discrete, the method in this paper is based on value function to design the guidance law and construct the action space. And we consider that the interceptor maneuvers in the way of constant maneuvering and solve the interception problem using DQN of DRL algorithm.

2.1. Deep Q Network Algorithm. The purpose of RL policy is to generate action instructions so that the interceptor can successfully intercept the target. To solve this problem, this paper uses the prioritized experience replay DQN to design a homing guidance law [35].

The DQN combines DL and classical Q-learning, and it can use a neural network to fit the value function, that is, Q network. The input of the Q network is the environment state, and the output is the expected cumulative return of various actions corresponding to the current state.

There are problems of data correlation during DQN training. The algorithm adopts the replay buffer and double network mechanism to solve the problems. In the training process, a small batch of data will be randomly obtained from the experience pool for training the neural network. The current network in the double network mechanism is used for predicting model, and the target network is used to calculate the value of tag. Both networks have the same structure, and the parameters of the target network need to be updated regularly during training. The basic framework of DQN is shown in Figure 1.

2.2. Prioritized Experience Replay DQN. The prioritized experience replay DQN optimizes the logic of the experience replay part based on the general DQN. It changes the way of random sampling from the sample pool and adds weight to the sample data so that the samples with large absolute value of temporal difference (TD) error have a greater probability of being selected. Therefore, the prioritized experience replay DQN can solve the problems of larger fluctuation and slow convergence in the training process, and the obtained policy is more stable [36].

The algorithm usually uses the binary tree structure of sum tree to store the samples in the experience replay pool with priority [35]. Due to the weight of the sample, the loss function of the network is also optimized. The improved equation is as follows:

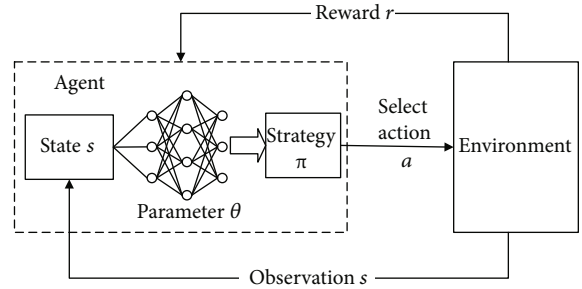

Figure 1: DQN basic framework.

$$\text{Loss} = \frac{1}{n}\sum_{j=1}^{n} \omega_j \left(y_j - Q(s_j, a_j, w)\right)^2, \tag{2}$$

where ω_j is the priority weight of the j -th sample, which is obtained by TD error. The meanings of other symbols are shown in Algorithm 1. After updating the gradient of network parameters, it is necessary to recalculate the TD error and update it to the sum tree structure. The prioritized experience replay DQN is the same as the DQN flow; then, its pseudocode is shown in Algorithm 1.

In Algorithm 1, s, a, r, and d, respectively, represent the environment state, action, reward, and termination state of the Markov decision process, T is the number of learning iterations, and C is the update frequency of the target network. β is the exponent in loss function.

3. RL Model of Homing Guidance

In order to design guidance law using deep reinforcement learning algorithm, the interception problem should be transformed into reinforcement learning framework. Then, the reinforcement learning model will be described in this section.

3.1. Model of Relative Motion. Considering the interception process in 3D space, the position vectors of the target and interceptor are defined as r_t and r_m in the launch inertial frame. The velocity vector of the target and interceptor is defined as v_t and v_m. The accelerations generated by the target and interceptor are a_t and a_m.

According to the relevant definitions of 3D interception process, the plane composed of the relative position vector r and the relative velocity vector v of the interceptor and target is called the intersection plane. The intersection plane may rotate with the relative motion of the target and interceptor [36–38]. The schematic diagram of the intersection plane is shown in Figure 2. The coordinate origin o_m is the centroid of interceptor, and e_r and e_θ, respectively, represent the unit vectors that are parallel and perpendicular to r in the intersection plane. q is the line-of-sight angle. The relative velocity vector v can be decomposed into v_r and v_θ, which represent closing velocity, y, and relative velocity which is perpendicular the to line-of-sight between interceptor and target. Also, v_θ is the cause of LOS rotation.

ω_s is defined as the rate of change of the line-of-sight vector in 3D space, and $\omega_s = \omega_s * e_\omega . e_\omega$ is perpendicular

1. Initialize Q network parameter w, target Q network parameter $w = w'$.
2. Initialize replay memory D with capacity N, the priority of all Sum Tree leaf nodes p_j=1.
3. For i=1to T do
4. Initialize s as the first state in the current state sequences of interceptor.
5. While s is not Termination:
6. a) Select an action a with ε-greedy.
7. b) Execute action a, transfer to the next state s', and get the immediate reward r. Judge whether it is in the termination state d.
8. c) Store transition $\{s, a, s', r, d\}$ in D. Replace the oldest tuple if $\|D\|>N$.
9. d) Sample n tuples from D, $\{ s_j, a_j, s'_j, r_j, d_j \}$, j=1,2,3,…,n. The sampling probability is $P_j = p_j/\sum_i p_i$. Compute the weight of loss function: $\omega_j = (N * P_j)^{-\beta}/\max_i \omega_i$.
10. e) Compute the current target Q value y_i.
$y_i = r_j + (1 - d)\gamma Q'(s'_j, \text{argmax}_{a'} Q(s'_j, a, w), w')$.
11. f) Compute the loss as equation (2). Updating Q network parameter w.
12. g) Compute TD error of all sample data: $\delta_j = y_j - Q(s_j, a_j, w)$. Update the priority of all Sum Tree nodes: $p_j = |\delta_j|$.
13. h) if T%C == 0, Update the target Q network parameter w'=w End if.
14. i) s=s'.
15. End For

Algorithm 1: Prioritized experience replay DQN for homing guidance.

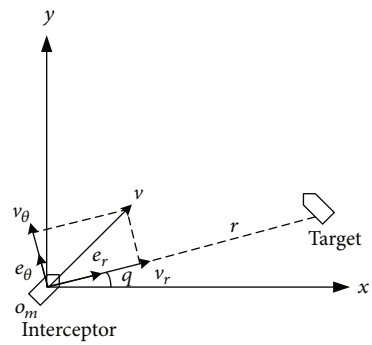

Figure 2: The intersection plane of interception geometry.

to e_r and e_θ, and they form the line-of-sight coordinate frame. We derive the relative position vector r and obtain the equation

$$\dot{r} = v_r e_r + v_\theta e_\theta = \dot{r} e_r + r\omega_s e_\omega \times e_r. \tag{3}$$

e_ω in equation (3) will rotate with the intersection plane in 3D space. We assume that Ωs is the angular velocity of e_ω. Then, the following equation can be obtained.

$$\dot{e}_\omega = \Omega_s e_r \times e_\omega = -\Omega_s e_\theta. \tag{4}$$

By deriving from the equation (3), the equation of relative motion in the line-of-sight coordinate frame can be obtained as the following equation:

$$\left(\ddot{r} - r\omega_s^2\right) e_r + (r\dot{\omega}_s + 2\dot{r}\omega_s) e_\theta + r\omega_s \Omega_s e_\omega = a_{tm}, \tag{5}$$

where $a_{tm} = a_t - a_m$. It represents the relative acceleration vector of the target and interceptor.

The line-of-sight direction can be expressed by the line-of-sight elevation angle q_ε and line-of-sight azimuth angle q_β in the launch inertial frame [37, 38]. q_ε and q_β are the LOS angle, and $\dot{q}_\varepsilon$ and $\dot{q}_\beta$ are the LOS angular rate in this paper. According to the relationship of coordinate conversion, the line-of-sight coordinate frame and the launch inertial frame can be mutually converted through the LOS azimuth angle and the LOS elevation angle. The line-of-sight angular velocity ω_s in the LOS coordinate frame is expressed as the following equation:

$$\omega_s = \dot{q}_\beta \sin q_\varepsilon \cdot x_S + \dot{q}_\beta \cos q_\varepsilon \cdot y_S + \dot{q}_\varepsilon z_S, \tag{6}$$

where x_S, y_S, and z_S are the coordinate axis unit vector of the LOS coordinate frame. Since e_r and x_S are in the same direction, the following equations can be obtained from the LOS angle and LOS angular rate.

$$\dot{e}_r = \dot{q}_\varepsilon y_S - \dot{q}_\beta \cos q_\varepsilon z_S, \tag{7}$$

$$e_\theta = \frac{\dot{q}_\varepsilon y_S - \dot{q}_\beta \cos q_\varepsilon z_S}{\sqrt{\left(\dot{q}_\beta \cos q_\varepsilon\right)^2 + \dot{q}_\varepsilon^2}}, \tag{8}$$

$$e_\omega = \frac{\dot{q}_\beta \cos q_\varepsilon y_S + \dot{q}_\varepsilon z_S}{\sqrt{\left(\dot{q}_\beta \cos q_\varepsilon\right)^2 + \dot{q}_\varepsilon^2}}. \tag{9}$$

To sum up, when the LOS elevation angle q_ε and LOS azimuth angle q_β are measured, and then, the line of sight angular rate is obtained by filtering, the intersection plane and the relative equation of motion can be determined according to equations (7)–(9).

3.2. Measurement Model. The measurement model of the interceptor includes information measurement and data processing. Its purpose is to obtain the LOS angle and LOS angular rate according to the current state of interceptor and target. The relative position vector and relative velocity

vector of the target and interceptor in the launch inertial frame are expressed as the following equations:

$$r = r_t - r_m = \left[r_x, r_y, r_z\right]^T, \tag{10}$$

$$v = v_t - v_m = \left[v_x, v_y, v_z\right]^T. \tag{11}$$

According to equations (10) and (11), the LOS angle and LOS angular rate can be obtained as the following equations [7]:

$$\begin{aligned} q_\varepsilon &= \tan^{-1}\left(\frac{r_y}{\sqrt{r_x^2 + r_z^2}}\right), \\ q_\beta &= \tan^{-1}\left(-\frac{r_z}{r_x}\right), \end{aligned} \tag{12}$$

$$\begin{aligned} \dot{q}_\varepsilon &= \frac{\left(r_x^2 + r_z^2\right)v_y - r_y\left(r_x v_x + r_z v_z\right)}{r^2\sqrt{r_x^2 + r_z^2}}, \\ \dot{q}_\beta &= \frac{r_z v_x - r_x v_z}{r_x^2 + r_z^2}. \end{aligned} \tag{13}$$

In this paper, the measurement model temporarily does not include the measurement errors of the relative range between the interceptor and the target, the LOS angle, and the closing velocity of the interceptor and the target but only considers the measurement errors of the LOS angular rate, which is set as the Gaussian white noise with the standard deviation of 1×10^{-4}rad/s. The measurement model in this section will be used to generate the simulation system.

3.3. MDP for Homing Guidance. The exoatmospheric interception process of interceptor and target is an MDP. And the three-dimensional relative motion model established before constitutes the environment of the process. In the environmental state design, we consider a more concise situation, that is, only the LOS angle needs to be measured and the LOS angular rate can be obtained through filtering. Therefore, we only use the LOS angle and LOS angular rate to design the state space [23], which is expressed as the following equation:

$$S = \left[\Delta q_\varepsilon, \Delta q_\beta, \dot{q}_\varepsilon, \dot{q}_\beta\right], \tag{14}$$

where Δq_ε and Δq_β are the errors of the LOS angle and the initial LOS angle. S is the variable of state space in the reinforcement learning problem we designed.

The output of DQN is each action value corresponding to each state. Because the control mode of interceptor is discrete, the action space in MDP is discrete. We consider that the interceptor maneuvers in the way of constant-maneuvering, and the maneuvers along y_S and z_S directions do not affect each other in the plane perpendicular to the LOS. The interceptor can maneuver with saturated overload in y_S direction, with saturated overload in y_S direction or without maneuver, that is, the interceptor's acceleration is zero. The maneuvering mode of interceptor in z_S direction is the same as that in y_S direction. We combine the actions in y_S and z_S to get the following action space, which is shown in the following equation:

$$A = [u, v], u, v \in \{-a_{\max}, 0, a_{\max}\}, \tag{15}$$

where u and v are the maneuvering accelerations of the interceptor along the y_S and z_S, respectively. We sort and code the 9 actions in the action space, which correspond to 9 action values output in the value network. The schematic diagram of its action space is shown in Figure 3.

For the interception problem, we assume that the interceptor carries detection equipment that can measure the position vector and velocity vector of the target. Through the measurement model, we can calculate the LOS angle errors Δq_ε and Δq_β and LOS angular rate $\dot{q}_\varepsilon$ and $\dot{q}_\beta$. Therefore, the observations of the model are expressed as equation (16). It can be seen from the above analysis that the MDP is observable in all states.

$$O = \left[r_t, v_t, r_m, v_m, \Delta q_\varepsilon, \Delta q_\beta, \dot{q}_\varepsilon, \dot{q}_\beta\right]. \tag{16}$$

3.4. Reward Function. The design of the reward function is the key factor for DQN to solve tasks. In many reinforcement learning environments, there is a problem of sparsity of reward, that is, there is no reward, resulting in slow convergence or even nonconvergence of the algorithm. Therefore, an effective solution is to introduce a reward shaping function to guide the agent to learn the final policy [39].

In the interception process, our ultimate goal is to intercept the target in a limited time. General guidance laws, such as PN, control the ZEM by restraining the divergence of LOS rate in the guidance process, while the OG takes minimizing the terminal miss distance as the optimization goal and obtains the optimal control law by solving the two-point boundary value problem. The reward function constructed in this paper is mainly based on LOS rate and final ZEM. The specific description is as follows.

Because the relative velocity which is perpendicular to line-of-sight between interceptor and target can reflect the LOS rate to a certain extent, the smaller the absolute value of the relative velocity which is perpendicular to line-of-sigh, the smaller the LOS rate and the smaller the ZEM of the interceptor. We design the shaping reward function in Gaussian form shown in the following equation:

$$R_1 = \exp\left(-\frac{|\theta|}{\sigma}\right), \tag{17}$$

where θ is the angle between the relative velocity of the interceptor and target and the LOS direction, which is called the velocity leading angle of the interceptor and target. σ is the reward coefficient, which is used to adjust the reward value. When σ is 0.01, the curve of the reward function is shown in Figure 4. It can be seen that the smaller the velocity leading angle of interceptor and target,

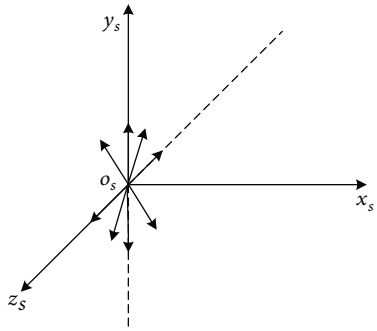

Figure 3: The discrete action space diagram.

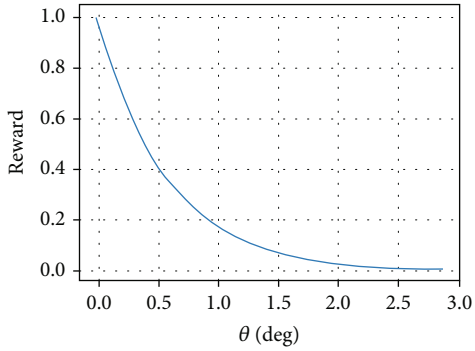

Figure 4: The Gaussian reward function.

the closer the relative velocity is to the direction of LOS, and the greater the reward value.

ZEM is the minimum distance between interceptor and target when both are not maneuvering. The process of intercepting the maneuvering target is to minimize the final miss distance of the interceptor. To ensure the interception effectively, another terminal reward is set in the paper. When the final ZEM generated by the interceptor at the time of termination is less than the given miss distance, the reward is a positive value. When the interceptor misses the target, the reward is 0. The form of the terminal reward function is shown as follows:

$$R_2 = \begin{cases} +10 & \text{if } \mathrm{ZEM}_f \le r_{\mathrm{Miss}}, \\ 0 & \text{else}, \end{cases} \tag{18}$$

where r_{Miss} is the maximum allowable miss distance. In general, the reward value obtained by the agent from the environment is the sum of shaping reward and terminal reward. It can be seen in the following equation:

$$R = R_1 + R_2. \tag{19}$$

4. DQN Agent

For the training process of DQN agent, the environment is a 3D kinematic system composed of interceptor and target, and the agent is generally an interceptor. Based on the 3D motion model of target and interceptor, a 3D interception environment can be established to train the guidance policy. Firstly, it is assumed that the target and interceptor are regarded as a particle, and the time lag and measurement error of the guidance control system is not considered during training. Secondly, the 4-order Runge Kutta method is used to compute the kinematic equations of interceptor and target, and we set the step of the simulation as 1 ms. Next, we will explain the design of the algorithm, including the selection of neural network structure, simulation parameters, and hyperparameter settings.

4.1. Algorithm Design. First, the initial conditions of interceptor and target in the training environment of simulation are given, as shown in Table 1.

In the training environment, it is assumed that the saturated overload of the interceptor along y_S and z_S is 6g. And g is the gravitational acceleration, taking 9.8 m/s^2. The total maximum overload of the interceptor is $6\sqrt{2}g$. We set the target's saturated overload along y_S and z_S as half of the interceptor, that is, $3g$, so the maximum overload is $3\sqrt{2}g$. In addition, the allowable miss distance r_{Miss} used for terminal reward is taken as 0.2 m in training.

In this paper, the prioritized experience replay DQN algorithm is used to train the agent, and the Tensorflow framework is used to build DQN model. Firstly, the model is established according to the basic parameters and neural network structure, which are shown as follows. There are mature built-in functions and basic variables in Tensorflow that can help build networks. Then, complete the design of the network interface and calling method of environment. Finally, the data is used for training the network and optimizing the parameters and network, which includes training method of network, parameter updating, and action selection policy.

The Q network of the algorithm adopts a three-layer fully connected neural network, and the network structure is shown in Table 2 and Figure 5. The target Q network has the same structure as the current Q network, and the initial network parameters are also consistent with the current network. According to the reinforcement learning model established in Section 2, the designed neural network has four inputs, that is, the state. There are 9 outputs, namely, action value. A relatively simple hidden layer is used in the middle, and tanh is used as our activation function in both layers, which can be better for algorithm convergence.

4.2. Hyperparameter Setting. DQN algorithm contains many hyperparameters, some of which are sensitive to the environment. If the parameters are set unreasonably, the algorithm will be difficult to converge. Therefore, in the training process of reinforcement learning, the hyperparameters need to be adjusted continuously to make the algorithm converge finally. For the interception environment established above, after a large number of numerical simulations, the algorithm hyperparameters set in this paper are shown in Table 3.

It can be seen from the table that we have trained 3000 episodes, and the discount factor is 0.996. The initial network parameters are trained with Adam optimizer, and the

Table 1: Initial conditions of interceptor and target in the training environment.

Physical parameters	Reference value
Line-of-sight range	100 km
Line of sight elevation angle	30 deg
Line of sight azimuth angle	40 deg
Position vector of interceptor	$[0, 0, 0]^T$ m
Target velocity	7 km/s
Velocity pitch angle of target	0 deg
Velocity yaw angle of target	220 deg
Interceptor velocity	5 km/s
Alignment deviation in intersection plane	2 deg
Alignment deviation perpendicular to intersection plane	2 deg

Table 2: Q network structure.

Network layer	Number of neurons	Activate function
Input layer	4	\
Hidden layer 1	100	tanh
Hidden layer 2	80	tanh
Output layer	9	\

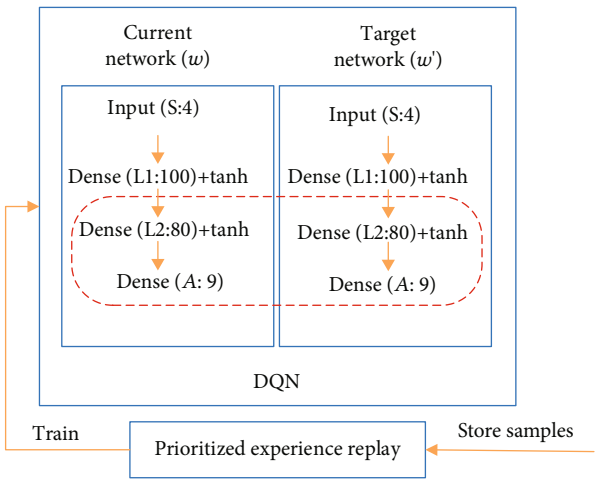

Figure 5: The structure of DQN network.

Table 3: DQN algorithm hyperparameter.

Hyperparameter	Parameter value
Maximum iterations	3000
Discount factor	0.996
Q network learning rate	0.001
Capacity of experience replay memory	100000
Minibatch size	64
Target network update rate	10
Initial exploration	0.8
Final exploration	0.01
Reward coefficient	0.05

learning rate is 0.001. The minibatch size is 64, and the weight of the target network is adjusted to the initial network every 10 steps. In addition, the exploration rate of our motion selection mechanism during training ranged from 0.8 simulated annealing to 0.01. Its purpose is to make the algorithm finally learn the optimal policy. The reward coefficient in the shaping reward function is set to 0.5.

5. Results and Analysis

The simulation results in this section mainly include the training results of the DQN agent, the test results of trained agent, and the comparison with TPN guidance methods. After each simulation result, we give a detailed analysis.

5.1. Training Simulation. The training process of DQN is actually the process of optimizing Q network. According to the training method of the neural network, the algorithm needs a lot of data to make the Q network converge. In each iteration, the agent interacts with the environment to generate experience data s, a, s', r, and d and save them to the experience replay pool. At the same time, a small batch of data is extracted from the replay pool to train the neural network. When the terminal state is reached, the iteration ends. After many iterations, the action selected by the agent through the Q network can maximize the value corresponding to the current state; it is indicated that the agent has learned an effective policy. The simulation in the paper is trained by a computer, which is with Intel(R) Gold 6226R: 2.90 GHz CPU, NVIDIA GeForce RTX 2080 Ti GPU, 64.0 GB RAM, and Windows7 operating system. We also use Python 3.7.6 and Tensorflow 1.15.0 to write algorithms.

The training process is visualized by Tensorboard. It takes about 14101.9053 s to train 3000 episodes and 45870.4302 s for 10000 episodes. It means that it uses nearly 3.9 hours to do full training. The results are shown in Figures 6 and 7. The horizontal axis of the coordinates represents the training times, the vertical axis of Figure 6 represents the Q network loss, and the vertical axis of Figure 7 represents the TD error. At the beginning of training, the network loss and TD error change very little, the agent is in the exploratory stage at the beginning of training, the experience data is relatively small, and the policy adopted by the agent has great randomness. With the increase of learning times, Q network loss and TD error change greatly, which indicates that the policy learned by the agent has been greatly improved. When the experience replay pool is full, the old experience data is of little significance to the agent's policy. The new experience data will replace the original historical experience. Therefore, the TD error of the experience data in the experience pool will gradually increase in the later stage of training, making the policy continuously optimized.

The reward changes during the training process are shown in Figure 8. The horizontal axis of the coordinates represents the episodes of training, and the vertical axis represents the episode reward and average reward. The blue curve in the figure is the cumulative reward after smoothing, and the orange curve represents the average reward of all

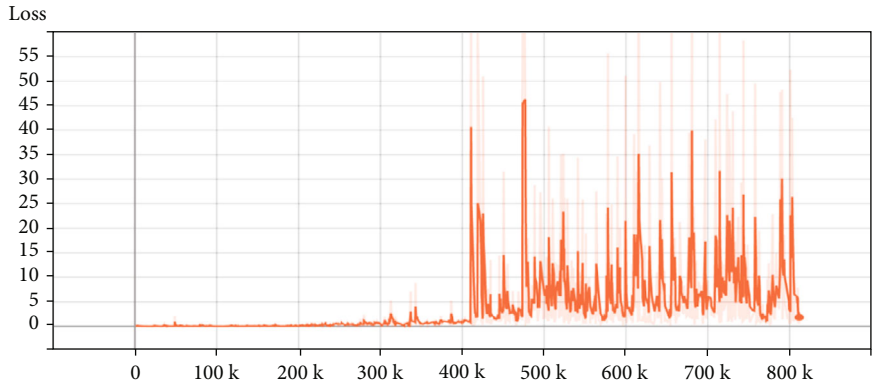

Figure 6: Q network loss change.

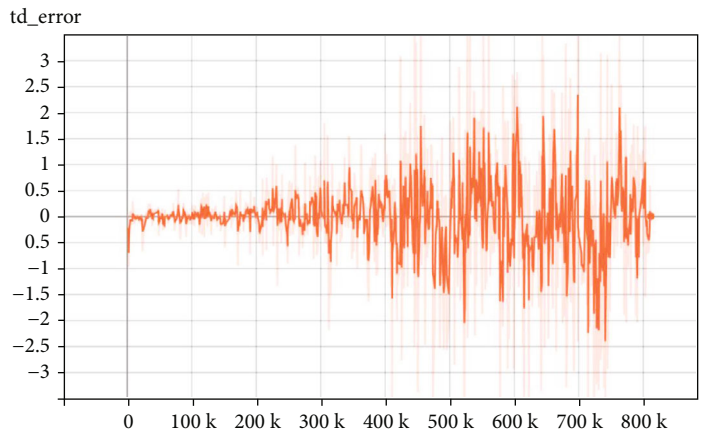

Figure 7: TD error change.

Figure 8: DQN cumulative reward change.

episodes. According to the change of the cumulative reward of each episode in the figure, the reward obtained by the agent in each episode in the early stage of learning is low, and the reward gradually increases after a period of learning. After 2000 rounds of iterations, the cumulative reward of the algorithm in each episode reaches the maximum, and after 3000 rounds of iterations, the cumulative reward is basically stable around the maximum, indicating that the algorithm has reached the convergence. The

Table 4: Simulation initial conditions.

	Position (km)	Velocity (m/s)
Target	[70, 50, -33.3]	[-6039, 610, 3486]
Interceptor	[0, 0, 0]	[338.7, 4984, -211]

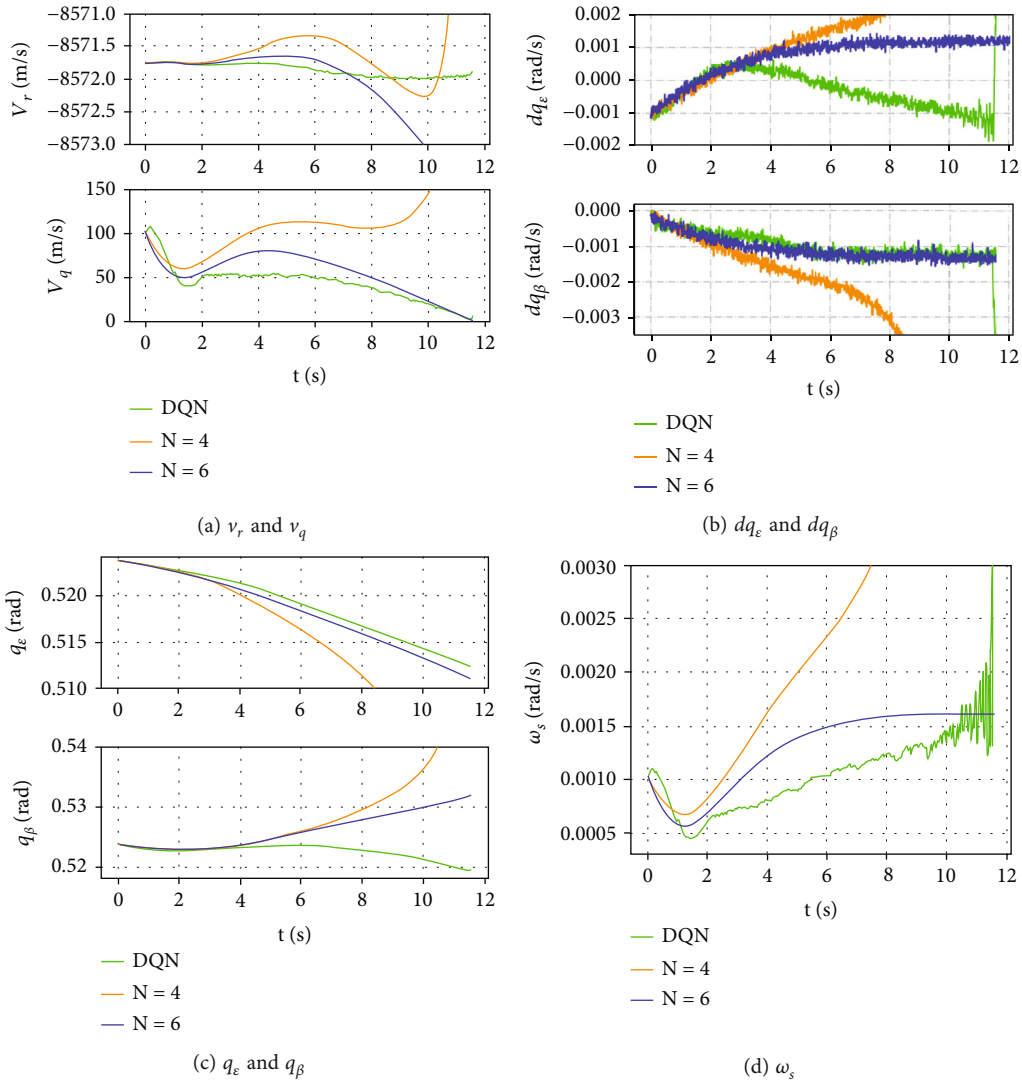

Figure 9: DQN simulation results of constant-maneuvering target.

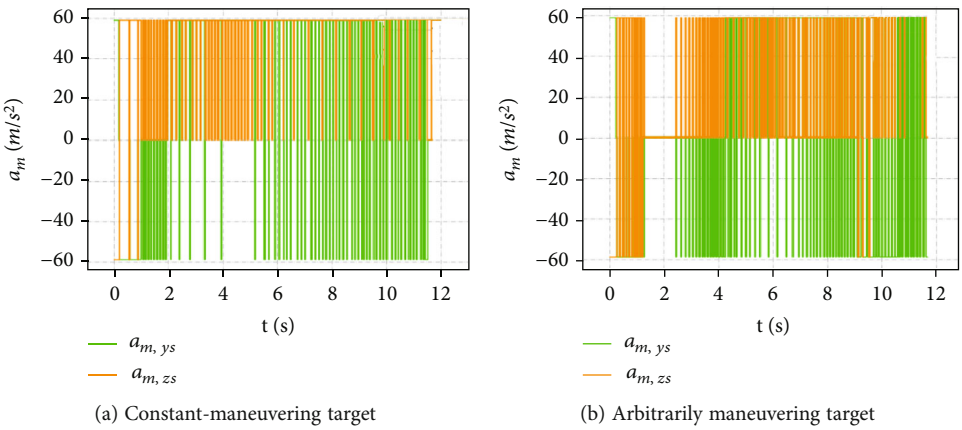

Figure 10: The engine switch curve of the exoatmospheric interceptor.

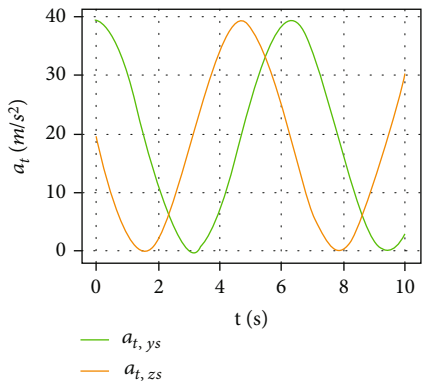

Figure 11: Acceleration of sinusoidal maneuvering target.

increasing average reward value also shows that the policy learned by the algorithm is improving.

5.2. Test and Comparative Analysis. In the training of agent, the measurement error and time delay were not considered. In order to be closer to the real environment, a 3D simulation system is established, and the measurement error of LOS angular rate is considered in the measurement model. Then, the guidance model is the guidance agent generated by training before. When we design the control system, it is assumed that the response speed of the interceptor is very fast, and the control command may lag 1~2 sampling cycles compared with the guidance command. Therefore, in the simulation, the response delay of the control system is the delay of the sampling period actually, which is 2 sampling periods later than the guidance instruction, that is, 20 ms. The DQN homing guidance law proposed in the paper and TPN with different coefficients are simulated and analyzed. We design two movements of target. They are target constant-maneuvering and arbitrarily maneuvering. Through simulation, the performance of the homing guidance law is further tested.

The longitude of the launch position in the geocentric inertial frame is 140°, the latitude is 60°, the altitude is 100 m, the launch azimuth is 90°, and the earth radius is 6378137 m. The initial position and velocity information of the target and interceptor in the simulation environment are shown in Table 4.

According to the initial conditions of the interceptor and the target, the relative range between the target and the interceptor is 100 km, the initial LOS azimuth is 30°, the initial LOS elevation angle is 30°, and the velocity of the target and the interceptor is 7.0 km/s and 5.0 km/s, respectively. For the interceptor, the main factors affecting its miss distance are the direction of target maneuvering and saturated overload. In the example, we consider the time step of simulation is 1 ms. And the sampling period of the interceptor's command acceleration is 10 ms. When the relative velocity between the interceptor and the target is greater than zero, the simulation ends, and the terminal miss distance is approximately the ZEM at this time.

5.2.1. Constant-Maneuvering Target. For the constant-maneuvering target, maneuvering in the plane perpendicular to the LOS direction can most effectively increase the line-of-sight rate, so we only consider the target maneuvering in the plane perpendicular to the LOS. We assume that the total overload of the target is $4\sqrt{2}g$, the acceleration in the LOS coordinate frame is expressed as $a_t = [0, 4g, 4g]^T$, and the total overload of the interceptor is $6\sqrt{2}g$. The interceptor is controlled by the obtained DQN homing guidance law and TPN, where the proportional guidance coefficient N of TPN is 4 and 6. The simulation results are shown in Figure 9, where v_r and v_q represent the closing velocity and relative velocity which is perpendicular to LOS between interceptor and target, respectively. q_ε and q_β are the elevation angle and azimuth angle of line of sight, and dq_ε and dq_β are their corresponding rate of change. And ω_s is 3D line-of-sight rate.

According to the calculation of final ZEM, the terminal miss distance of DQN homing guidance law is 0.38 m, the terminal miss distance of TPN is 206 m when $N = 4$, and the terminal miss distance of TPN is 5.3×10^{-4} m when $N = 6$. It can be seen from Figure 9(a) that both DQN homing guidance law and TPN with guidance coefficient $N = 6$ can reduce v_q to zero in a limited time, while v_q increases in the later period when $N = 4$. Figure 9(b) shows the of LOS angular rate with noise. It can be seen that TPN with guidance coefficient $N = 6$ and DQN homing guidance law can finally make LOS angular rate stable. From the change of 3D LOS rate described in Figure 9(d), both DQN homing guidance law and TPN with guidance coefficient $N = 6$ can effectively limit the divergence of LOS rate. However, when $N = 4$, LOS rate gradually diverges. Figure 10(a) shows the engine switch curve when the exoatmospheric interceptor intercepts the constant-maneuvering target, which corresponds to the accelerations in y_s and z_s directions, respectively.

5.2.2. Arbitrarily Maneuvering Target. For arbitrarily maneuvering target, it is assumed that the target's maneuvering mode is sinusoidal maneuvering, the total saturated overload is $4\sqrt{2}g$, and the acceleration in the LOS coordinate frame is $a_t = [0, 2g + 2g\cos(t), 2g - 2g\sin(t)]^T$. Other simulation settings are the same as those of the constant-maneuvering target, and the change of maneuvering acceleration with time is shown in Figure 11.

Similar to the target with constant-maneuvering, the interceptor is controlled by DQN homing guidance law for guidance and compares it with TPN simulation. The guidance coefficient N is also taken as 4 and 6. The simulation results are shown in Figure 12. And the meanings of symbols in the figure are the same as those in Figure 9. The calculation of final ZEM shows that the terminal miss distance of DQN homing guidance law is 0.29 m, the terminal miss distance of TPN is 4.13 m when $N = 4$, and the terminal miss distance of TPN is 6.3×10^{-4} m when $N = 6$.

It can be seen from Figure 12(a) that both DQN homing guidance law and TPN has a small impact on the closing velocity. But when $N = 4$, the relative velocity which is

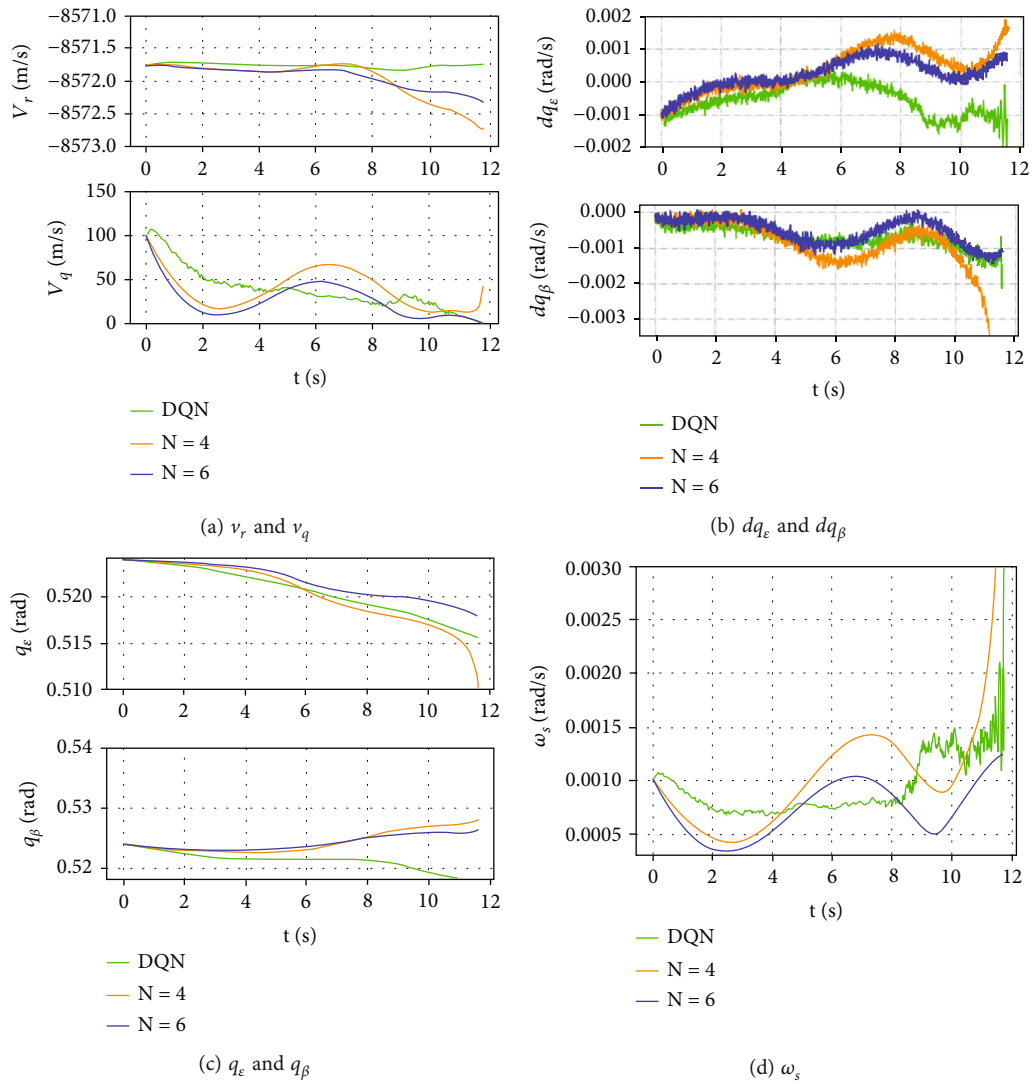

Figure 12: DQN simulation results of sinusoidal maneuvering target.

perpendicular to LOS between interceptor and target of TPN does not decrease to 0 at the terminal time. From the change of dq_ε and dq_β in Figure 12(b), we can find that the change of LOS angular rate in the three cases is not very dramatic, but the change of LOS angular rate is more stable when using DQN homing guidance law and TPN guidance coefficient $N = 6$. In addition, the 3D LOS rate curve in Figure 12(d) also shows that DQN homing guidance law can effectively reduce LOS rate compared with TPN. Since the target's maneuvering mode is sinusoidal, it can also be seen that the changes of LOS angular rate and 3D LOS rate also have some volatility. In this test, the engine switch curve of the interceptor is given in Figure 10(b).

The simulation results of the above groups show that, after adding the measurement error and guidance time delay, the designed RL homing guidance law based on DQN can intercept the target with certain maneuverability more effectively than the general guidance law of TPN in the 3D interception simulation system. At the same time, the comparative analysis shows that whether the target is constant-maneuvering or arbitrarily maneuvering, the homing guidance law can reduce the relative velocity which is perpendicular to LOS between interceptor and target in a limited time. It can also restrain the divergence of LOS rate and ensure that the final miss distance is very small.

6. Conclusion

A reinforcement learning homing guidance law based on DQN is proposed in this paper. The DQN homing guidance law only needs the LOS angle and angular rate between the interceptor and the target. The continuous state space, discrete action space, and Gaussian reward function in the MDP process are designed. Then, the DQN agent is built in the Tensorflow framework, which is trained and optimized in the interception environment. The obtained homing guidance law is numerically simulated and compared with the classical TPN. The simulation results show that the reinforcement learning homing guidance law proposed in this paper has a better performance compared with TPN.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was funded by the National Natural Science Foundation of China (No.: 12002370).

References

[1] Y. W. Fang, T. B. Deng, and W. X. Fu, "Review of intelligent guidance law," *Unmanned Systems Technology*, vol. 3, no. 6, pp. 36–42, 2020.

[2] W. D. Yin, "Design of guidance law for anti-high speed maneuvering target," *Harbin: Harbin Institute of Technology*, pp. 1–12, 2018.

[3] C. D. Yang and C. C. Yang, "A unified approach to proportional navigation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, no. 2, pp. 557–567, 1997.

[4] D. Ghose, "Capture region for true proportional navigation guidance with nonzero miss-distance," *Journal of Guidance, Control, and Dynamics*, vol. 17, no. 3, pp. 627-628, 1994.

[5] D. Ghose, "True proportional navigation with maneuvering target," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 30, no. 1, pp. 229–237, 1994.

[6] Z. H. Bai, K. B. Li, W. S. Su, and L. Chen, "Real true proportional guidance intercepts the capture area of any maneuvering target," *Acta Aeronautica et Astronautica Sinica*, vol. 41, no. 8, pp. 338–348, 2020.

[7] K. B. Li, Z. H. Bai, H. S. Shin, A. Tsourdos, and M. J. Tahk, "Capturability of 3D RTPN guidance law against true-arbitrarily maneuvering target with maneuverability limitation," *Chinese Journal of Aeronautics*, vol. 644, pp. 4511–4528, 2021.

[8] D. Zhou, "New guidance laws for homing missile," *Beijing: National Defense Industry Press*, vol. 29, pp. 97–110, 2020.

[9] X. S. Huang, "Missile guidance and control systems design," *Changsha: National University of Defense Technology Press*, pp. 180–215, 2013.

[10] K. B. Li, L. Chen, and X. Z. Bai, "Differential geometry modeling of interceptor guidance," *SCINCE CHINA: Technological Sciences*, vol. 41, no. 9, pp. 1205–1217, 2011.

[11] S. M. He, "Research on precision guidance technology based on sliding mode control," *Beijing: Beijing Institute of Technology*, pp. 1–15, 2016.

[12] K. Shi, C. Liu, Z. Sun, and X. Yue, "Coupled orbit-attitude dynamics and trajectory tracking control for spacecraft electromagnetic docking," *Applied Mathematical Modelling*, vol. 101, pp. 553–572, 2022.

[13] R. S. Sutton and A. G. Barto, "Reinforcement learning: an introduction," *Cambridge: MIT Press*, pp. 106–112, 1998.

[14] B. Gaudet and R. Furfaro, "Missile homing-phase guidance law design using reinforcement learning," in *AIAA Guidance, Navigation, and Control Conference*, p. 4470, Minneapolis, Minnesota, 2012.

[15] Q. H. Zhang, B. Q. Ao, and Q. X. Zhang, "Q-learning reinforcement learning guidance law," *Systems Engineering and Electronics*, vol. 42, no. 2, pp. 414–419, 2020.

[16] D. Silver, A. Huang, C. J. Maddison et al., "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[17] D. Silver, T. Hubert, J. Schrittwieser et al., "A general reinforcement learning algorithm that master chess, shogi, and Go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[18] X. L. Hou, H. Li, Z. Wang, Z. X. Wu, and H. Wen, "Design of missile terminal guidance law based on DDPG algorithm," *Tactical Missile Technology*, vol. 4, pp. 110–116, 2021.

[19] S. Y. Liu, R. L. Wu, and B. Z. Zhou, "Research on midcourse guidance of exo-atmospheric interceptor," *Journal of Astronautics*, vol. 2, pp. 156–163, 2005.

[20] Y. Tian and Z. Ren, "Design of terminal guidance law of exo-atmospheric kinetic kill vehicle," *Journal of Astronautics*, vol. 30, no. 2, pp. 474–480, 2009.

[21] C. Liu, X. Yue, J. Zhang, and K. Shi, "Active disturbance rejection control for delayed electromagnetic docking of spacecraft in elliptical orbits," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 3, pp. 2257–2268, 2022.

[22] C. Liu, X. Yue, and Z. Yang, "Are nonfragile controllers always better than fragile controllers in attitude control performance of post-capture flexible spacecraft?," *Aerospace Science and Technology*, vol. 118, p. 107053, 2021.

[23] C. Liang, W. Wang, Z. Liu, C. Lai, and B. Zhou, "Learning to guide: guidance law based on deep meta-learning and model predictive path integral control," *IEEE Access*, vol. 7, pp. 47353–47365, 2019.

[24] M. Du, C. Peng, and J. Ma, "Deep reinforcement learning based missile guidance law design for maneuvering target interception," in *Proceedings of the 40th Chinese Control Conference*, pp. 3733–3738, IEEE, Shanghai, 2021.

[25] S. Gutman, "Exo-atmospheric interception via linear quadratic optimization," *Journal of Guidance Control and Dynamics*, vol. 42, no. 3, pp. 624–631, 2019.

[26] A. Ratnoo and D. Ghose, "Collision-geometry-based pulsed guidance law for exo-atmospheric interception," *Journal of Guidance Control & Dynamics*, vol. 32, no. 2, pp. 669–675, 2009.

[27] B. Gaudet, R. Furfaro, R. Linares, and A. Scorsoglio, "Reinforcement meta-learning for interception of maneuvering exo-atmospheric targets with parasitic attitude loop," *Journal of Spacecraft and Rockets*, vol. 2, pp. 1–14, 2020.

[28] B. Gaudet, R. Furfaro, and R. Linares, "Reinforcement learning for angle-only intercept guidance of maneuvering targets," *Aerospace Science and Technology*, vol. 99, p. 105746, 2020.

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

[30] L. Busoniu, T. De Bruin, D. Tolic, J. Kober, and I. Palunko, "Reinforcement learning for control: performance, stability, and deep approximators," *Annual Review in Control*, vol. 46, pp. 8–28, 2018.

[31] M. Sewak, "Deep reinforcement learning: Frontiers of artificial intelligence," *Singapore: Springer Singapore*, pp. 127–140, 2019.

[32] J. X. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, and M. Botvinick, "Learning to reinforcement learn," 2016, https://arxiv.org/abs/1611.05763.

[33] R. S. Sutton, D. Mcallester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function ap-proximation," *Advances in Neural Information Processing System*, vol. 12, pp. 1057–1063, 1999.

[34] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," 2018, https://arxiv.org/abs/1802.09477.

[35] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *ICLR: San Juan Puerto Rico*, 2015, https://arxiv.org/abs/1511.05952.

[36] K. B. Li, S. Hyo-Sang, T. Antonios, and T. Min-Jea, "Performance of 3-D PPN against arbitrarily maneuvering target for homing phase," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 5, pp. 3878–3891, 2020.

[37] K. B. Li, S. Hyo-Sang, and T. Antonios, "Capturability of a sliding-mode guidance law with finite-time convergence," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 3, pp. 2312–2325, 2019.

[38] K. B. Li, S. Hyo-Sang, T. Antonios, and T. Min-Jea, "Capturability of 3D PPN against lower-speed maneuvering target for homing phase," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 1, pp. 711–722, 2019.

[39] W. Y. Yang, C. J. Bai, and C. Cai, "Review on sparse reward in deep reinforcement learning," *Computer Science*, vol. 47, no. 3, pp. 183–191, 2020.