Hindawi
International Journal of Aerospace Engineering
Volume 2023, Article ID 4188037, 12 pages
https://doi.org/10.1155/2023/4188037

Research Article

Homing Guidance Law Design against Maneuvering Targets Based on DDPG

Yangang Liang,[1,2] Jin Tang,[1,2] Zhihui Bai,[3] 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*

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

Received 6 March 2023; Revised 28 April 2023; Accepted 18 May 2023; Published 13 June 2023

Academic Editor: Shaoming He

A novel homing guidance law against maneuvering targets based on the deep deterministic policy gradient (DDPG) is proposed. The proposed guidance law directly maps the engagement state information to the acceleration of the interceptor, which is an end-to-end guidance policy. Firstly, the kinematic model of the interception process is described as a Markov decision process (MDP) that is applied to the deep reinforcement learning (DRL) algorithm. Then, an environment of training, state, action, and network structure is reasonably designed. Only the measurements of line-of-sight (LOS) angles and LOS rotational rates are used as state inputs, which can greatly simplify the problem of state estimation. Then, considering the LOS rotational rate and zero-effort-miss (ZEM), the Gaussian reward and terminal reward are designed to build a complete training and testing simulation environment. DDPG is used to deal with the RL problem to obtain a guidance law. Finally, the proposed RL guidance law's performance has been validated using numerical simulation examples. The proposed RL guidance law demonstrated improved performance compared to the classical true proportional navigation (TPN) method and the RL guidance policy using deep-Q-network (DQN).

1. Introduction

Intercepting the maneuvering targets is a particular challenge due to the complexity of the engagement [1, 2]. The traditional guidance and control system for interception show its weakness when facing high maneuvering targets, but intelligent methods can solve the problem [3]. In the field of guidance, proportional navigation (PN) has found widespread applications because of the features of simplicity and robustness [4]. PN is mainly divided into true proportional navigation (TPN) [5] and pure proportional navigation (PPN) [6]. For maneuvering targets, Ref. [7] investigated the capture region of the realistic true proportional navigation (RTPN) in three-dimensional (3D) space, taking into account the nonlinearity of the interceptor-target relative kinematics, resulting in more general findings. However, when targets exhibit large maneuvering, the performance of proportional navigation (PN) can significantly decline. This is mainly due to the commanded acceleration of PN often exceeding the capability of the interceptor, resulting in large miss distances [8]. Optimal guidance law (OGL) can intercept or strike a target with a specific optimized performance index [9]. However, the time-to-go needs to be accurately estimated in OGL; otherwise, the performance may decline. Many new guidance methods such as differential geometry [10], sliding mode control [11], and other dynamic and control theories have also been proposed. However, these guidance laws are often too complex in form, which usually require too much measurement information and involve plenty of guidance parameters, and, hence, are difficult to be applied in practice.

Reinforcement learning (RL) [12] provides a new idea for the homing guidance law design problem. For example, Q-learning is used for the adaptive determination of parameters in [13] and [14] by training. In [15], a guidance framework designed by RL-based guidance law is proposed. It has been proven that guidance laws based on RL are much better than PNG, according to plenty of numerical simulation results. However, these traditional RL-based algorithms promote the guidance performance only through selecting suitable coefficients of the controller [16], which cannot achieve precise guidance under realistic disturbed conditions. Moreover, the state and action set of the traditional RL method are discrete, and the dimension is low, while the actual interception engagement is continuous and high-dimensional [17].

As deep learning (DL) continued to advance, a new class of algorithms known as DRL, combining both DL and RL techniques, emerged [18]. The DRL method can effectively overcome the difficulties of complex space and high dimensions [19, 20], so it may have advantages in homing guidance. Ref. [21] proposed deep Q-Network (DQN), which solves the problem of high-dimensional input. Aiming at the problem of exoatmospheric homing guidance, a novel guidance method using DQN is proposed in [22]. However, DQN is more suitable for the problem of discrete control, while the actual interceptor's acceleration is usually continuous. The discrete action command might lead to a large deviation and also a big miss distance.

The DDPG algorithm, introduced in [18], is an actor-critic (AC) [23] algorithm that is well suited for the homing guidance problem in continuous state and action space environments. Ref. [24] explored the possibility of applying DDPG to the design of homing guidance law. By comparing the two learning modes of learning from zero and learning with prior knowledge, it is proven that the latter helps to improve learning efficiency. In [25] and [26], the terminal guidance law of missiles is also advanced based on DDPG. The result shows that the proposed policy has stronger robustness and a smaller miss distance compared with PN. However, most DRL-based guidance laws need to measure and estimate the relative velocity and position between the target and interception and the information on target acceleration [27, 28]. The involved measurements are too many and are usually with lags and large errors. An RL-based guidance law was proposed in [29] and [30] to solve this problem, which only uses the LOS angle measurements and their change rates as the observation information. The problem of state estimation is simplified, and the bad influences caused by the estimation biases of position and velocity may be eliminated. Ref. [29] introduces proximal policy optimization (PPO) to propose a homing guidance law to intercept exoatmospheric maneuvering targets, combing with metalearning [31, 32]. Experimental results have shown that this guidance method outperforms the augmented ZEM [30] guidance method. Ref. [33] proposed a model-based DRL method, which uses deep neural networks and metalearning to approximate the predictive model of the guidance dynamics and incorporates it into the control framework of path integration. It introduces a general framework for guidance, but it is complex in form, and the problem of estimation errors is still not solved.

A novel homing guidance law against maneuvering targets using the DDPG algorithm is proposed in this paper, which directly maps an engagement state information to the commanded acceleration, which is an end-to-end and model-free guidance policy. The homing guidance law we proposed only takes the information of LOS angles and LOS rates between the target and interceptor as observation and state inputs and does not require prior estimation of the target's acceleration. DDPG algorithm can effectively solve a continuous and high dimensional dynamic environment. Continuous action space is designed based on the interceptor's acceleration overload. The LOS rate and ZEM are mainly considered in the design of reward, and the agent is trained in a 3D environment. The results of comparison with TPN and DQN-based RL guidance law show that the proposed guidance method is with strong environmental adaptability and better guidance performance.

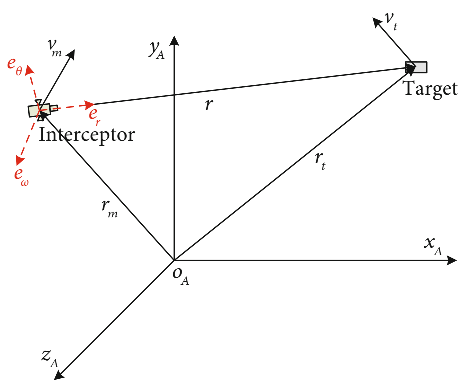


Figure 1: The interception geometry in 3D.

The paper is structured as follows: Section 2 presents the problem formulation, including the engagement scenario and the model of motion and measurement. Section 3 mainly introduces the DDPG algorithm, and the details of RL guidance law are described. The results are given in Section 4, and Section 5 presents the conclusion.

2. Problem Formulation

2.1. Engagement Scenario. The interception process, a simplified engagement scenario, is used. Referring to Figure 1, the target's and the interceptor's position vectors are r_t and r_m. r is the relative position in the launch inertial coordinate system. The velocity vectors are v_t and v_m, the relative velocity vector is v. The accelerations are defined as a_t and a_m, and the relative acceleration vector is $a = a_t - a_m$.

For the process of interception, the closing velocity is usually large. If the target and interceptor maneuver along the LOS direction, it can be challenging to alter the miss distance outcome. Therefore, we assume that the interceptor maneuvers only in a plane perpendicular to the direction of LOS in the LOS coordinate system, without considering its maneuver along the LOS direction.

2.2. Motion Model of Interception. The intersection plane is formed by r and v which are shown in Figure 1, and the details are illustrated in Figure 2. The plane between the target and interceptor will rotate as a function of their relative motion [34]. Figure 2 illustrates the centroid of the interceptor at the origin o_m, with unit vectors perpendicular and

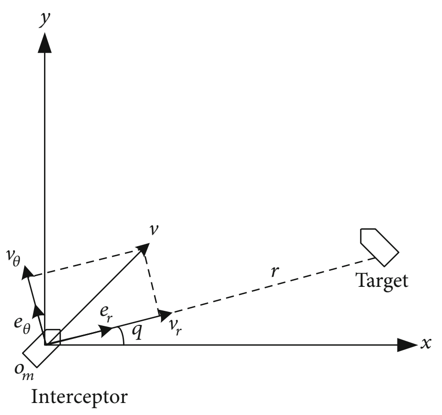


Figure 2: The geometric relationship of the intersection plane.

parallel to r denoted by $\boldsymbol{e}_\theta$ and e_r, respectively. Additionally, q represents the LOS angle within the plane.

The relative velocity is decomposed into two components: $\boldsymbol{v}_r$ represents the closing velocity, while $\boldsymbol{v}_\theta$ represents relative velocity perpendicular to the LOS. $\boldsymbol{v}_\theta$ causes the rotation of the LOS. Additionally, $\boldsymbol{\omega}_s$ denotes the angular velocity of the LOS within 3D space, $\boldsymbol{\omega}_s = \omega_s \boldsymbol{e}_\omega$ and $\boldsymbol{e}_\omega$ is perpendicular to $\boldsymbol{e}_r$ and $\boldsymbol{e}_\theta$, forming the LOS coordinate system. According to [6],

$$\dot{r} = v_r e_r + v_\theta e_\theta = \dot{r} e_r + r\omega_s e_\omega \times e_r, \tag{1}$$

$$\dot{e}_\omega = \Omega_s e_r \times e_\omega = -\Omega_s e_\theta. \tag{2}$$

Ωs represents the angular velocity. By deriving from equation (1),

$$\left(\ddot{r} - r\omega_s^2\right)e_r + \left(r\dot{\omega}_s + 2\dot{r}\omega_s\right)e_\theta + r\omega_s\Omega_s e_\omega = a. \tag{3}$$

The LOS direction can be represented using q_β and q_ε and within the launch inertial system [35]. Referring to [6], equation (4) can be obtained.

$$\omega_s = \dot{q}_\beta \sin q_\varepsilon \cdot x_S + \dot{q}_\beta \cos q_\varepsilon \cdot y_S + \dot{q}_\varepsilon z_S, \tag{4}$$

where x_S, y_S, and z_S are the coordinate axis unit vectors in the LOS coordinate system. According to [36] and [22],

$$\dot{e}_r = \dot{q}_\varepsilon y_S - \dot{q}_\beta \cos q_\varepsilon z_S, \tag{5}$$

$$e_\theta = \frac{\dot{q}_\varepsilon y_S - \dot{q}_\beta \cos q_\varepsilon z_S}{\sqrt{\left(\dot{q}_\beta \cos q_\varepsilon\right)^2 + \dot{q}_\varepsilon^2}}, \tag{6}$$

$$e_\omega = \frac{\dot{q}_\beta \cos q_\varepsilon y_S + \dot{q}_\varepsilon z_S}{\sqrt{\left(\dot{q}_\beta \cos q_\varepsilon\right)^2 + \dot{q}_\varepsilon^2}}. \tag{7}$$

In summary, when q_ε and q_β are measured, and then their rates of change are obtained by filtering, the equation of motion and the intersection plane can be determined according to equations (5)–(7).

ZEM is the final miss distance generated by the interceptor when the target and missile are not maneuvering [7, 34]. ZEM and time-to-go are calculated as follows.

$$\begin{aligned}\overrightarrow{\text{ZEM}} &= r - v \cdot t_{go},\\ t_{\text{go}} &= \frac{r \cdot v}{v^2}.\end{aligned} \tag{8}$$

2.3. Measurement Model of Interception. The measurement model is mainly to process the information measured by the interceptor and is developed to calculate the LOS angles and LOS rates of change based on the current missile-target state [37]. Referring to Section 2.1, the relative position and velocity vector are as follows:

$$\boldsymbol{r} = \left[r_x, r_y, r_z\right]^{\mathrm{T}}, \tag{9}$$

$$\boldsymbol{v} = \left[v_x, v_y, v_z\right]^{\mathrm{T}}. \tag{10}$$

By utilizing equations (9) and (10),

$$\begin{aligned}q_\varepsilon &= \tan^{-1}\left(\frac{r_y}{\sqrt{r_x^2 + r_z^2}}\right),\\ q_\beta &= \tan^{-1}\left(\frac{-r_z}{r_x}\right),\\ \dot{q}_\varepsilon &= \frac{\left(r_x^2 + r_z^2\right)v_y - r_y\left(r_x v_x + r_z v_z\right)}{r^2\sqrt{r_x^2 + r_z^2}},\\ \dot{q}_\beta &= \frac{r_z v_x - r_x v_z}{r_x^2 + r_z^2}.\end{aligned} \tag{11}$$

This paper's simulation neglects the effects of error related to the relative distance, closing velocity, and LOS angle measurements in the measurement model. Only the errors of measurement in the LOS angular rates are introduced. A Gaussian noise with zero mean and a specified standard deviation 1×10^{-4}rad/s is assigned.

3. RL Homing Guidance Law

Establishing a Markov decision model [38] of the problem is a prerequisite for designing the homing guidance law using the DRL algorithm [12]. Then, the interception problem needs to be transformed into the RL framework.

3.1. The Overview of RL. Reinforcement learning is an iterative process [39] that involves an agent interacting with the environment, observing state S_t and receiving an instantaneous reward R_t for each action A_t taken during a single episode of training. Then, it executes an action and feeds back to the environment, so that the agent learns the better policy.

Reinforcement learning algorithms are broadly categorized into two methods: value function and policy gradient [40]. The methods of the former, such as Q-learning and DQN, estimate the value of state-action pairs. The latter's methods, such as policy gradient and AC algorithm, directly

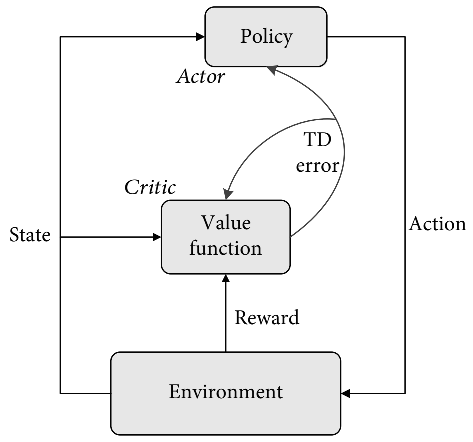


Figure 3: DDPG basic framework.

learn the policy which maps states to actions. DRL algorithms, such as DDPG and A3C, combine deep learning with these methods. However, value function methods are not suitable for problems with high dimensions and continuous action spaces, and the policy gradient method based on AC architecture has more advantages for such problems. DDPG is used for solving the problem and is compared against the TPN and DQN algorithms in this paper.

3.2. DDPG Algorithm. DDPG is based on AC architecture for solving RL problems with continuous spaces in state and action. The algorithm uses neural networks to approximate the functions, which are represented by the value network (part of the critic) and policy network (part of the actor). The value network calculates the state or action values of the corresponding state, while the policy network calculates the action values of the policy. The DDPG framework is shown in Figure 3.

A dual network is also used in the DDPG algorithm, namely, the current and the target network. The AC-type algorithm generally includes a policy and value network, so DDPG has a total of four networks after using a dual network [18].

DDPG also uses the replay buffer to reduce the correlation between training data. During training, the agent randomly selects small batches of data from the experience replay pool to calculate network loss and gradient and then updates the current policy network and value network through gradient backpropagation. DDPG differs from DQN in that it implements a soft update approach to update the target network, as opposed to periodically copying parameters from the current network. The soft update slowly updates the parameters each time, and it is mathematically expressed as follows:

$$\begin{aligned} w' &\longleftarrow \tau w + (1-\tau)w', \\ \theta' &\longleftarrow \tau\theta + (1-\tau)\theta', \end{aligned} \tag{12}$$

where τ is the update coefficient. To avoid the local optimum in the process of exploring the state space, random noise μ is added to the action, which is expressed as follows:

$$a = \pi_\theta(s) + \mu, \tag{13}$$

where $\pi_\theta(s)$ is the output of actor network. The loss is also obtained by temporal-difference (TD) training. The pseudocode is shown in Algorithm 1.

3.3. RL Model of Interception. To solve the problem of interception using DDPG, the original problem needs to be transformed into the framework of RL. First, the corresponding MDP is established, and the elements of reinforcement learning are designed according to the motion model in Section 2.2.

3.3.1. State. The process of interception can be described by an MDP. The environment of this process is composed of the 3D motion model established in Section 2. The state space designed mainly includes LOS angle and their change of rate [29], which is expressed as follows:

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

where Δq_ε and Δq_β are the LOS angle differences. Therefore, this information can be used for the state input of the agent only by measuring the LOS angles and their rates. It is assumed that the interceptor has a detection capability. The process is observable, and the variables in this paper are defined as follows:

$$O = \left[\boldsymbol{r}_\mathrm{t}, \boldsymbol{v}_\mathrm{t}, \boldsymbol{r}_\mathrm{m}, \boldsymbol{v}_\mathrm{m}, \Delta q_\varepsilon, \Delta q_\beta, \dot{q}_\varepsilon, \dot{q}_\beta\right]. \tag{15}$$

3.3.2. Action. The DDPG algorithm is particularly appropriate for problems with continuous actions. Considering the continuous maneuvering form of the interceptor, without considering the maneuvering along the LOS, that is, the interceptor maneuvers along the plane perpendicular to LOS. Therefore, if the interceptor's acceleration in y_S and z_S directions is u_1 and u_2, respectively, the continuous action space is expressed as follows:

$$A = [u_1, u_2], u_1, u_2 \in [-a_{\max}, a_{\max}]. \tag{16}$$

The total acceleration acting on the interceptor is as follows:

$$a_m = \sqrt{{u_1}^2 + {u_2}^2}. \tag{17}$$

We assume that the maneuvering target's maximal overload and the interceptor in a certain direction are 3 g and 6 g, respectively, so the target's and the interceptor's maximum total overload are $3\sqrt{2}$ g and $6\sqrt{2}$ g.

3.3.3. Reward. The reward design is the key to RL problems. To ensure the training converges to the optimum, the method of reward shaping [41] is used to avoid the problem of reward sparsity and learn the optimal policy.

The LOS rate and ZEM are considered in the reward function of the model. During the interception, the LOS rate is positively correlated with the relative velocity. The smaller

```
1. Initialize network parameters and target Q network parameters $w, \theta, w'=w, \theta'=\theta$.
2. Initialize replay pool $D$.
3. For $episode = 0$ to T
4.   Interceptor's state $s_0$ is initialized
5.   For $s = s_0$ to Terminations:
6.     a) Output action $a = \pi_\theta(s) + \mu$ according to state $s$ in policy network.
7.     b) Execute $a$, transfer to $s'$, and get reward $r$. Judge termination $d$.
8.     c) Store transitions $\{s, a, s', r, d\}$.
9.     d) Sample $n$ transitions from $D$.
10.     e) Compute the current target Q value $y_i$.
         $y_i = r_j + (1-d)\gamma Q'(s_j', \pi_{\theta'}(s_j'), w')$.
11.     f) Compute the loss $c\_loss = 1/n\sum_{j=1}^{n}(y_j - Q(s_j, a_j, w))^2$. Update value network parameter $w$ by backpropagation.
12.     g) Compute the loss of policy network $a_{loss} = -1/n\sum_{j=1}^{n}(s_j, \pi_\theta(s_j), \theta))$. Update policy network parameter $\theta$ by backpropagation.
13.     h) Update parameters in target networks with equation (12).
14.     i) Update state: $s = s'$.
```

Algorithm 1: DDPG for homing guidance law.

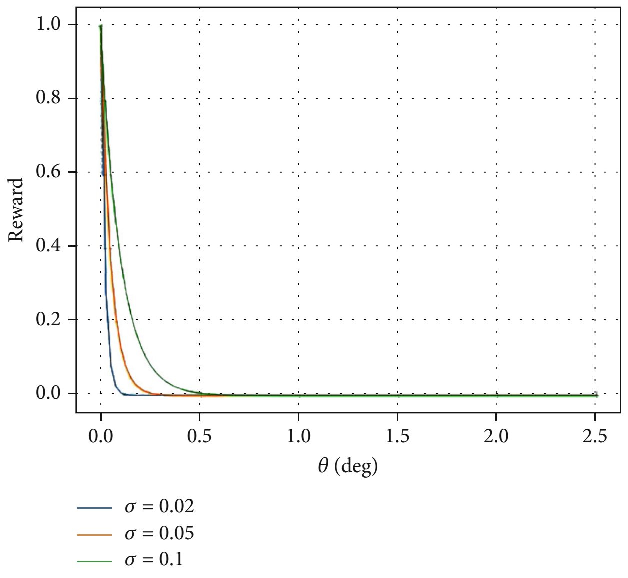


Figure 4: The Gaussian reward when σ takes different values.

the absolute value of the relative velocity, the smaller the ZEM. The Gaussian reward [30] is designed as follows:

$$R_1 = \exp(-|\theta|/\sigma). \tag{18}$$

The reward is a shaping reward that depends on the velocity-leading angle θ between the LOS direction and the relative velocity. The reward coefficient σ is used for adjusting the reward value. Figure 4 illustrates the effect of different σ values on the reward. Smaller values of σ result in smaller rewards under the same conditions. When θ is smaller, it indicates that the corresponding reward is larger.

To ensure effective interception of the target, a terminal reward constraint is required. Therefore, a terminal reward function is designed. If the ZEM meets with the allowable miss distance, then a positive reward (+10) is given. Otherwise, there is no reward value (+0). Specifically expressed as:

Table 1: The structure of the value network.

Layers	Neurons	Activation functions
Input of state	4	\
Input of action	2	\
Hidden layer 1	60	Relu
Hidden layer 2	40	Relu
Output	1	\

$$R_2 = \begin{cases} +10 & \text{if } \mathrm{ZEM}_f \le r_{\mathrm{Miss}} \\ 0 & \text{else} \end{cases}. \tag{19}$$

To sum up, the total reward is as follows:

$$R = R_1 + R_2. \tag{20}$$

3.4. Create the Agent. Based on the established framework of interception, the network is further designed, the algorithm hyperparameters are debugged, and the DDPG agent is trained.

3.4.1. The Neural Network. The Tensorflow framework is used for building the neural network of DDPG. DDPG contains two parts: value and policy network. For the value network, the output is the action value corresponding to the state and action, which is different from the Q network in DQN. The value network uses a three-layer backpropagation (BP) neural network [42], which is shown in Table 1. Relu and tanh activation functions are used in the network [43]. The policy network structure is described in Table 2.

3.4.2. Hyper Parameter. The parameters are important to the performance of training. And this adjustment process is different in different application ranges. Different problems have different parameter sets. If the parameter setting is

Table 2: The structure of the policy network.

Layers	Neurons	Activation functions
Input of state	4	\
Hidden layer 1	60	Relu
Hidden layer 2	40	Relu
Output	1	Tanh

Table 3: The hyperparameters of DDPG.

Hyperparameter	Parameter value
Maximum iterations	2000
Discount factor	0.995
Coefficient of soft update	0.001
Reward coefficient	0.05
Capacity of experience replay pool	100000
Minibatch size	64
Environmental noise variance	1.0
Noise attenuation rate	0.99
Value network learning rate	0.002
Policy network learning rate	0.001

unreasonable, the algorithm cannot converge. Therefore, hyperparameters need to be continually adjusted during training. The hyperparameters used in this problem are determined by conducting numerous numerical simulations in the established interception environment. Table 3 shows the hyperparameters that are ultimately chosen for this study.

4. Simulations and Analysis

During the training, the state measurement errors and time constant are not considered. The motion equation of each episode is integrated by the Runge-Kutta method, whose order is 4 and the simulation step is 1 ms. Table 4 shows the initial conditions.

During training, a terminal reward with an allowable miss distance of 0.2 m was used. The results presented include the training results, a comparison with TPN, and a comparison with a homing guidance law based on DQN [22].

4.1. Results of Training. The DDPG environment is built in Tensorflow, and then the agent generates a large volume of data which is used to optimize its policy. We train the agent by a computer with NVIDIA GeForce RTX 2080 Ti GPU, Gold 6226R: 2.90GHz CPU. The versions of Python and Tensorflow are 3.7.6 and 1.15.0.

Tensorboard is used to show the process of training, and it took approximately 9401.2702 seconds to train 2000 episodes, equivalent to about 2.6 hours for full training. Figures 5 and 6 depict the change in the loss for the policy and value networks, respectively, with the horizontal axis representing the iterations of training. A decrease in the policy network loss corresponds to an increase in the Q-value output of the value network, as demonstrated in Figure 5.

Table 4: The initial conditions for training.

Physical parameters	Reference value
Azimuth angle of LOS	40 deg
Elevation angle of LOS	30 deg
LOS range	100 km
Interceptor's position vector	$[0, 0, 0]^T$ m
Velocity yaw angle of target	220 deg
Interceptor velocity	5 km/s
Target velocity	7 km/s
Alignment deviation perpendicular to intersection plane	2 deg
Velocity pitch angle of the target	0 deg
Alignment deviation in the intersection plane	2 deg

This indicates that the parameters of the policy network are continuously optimized, resulting in maximum action value output. The loss function is in the value network, and it can be observed that in the early stages of training, TD error is relatively small. As training progresses, the network becomes increasingly optimized, with lower TD error values being more beneficial for algorithm training.

Figure 7 illustrates the changes in rewards, with the horizontal axis representing episodes and the vertical axis representing the cumulative reward (in blue) and average reward (in orange) for each round after smoothing. It can be observed that the maximum cumulative reward is achieved after 250 episodes. Since DDPG contains two networks, the stability of the algorithm is affected, and there may be fluctuations. Therefore, there is a certain range of change in the cumulative reward after convergence. However, the policy of the agent is optimized during training, and the convergence speed is fast.

4.2. Comparison with TPN. During training in Section 4.1, the effects of measurement errors and time delays are not considered. The agent is compared with TPN with different guidance coefficients in two ways, that is constant maneuvering and sinusoidal maneuvering. The simulation takes into account the measurement error. Additionally, the control system's response delay is assumed to be equivalent to two sampling periods (20 ms) after the guidance command.

The simulation is conducted under the following conditions: the launch location's latitude is 60°, longitude is 140°, launch azimuth is 90°, and altitude is 100 m. The target's and interceptor's initial information is presented in Table 5, indicating an initial relative distance between them of 100 km with q_ε and q_β both being 30°. The guidance acceleration's sampling period is 10 ms, with the time step being 1 ms in the test. When the relative velocity is greater than 0, the terminal miss distance approximates the ZEM.

4.2.1. Constant-Maneuvering Target. In the case of constant-maneuvering targets, the maneuvering is only considered in the LOS vertical plane. To verify the DDPG guidance law's

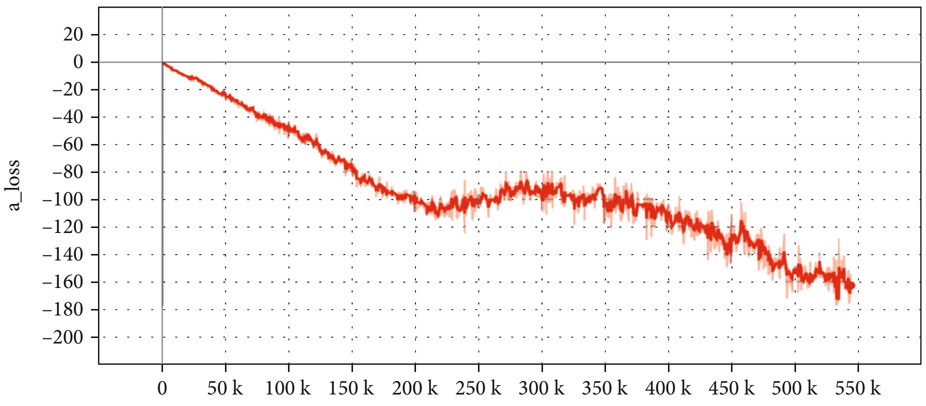


FIGURE 5: The loss of policy network.

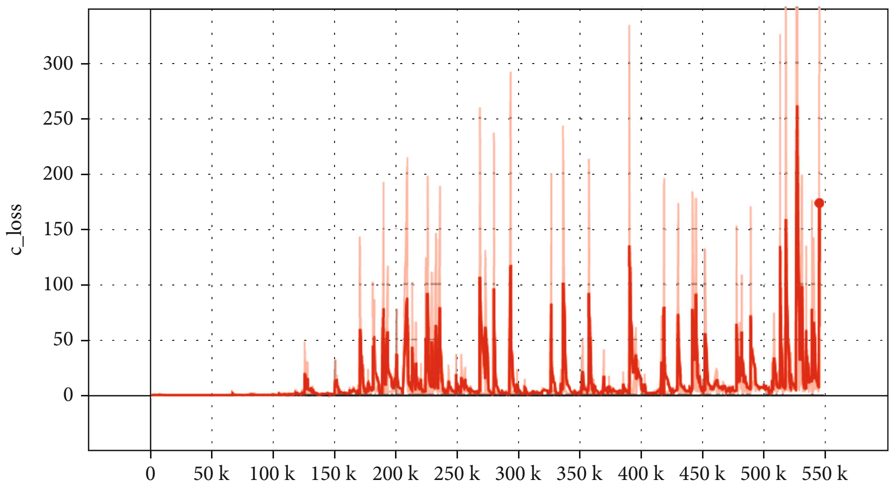


FIGURE 6: The loss of value network.

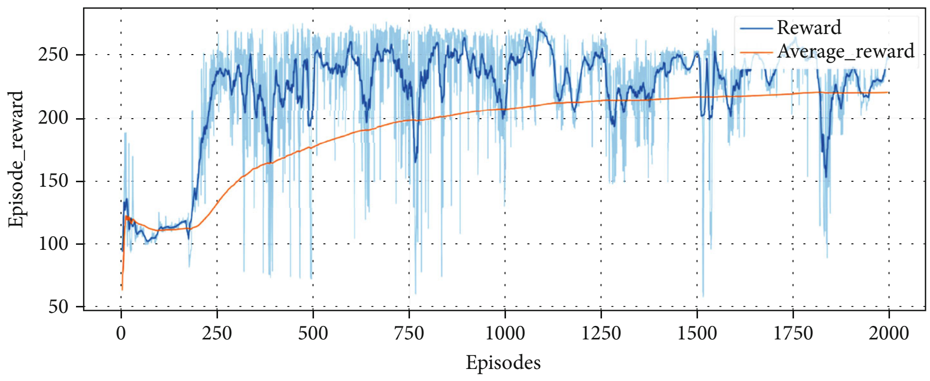


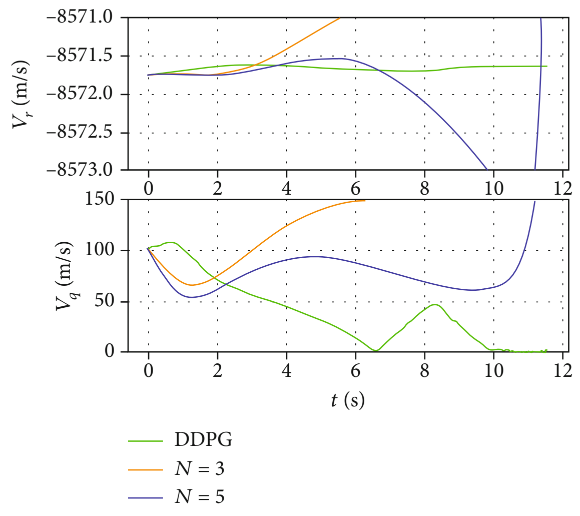
FIGURE 7: DDPG cumulative reward change.

TABLE 5: The initial conditions of the simulation.

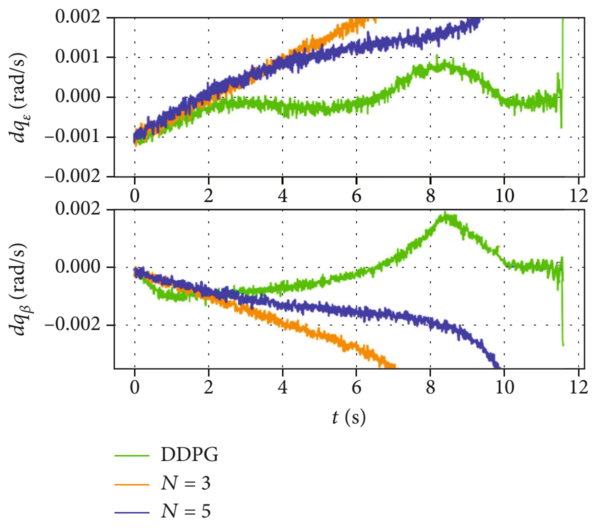
	Position (km)	Velocity (m/s)
Interceptor	(0, 0, 0)	(338.7, 4984, -211)
Target	(70, 50, -33.3)	(-6039, 610, 3486)

generalization ability, we assume the target's acceleration is $\boldsymbol{a}_t = [0, 4g, 4g]^T$. The interceptor's total overload is set to $6\sqrt{2}$ g, which differs from the target and interceptor setting during training. The guidance coefficient is 3 and 5. Figure 8 shows the results.

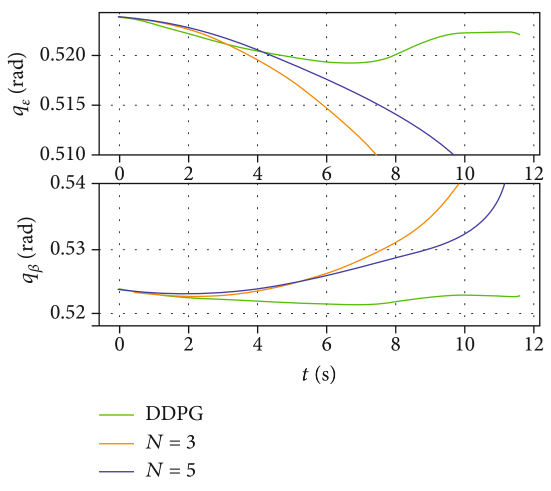
The terminal miss distance is given in Table 6. In Figure 8(a), the TPN in both cases of N taking 3 and 5 cannot reduce the vertical velocity. When the time-to-go



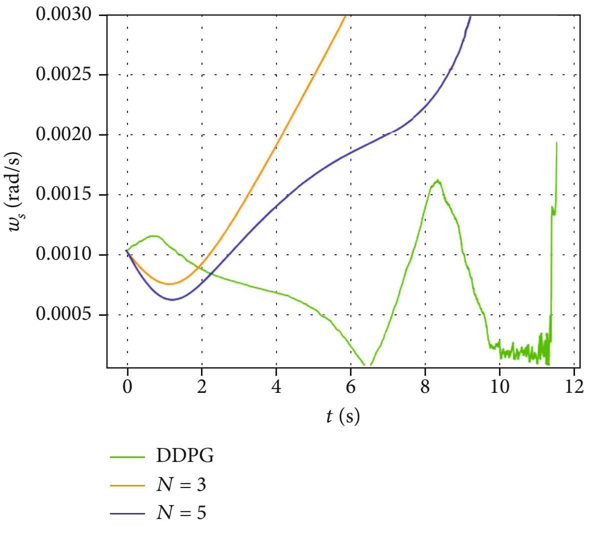
(a) v_r and v_q



(b) dq_ε and dq_β



(c) q_ε and q_β



(d) ω_s

FIGURE 8: DDPG results of the constant-maneuvering target.

TABLE 6: The comparison results of terminal miss distance.

	TPN ($N = 3$)	TPN ($N = 5$)	DDPG guidance law
Constant maneuvering	414 m	68 m	0.16 m
Sinusoidal maneuvering	12.5 m	0.93 m	0.1 m

decreases, v_q increases continuously, and the closing velocity also changes greatly. In Figure 8(d), the guidance law based on DDPG effectively suppresses the LOS rate. According to the results, when N is 3 and 5, TPN eventually produces a large miss distance, while DDPG guidance law produces a small miss distance.

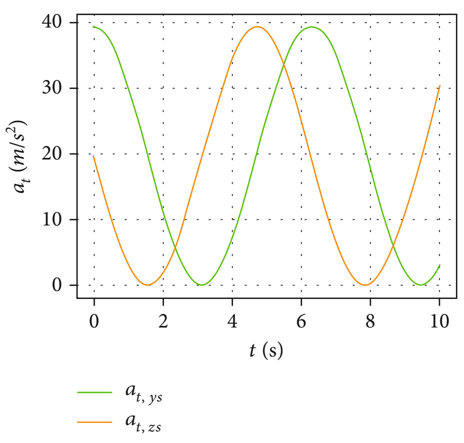


FIGURE 9: The acceleration in sinusoidal maneuvering target.

4.2.2. Sinusoidal-Maneuvering Target. The target's acceleration is $a_t = [0, 2\mathrm{g} + 2\mathrm{g}\cos(t), 2\mathrm{g} - 2\mathrm{g}\sin(t)]^T$, and the over-

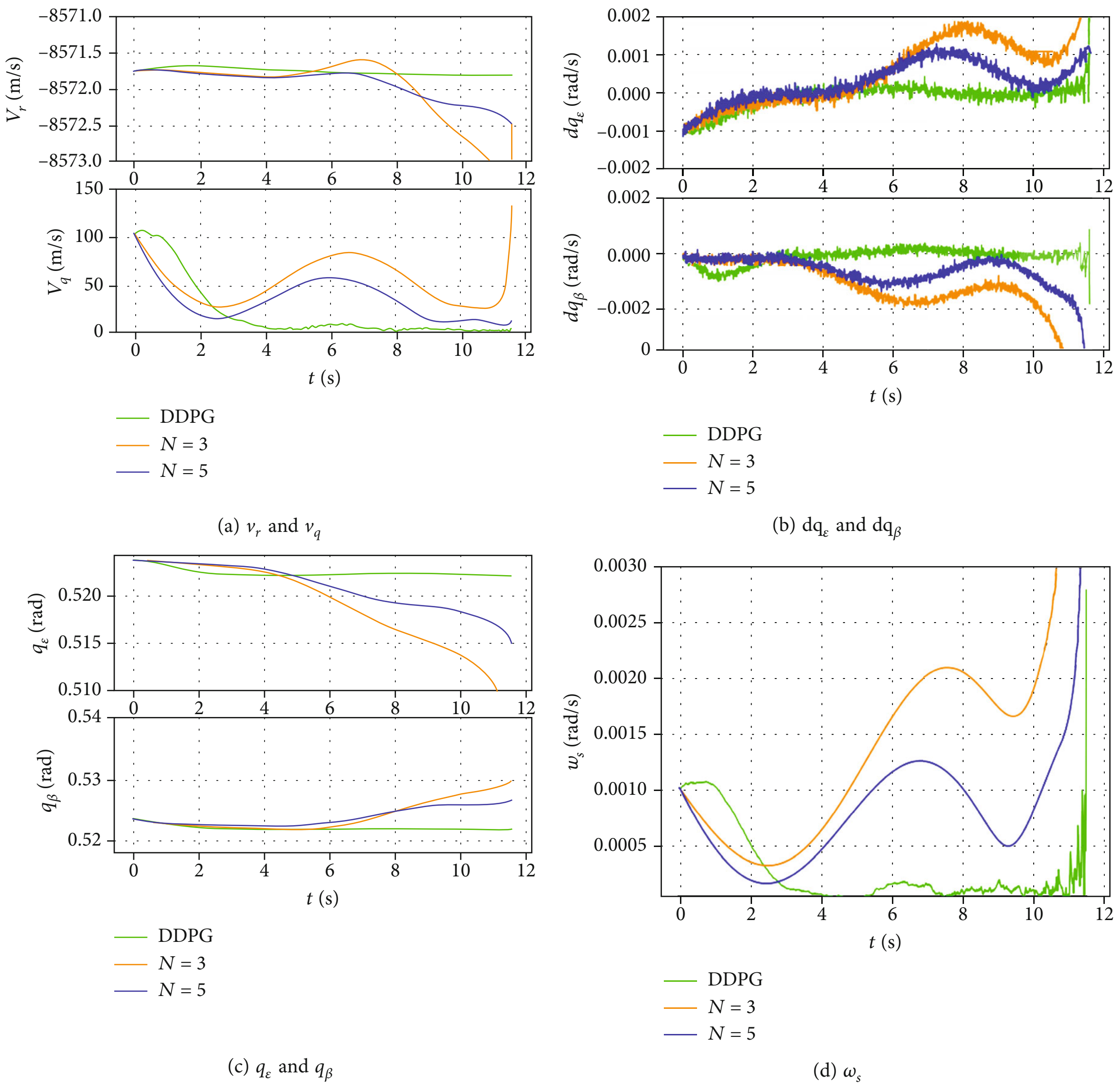


Figure 10: DDPG results of sinusoidal maneuvering target.

Table 7: The initial conditions of the target and interceptor.

	Position (km)	Velocity (m/s)
Target	(66.34, 50, -55.67)	(-5362, 0, 4499)
Interceptor	(0, 0, 0)	(872, 4860, -785)

load saturation is $4\sqrt{2}$g. Figure 9 shows the acceleration changing with time.

The guidance coefficient also takes 3 and 5. Figure 10 gives the simulation results. In Figure 10(a), the guidance law based on DDPG can reduce the vertical velocity more fully than TPN. In Figure 10(d), the change of the LOS rate also shows that DDPG can reduce the LOS rate more effectively during the guidance process.

The coefficient N takes 3 and 5 in this case as well. The results are presented in Figure 10. Table 6 reveals the terminal miss distance for the DDPG-based guidance law and TPN. Figure 10(a) suggests that the DDPG method can reduce the vertical velocity to a greater extent than TPN. Additionally, Figure 10(d) illustrates that DDPG is more effective in reducing the LOS rate during the guidance process.

The above results show that the proposed RL method is more effective than the TPN to intercept targets with certain maneuvering ability. The DDPG guidance law is capable of effectively reducing the vertical relative velocity, ensuring a very small final miss distance, and mitigating the divergence of the LOS rate.

4.3. Comparison with DQN. During the training process in Section 4.1, the target's maximum overload is $3\sqrt{2}$ g. In the test of this section, the initial conditions are kept unchanged. The target moves in a mode of constant maneuvering, and its maximum overload is also $3\sqrt{2}$ g. The acceleration form is $\boldsymbol{a}_t = [0, 3g, 3g]^T$. The interceptor is guided by the guidance

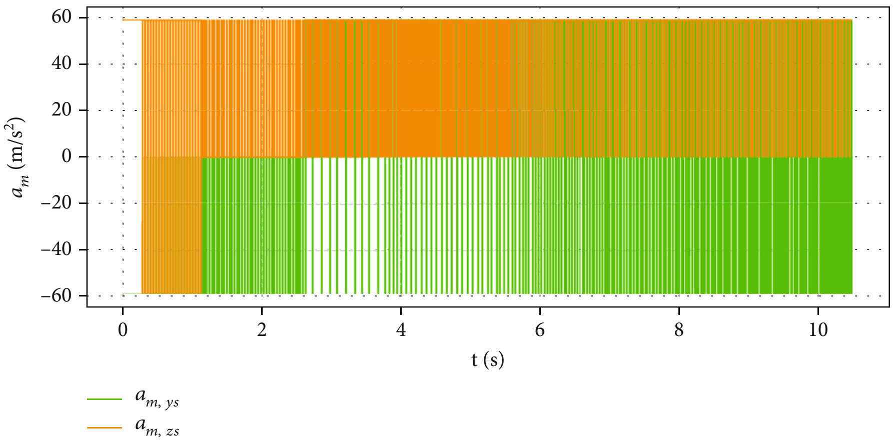


FIGURE 11: Acceleration of DQN command.

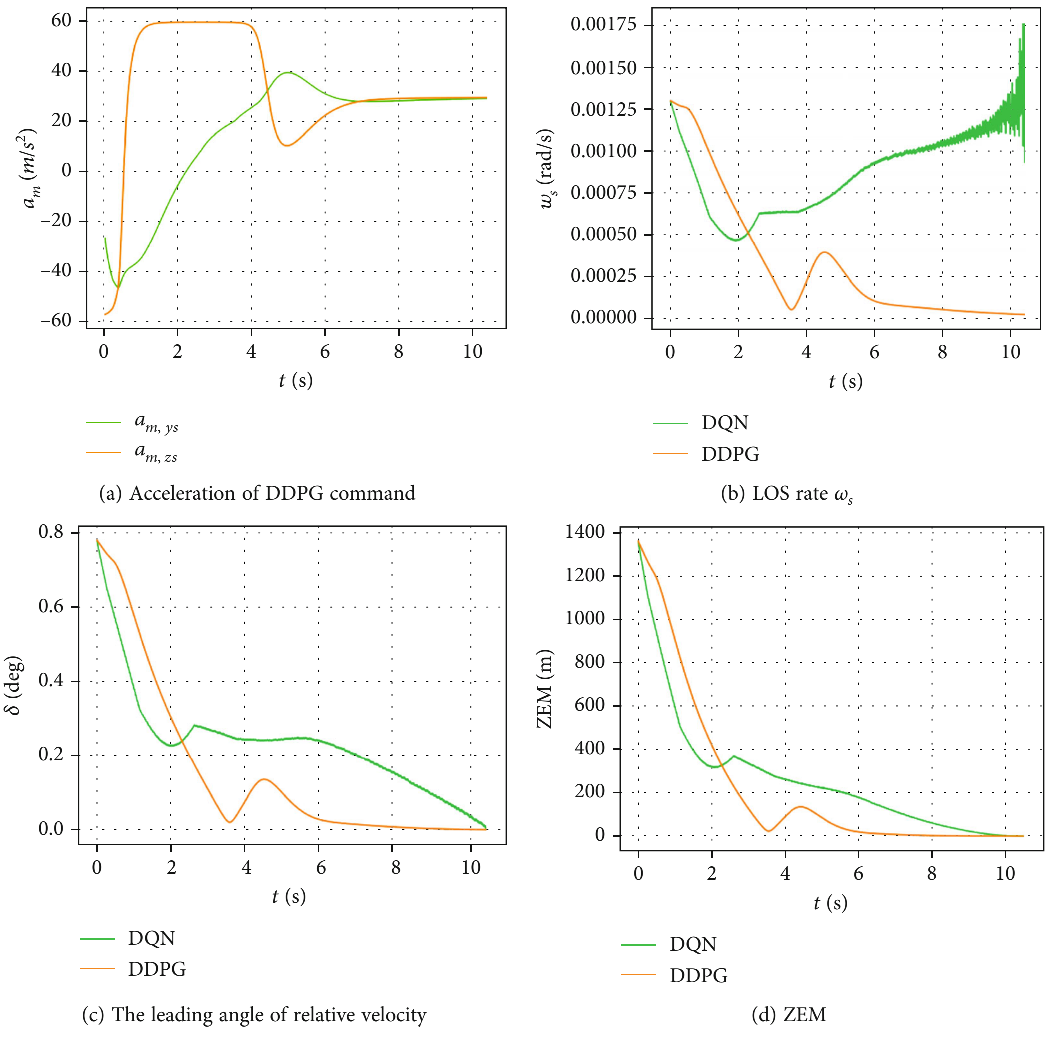


FIGURE 12: Comparative simulation results of DQN and DDPG.

laws based on DDPG and DQN [22]. The initial conditions are presented in Table 7, and Figures 11 and 12 show the results.

Figures 11 and 12(a) show that the two methods are suitable for discrete (DQN) and continuous (DDPG) accelerations, respectively. As shown in Figure 12(d), the terminal miss distances in DQN and the DDPG guidance law are less than the allowable miss distance, and both are less than 0.01 m after calculating. When the target's overload saturation is $3\sqrt{2}$ g, both guidance laws can fully suppress the

LOS rate's divergence. In Figure 12(b), the LOS rate of the DQN guidance law fluctuates in the end, while the LOS rate of the DDPG guidance law decreases to 0. Therefore, the DDPG guidance law performs better than the DQN. In Figures 12(c) and 12(d), the velocity leading angle of the two guidance laws decreases continuously during the interception process, that is, v_q decreases continuously. At the same time, the ZEM generated by the interceptor also decreases, and it effectively hits the target with a small miss distance at the end time.

In addition, when the total overload saturation of the target is less than $3\sqrt{2}\,g$, the simulation is compared and verified. The two RL-based guidance laws can effectively intercept the target from the results, and the final miss distance is within the allowable miss distance. However, the DDPG guidance law can deal with the continuous action space, which is more realistic. Moreover, the DDPG guidance law can effectively suppress the LOS rate (see Figure 12(b)), while the DQN guidance law diverges at the end of time, indicating that the DDPG method performs better. Based on the simulations conducted above, it can be observed that when the target possesses some maneuvering capability, both RL-based guidance laws can ensure effective target interception, with the DDPG guidance law outperforming the DQN guidance law.

5. Conclusion

We propose a DDPG-based guidance law for the guidance and control of interceptors with continuous maneuvering capabilities in this paper. The DDPG agent is developed using TensorFlow and optimized in the interception engagement scenario. Taking into account the effects of measurement errors and time delays in guidance control, the effectiveness of the proposed guidance law is compared with TPN and DQN-based RL guidance law through simulations of typical examples. The findings suggest that the DDPG-based guidance law outperforms the other two in terms of guidance performance. Future research could consider more complex interception scenarios, exploring more suitable intelligent guidance methods with potential implementation in real interception processes.

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 supported by the National Natural Science Foundation of China (no.: 12002370).

References

[1] D. Hong, M. Kim, and S. Park, "Study on reinforcement learning-based missile guidance law," *Applied Sciences*, vol. 10, no. 18, p. 6567, 2020.

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

[3] Y. F. Nie, Q. J. Zhou, and T. Zhang, "Research status and prospect of guidance law," *Flight Mechanics*, vol. 19, no. 3, pp. 7–11, 2001.

[4] P. Zarchan, *Tactical and Strategic Missile Guidance*, American Institute of Aeronautics and Astronautics, New York, NY, USA, 2012.

[5] K. B. Li, W. S. Sun, and L. Chen, "Performance analysis of realistic true proportional navigation against maneuvering targets using Lyapunov-like approach," *Aerospace Science and Technology*, vol. 69, no. 10, pp. 333–341, 2017.

[6] 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.

[7] K. B. Li, Z. H. Bai, H. S. Shin, A. Tsourdos, and M. J. Tahk, "Capturability of 3D RTPN against true-arbitrarily maneuvering target with maneuverability limitation," *Chinese Journal of Aeronautics*, vol. 644, pp. 4511–4528, 2022.

[8] 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.

[9] X. S. Huang, *Missile Guidance and Control Systems Design*, National University of Defense Technology Press, Changsha, China, 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] Z. X. Li and R. Zhang, "Time-varying sliding mode control of missile based on suboptimal method," *Journal of Systems Engineering and Electronics*, vol. 32, no. 3, pp. 700–710, 2021.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, 1998.

[13] T. R. Li, B. Yang, R. Wang, and J. P. Hui, "Guidance method of reentry vehicle based on Q-learning algorithm," *Tactical Missile Technology*, vol. 5, pp. 44–49, 2019.

[14] 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.

[15] 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.

[16] G. L. Han, *Design of Terminal Guidance Guidance Law Based on Reinforcement Learning*, Harbin Institute of Technology, Harbin, 2019.

[17] H. Y. Li, J. Wang, S. M. He, and C. H. Lee, "Nonlinear optimal impact-angle-constrained guidance with large initial heading error," *Journal of Guidance, Control, and Dynamics*, vol. 44, no. 9, pp. 1663–1676, 2021.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel et al., "Continuous control with deep reinforcement learning," *Computer Science*, vol. 8, no. 6, p. 187, 2015.

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

[20] 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.

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

[22] J. Tang, Z. H. Bai, Y. G. Liang, F. Zheng, and K. Li, "An exoatmospheric homing guidance law based on deep Q network," *International Journal of Aerospace Engineering*, vol. 2022, Article ID 1544670, 13 pages, 2022.

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

[24] S. M. He, H. S. Shin, and A. Tsourdos, "Computational missile guidance: a deep reinforcement learning approach," *Journal of Aerospace Information Systems*, vol. 18, no. 8, pp. 571–582, 2021.

[25] 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.

[26] Y. Liu, Z. Z. He, C. Y. Wang, and M. Z. Guo, "Research on terminal guidance law design based on DDPG algorithm," *Journal of Computer Science*, vol. 44, no. 9, pp. 1854–1865, 2021.

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

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

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

[30] 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. 58, no. 2, pp. 386–399, 2021.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, https://arxiv.org/abs/1707.06347.

[32] H. Xiang, J. Lin, C. H. Chen, and Y. Kong, "Asymptotic meta learning for cross validation of models for financial data," *IEEE Intelligent Systems*, vol. 35, no. 2, pp. 16–24, 2020.

[33] 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.

[34] 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.

[35] 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, 2020.

[36] 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, 2020.

[37] S. Hyo-Sang and K. B. Li, "An improvement in three-dimensional pure proportional navigation guidance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 5, pp. 3004–3014, 2021.

[38] S. Kieninger, L. Donati, and B. G. Keller, "Dynamical reweighting methods for Markov models," *Current Opinion in Structural Biology*, vol. 61, pp. 124–131, 2020.

[39] 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.

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

[41] 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.

[42] N. Fatema, S. G. Farkoush, M. H. Hasan, and H. Malik, "Deterministic and probabilistic occupancy detection with a novel heuristic optimization and back-propagation (BP) based algorithm," *Journal of Intelligent Fuzzy Systems*, vol. 42, no. 2, pp. 779–791, 2022.

[43] A. Maniatopoulos and N. Mitianoudis, "Learnable leaky ReLU (LeLeLU): an alternative accuracy-optimized activation function," *Information*, vol. 12, no. 12, p. 513, 2021.