Hindawi
International Journal of Aerospace Engineering
Volume 2023, Article ID 5759135, 15 pages
https://doi.org/10.1155/2023/5759135

Research Article

Two-Loop Acceleration Autopilot Design and Analysis Based on TD3 Strategy

Junfang Fan,[1,2] Denghui Dou,[1,2] Yi Ji,[1] Ning Liu,[2] Shiwei Chen,[3] Huajie Yan,[1,2] and Junxian Li[1,2]

[1]*School of Automation, Beijing Information Science & Technology University, Beijing 100192, China*
[2]*Beijing Key Laboratory of High Dynamic Navigation Technology, Beijing Information Science & Technology University, Beijing 100192, China*
[3]*School of Aerospace Engineer, Beijing Institute of Technology, Beijing 100081, China*

Correspondence should be addressed to Junfang Fan; wyhffjf@bistu.edu.cn

Received 12 January 2023; Revised 3 March 2023; Accepted 9 March 2023; Published 23 March 2023

Academic Editor: Xingling Shao

A two-loop acceleration autopilot is designed using the twin-delayed deep deterministic policy gradient (TD3) strategy to avoid the tedious design process of conventional tactical missile acceleration autopilots and the difficulty of meeting the performance requirements of the full flight envelope. First, a deep reinforcement learning model for the two-loop autopilot is developed. The flight state information serves as the state, the to-be-designed autopilot control parameters serve as the action, and a reward mechanism based on the stability margin index is designed. The TD3 strategy is subsequently used to offline learn the control parameters for the entire flight envelope. An autopilot control parameter fitting model that can be directly applied to the guidance loop is obtained. Finally, the obtained fitting model is combined with the impact angle constraint in the guidance system and verified online. The simulation results demonstrate that the autopilot based on the TD3 strategy can self-adjust the control parameters online based on the real-time flight state, ensuring system stability and achieving accurate acceleration command tracking.

1. Introduction

The autopilot is a critical component of the guidance and control system, capable of producing control forces and torques by driving actuators in response to control commands. Thus, the speed and direction of the missile's flight can be changed, the stability of the missile's centroid and attitude can be maintained, and the guided missile can hit the target based on the required flight trajectory and attitude [1, 2]. The key to designing an autopilot is identifying the control parameters that meet specific performance indices. Traditional autopilot design methods include pole placement [3, 4], optimal control [5, 6], sliding mode variable structure [7], event-triggered attitude control policy [8], and active disturbance rejection [9]. Due to the diversification of tactical missile combat scenarios and the intelligent development of missiles, the design of autopilots must meet more stringent requirements, such as improved mobility and stability at large attack angles and large-scale flight scenarios [10, 11]. However, traditional autopilot design methods select typical state feature points in the flight envelope to design the control parameters [12]. Consequently, it is difficult to meet the requirements for the entire control domain of the flight envelope.

In recent years, deep learning (DL) and reinforcement learning (RL) have become hot topics in artificial intelligence technology, providing new design options for aircraft guidance and control systems [13–15]. The principle of RL originates from the process of intelligent species learning new things. For a specific task, agents learn through real-time interaction with the external environment and continuous trial and error; ultimately, a task-appropriate action strategy is obtained [16]. Deep reinforcement learning (DRL) combines the autonomous decision-making capability of RL with the feature extraction capability of deep neural networks (DNNs) and has demonstrated outstanding performance in high-dimensional data control problems. Unlike conventional control methods, DRL is less dependent on models and more transferable [17].

In order to solve the strong coupling problem between the adaptive updating loop and the strict-feedback control loop in traditional control, a low-frequency learning structure consisting of a low-pass filter, a state estimator, and a multilayer perceptron (MLP) neural network was proposed in [18], which reduced the computation complexity and effectively avoided the update explosion problem. Literature [19] designed a two-loop autopilot based on a missile longitudinal channel model employing the RL principle and solved the quadratic optimal tracking control problem employing an actor-critic structure. The results demonstrated that the designed controller possessed an excellent tracking effect and dynamic performance. Different from literature [19], literature [20] transformed the aircraft control procedure into a Markov decision procedure. Utilizing the deep deterministic policy gradient (DDPG) algorithm [21], the optimal PID controller control parameters were determined iteratively. Compared to the LQR controller, the result of the PID controller exhibited better control effects and robustness. Similar to literature [20], based on the acceleration autopilot, literature [22] developed a control parameter design method for acceleration autopilots using the policy gradient algorithm [23]. In order to reduce the time and workload for identifying model information during the autopilot design process, literature [24] developed a learning-based design method for UAV autopilot using the DDPG algorithm by designing appropriate observation and reward functions. In addition, literature [25] made a comparative analysis of PID neural network controller and DDPG controller and also provided a conceptual proof that reinforcement learning can effectively solve the adaptive optimal control problem of nonlinear dynamic systems. However, most of the methods in [19, 20, 22, 24] and [25] were designed and analyzed based on nonglobal profile state rather than entire flight envelope, which led to insufficient consideration of global constraints and performance indicators. When there was great uncertainty in the flight environment, the autonomy and robustness of the nominal trajectory tracking guidance mode were poor.

In order to ensure that the missile can be well controlled in the entire flight envelope, literature [26] used wavelet analysis to monitor the incremental RL's attitude control stability online. Then, they adaptively modified the learning rate of the RL algorithm based on the gradient descent principle, which effectively enhanced the aircraft's control stability under large-scale dynamic changes. Literature [27] remodeled the autopilot in the RL framework, trained a two-degree-of-freedom- (DOF-) linearized dynamic model for missiles using the DDPG algorithm. By taking into account different flight conditions and uncertainties in the aerodynamic coefficients, literature [27] validated the ability of the designed controller to maintain closed-loop stability in the presence of uncertain models. Literature [28] developed a method for aircraft control using the DDPG algorithm, and the complete flight control was achieved by controlling the position, speed, and attitude angle. By adding a PD controller, the stability in the early stage of training was effectively improved. Literature [29] proposed a model-free coupling dynamic controller design method for jet aircrafts capable of withstanding multiple types of faults. After offline training, adequate results were achieved under highly coupled maneuvers and were robust to various failure situations. To surmount the reliance on global position data in hostile surroundings where GPS was being attacked or disrupted, literature [30] designed a new GPS-free cooperative elliptical circling controller, in which not only the energy consumption was reduced but also the dependency on global position was removed when multiple nonholonomic vehicles moved together. Literature [31] fixes the autopilot structure as typical three-loop autopilot, and deep reinforcement learning is utilized to learn the autopilot gains, and the state-of-the-art deep deterministic policy gradient algorithm is utilized to learn an action policy that maps the observed states to the autopilot gains. Influenced by literature [31], literature [32] used the pole placement algorithm of a three-loop autopilot as the foundation and designed an intelligent training method for converting three-dimensional control parameters into one-dimensional design parameters using the proximal policy optimization algorithm. The simulation results demonstrated that its control effect was good. However, these design methods had some problems such as poor stability and low strategy learning efficiency. How to improve the practicability of deep reinforcement learning method in the field of autopilot has become an urgent problem to be solved.

Motivated by the previous investigations, this paper employs the DRL principle to design a parameter tuning model for a two-loop autopilot using the TD3 algorithm [33], which has the following striking advantages over existing autopilot design schemes:

(1) Contrasting to the previous alternatives [19, 20, 22, 24, 25] based on nonglobal profile state, herein the proposed method takes the entire flight envelope state space as the research object. The TD3 algorithm is used to offline learn the control parameters for the entire flight envelope, which can ensure the autopilot has good performance during the flight envelope. In addition, after offline training, a fitting model with the real-time flight state of the aircraft as the input and the autopilot parameters as the output is obtained through the policy network, which can ensure the rapid online tuning under the condition of large-scale flight state changes

(2) Different from the reward design mechanism in literatures [26–29] and [31], this paper considers the stability margin when designing the reward mechanism to ensure the autopilot has a desired margin space in the entire flight envelope, which makes the autopilot has stronger robustness. In addition, compared to [31, 32] which randomly selected states during training, this paper arranges the flight

states in an orderly manner and then samples them sequentially, which can improve the strategy learning efficiency while ensuring the convergence of the algorithm

The remainder of this paper is organized as follows. Problem formulation is stated in Section 2. The specific process of two-loop acceleration autopilot design method based on TD3 strategy is given in Section 3. Section 4 is the simulation analysis of the proposed method. Conclusion of this paper is described in Section 5.

2. Problem Description

2.1. Two-Loop Autopilot Model. This paper uses an air-to-ground guided missile as the research object. The missile model is illustrated in Figure 1, where V represents the velocity, $F_y/F_x/F_g$ represent the lift, resistance, and gravity, respectively, M_z is the pitching moment, θ denotes the ballistic inclination, ϑ is the pitch angle, α represents the angle of attack (AOA), δ_z is the elevator deflection angle, L denotes the reference length, D is the projectile diameter, $c.p$ is the pressure center position of the whole missile, and $c.g$ is the centroid position.

The expressions of aerodynamic force, moment, and gravity are as follows:

$$\begin{cases} F_x = c_x q_d S, \\ F_y = c_y q_d S, \\ M_z = m_z q_d S L, \\ F_g = mg, \end{cases} \tag{1}$$

where c_x, c_y, and m_z denote the resistance coefficient, lift coefficient, and pitching moment coefficient, respectively, S denotes the reference area, q_d represents the incoming flow pressure ($q_d = \rho V^2/2$, where ρ represents the air density at the flying altitude of the missile), m is the missile mass, and g is the gravitational acceleration.

The projectile dynamics at the longitudinal plane can be described as follows:

$$\begin{cases} \dot{\alpha} = \omega_z - \dfrac{P + c_y^{\alpha} q_d S}{mV}\alpha - \dfrac{c_y^{\delta} q_d S}{mV}\delta_z + d_{\alpha}, \\ \dot{\omega}_z = \dfrac{m_z^{\bar{\omega}} q_d S L}{J_z}\omega_z + \dfrac{m_z^{\alpha} q_d S L}{J_z}\alpha + \dfrac{m_z^{\delta} q_d S L}{J_z}\delta_z + d_{\omega}, \\ a_y = \dfrac{q_d S\left(c_y^{\alpha}\alpha + c_y^{\delta}\delta_z\right)}{m} + d_q, \end{cases} \tag{2}$$

where ω_z represents the pitch angle rate, P is the axial thrust of the missile, c_y^{α} represents the derivative of the lift coefficient to the AOA, c_y^{δ} denotes the derivative of the lift coefficient to the elevator deflection angle, m_z^{α} represents the derivative of the pitching moment coefficient to the AOA, $m_z^{\bar{\omega}}$ represents the derivative of the pitching moment coefficient to the dimensionless pitch angle rate, m_z^{δ} denotes the derivative of the pitching moment coefficient to the elevator deflection angle, J_z is the pitching rotational inertia, and d_{α}, d_{ω}, and d_q represent the aerodynamic noise interference. The aerodynamic noise obeys the following bounded Gaussian distribution:

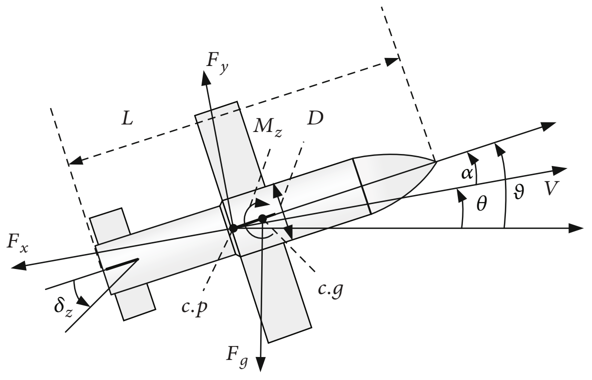


FIGURE 1: Missile model and characteristic parameters.

$$d_{\alpha/\omega/q} \sim \operatorname{clip}\left(\mathcal{N}\left(0, \sigma_{\alpha/\omega/q}\right), -e_{\alpha/\omega/q}, e_{\alpha/\omega/q}\right), \tag{3}$$

where $e_{\alpha/\omega/q}$ is the upper bound of noise, $\sigma_{\alpha/\omega/q}$ represents the variance of the Gaussian distribution $\mathcal{N}(0, \sigma_{\alpha/\omega/q})$, and the clip function is defined as follows:

$$\operatorname{clip}(x_1, y_1, z_1) = \begin{cases} x_1 & \text{if } y_1 \le x_1 \le z_1, \\ y_1 & \text{if } x_1 < y_1, \\ z_1 & \text{if } x_1 > z_1. \end{cases} \tag{4}$$

The research object of this paper is a thrust-free missile; thus, $P = 0$. Figure 2 shows the two-loop acceleration autopilot model composed of an accelerometer and a velocity gyroscope [34]. In Figure 2, a_{yc} denotes the acceleration command based on the guidance command, δ_{zc} is the actuator command with the actuator dynamic delay being temporarily ignored, and k_A and k_g are the control parameters to be designed.

The geometric relationship between the two-dimensional missile plane and the target is illustrated in Figure 3, where M represents the missile position, T is the target position, q denotes the line-of-sight angle of the missile and the target, φ is the lead angle, and R is the relative distance between missile and target.

For stationary targets,

$$\begin{cases} m\dot{V} = -F_x - F_g \sin\theta, \\ \dot{R} = -V\cos\varphi, \\ R\dot{q} = V\sin\varphi. \end{cases} \tag{5}$$

It is well-known that the dynamics of the airframe change with the flight state during the flight process. To maintain control stability and accurate acceleration tracking throughout the entire guidance procedure, the control

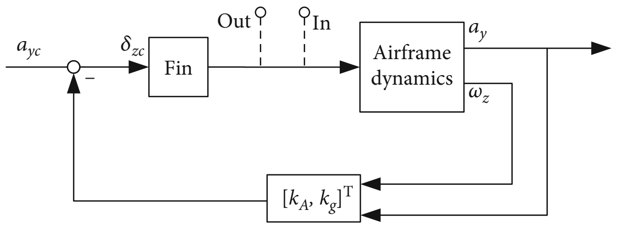


FIGURE 2: Two-loop autopilot structure.

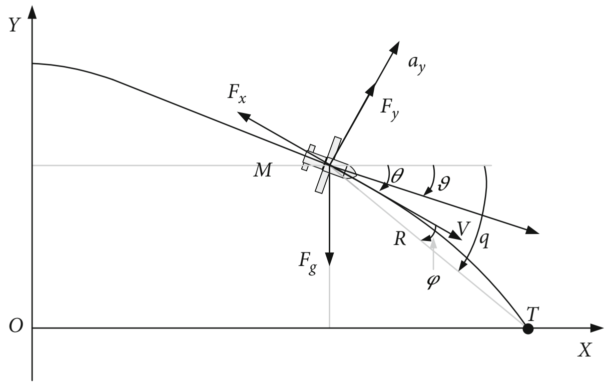


FIGURE 3: Missile-target geometric relationships.

parameters must be continuously adjusted based on the flight state. A missile guidance control model was designed in combination with the impact angle constraint guidance (Figure 4). In this paper, a two-loop autopilot with self-adjustable control parameters based on the model depicted in Figure 4 will be developed.

2.2. Autopilot Parameter Design Method and Online Application Framework. The key to the design of the autopilot is to ensure that the control parameters k_A and k_g meet the expected performance indices of the control system, so that the aircraft can accurately and robustly track the guidance acceleration command, accelerate the response speed, and improve the damping of the missile. Among the frequency domain indices of the system, the amplitude margin h_m and phase margin γ_m are often used as important indices to assess the control system performance. In general, the amplitude and phase margins of the autopilot should not be less than 6.5 dB and 40°, respectively. Therefore, in this paper, the amplitude margin h_m and phase margin γ_m are selected as the performance indices that must be satisfied during the design process of the two-loop autopilot. According to Figure 2, the conventional open-loop system $G(s)$ is the transfer function from "in" to "out," which is disconnected at the actuator. As a result, h_m and γ_m can be calculated as follows:

$$\begin{cases} h_m = \dfrac{1}{|G(j\omega_x)|}, \\ \gamma_m = 180 \circ + \angle G(j\omega_c), \end{cases} \tag{6}$$

where ω_x represents the phase crossover frequency, meeting $\angle G(j\omega_x) = -\pi$, and ω_c is the cut-off frequency, meeting $|G(j\omega_c)| = 1$.

In this paper, the guidance control model described in Section 2.1 is modeled by RL. The TD3 algorithm was utilized to offline learn the driving control parameters for each flight state across the entire flight envelope. At the same time, a MLP neural network is used to model the nonlinear relationship between different flight states and control parameters in the TD3 algorithm model. As depicted in Figure 5, the MLP neural network fitting can be directly applied to the missile guidance control loop. The corresponding autopilot control parameters can be self-adjusted online during flight based on the real-time flight state.

3. DRL Design Method for the Autopilot

3.1. Markov Decision Process. The flight state of the missile at two adjacent moments can be approximately regarded as the transfer between the states under a given control command, and the state of the next moment is only related to the state at the current moment, so the flight state of the missile has Markov property. By discretizing the flight process in the time dimension, the guidance process of the missile can be approximately modeled as a discrete-time Markov chain (DTMC). When the guidance law is determined, the control parameters determine whether the autopilot can follow the guidance command. Therefore, the design process of the autopilot is to add decision-making command to the Markov chain of the missile flight process, so the design process of the missile autopilot can be modeled as a Markov decision process (MDP).

MDP is a sequential decision process, which can be described by a 5-tuple $\langle S, A, P(.), R(.), \gamma \rangle$. The specific model definition is as follows:

$$\text{MDP}\begin{cases} S: \text{state set}, S = \{s_1, s_2, \cdots s_n\}, \\ A: \text{action set}, A = \{a_1, a_2, \cdots a_n\}, \\ P(s_{t+1}|s_t, a_t): t\text{he state transition function of the environment}, \\ R(s_t, a_t, s_{t+1}): \text{the reward function of the environment}, \\ \gamma: \text{discount factor}. \end{cases} \tag{7}$$

In this paper, the agent cannot directly access the transition function $P(s_{t+1}|s_t, a_t)$ and the reward function $R(s_t, a_t, s_{t+1})$ but can only obtain specific information about the state s_t, action a_t, and reward r_t by interacting with the environment. The interaction process between the agent and the environment in the MDP is depicted in Figure 6.

The autopilot design problem is a continuous control problem. In each round, the agent observes the state s_t at each time t and decides the action a_t to be taken according to the current strategy. The strategy π is the mapping from the state to the action ($\pi : a \sim \pi(s)$). Autopilot model gets the next state under the action and gets a reward from the environment. The trajectory generated by the control loop from s_0 to the termination state s_T is expressed as $\tau : (s_0, a_0, r_0), (s_1, a_1, r_1), \cdots, (s_1, a_1, r_1)$.

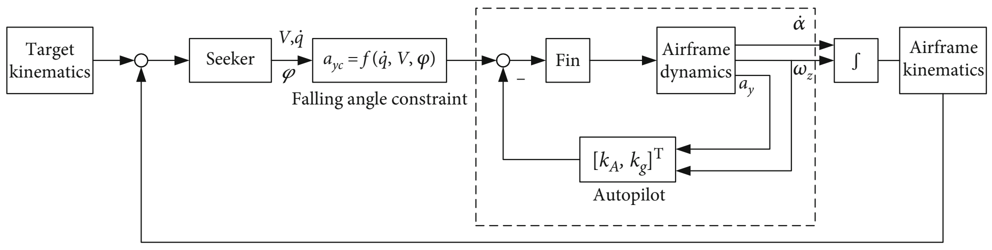


FIGURE 4: Guidance control model.

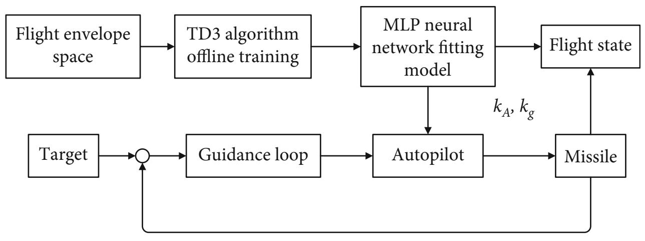


FIGURE 5: Design and online application framework of the autopilot based on the TD3 algorithm.

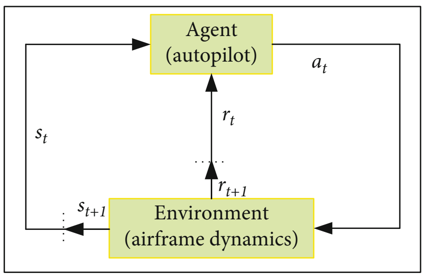


FIGURE 6: The interaction process between the agent and the environment in the MDP.

Formalizing the optimization objective, the reward is defined as the weighted sum of all rewards in the trajectory:

$$R(\tau) = \sum_{t=0}^{T} \gamma^t r_t, \tag{8}$$

where $\gamma \in [0, 1]$ represents discount factor and T represents the state number of a scene data trajectory. The objective function $J(\tau)$ is defined as the expectation of the trajectory return:

$$J(\tau) = E_{\tau\sim\pi}[R(\tau)] = E_{\tau\sim\pi}\left[\sum_{t=0}^{T} \gamma^t r_t\right]. \tag{9}$$

With $Q(s_t, a_t)$ defined as the state action-value function of $J(\tau)$, the Behrman equation of the state action-value function can be expressed as

$$Q(s_t, a_t) = E_\pi[r_t + \gamma E_\pi[Q(s_{t+1}, a_{t+1})]]. \tag{10}$$

Throughout the interaction between agent and environment, the optimal policy is continuously updated to maximize the state action-value function.

$$\pi^* = \arg\max_{\pi} Q(s_t, a_t). \tag{11}$$

The Behrman optimal equation of the state action-value function can be obtained through the Behrman optimal principle:

$$Q^*(s_t, a_t) = E_{\pi^*}[r_t + \gamma E_{\pi^*}[Q^*(s_{t+1}, a_{t+1})]]. \tag{12}$$

3.2. TD3 Algorithm. The TD3 algorithm is an off-policy algorithm for continuous action space. Based on the DDPG algorithm, the actor- and critic-networks are simultaneously improved, thereby resolving the problem of the critic-network overestimating the Q value and enhancing the algorithm's stability. To increase the agent's ability to explore the environment, the TD3 algorithm chooses actions based on the current policy and exploration noise:

$$a_t = \mathrm{clip}\left(\pi_\phi(s_t) + \epsilon, a_{\min}, a_{\max}\right), \tag{13}$$

where ϕ represents the actor-network parameter, $a_{\min}$ is the lower limit of the action space, $a_{\max}$ denotes the upper limit of the action space, and ϵ is the exploration noise which obeys the Gaussian distribution: $\epsilon \sim \mathcal{N}(0, \sigma_t)$. In addition, to ensure the algorithm convergence in the later stage, the Gaussian distribution variance σ_t obeys the following: $\sigma_{t+1} = \sigma_t(1-\varepsilon)$, where $0 \le \varepsilon \le 1$ is the attenuation factor.

The experience replay mechanism is adopted for the TD3 algorithm. The agent inputs the trial data (s_t, a_t, r_t, s_{t+1}) into the experience buffer pool R and then randomly selects the data (s_i, a_i, r_i, s_{i+1}) of the batch N from R to train the network and update its parameters. To solve the overestimation problem of the Q value, the TD3 algorithm uses two sets of critic-networks to represent different Q values and estimates the target Q value using the following equation:

$$y_i = r_i + \gamma \min_{k=1,2} Q_{\theta'_k}(s_{i+1}, \tilde{a}_{i+1}), \tag{14}$$

```
TD3: Twin Delayed Deep Deterministic Policy Gradient
Randomly initialize the network parameters $\theta_1, \theta_2$, and $\phi$
Initialize the target network parameters $\theta'_1 \longleftarrow \theta_1, \theta'_2 \longleftarrow \theta_2, \phi' \longleftarrow \phi$
Initialize the replay buffer $R$
**for** episode = 1, M **do**
  Initialize an exploration noise $\epsilon \sim \mathcal{N}(0, \sigma)$ for action exploration
  Receive the initial environmental state quantity $s_0$
  **for** t = 1, T **do**
    Select an action according to the current policy and exploration noise:
              $a_t = \mathrm{clip}(\pi_\phi(s_t) + \epsilon, a_{\min}, a_{\max})$
    Execute the action $a_t$ and observe the reward $r_t$ and the next state $s_{t+1}$
    Store the explored transition array $(s_t, a_t, r_t, s_{t+1})$ in $R$
    Extract sample data $(s_i, a_i, r_i, s_{i+1})$ of the batch $N$ from $R$
    $\tilde{a}_{i+1} = \mathrm{clip}(\pi_{\phi'}(s_{i+1}) + \tilde{\epsilon}, a_{\min}, a_{\max}), \tilde{\epsilon} \sim \mathrm{clip}(\mathcal{N}(0, \tilde{\sigma}), -c, c)$
    $y_i = r_i + \gamma \min_{k=1,2} Q_{\theta'_k}(s_{i+1}, \tilde{a}_{i+1})$
    Update the critic-network parameters:
          $\theta_k \longleftarrow \arg\min_{\theta_k} (1/N)\sum_i (y_i - Q_{\theta_k}(s_i, a_i))^2$ for $k \in \{1, 2\}$
    **if** t mod d **then**
      Update the actor-network parameters through deterministic policy gradients:
          $\nabla_\phi J(\pi_\phi) \approx 1/N \sum_i \nabla_a Q_{\theta_1}(s_i, a)|_{a=\pi_\phi(s_i)} \nabla_\phi \pi_\phi(s_i)$
      Update the target network:
          $\theta'_k \longleftarrow \mu\theta_k + (1-\mu)\theta'_k$ for $k \in \{1, 2\}$
          $\phi' \longleftarrow \mu\phi + (1-\mu)\phi'$
    **end if**
  **end for**
**end for**
```

ALGORITHM 1: TD3 algorithm.

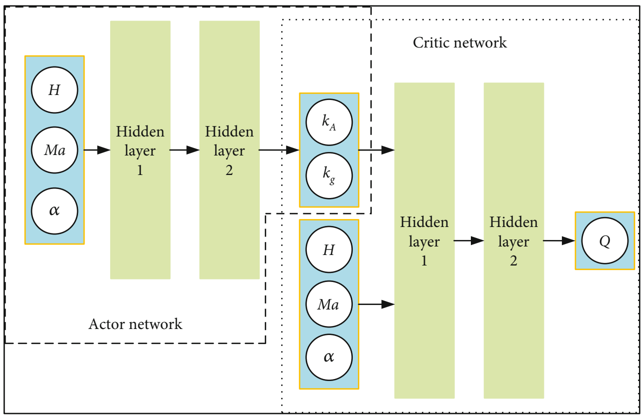


FIGURE 7: TD3 algorithm network layer structure.

where $\theta'_k(k=1,2)$ is the parameters of the target critic-network and $\tilde{a}_{i+1}$ is expressed as follows:

$$\tilde{a}_{i+1} = \mathrm{clip}\left(\pi_{\phi'}(s_{i+1}) + \tilde{\epsilon}, a_{\min}, a_{\max}\right), \tag{15}$$

where ϕ' represents the parameter of the target actor-network. Random noise $\tilde{\epsilon}$ is introduced to enhance the stability of the target policy. The noise $\tilde{\epsilon}$ follows an independent distribution controlled by parameters different from those of the exploration noise ϵ, i.e., $\tilde{\epsilon} \sim \mathrm{clip}(\mathcal{N}(0, \tilde{\sigma}), -e, e)$, where e is the upper limit of the noise.

The TD3 algorithm updates the critic-network parameter θ_k by minimizing the temporal difference (TD) error:

$$\theta_k \longleftarrow \arg\min_{\theta_k} \frac{1}{N}\sum_i \left(y_i - Q_{\theta_k}(s_i, a_i)\right)^2, \text{for } k \in \{1, 2\}, \tag{16}$$

where $\theta_k(k=1,2)$ is the critic-network parameter.

To reduce the number of incorrect updates and improve the algorithm's stability, the TD3 algorithm does not update the actor-network until the Q value becomes stable. Therefore, the actor-network in the TD3 algorithm has a slightly lower update frequency than the critic-network. Through a deterministic policy gradient, the TD3 algorithm updates the actor-network parameters.

$$\nabla_\phi J(\pi_\phi) \approx \frac{1}{N}\sum_i \nabla_a Q_{\theta_1}(s_i, a)\Big|_{a=\pi_\phi(s_i)} \nabla_\phi \pi_\phi(s_i). \tag{17}$$

Table 1: Dimensions of each layer of the actor- and critic-networks.

Layer	Actor-network	Critic-network
Input layer	3 (state dimension)	5 (state dimension+action dimension)
Hidden layer 1	256	256
Hidden layer 2	256	256
Output layer	2 (action dimension)	1 (action value function dimension)

Table 2: TD3 algorithm hyperparameters.

Parameter	Value
Maximum permissible episodes	1600
Maximum permissible steps of each episode	572
Actor-network learning rate	10^{-4}
Critic-network learning rate	10^{-4}
Regularization constant	6×10^{-3}
Discounting factor γ	0.99
Sampling size N	256
Variance σ_t for the initial exploration noise ϵ	0.25
Variance fading factor ε	0.005
Variance $\tilde{\sigma}$ for random noise $\tilde{\epsilon}$	0.2
Upper limit e of the random noise $\tilde{\epsilon}$	0.25

To ensure the stability of training the neural network, the soft update strategy is adopted for the target network parameters:

$$\begin{cases} \theta'_k \longleftarrow \mu\theta_k + (1-\mu)\theta'_k \text{ for } k \in \{1,2\}, \\ \phi' \longleftarrow \mu\phi + (1-\mu)\phi', \end{cases} \quad (18)$$

where $0<\mu<1$ is a smoothing constant that represents the update speed. The pseudocode of the TD3 algorithm is given in Algorithm 1.

3.3. RL Model for Autopilot Design. To use the TD3 algorithm to solve the problem of two-loop autopilot parameter design, it is necessary to transform the autopilot design process into an RL problem and to design the various components of the RL model, i.e., state, action, and reward. At the same time, the algorithm's network structure and hyperparameters must be determined to ensure its effectiveness.

The design of the two-loop autopilot requires information regarding the flight state feature points in the full flight envelope. The Mach number and α can affect the change of aerodynamic characteristics, and the flight altitude and Mach number can affect the change of dynamic pressure, so the state vector can be designed as $s_t : [H, Ma, \alpha]^{\mathrm{T}}$, where Ma represents the Mach number and H represents the flight height. The objective of the design of a two-loop autopilot is that the control parameters satisfy the performance index requirements. Thus, the action vector is designed as $a_t : [k_A, k_g]^{\mathrm{T}}$.

The reward signal is the objective that must be maximized by the agent. A reasonable reward mechanism is crucial for training effectiveness. The frequency domain stability margin of the control system is selected as the performance index, and the following reward function is designed:

$$r = -\left(k_1(h_m - h_{m0})^2 + k_2(\gamma_m - \gamma_{m0})^2\right), \quad (19)$$

where h_{m0} is the expected magnitude margin, γ_{m0} represents the expected phase margin, and k_1 and k_2 are the influence weights of the magnitude and phase margins on the reward, respectively, meeting $0 \le k_1, k_2 \le 1$, and $k_1 + k_2 = 1$.

As shown in Figure 7, both the actor- and critic-networks of the TD3 algorithm employ an MLP neural network with four layers, and an activation function is connected behind the neurons of each layer besides the input layer. To prevent control input saturation, the tanh function activates the actor-network of the TD3 algorithm:

$$\mathrm{Tanh}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}. \quad (20)$$

The critic-network is activated by the ReLu function:

$$\mathrm{Re\,Lu}(x) = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{if } x \le 0. \end{cases} \quad (21)$$

The dimensions of each layer of the actor- and critic-networks are defined in Table 1.

The inputs of the actor- and critic-networks include state vectors. The state vectors must be normalized to eliminate the effect of input data dimensions on the neural network training procedure. In this paper, the state vectors were processed by (0,1) normalization:

$$\begin{cases} \bar{H} = \dfrac{H - H_{\min}}{H_{\max} - H_{\min}}, \\ \bar{Ma} = \dfrac{Ma - Ma_{\min}}{Ma_{\max} - Ma_{\min}}, \\ \bar{\alpha} = \dfrac{\alpha - \alpha_{\min}}{\alpha_{\max} - \alpha_{\min}}, \end{cases} \quad (22)$$

where $\bar{H}$, $\bar{Ma}$, and $\bar{\alpha}$ are the normalized network input values, in which $H \in [H_{\min}, H_{\max}]$, $Ma \in [Ma_{\min}, Ma_{\max}]$, and $\alpha \in [\alpha_{\min}, \alpha_{\max}]$.

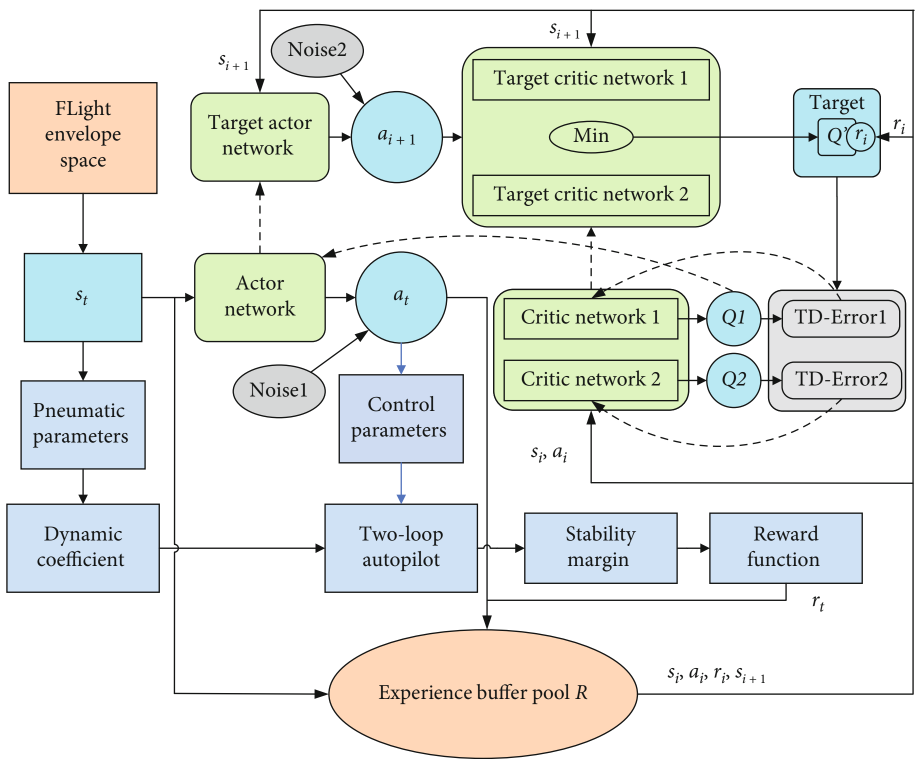


FIGURE 8: Two-loop autopilot offline training framework based on the TD3 algorithm.

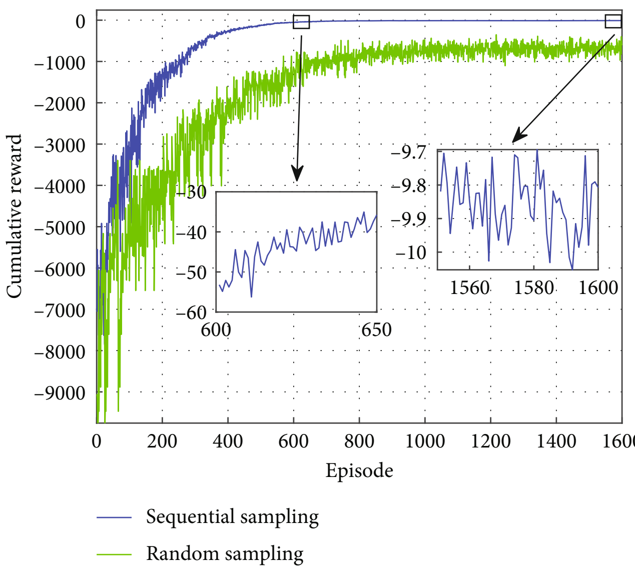


FIGURE 9: Episode cumulative reward change curve.

TABLE 3: Flight state at three different feature points.

	H (m)	Ma	α (°)
Feature point 1	600	0.5	6
Feature point 2	200	0.4	2
Feature point 3	1600	0.6	-2

Since the tanh activation function limits the output of the actor-network to the range (-1, 1), the output value of the actor-network needs to be denormalized to obtain the action vector value:

$$\begin{cases} k_A = \dfrac{\left[\bar{k}_A (k_{A\ \max} - k_{A\ \min}) + (k_{A\ \max} + k_{A\ \min})\right]}{2}, \\ k_g = \dfrac{\left[\bar{k}_g \left(k_{g\ \max} - k_{g\ \min}\right) + \left(k_{g\ \max} + k_{g\ \min}\right)\right]}{2}, \end{cases} \quad (23)$$

where $\bar{k}_A$ and $\bar{k}_g$ are the output values of the actor-network, $k_{A\ \max}, k_{A\ \min}$ are the maximum and minimum values of the control parameters, respectively, and $k_{g\ \max}, k_{g\ \min}$ are the maximum and minimum values of the control parameters, respectively.

To prevent the overfitting problem, both the actor- and critic-networks are trained by the Adam optimizer with L_2 regularization. The hyperparameter settings significantly affect the performance of the TD3 algorithm. Table 2 shows the hyperparameters suitable for the application scope of this paper.

3.4. Offline Training Framework. Before training, computational fluid dynamics (CFD) software was used to calculate the missile's pneumatic parameters throughout its entire flight envelope, and a library of pneumatic parameters was compiled.

Table 4: Design results of the TD3 algorithm at the feature points.

	Feature point 1	Feature point 2	Feature point 3
k_A	0.001590	0.002485	0.001822
k_g	0.10344	0.12109	0.108623
k_{dc}	3.6646	3.5354	3.5850
h_m	8.6694	8.5004	8.5971
γ_m	79.9758	79.7898	79.8691

Table 5: Pole assignment results at the feature points.

	Feature point 1	Feature point 2	Feature point 3
k_A	0.003824	0.00663	0.004038
k_g	0.08471	0.10398	0.08859
k_{dc}	2.0789	1.9312	2.1368
h_m	3.6045	3.1859	3.8796
γ_m	59.0424	55.6092	60.9565

In the offline training process, the flight speed V can be obtained using the following atmospheric model:

$$\begin{cases} \rho = \rho_0 e^{-(g/287.05T)H}, \\ c = \sqrt{1.4 \times 287.05T}, \end{cases} \tag{24}$$

where c represents acoustic velocity and $\rho_0 = 1.225\,\text{kg/m}^3$ is the atmospheric density in the sea level. The correlation between the temperature T and the troposphere H can be approximately expressed as

$$T = T_0 - 0.0065H, \tag{25}$$

where $T_0 = 288.15\,\text{K}$ is the reference temperature at sea level.

Firstly, the actor-network and the critic-network are initialized. Then, the experiment begins, the initial state s_t is selected and brought into the library of pneumatic parameters to obtain the corresponding pneumatic parameters, and the dynamic coefficient of the two-loop autopilot is computed. In this paper, thrust $P = 0$, so the dynamic coefficient is defined as follows: $a_\alpha = -m_z^\alpha qSL/J_z$, $a_\omega = -m_z^{\bar{\omega}} qSL/J_z$, $b_\alpha = c_y^\alpha qS/(mV)$, $b_\delta = c_y^\delta qS/(mV) + d_\alpha$, and $a_\delta = -m_z^\delta qSL/J_z + d_\omega$.

According to the structure of the two-loop autopilot, the open-loop transfer function $G(s)$ can be derived as follows [34]:

$$G(s) = \frac{M_2 s^2 + M_1 s + M_0}{s^2/\omega_m^2 + 2\mu_m s/\omega_m + 1}, \tag{26}$$

where $\omega_m = \sqrt{a_\alpha + a_\omega b_\alpha}$ and $\mu_m = (a_\omega + b_\alpha)/(2\omega_m)$. M_0, M_1, and M_2 are expressed as follows:

$$\begin{cases} M_0 = \dfrac{(a_\delta b_\alpha - a_\alpha b_\delta)(k_g + k_A V)}{(a_\alpha + a_\omega b_\alpha)}, \\ M_1 = \dfrac{(k_g a_\delta - k_A a_\omega b_\delta V) + (a_\delta b_\alpha - a_\alpha b_\delta) c k_A}{(a_\alpha + a_\omega b_\alpha)}, \\ M_2 = \dfrac{k_A (c a_\delta - b_\delta V)}{(a_\alpha + a_\omega b_\alpha)}. \end{cases} \tag{27}$$

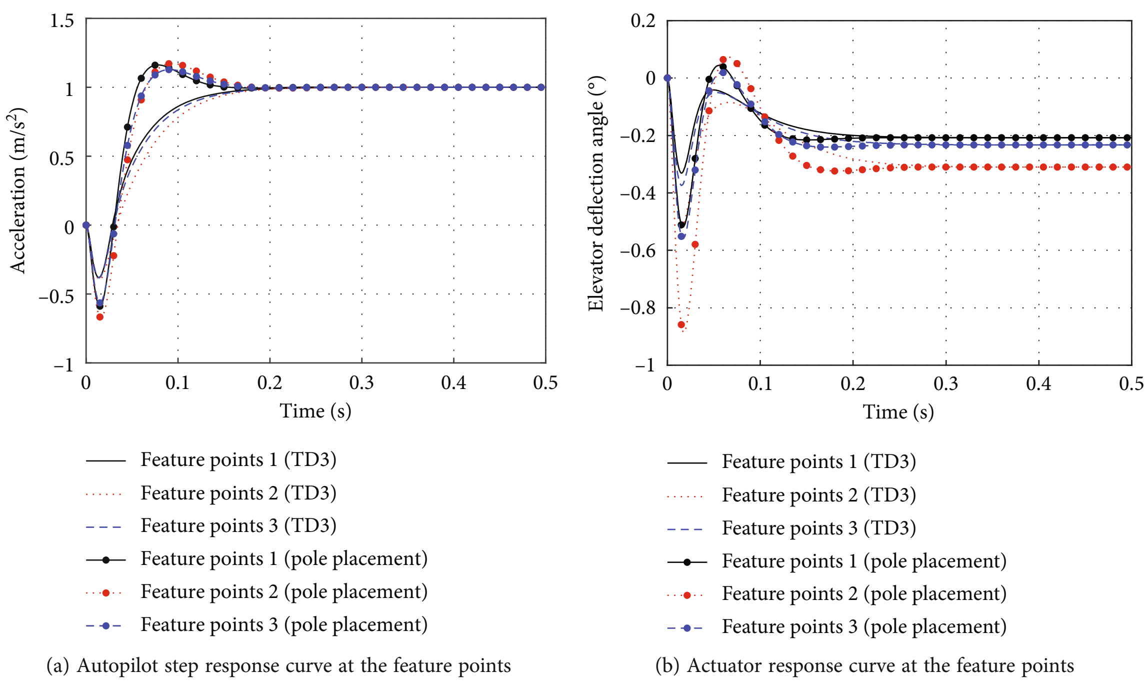
At the same time, the flight state s_t determined the control parameters k_A and k_g through the actor-network and exploration noise ϵ and output the action a_t. Then, the system stability margin is calculated according to Equation (6) and Equation (26), and the real-time reward r_t is calculated according to Equation (19).

It is worth noting that the offline design of the autopilot parameters in the whole flight envelope is not a traditional sequential decision-making problem but can be regarded as a problem to obtain the optimal value of control parameters in the flight envelope. Therefore, the acquisition of the next flight state s_{t+1} in the interaction with the environment needs to be designed. After several tests, it is determined that, compared to random sampling for training, the orderly arrangement of flight states and sequential sampling can improve the efficiency of the TD3 algorithm and the rate of convergence of the reward function. Therefore, in this study, all flight states in the library of pneumatic parameters are sequentially sorted. The ranking rule is fixed H first, and then, Ma and α are determined in turn. The next flight state s_{t+1} is determined according to the ranking rule.

The explored data (s_t, a_t, r_t, s_{t+1}) are imported into the experience buffer pool R. Then, the data (s_i, a_i, r_i, s_{i+1}) of the batch N are selected from the experience buffer pool R to train the network according to the TD3 algorithm. It should be noted that this sampling method does not affect the neural network's ability to generalize, as the network model can "remember" the order of the samples. This is because the TD3 algorithm employs the experience replay mechanism, and random selection of data from the experience pool during training breaks the correlation between sequences. The offline training framework for the two-loop autopilot parameter design based on the TD3 algorithm is illustrated in Figure 8.

4. Simulation Analysis

The sample space for the two-loop autopilot parameter training process was determined based on the application scene of the selected missile in this project. Subsequently, the TD3 algorithm is used to train the model offline, and the training results are analyzed. Finally, in conjunction with the impact angle constraint guidance problem, the neural network fitting model obtained after training was directly implemented in the guidance control loop for trajectory simulation to demonstrate its performance.



(a) Autopilot step response curve at the feature points

(b) Actuator response curve at the feature points

FIGURE 10: Step response curves at the feature points.

4.1. Autopilot Design Simulation Based on the TD3 Algorithm. First, the design state space of the autopilot parameters is determined: $S : [H, Ma, \alpha]^{\mathrm{T}}$, $H \in [0\,\mathrm{m}, 3000\,\mathrm{m}]$, $Ma \in [0.3, 1]$, and $\alpha \in [-12\circ, 12\circ]$. Then, the design action space of the autopilot parameters is set as follows: $a_t : [k_A, k_g]^{\mathrm{T}}$, $k_A \in [0, 0.01]$, and $k_g \in [0, 0.5]$.

Subsequently, the expected performance indices are selected, $h_{m0} = 8.5\mathrm{dB}$, $\gamma_{m0} = 80\circ$, $k_1 = 0.5$, and $k_2 = 0.5$, and the hyperparameters are set according to Table 2 for offline training. The change process of the episode cumulative reward during the offline training process is exhibited in Figure 9.

It can be seen that compared with random sampling, sequential sampling improved the strategy learning efficiency of the algorithm, and the stable value of cumulative reward was higher than random sampling. This is because random sampling in the training process leads to no correlation in state transfer, which makes the agent unable to determine how the state transfers through learning, so that there will be a large difference in each calculation of the Q value, and it is difficult to obtain a stable Q value. However, with sequential sampling, the state transition is determined, which greatly reduces the learning difficulty of the agent and enables faster convergence of cumulative reward. In the early stages of training, the cumulative reward value was small, with relatively large fluctuation due to the random selection of actions within the set range. Nevertheless, as the training progressed, the agent gradually chose better actions; as a result, the cumulative reward value gradually increased, and the fluctuation decreased. At 600 epochs of training, the cumulative reward value of sequential sampling was greater than -60 but still exhibited an upward trend. When the training reached 1550 epochs, the cumulative reward value was stable at approximately -9.9, with a small fluctuation range. It can be considered that the training process achieved the ideal results.

The design results of three characteristic points in the flight envelope are selected for analysis.

The feature points 1 and 2 in Table 3 are selected according to the predicted trajectory. To verify the generalization ability of the algorithm, feature point 3 with a negative AOA is selected for analysis. Table 4 lists the design results and stability margin of the TD3 algorithm for the autopilot control parameters at the feature points.

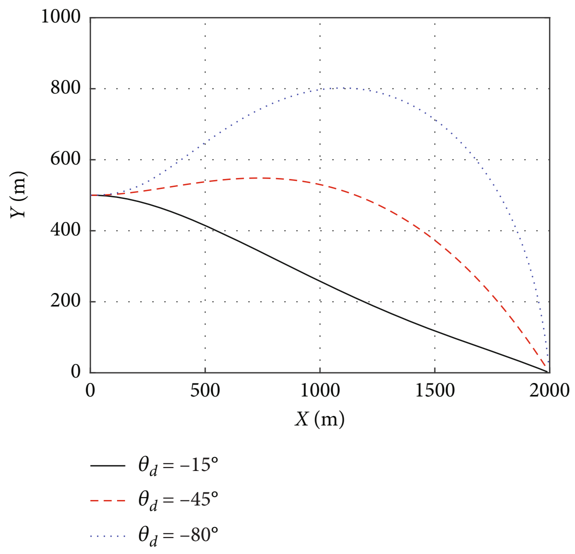
According to Table 4, the autopilot design results at the feature points met the design requirements of the amplitude and phase margins. Subsequently, the pole placement [27] method is used to determine the autopilot control parameters at the feature points, the damping coefficient μ is taken as 0.7, and the natural frequency ω is taken as 35 rad/s. The design results are listed in Table 5.

The autopilot actuator adopts the second-order actuator model:

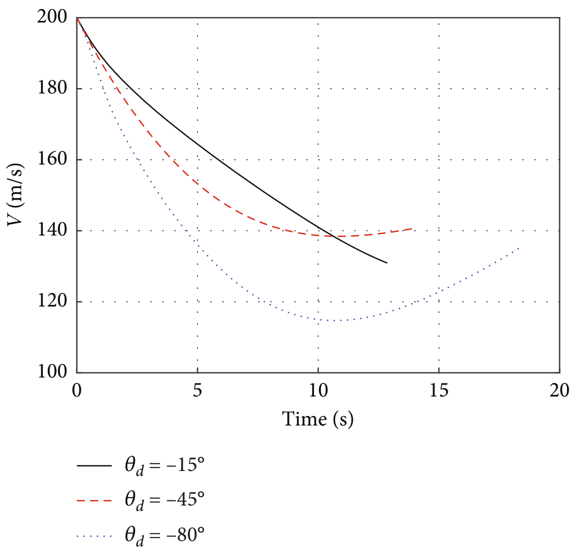
$$W_\delta(s) = \frac{\omega_{\mathrm{act}}^2}{s^2 + 2\mu_{\mathrm{act}}\omega_{\mathrm{act}} + \omega_{\mathrm{act}}^2}, \tag{28}$$

where $\omega_{\mathrm{act}} = 220\,\mathrm{rad/s}$ represents the actuator natural frequency and $\mu_{\mathrm{act}} = 0.65$ is the damping coefficient.

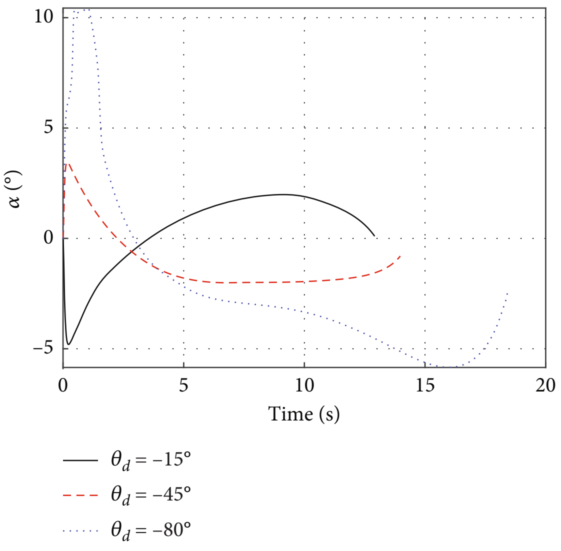
The design results of the control parameters at the feature points are incorporated into the two-loop autopilot model, and Figure 10 depicts the step response curves. According to the simulation results, the performance of the autopilot designed by the TD3 algorithm was superior to that obtained by the conventional pole placement method. Under the RL framework, the autopilot designed by the



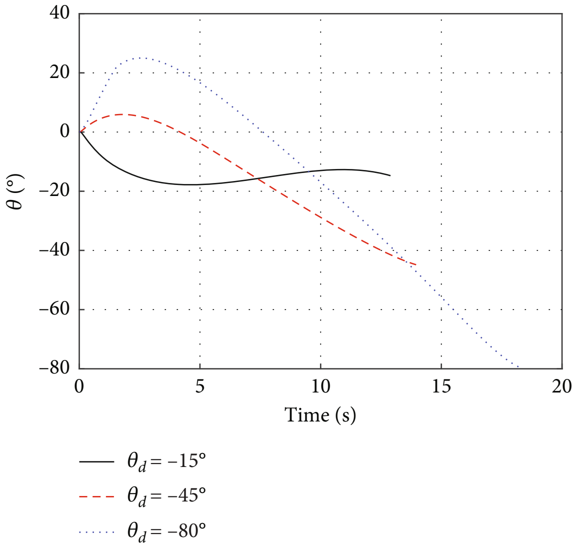
(a) Flight trajectory curve



(b) Speed change curve

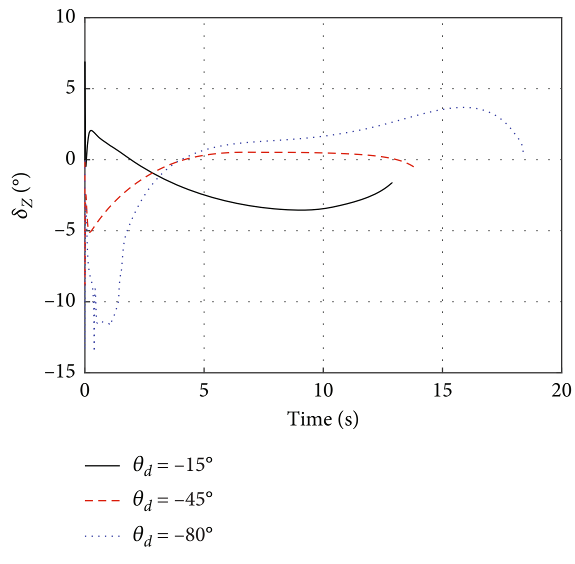


(c) AOA curve

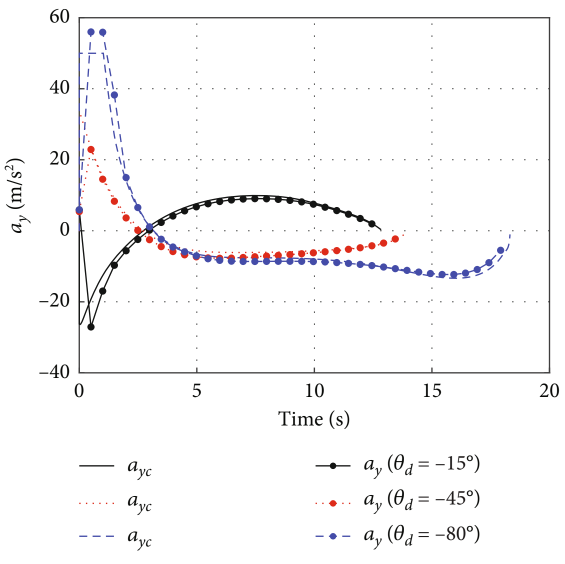


(d) Trajectory inclination curve

FIGURE 11: Continued.



(e) Elevator deflection variation curve



(f) Acceleration tracking curve

FIGURE 11: Control ballistic simulation results.

TD3 algorithm can obtain the control parameters that satisfy the ideal performance indices through autonomous learning, and the actuator's performance requirements are more reasonable.

4.2. Online Application of the Autopilot Parameter Fitting Model. After offline training of the TD3 algorithm, the MLP neural network fitting model with the flight state $[H, Ma, a]$ as input and the autopilot control parameters $[k_A, k_g]$ as the output in the full flight envelope is obtained by the actor-network. This model can be directly incorporated into the guidance control loop to automatically adjust the autopilot control parameters based on the current flight state. To further validate the effectiveness and adaptability of the proposed method, a flight experiment was designed with the guidance and impact angle constraint control problem in mind. The three-DOF motion equation in the plumb plane from Ref. [35] is chosen as the model for the theoretical flight simulation. To ensure the end impact angle, proportional guidance plus an offset term is required to meet the end impact angle constraint:

$$a_{yc} = \left(N_y \dot{q} + a_b\right) V, \tag{29}$$

where a_b represents the control item to be designed and N_y is the proportional coefficient, for which $N_y \geq 2$. The following relationship can be easily obtained:

$$\dot{\theta} = N_y \dot{q} + a_b. \tag{30}$$

Assuming that t_f is the end time of guidance, Equation (30) can be integrated on the interval $[t, t_f]$, and the following relationship can be obtained:

$$\theta\left(t_f\right) = \theta(t) + N\left(q\left(t_f\right) - q(t)\right) + \int_t^{t_f} a_b \mathrm{d}t. \tag{31}$$

Then, θ_d is set to a specified impact angle, and it must meet

$$\theta\left(t_f\right) = q\left(t_f\right) = \theta_d. \tag{32}$$

From Equations (31) and (32), the following relationship can be deduced:

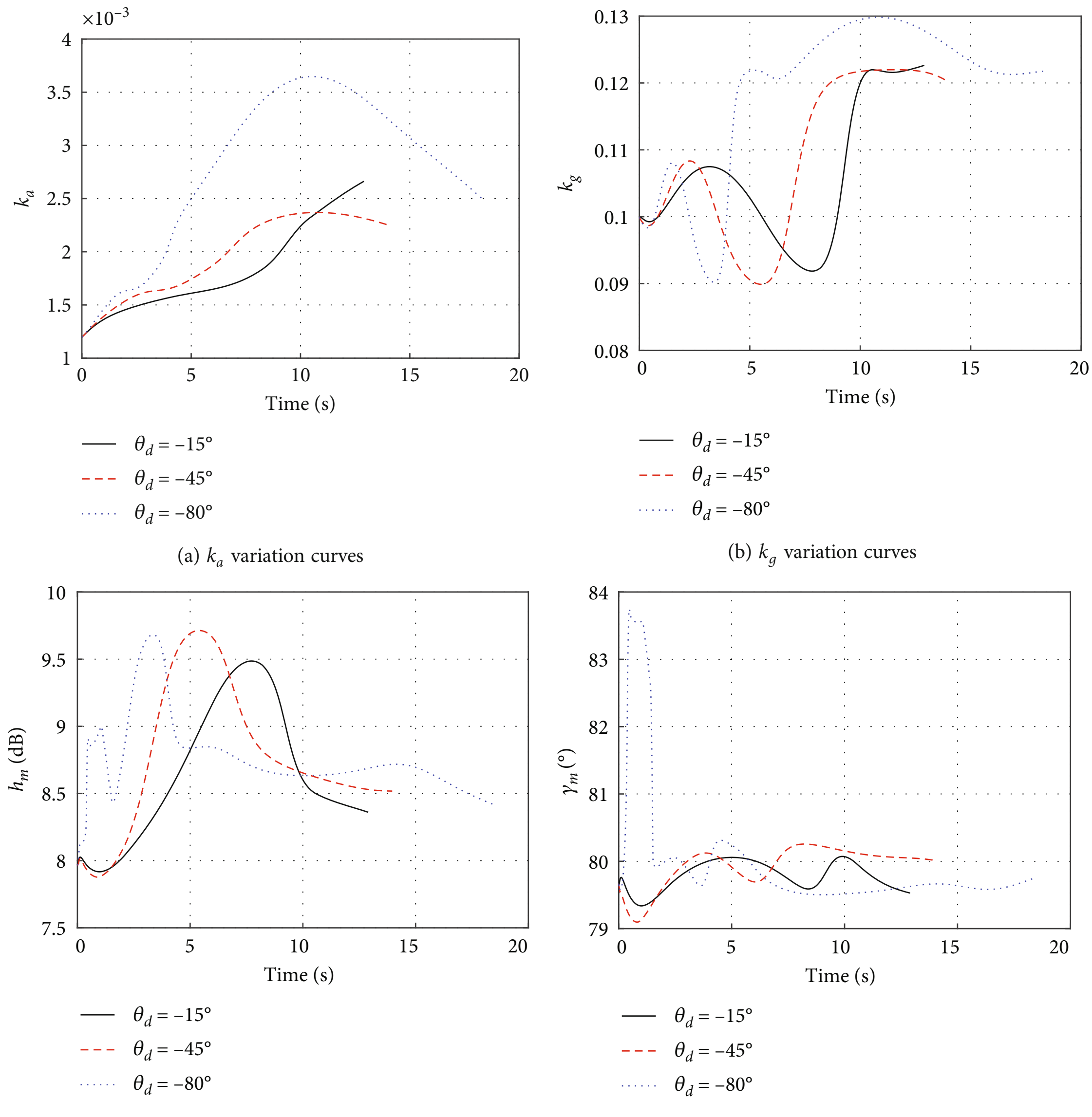
$$\int_{t_f}^{t} a_b \mathrm{d}t = (N-1)\theta_d + \theta(t) - Nq(t). \tag{33}$$

Consequently, $\eta(t) = \int_{t_f}^{t} a_b \mathrm{d}t$ and $\eta(t_f) = 0$ can be obtained. It is known that, as long as η converges to zero during the guidance process, the impact angle requirements can be met. Therefore, the following equation can be formulated [36]:

$$\dot{\eta} = -\frac{KV\cos\varphi}{R}\eta, \tag{34}$$

where K is an adjustable parameter, $K \geq 0$. Based on Equation (33), the following relationship can be obtained:

$$\dot{\eta} = \dot{\theta} - N_y \dot{q} = \frac{a_{yc}}{V_y} - N_y \dot{q}. \tag{35}$$



(a) k_a variation curves

(b) k_g variation curves

(c) Amplitude margin variation curve

(d) Phase margin variation curve

Figure 12: Variation curves of control parameters and stability margins.

The offset proportional guidance relationship with the impact angle constraint can be obtained by combining Equations (34) and (35):

$$a_{yc} = N_y V \dot{q} - \frac{KV^2 \cos \varphi}{R} \eta. \tag{36}$$

Based on the structure of the two-loop autopilot, the expression for the actuator command can be deduced as follows:

$$\delta_{zc} = -\left(\left(a_{yc} - a_y\right) K_A - k_g \omega_z\right). \tag{37}$$

Set the initial missile position to ($X_0 = 0$ m, $Y_0 = 500$ m), the initial pitch angle to $\vartheta_0 = 0°$, and the initial velocity to $V_0 = 200$ m/s. Set the target position to ($X_t = 2000$ m, $Y_t = 0$ m), $N_y = 3$, $K = 3$, and the acceleration command is constrained to [-50 m/s^2, 50 m/s^2]. The desired attack angles are set to -15°, -45°, and -80°, respectively. The simulation results are obtained and exhibited in Figure 11.

According to simulation results, the missile can consistently strike the target at the preset impact angle. Figure 11(f) illustrates that during the flight process, the control parameters obtained by the online adjustment of the real-time flight state via the neural network fitting model are able to effectively implement acceleration tracking. This indicates that the autopilot parameter fitting model trained by the TD3 algorithm is robust and capable of completing the flight task specified.

Figure 12 depicts the change processes of the control parameters and stability margins.

Figure 12 demonstrates that the autopilot parameter fitting model trained by the TD3 algorithm can self-adjust the autopilot control parameters online based on the real-time flight state, ensuring that the amplitude margin h_m and phase margin γ_m of the control system are within the expected range. In addition, it has strong generalizability and can be implemented in the guidance control loop to implement acceleration instruction tracking.

5. Conclusions

In this paper, the TD3 algorithm is used to design a two-loop autopilot for the full flight envelope that can be deployed directly to the guidance control loop, allowing online self-adjustment of the control parameters. First, a deep reinforcement learning model is constructed, with the flight state $[H, Ma, \alpha]^T$ taken as the state and the control parameters to be designed $[k_A, k_g]^T$ taken as the actions. According to the amplitude and phase margins in the frequency domain index of the control system, a reasonable reward mechanism is designed. The TD3 algorithm is then used offline to learn the control parameters of the complete flight envelope, and an MLP neural network fitting model is obtained. Finally, a verification study for the designed autopilot is conducted in combination with the impact angle constraint guidance problem. The results indicate that the TD3 algorithm-based autopilot can satisfy the performance index requirements. The fitting model can self-adjust the control parameters of the autopilot based on the real-time flight state, ensuring that the control system's stability margin is within the expected range and can accurately track acceleration. In addition, the fitting model has strong generalization capability and robustness, eliminating the robustness problem of the conventional overload autopilot, which is insufficiently robust to meet the performance requirements for the entire flight envelope. The method proposed in this paper for designing two-loop autopilots based on the TD3 algorithm is also applicable to the design of three-loop autopilots.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Acknowledgments

This work is supported in part by the National Key Research and Development Program under Grant No. 2020YFC1511705, the National Natural Science Foundation of China under Grant No. 61801032, and the Project of Construction and Support for High-Level Innovative Teams of Beijing Municipal Institutions (BPHR20220123).

References

[1] O. Zhang and S. X. Yan, "Study and analysis of several autopilots for air defense missile," *Aerospace Control*, vol. 38, no. 4, pp. 34–39, 2020.

[2] S. Snyder, "Autopilot design with learn-to-fly," in *AIAA Scitech 2020 Forum*, p. 0763, Orlando, FL, USA, 2020.

[3] D. F. Lin, J. F. Fan, Z. K. Qi, and Y. Mou, "Analysis and improvement of missile three-loop autopilots," *Journal of Systems Engineering and Electronics*, vol. 20, no. 4, pp. 844–851, 2009.

[4] H. Wang, T. Li, and D. Tang, "Analysis on stability of three-loop autopilot with an acceleration feedback augmentation loop," in *2021 40th Chinese Control Conference*, pp. 7538–7543, Shanghai, China, 2021.

[5] J. Lee, N. Cho, and Y. Kim, "Analysis of missile longitudinal autopilot based on the state-dependent Riccati equation method," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 10, pp. 2183–2196, 2019.

[6] L. D. Xing, W. C. Chen, and X. L. Yin, "Design three-loop autopilot for BTT missile using combined optimal/classical approach," *Journal of Astronautics*, vol. 29, no. 1, pp. 183–187, 2008.

[7] R. B. Sankar, B. Bandyopadhyay, and H. Arya, "Roll autopilot design of a tactical missile using higher order sliding mode technique," in *2016 Indian Control Conference (ICC)*, pp. 298–303, Hyderabad, India, 2016.

[8] X. L. Shao, J. T. Zhang, L. X. Xu, and W. D. Zhang, "Appointed-time guaranteed adaptive fault-tolerant attitude tracking for quadrotors with aperiodic data updating," *Aerospace Science and Technology*, vol. 132, article 107881, 2023.

[9] Y. T. Bi, Y. H. Wang, and Y. Yao, "Attitude control design of missiles with dual control based on model predictive control and active disturbance rejection control," *Journal of Astronautics*, vol. 36, no. 12, pp. 1373–1383, 2015.

[10] L. Y. Huang, R. Abbott, X. Qu, Y. H. Fan, and J. B. Wang, "Integrated guidance and control design for spiral maneuvering missile with multiple monstraints," *Journal of Astronautics*, vol. 42, no. 9, pp. 1108–1118, 2021.

[11] W. H. Ma, "Review and prospect of missile/launch vehicle guidance, navigation and control technologies," *Journal of Astronautics*, vol. 41, no. 7, pp. 860–867, 2020.

[12] H. Wang, D. F. Lin, and Z. K. Qi, "Design and analysis of missile three-loop autopilot with pseudo-angle of attack feedback," *Systems Engineering and Electronics*, vol. 34, no. 1, 2012.

[13] L. Graesser and W. L. Keng, *Foundations of Deep Reinforcement Learning: Theory and Practice in Python*, Addison-Wesley Professional, 2019.

[14] X. Zeng, Y. W. Zhu, L. P. Yang, and C. M. Zhang, "A guidance method for coplanar orbital interception based on reinforcement learning," *Journal of Systems Engineering and Electronics*, vol. 32, no. 4, pp. 927–938, 2021.

[15] D. Hong and S. Park, "Avoiding obstacles via missile real-time inference by reinforcement learning," *Applied Sciences*, vol. 12, no. 9, p. 4142, 2022.

[16] Y. Li, X. H. Qiu, X. D. Liu, and Q. L. Xia, "Deep reinforcement learning and its application in autonomous fitting optimization for attack areas of UCAVs," *Science in China Series F: Journal of Systems Engineering and Electronics*, vol. 31, no. 4, pp. 734–742, 2020.

[17] M. Piccinin and M. R. Lavagna, "Deep reinforcement learning approach for small bodies shape reconstruction enhancement," in *AIAA Scitech 2020 Forum*, p. 1909, Orlando, FL, USA, 2020.

[18] X. L. Shao, H. N. Si, and W. D. Zhang, "Low-frequency learning quantized control for MEMS gyroscopes accounting for full-state constraints," *Engineering Applications of Artificial Intelligence*, vol. 115, article 104724, 2022.

[19] J. F. Fan and X. Zhang, "Design two-loop autopilot based on reinforcement learning for miniature munition," *Tactical Missile Technology*, vol. 1, no. 4, pp. 48–54, 2019.

[20] Y. Zhen and M. R. Hao, "Research on intelligent PID control method based on deep reinforcement learning," *Tactical Missile Technology*, vol. 1, no. 5, pp. 37–43, 2019.

[21] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International Conference on Machine Learning*, pp. 387–395, Beijing, China, 2014.

[22] X. R. Liu and W. J. Yao, "Design of autopilot control parameters based on policy gradient," *Automation & Instrumentation*, vol. 1, no. 2, pp. 1–4, 2021.

[23] T. T. Zhao, H. Hachiya, V. Tangkarat, J. Morimoto, and M. Sugiyama, "Efficient sample reuse in policy gradients with parameter-based exploration," *Neural Computation*, vol. 25, no. 6, pp. 1512–1547, 2013.

[24] C. H. Lee and M. J. Tahk, "Autopilot design for unmanned combat aerial vehicles (UCAVs) via learning-based approach," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 476–481, Xiamen, China, 2019.

[25] D. M. Milz and G. Looye, "Design and evaluation of advanced intelligent flight controllers," in *AIAA Scitech 2020 Forum*, p. 1846, Orlando, FL, USA, 2020.

[26] J. H. Liu, J. Y. Shan, J. L. Rong, and X. Zheng, "Incremental reinforcement learning flight control with adaptive learning rate," *Journal of Astronautics*, vol. 43, no. 1, pp. 111–121, 2022.

[27] A. Candeli, G. D. Tommasi, D. G. Lui, A. Mele, S. Santini, and G. Tartaglione, "A deep deterministic policy gradient learning approach to missile autopilot design," *IEEE Access*, vol. 10, pp. 19685–19696, 2022.

[28] X. Huang, J. R. Liu, C. H. Jia, Z. L. Wang, and J. Zhang, "Deep deterministic policy gradient algorithm for UAV control," *Acta Aeronautica et Astronautica Sinica*, vol. 42, no. 11, pp. 404–414, 2021.

[29] K. Dally and E. J. V. Kampen, "Soft actor-critic deep reinforcement learning for fault tolerant flight control," in *AIAA SCITECH 2022 Forum*, p. 2078, San Diego, CA, USA, 2022.

[30] X. L. Shao, S. X. Li, J. T. Zhang, F. Zhang, W. D. Zhang, and Q. Z. Zhang, "GPS-free collaborative elliptical circumnavigation control for multiple non-holonomic vehicles," *IEEE Transactions on Intelligent Vehicles*, pp. 1–12, 2023.

[31] H. S. Shin, S. M. He, and A. Tsourdos, "A domain-knowledge-aided deep reinforcement learning approach for flight control design," 2019, https://arxiv.org/abs/1908.06884.

[32] Q. T. Wan, B. G. Lu, Y. X. Zhao, and Q. Q. Wen, "Autopilot parameter rapid tuning method based on deep reinforcement learning," *Systems Engineering and Electronics*, vol. 44, no. 10, pp. 3190–9199, 2022.

[33] Q. Cui, G. Kim, and Y. Weng, "Twin-delayed deep deterministic policy gradient for low-frequency oscillation damping control," *Energies*, vol. 14, no. 20, p. 6695, 2021.

[34] J. F. Fan, D. F. Lin, Z. K. Qi, and H. Zhang, "Design and analysis of two loop autopilot," *Systems Engineering and Electronic*, vol. 30, no. 12, pp. 2447–2450, 2008.

[35] X. F. Qian, R. X. Lin, and Y. N. Zhao, *Missile Flight Mechanics*, Beijing University of Technology Press, Beijing, China, 2013.

[36] Y. A. Zhang, G. X. Ma, and H. L. Wu, "A biased proportional navigation guidance law with large impact angle constraint and the time-to-go estimation," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 228, no. 10, pp. 1725–1734, 2014.