



Exoatmospheric Evasion Guidance Law with Total Energy Limit via Constrained Reinforcement Learning

Mengda Yan¹ · Rennong Yang¹ · Yu Zhao¹ · Longfei Yue² · Xiaoru Zhao¹

Received: 22 October 2023 / Revised: 2 January 2024 / Accepted: 14 February 2024
© The Author(s) 2024

Abstract

Due to the lack of aerodynamic forces, the available propulsion for exoatmospheric pursuit-evasion problem is strictly limited, which has not been thoroughly investigated. This paper focuses on the evasion guidance in an exoatmospheric environment with total energy limit. A Constrained Reinforcement Learning (CRL) method is proposed to solve the problem. Firstly, the acceleration commands of the evader are defined as cost and an Actor-Critic-Cost (AC2) network structure is established to predict the accumulated cost of a trajectory. The learning objective of the agent becomes to maximize cumulative rewards while satisfying the cost constraint. Secondly, a Maximum-Minimum Entropy Learning (M2EL) method is proposed to minimize the randomness of acceleration commands while preserving the agent's exploration capability. Our approaches address two challenges in the application of reinforcement learning: constraint specification and precise control. The well-trained agent is capable of generating accurate commands while satisfying the specified constraints. The simulation results indicate that the CRL and M2EL methods can effectively control the agent's energy consumption within the specified constraints. The robustness of the agent under information error is also validated.

Keywords Pursuit-evasion · Exoatmospheric vehicle · Guidance law · Reinforcement learning · Constraint · Energy consumption

1 Introduction

With the development of technologies such as propulsion systems and sensors, the intelligent game between exoatmospheric vehicles has become a worthwhile research topic. Due to the extremely high speeds and the lack of aerodynamic forces, maneuverability for these vehicles is limited by their finite energy, making exoatmospheric pursuit-evasion game a challenging problem.

Many researchers have studied pursuit-evasion problems using geometry analysis method, optimal control, and differential game methods. Extensive research has been conducted on the interception problem for pursuers, focusing on aspects such as convergence of line-of-sight angle, time constraints,

terminal angle constraints, and energy consumption. The Apollonius circle geometry analysis method is perhaps the most classic research approach in this field. However, this method significantly simplifies problems by assuming a constant velocity and neglecting realistic complex dynamics. In recent years, some researchers have explored Apollonius circles geometric analysis method to address pursuit-evasion problems with multiple participants [1]. However, the problem is still modeled as a simplified two-dimensional scene. Liu et al. [2] addressed the problem of a low-speed missile intercepting a hypersonic vehicle in the longitudinal plane. The guidance system is established by defining the LOS angular rate as the state variable. He et al. [3] and Reisner et al. [4] studied exoatmospheric the interception problem based on optimal control theory, taking into account finite time and interception angle constraints. Liang et al. [5, 6] investigated the guidance problem for interceptors against spacecraft with active defense, considering fuel cost, control saturation, chattering phenomenon and parameters selection. Wang et al. [7] proposed a cooperative augmented proportional guidance for proximity to uncooperative space targets.

✉ Yu Zhao
374602855@qq.com

¹ School of Air Traffic Control and Navigation, Air Force Engineering University, Xian 710051, China

² National Key Laboratory of Science and Technology on Vessel Integrated Power System, Naval University of Engineering, Wuhan 430033, China

For the evader, the primary objective is to avoid entering the pursuer's kill radius. However, the evader also has an ultimate goal of striking a specific target or flying to a particular area. Evading the pursuer becomes a necessary condition for accomplishing the final goal. Therefore, the evader often needs to compromise between the evasion task and the ultimate objective. Consequently, it becomes difficult for the evader to choose a suitable evasion strategy. Therefore, several literatures have investigated scenarios in which the evader carries a defender for active defense [8–10]. With assistance, the evasion task for the evader becomes significantly simpler. However, only a few studies have focused on investigating a single evader's strategies. Shaferman et al. [11] proposed a Near-optimal evasion strategies for an evader, exploiting the inherent time delay associated with the pursuer's estimate of the evader's acceleration to maximize the miss distance. Carr Ryan et al. [12] investigated a scenario where proportional navigation is nearly optimal for the pursuer and solves the differential game solution for the evader in this scenario. Fonod Robert et al. [13] proposed a multiple model adaptive evasion strategies and analyzed several specific limiting cases in which the attacking missile uses proportional navigation, augmented proportional navigation, or an optimal guidance law.

The references mentioned above mostly employ optimal control or differential game approaches which had two shortcomings. Firstly, simplifying the problem to two dimensions and linearize complex dynamic equations [14] due to the difficulty of solving nonlinear models. However, exoatmospheric missile engagements are characterized by high speeds, short durations, and small kill radii. Achieving high simulation accuracy is crucial, and any model simplification may result in significant errors in terminal miss distance. Secondly, previous research had focused on minimizing energy consumption as one of the evader's goals, but no literature considered the total energy limit as a constraint for the evader. The two approaches are different. The former aims to minimize energy consumption while ensuring a successful evasion, whereas the latter tries to find the optimal evasion strategy within the energy constraints. In exoatmospheric environment, only direct force can be utilized as there is no aerodynamic force. Consequently, the evader must consider the constraint of energy consumption. Therefore, it is necessary to develop a real-time decision-making method that can directly solve the evasion guidance law with nonlinear dynamic equations and strict energy constraint.

Reinforcement learning (RL) is an iterative decision-making method that allows an agent to interact with its environment in real-time, which can solve the two problems above effectively. The decision-making entity, referred to as an agent, observes the current state of the environment and makes decisions based on these observations. The agent then evaluates and enhances its strategy by receiving feedback

in the form of reward values from the environment. However, applying reinforcement learning methods to address the problem studied in this paper encounters two significant challenges: firstly, how to effectively handle constraints, and secondly, how to achieve precise control. Traditional reinforcement learning methods typically optimize an agent's strategy through a reward function. However, when energy consumption is incorporated into the reward function, ensuring compliance with total energy constraints that the agent must satisfy becomes challenging. In certain related research, some researchers refer to the constraints an agent must satisfy as cost. Subsequently, the agent's objective becomes twofold: to maximize the expected reward value while accumulating sufficient cost to meet the constraint. This approach is known as constrained reinforcement learning (CRL) [15–17]. On one hand, the evader's acceleration command was defined as cost. The accumulated cost of a trajectory must satisfy the total energy constraints. On the other hand, reinforcement learning is commonly employed with stochastic policies to enhance the exploration capability of an agent. However, in missile guidance problems, excessive randomness can result in unnecessary energy consumption. Therefore, a maximum-minimum entropy learning method is proposed to reduce the randomness of acceleration commands without compromising the exploration capability of the agent. In this game, evader's maneuvering capability is less than that of the pursuer. Simulation results have shown that constrained reinforcement learning can effectively address such constrained decision-making problems. The evader agent is capable of completing the evasion task while satisfying the energy consumption constraint.

The main contributions of this paper are as follows:

1. Constrained reinforcement learning method was proposed to solve the problem of exoatmospheric evasion guidance with total energy constraints.
2. To minimize the randomness of acceleration commands while preserving the agent's exploration capability, we introduce the Maximum-Minimum Entropy Learning method, which is integrated into the agent's learning objective as a constraint term.
3. The effectiveness and robustness of the proposed method have been validated on a randomly generated dataset.

2 Related Work

In recent years, the remarkable advancements of RL in various domains have prompted researchers to explore its application in computational missile guidance. The researchers first demonstrated that reinforcement learning guidance laws

have certain advantages compared to proportional navigation methods. He et al. [18] and Hong Daseon et al. [19] studied the comparison between the reinforcement learning guidance law and the traditional proportional guidance law, and the experiment verified that the guidance law based on reinforcement learning can be applied to the missile guidance law. Gaudet et al. [20] showed that the guidance law of reinforcement learning performs better than the proportional guidance method and the enhanced proportional guidance method when considering the noise and time delay of the sensors and actuators.

Reinforcement learning method is also capable of addressing constraint problems in missile guidance, like interception angle constraints. Gong et al. [21] presented an all-aspect attack guidance law for agile missiles based on deep reinforcement learning (DRL), which can effectively cope with the aerodynamic uncertainty and strong nonlinearity in the high angle-of-attack flight phase. Li et al. [22] proposed an assisted deep reinforcement learning (ARL) algorithm to optimize the neural network-based missile guidance controller for head-on interception. Based on the relative velocity, distance, and angle, ARL can control the missile to intercept the maneuvering target and achieve large terminal intercept angle.

Subsequently, reinforcement learning has been further applied to missile active defense [23, 24], spacecraft pursuit-evasion games [25–28], as well as exoatmospheric missile guidance [29–31]. Shalumov et al. [24] tried to find an optimal launch time of the defender and an optimal target guidance law before and after launch using DRL. A policy suggesting at each decision time the bang-bang target maneuver and whether or not to launch the defender was obtained and analyzed via simulations. Simulations showed the ability of the reinforcement learning based method to obtain a close to optimal level of performance in terms of the suggested cost function. Yang et al. [25] proposed a closed-loop pursuit approach using reinforcement learning algorithms to solve and update the pursuit trajectory for the incomplete-information impulsive pursuit-evasion missions. Brandansio et al. [26] focused towards enhanced autonomy on-board spacecrafts for on-orbit servicing activities using deep reinforcement learning method. Zhao et al. [27] investigated the problem of impulsive orbital pursuit-evasion games using Multi-Agent Deep Deterministic Policy Gradient approach. A closed-loop pursuit approach using reinforcement learning algorithms was proposed to solve and update the pursuit trajectory for the incomplete-information impulsive pursuit-evasion missions. Zhang et al. [28] studied an one-to-one orbital pursuit-evasion problem. A near-optimal guidance law using deep learning is proposed to intercept the evader inside the capture zone. For the games that start outside the barrier, a learning algorithm for the capture zone embedding strategy is presented based on deep reinforcement learning

to help the game state cross the barrier surfaces. In [30], reinforcement learning algorithm was applied to the mid-course penetration of extra-atmospheric ballistic missiles. In [31], reinforcement learning combined with meta-learning is applied to the guidance law of extra-atmospheric interceptor. The reinforcement learning algorithm outputs four propulsion instructions for steering thrusters. The results show that reinforcement learning guidance law is superior to the traditional ZEM guidance law in interception rate and energy consumption.

3 Problem Formulation

This paper aims to analyze the terminal guidance phase of a 3D exoatmospheric pursuit-evasion problem. The 3D relative kinematics relationship is shown in Fig. 1. The centroid of the evader is represented by the red dot, while the centroid of the pursuer is represented by the blue dot. The inertial coordinate system is denoted as XYZ , while the virtual inertial coordinate system, obtained by shifting the inertial coordinate system to the centroid of the evader, is denoted as $X'Y'Z'$. The missile body coordinate system is denoted as $X_m Y_m Z_m$, where the X_m axis aligns with the missile axis, the Y_m axis is perpendicular to the missile in an upward direction, and the Z_m axis follows the right-hand rule. The line-of-sight (LOS) coordinate system is denoted as $X_l Y_l Z_l$, with the X_l axis coinciding with the line of sight, the Y_l axis pointing upward in the vertical plane of the X_l axis, and the Z_l axis following the right-hand rule. θ and φ represent the pitch angle and yaw angle, respectively, of the moving coordinate system in relation to the inertial system.

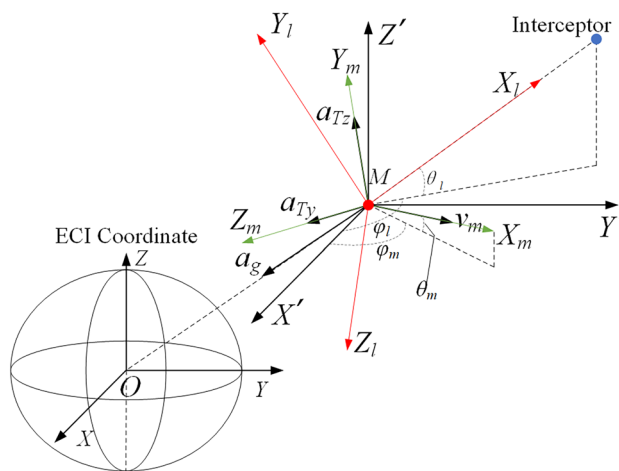


Fig. 1 3D relative kinematics relationship

3.1 Kinematic and Dynamic Formulas in the Inertial Coordinate System

The kinematic formulas for the evader and pursuer in the inertial frame are:

$$\begin{cases} \dot{\mathbf{r}}_e = \mathbf{v}_e \\ \dot{\mathbf{v}}_e = \mathbf{a}_e \end{cases}, \quad (1)$$

$$\begin{cases} \dot{\mathbf{r}}_p = \mathbf{v}_p \\ \dot{\mathbf{v}}_p = \mathbf{a}_p \end{cases}, \quad (2)$$

where $\mathbf{r} = [x, y, z]$, $\mathbf{v} = [v_x, v_y, v_z]$, $\mathbf{a} = [a_x, a_y, a_z]$. The subscripts e and p represent the evader and pursuer, respectively.

Exoatmospheric hypersonic vehicles usually rely on the divert control system located on the side of the vehicle body and near the center of gravity to provide direct forces during the orbit flight phase [32]. As depicted in Fig. 1, the evader's thrust accelerations are applied along the Y_m and Z_m axes, and the thrust acceleration of the evader in the missile coordinate system can be expressed as $\mathbf{a}_t^m = [0, a_{ty}, a_{tz}]^m$. Suppose there is a maximum limit on the thrust acceleration of the evader, denoted as a_{t_max} , that is $|\mathbf{a}_{ty}| \leq a_{t_max}$ and $|\mathbf{a}_{tz}| \leq a_{t_max}$. The thrust acceleration of the evader in the inertial frame is denoted as $\mathbf{a}_t^i = \mathbf{C}_{mi} \mathbf{a}_t^m$, where \mathbf{C}_{mi} is the transformation matrix from the missile body coordinate system to the inertial coordinate system.

The exoatmospheric dynamics equation of the evader is:

$$\mathbf{a}_e^i = \mathbf{a}_g^i + \mathbf{a}_t^i, \quad (3)$$

where \mathbf{a}_g represents the gravitational acceleration of the Earth in the inertial frame, and its expression is:

$$\mathbf{a}_g = \begin{bmatrix} \frac{GM \cdot x}{\sqrt{(x^2 + y^2 + z^2)^3}} \\ \frac{GM \cdot y}{\sqrt{(x^2 + y^2 + z^2)^3}} \\ \frac{GM \cdot z}{\sqrt{(x^2 + y^2 + z^2)^3}} \end{bmatrix}, \quad (4)$$

where GM is the gravitational constant, with a value of 3.9753×10^{14} .

Assuming that the pursuer uses proportional guidance, the formula is [33]:

$$\mathbf{a}_c^i = N \frac{z_{em}}{t_{go}^2}, \quad (5)$$

where N is a guidance gain, z_{em} is the Zero Effort Miss (ZEM) [33] that is perpendicular to the LOS and t_{go} represents the remaining flight time, which is approximated as:

$$t_{go} = -\frac{R}{\dot{R}}. \quad (6)$$

The exoatmospheric dynamics equation of the pursuer is:

$$\mathbf{a}_p^i = \mathbf{a}_g^i + \mathbf{a}_c^i. \quad (7)$$

In this paper, it is assumed that the pursuer's initial ZEM is 0 [34], and the initial velocity vector of the pursuer is determined by solving the two-body Lambert equation.

3.2 Kinematic Formulas in the LOS Coordinate System

Denote the rotation matrices that transform the ECI coordinate system to the LOS coordinate system around the Z , Y , and X axes as \mathbf{C}_Z , \mathbf{C}_Y , and \mathbf{C}_X , respectively. The matrix representing the angular velocities of rotation is denoted as Ω^i , which can be expressed as:

$$\Omega^i = \mathbf{C}_X \mathbf{C}_Y \mathbf{C}_Z \begin{bmatrix} 0 \\ 0 \\ \dot{\phi}_l \end{bmatrix} + \mathbf{C}_X \mathbf{C}_Y \begin{bmatrix} 0 \\ \dot{\theta}_l \\ 0 \end{bmatrix} = \begin{bmatrix} -\dot{\phi}_l \sin \theta_l \\ \dot{\phi}_l \cos \theta_l \\ -\dot{\theta}_l \end{bmatrix}. \quad (8)$$

The kinematic formula in the LOS coordinate system is:

$$\dot{\mathbf{V}}_r = \begin{bmatrix} \delta \mathbf{R} \\ \delta t \end{bmatrix}^l + [\Omega^i]^l \times \mathbf{R}. \quad (9)$$

Substituting Eq. 8 into Eq. 9 yields:

$$\dot{\mathbf{V}}_r = \begin{bmatrix} \dot{R} \\ -R\dot{\theta}_l \\ -R\dot{\phi}_l \cos \theta_l \end{bmatrix}. \quad (10)$$

Assuming the pursuer utilizes a direct collision interception method with a kill radius of 0.5 m, successful interception by the pursuer is defined when $R \leq 0.5$. Conversely, successful evasion by the evader is defined when $\dot{R} \geq 0$.

The decision time interval and simulation interval is crucial for the convergence of the algorithm and the accuracy of simulation. Considering the algorithm search space and practical engineering conditions, we set the decision time interval to 0.1 s, which means that an acceleration command is output every 0.1 s. For ballistic simulation, a smaller simulation interval can improve simulation accuracy, but it also increases the training time cost. To accelerate the training process, we set the initial simulation step size to be the same

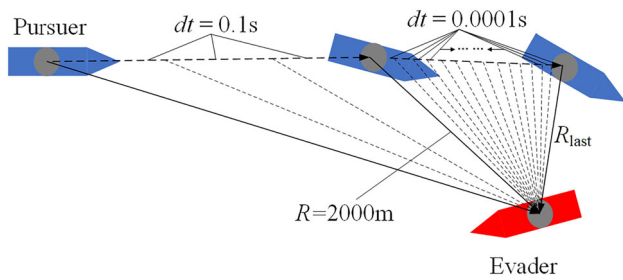


Fig. 2 Diagram illustrating simulation accuracy

as the decision step size, which is 0.1 s. Meanwhile, to accurately calculate the miss distance at the end, the simulation interval is adjusted to 0.0001 s when the relative distance is less than 2000 m, as depicted in Fig. 2. If the simulation interval is directly set to 0.0001 s, the training time cost will increase by more than 40 times, which would be unacceptable. Additionally, the motion differential equation is updated using the 4th order Runge–Kutta method.

4 Method

To solve the constrained evasion problem in RL framework, we firstly build a Constrained Markov Decision Process (CMDP) and then present the Constrained Proximal Policy Optimization (CPPO) based evasion guidance law.

4.1 CMDP

CMDP differs from the classical MDP by incorporating cost as feedback during the decision-making process. A CMDP can be represented by $(\mathbf{S}, \mathbf{A}, \mathbf{P}, \mathbf{R}, \mathbf{C})$, where \mathbf{S} is the state space, \mathbf{A} is the action space, \mathbf{R} denote the reward function and \mathbf{C} denote the cost function, respectively, and \mathbf{P} is the transition probability function with $p(s'|s, \mathbf{a})$ denoting the transition probability from state \mathbf{s} to state \mathbf{s}' given action \mathbf{a} . A stochastic policy $\pi : S \rightarrow A$ is a mapping from states to probabilities of selecting each possible action. The goal is to find the optimal policy π^* that provides the highest expected sum of rewards:

$$\pi^* = \arg \max_{\pi} E_{\pi} \left\{ \sum_t^T \gamma^t r_{t+1} | s_0 = s \right\}, \tag{11}$$

where $\gamma \in [0, 1]$ is the discount factor. The CMDP problem can be generalized into:

$$\begin{aligned} \max_{\pi} E_{\pi} \left\{ \sum_t^T \gamma^t r_{t+1} | s_0 = s, a_0 = a \right\} \\ \text{s.t. } \sum_t^T c_{t+1} \leq C, \end{aligned} \tag{12}$$

where C is the cost limit, and the $\sum_t^T c_{t+1}$ is the total cost during a trajectory.

This paper assumes that the evasion agent can only observe partial information about the engagement state. Therefore, in the subsequent description, the state \mathbf{s} is replaced by the observed variable \mathbf{o} . The observation space, action space, cost function, and reward function of CMDP are set as follows.

4.1.1 Environment States and Agent Observations

The environment states can be described in both the inertial coordinate system and the LOS coordinate system. In the inertial coordinate system, the environmental state includes the positional coordinates, velocity vectors, and acceleration vectors of both the pursuer and the evader. In the LOS coordinate system, the environmental state can be described by relative distance and its rate, line-of-sight angles and its rate, as well as elevation angle and its rate.

In this paper, the environment information is only partially observable for the evader. Assuming that the evader is equipped with an infrared sensor, it can directly measure the line-of-sight angles in the missile body coordinate system, as depicted in Fig. 3. Although these two angles can be transformed into the inertial coordinate system or the LOS coordinate system, we choose to directly use these two angles as observations of the agent to minimize errors. However, the information provided by these two angles is insufficient. Therefore, it is assumed that the agent can estimate the relative distance to the pursuer using a filtering algorithm, resulting in a total of three observations, denoted as $[\varphi_m, \theta_m, R]$. Furthermore, the evader does not require knowledge of the rate of line-of-sight angle and the rate of relative distance.

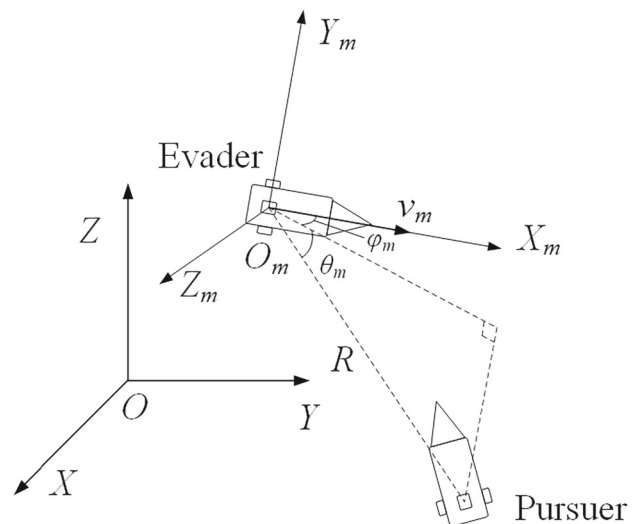


Fig. 3 Diagram of observations for the evader

4.1.2 Action Space

The action output of the actor network is the thrust acceleration in the y -axis and z -axis directions defined in the missile body coordinate system. Specifically, it can be represented as $\mathbf{a} = [a_y, a_z]^m$, where the superscript m denotes the missile body coordinate system.

4.1.3 Cost Function

The role of the cost function is to calculate the cumulative constraints. For each interaction during training, the single-step constraint is determined by summing the absolute values of the current thrust acceleration, which can be represented as $c = \tau(|a_y| + |a_z|)$, where τ is a hyperparameter and is set to 0.1 in this paper.

4.1.4 Reward Function

The reward function plays a crucial role in the reinforcement learning environment as it guides the agent's learning process. Reward functions can be categorized into shaping rewards and end rewards. Shaping rewards are used to guide the agent's exploration during training, while end rewards indicate whether the task has been successfully accomplished or not. In our case, since the acceleration constraint is already accounted for by the cost function, we only employ a sparse reward approach, specifically using an end reward function. If the evader is hit, the task fails and a penalty is given; if the evader successfully avoids being hit, the task succeeds, and a positive reward is given. The expression is as follows:

$$\begin{cases} r = -\left(1 - \frac{|R|}{R_{kill}}\right) & \text{if } R \leq R_{kill} \\ r = 0.2 \log(2R) & \text{if } R > R_{kill} \text{ and } \dot{R} \geq 0 \end{cases} \quad (13)$$

The reward function curve is shown in Fig. 4. The relationship between terminal miss distance and reward is a continuous curve, and the range of reward values is approximately from -1 to 1. When the miss distance is equal to the kill radius, the reward is 0. If the miss distance is smaller than the kill radius, a penalty value is assigned, and the magnitude of the penalty increases as the miss distance decreases. As the miss distance approaches 100 m, the reward value gradually increases, but the rate of change becomes smaller, which informs the agent that the current miss distance is sufficiently large.

4.2 Traditional PPO Algorithms

The PPO [35] algorithm, which is a state-of-art reinforcement learning algorithm, is used as the basic algorithm in

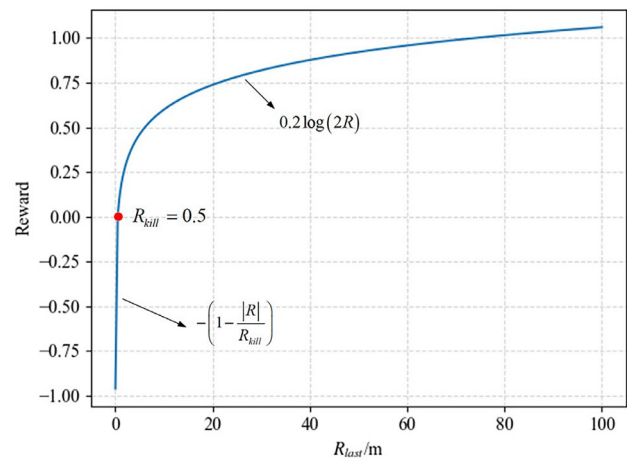


Fig. 4 Reward function curve

this paper. PPO is an on-policy algorithm that operates within the Actor-Critic (AC) framework. The traditional framework of continuous PPO algorithm has two main components: the actor network and the critic network. The actor network is responsible for outputting the distribution of actions, while the critic network evaluates the state value function.

The PPO algorithm utilizes importance sampling method to calculate the ratio of the old policy to the new policy to measure the quality of the new policy, which as shown in formula (14),

$$p_n(\theta) = \frac{\pi_\theta(\mathbf{a}_n | \mathbf{o}_n)}{\pi_{\theta_{old}}(\mathbf{a}_n | \mathbf{o}_n)}. \quad (14)$$

The samples obtained through importance sampling can be reused multiple times, and the number of sample usages, denoted as n_{reuse} in this paper, is a crucial hyperparameter in the PPO algorithm. Following the proposal of the PPO algorithm, several versions have been developed, with the clip version being the most commonly used. The clip function is responsible for controlling the gap between the old policy and the new policy. The objective of the PPO algorithm is to maximize the expected value of the advantage function through importance sampling. The objective function is shown in formula (15).

$$\begin{aligned} J(\theta) \\ = E_{p(\tau)}[\min[p_n(\theta), \text{clip}(p_n(\theta), 1 - \varepsilon, 1 + \varepsilon)] A_{\mathbf{w}}^\pi(\mathbf{o}_n, \mathbf{a}_n)]. \end{aligned} \quad (15)$$

The advantage function is defined as the difference between the state-action value function and the state value function, and it encourages actions with values greater than the average value. There are various forms to express the advantage function, and the Generalized Advantage Estimation (GAE) is an effective method that provides a good

balance between estimation bias and variance. Its expression is given by formula (16).

$$A_w^\pi(\mathbf{o}_t, \mathbf{a}_t)_{GAE} = \sum_{t=0}^{T-n-1} (\gamma\lambda)^n [r_{t+n} + \gamma V(\mathbf{o}_{t+n+1}) - V(\mathbf{o}_{t+n})]. \quad (16)$$

The objective of the critic network is to predict the value of a given state. The loss function of the critic network is shown in formula (17).

$$L(\mathbf{w}) = \sum_{i=1}^M \left(V_w^\pi(\mathbf{o}_n^i) - \left[\sum_{t=n}^T \gamma^{t-n} r(\mathbf{o}_t^i, \mathbf{a}_t^i) \right] \right)^2. \quad (17)$$

4.3 Maximum-Minimum Entropy Learning method

The formulas (15) to (17) represent the traditional PPO algorithm, where the policy is typically described using a Gaussian distribution. The acceleration command is generated by sampling from the policy distribution, resulting in a certain level of randomness. In general application scenarios like competitive games, random policies are favored because they enhance exploration capabilities, reduce the likelihood of getting stuck in local optima, and exhibit better robustness. For instance, the authors of the Soft Actor-Critic (SAC)[36] algorithm propose incorporating the maximization of entropy learning into the objective of the agent. However, in missile guidance problems, where there are strict limitations on the sum of acceleration commands, excessive randomness can lead to energy waste and hinder the learning of the optimal policy under total energy constraints. Therefore, this paper introduces the maximum-minimum entropy learning method to address this issue. The idea is to encourage exploration during the early stages of training and gradually reduce the randomness of the policy in later stages, thus reducing the randomness of acceleration commands without compromising the exploration capability of the agent.

The classic definition of entropy is $-\sum p(x) \log p(x)$, and Ref. [36] defined the entropy as $E_{\mathbf{a}_n \sim \pi} [-\log(\pi(\mathbf{a}_n | \mathbf{s}_n))]$. However, the former has excessively large gradients in the direction of entropy reduction, while the latter has excessively small gradients in the direction of entropy reduction. Therefore, this paper constructs a logistic entropy function as shown in formula (18).

$$H(p(\mathbf{a}_n | \mathbf{o}_n)) = E_{\mathbf{a}_n \sim \pi} \left[-\frac{k_n e^{-\eta p(\mathbf{a}_n | \mathbf{o}_n)}}{(1 + (k_n/k_0 - 1))} \right], \quad (18)$$

where $k_n = 100$, $k_0 = 2$, $\eta = 0.2$, and $p(\mathbf{a}_n | \mathbf{o}_n)$ is the value of probability density. The images of the three entropy functions are shown in Fig. 5. It can be observed that neither $p(x) \log p(x)$

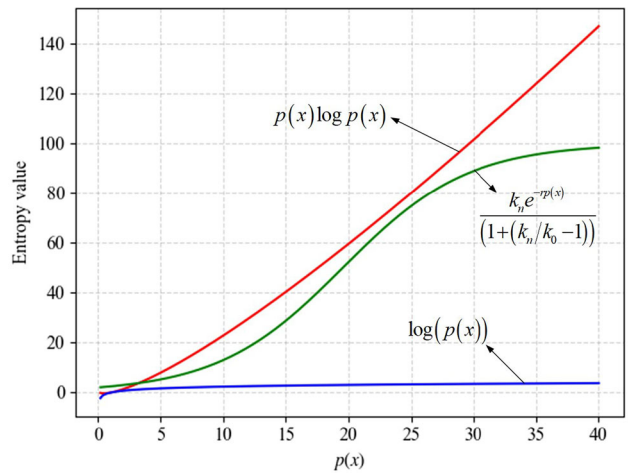


Fig. 5 Comparison of three entropy functions

(x) nor $\log(p(x))$ is suitable when the learning objective is entropy reduction. $p(x) \log p(x)$ has a derivative that is too large in the entropy reduction direction, while $\log(p(x))$ has a derivative that is too small. Function $p(x) \log p(x)$ exhibits an excessive derivative in the direction of entropy reduction, whereas derivative of $\log(p(x))$ is insufficient. In contrast, the logistic entropy function exhibits a derivative in the direction of entropy reduction that initially progresses slowly but gradually accelerates, eventually converging. This is in line with the requirements of the training objective.

The logistic entropy function is incorporated as a constraint term in the objective function of the policy network, as shown in formula (19).

$$J(\theta) = -E_{p(\tau)} \left[\min[p_n(\theta), \text{clip}(p_n(\theta), 1 - \epsilon, 1 + \epsilon)] A_w^\pi(\mathbf{o}_n, \mathbf{a}_n) \right] - \alpha H(p(\mathbf{a}_n | \mathbf{o}_n)), \quad (19)$$

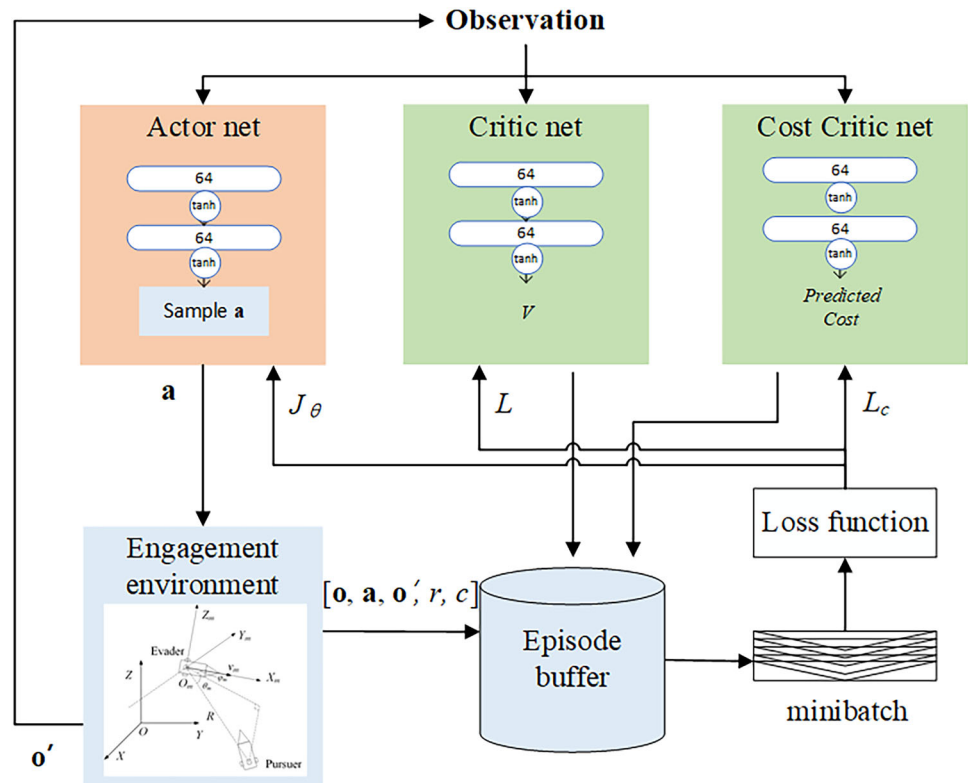
where α is an adaptive parameter, and its value affects whether the policy distribution increases or decreases in entropy. The objective function of α is given by formula (20).

$$J(\alpha) = \begin{cases} E_{(\mathbf{o}_t, \mathbf{a}_t) \sim \rho_z} \left[\alpha \sum_t \gamma^t r(\mathbf{o}_t, \mathbf{a}_t) \right], & \text{if } E(\gamma^t r(\mathbf{o}_t, \mathbf{a}_t)) \leq 0 \\ E_{(\mathbf{o}_t, \mathbf{a}_t) \sim \rho_z} \left[\alpha (H_0 - \log(p(\mathbf{a}_n | \mathbf{o}_n))) \right], & \text{otherwise} \end{cases}, \quad (20)$$

where H_0 is an objective entropy, and its value is set to 3 in this paper. The explanation of the influence of exploration randomness and H_0 value can be found in Appendix.

4.4 CPPO Evasion Guidance Law

After controlling the randomness of the policy, we can proceed to solve the constrained optimization problem described

Fig. 6 CPPO algorithm framework


in formula (12). The challenge lies in incorporating the cost and cost limit within the reinforcement learning algorithm framework. In the CMDP problem, it is crucial to calculate the cumulative cost following a specific state. Therefore, we propose an Actor-Critic-Cost (AC2) structure by incorporating a cost network into the traditional AC framework to predict the cumulative cost value. The AC2 framework is illustrated in Fig. 6.

In general, a constrained optimization problem can be solved by the Lagrange Multiplier method. By introducing the Lagrange Multiplier β , formula (12) becomes an unconstrained optimization problem:

$$\begin{aligned} \max_{\pi} \mathcal{L}(\pi, \beta) &\doteq f(\pi) - \beta g(\pi) \\ f(\pi) &= \mathbb{E}_{(\mathbf{o}_t, \mathbf{a}_t) \sim \rho_{\pi}} \left[\sum_t \gamma^t r(\mathbf{o}_t, \mathbf{a}_t) \right] \\ g(\pi) &= \mathbb{E}_{(\mathbf{o}_t, \mathbf{a}_t) \sim \rho_{\pi}} \left[\sum_t c(\mathbf{o}_t, \mathbf{a}_t) \right] - C. \end{aligned} \quad (21)$$

As a result, the complete objective function of the actor network in this paper can be described by formula (22).

$$\begin{aligned} J(\theta) &= \mathbb{E}_{p(\tau)} \left[\min[p_k(\theta), \text{clip}(p_k(\theta), 1 - \varepsilon, 1 + \varepsilon)] \right. \\ &\quad \left. \times (A_{\mathbf{w}}^{\pi}(\mathbf{o}_k, \mathbf{a}_k) + \beta Q^c(\mathbf{o}_k, \mathbf{a}_k)) \right] - \alpha H(p(\mathbf{a}_n | \mathbf{o}_n)), \end{aligned} \quad (22)$$

where $Q^c(\mathbf{o}_k, \mathbf{u}_k)$ is obtained from the cost critic network. Similar to the critic network, the loss function of the cost critic network is described by formula (23).

$$L^c(\mathbf{w}) = \sum_{i=1}^M \left(C_{\mathbf{w}}^{\pi}(\mathbf{o}_k^i) - \left[\sum_{t=k}^T \gamma^{t-k} c(\mathbf{o}_t^i, \mathbf{a}_t^i) \right] \right)^2. \quad (23)$$

It is worth noting that the β in formula (21) is continuously updated based on the relationship between the cost limit and the average cumulative cost of the current episode. The objective function of β is shown in formula (24).

$$J(\beta) = \beta \left(C - \max \left(\sum c(\mathbf{o}_t, \mathbf{a}_t) \right) \right) j, \quad (24)$$

where j is an adaptive parameter that adjusts the update speed of β under different conditions. The updating process of β is outlined in Algorithm 1.

Algorithm 1 Adaptive update algorithm for β

Initialize: Cost limit C ; the maximum cumulative cost of the previous episode $cost_{pre} = 0$
 Define: The objective function for updating β : $J(\beta) = \beta(C - \max(\sum c(o, a)))j$;
 The difference between the maximum cumulative cost and C :
 $cost_{dev} = \max(\sum c(o, a)) - C$;
 The amount of change in cumulative cost: $dcost = \max(\sum c(o, a)) - cost_{pre}$

```

1:   IF  $cost_{dev} > 0$ 
2:     IF  $dcost \geq 0$ 
3:        $j = j_1$ 
4:     ELSE
5:        $j = j_2$ 
6:     ELSE
7:       IF  $dcost > 0$ 
8:          $j = j_3$ 
9:       ELSE
10:         $j = j_4$ 
11:      Update penalty parameter  $\beta$  by using objective function  $J(\beta)$ 
12:       $cost_{pre} = \max(\sum c(s, a))$ 

```

In Algorithm 1, the values of j_1, j_2, j_3 and j_4 need to be tuned, which are set to 1, - 0.1, - 1 and 5 in this paper. The updates of parameters α and β are not synchronized with the updates of network parameters. Network parameters are updated multiple times within each episode based on the value of parameter n_{reuse} , while parameters α and β are updated at a slower frequency, only once at each episode. The complete process of the proposed CPPO algorithm is shown in Algorithm 2.

Algorithm 2 CPPO algorithm

Input: Initial policy parameters θ_0 , critic network parameters θ_{v0} , cost network parameters θ_{c0} , clipping threshold ϵ , reuse time n_{reuse} , cost limit C , entropy penalty parameter initial value α_0 , cost penalty parameter initial value β_0 ,

```

1:   FOR episode  $i = 1, 2, 3, \dots$  DO
2:     Initial replay buffer  $M$ 
3:     FOR each trajectory DO
4:       Sample action from policy  $\pi(\theta_t)$ 
5:       Execute action  $a_t$  and store the tuple  $s_t, a_t, r_t, c_t, s_{t+1}$  to  $M$ 
6:     END FOR
7:     FOR  $n = 1, 2, 3, \dots, n_{reuse}$  DO
8:       Sample minibatch in  $M$  randomly
9:       Update policy parameters  $\theta_t$  by using formula (22)
10:      Update critic network parameters  $\theta_{vt}$  by using formula (17)
11:      Update cost network parameters  $\theta_{ct}$  by using formula (23)
12:    END FOR
13:    Update entropy penalty parameter  $\alpha$  by using formula (20)
14:    Update cost penalty parameter  $\beta$  by using Algorithm 1
15:  END FOR

```

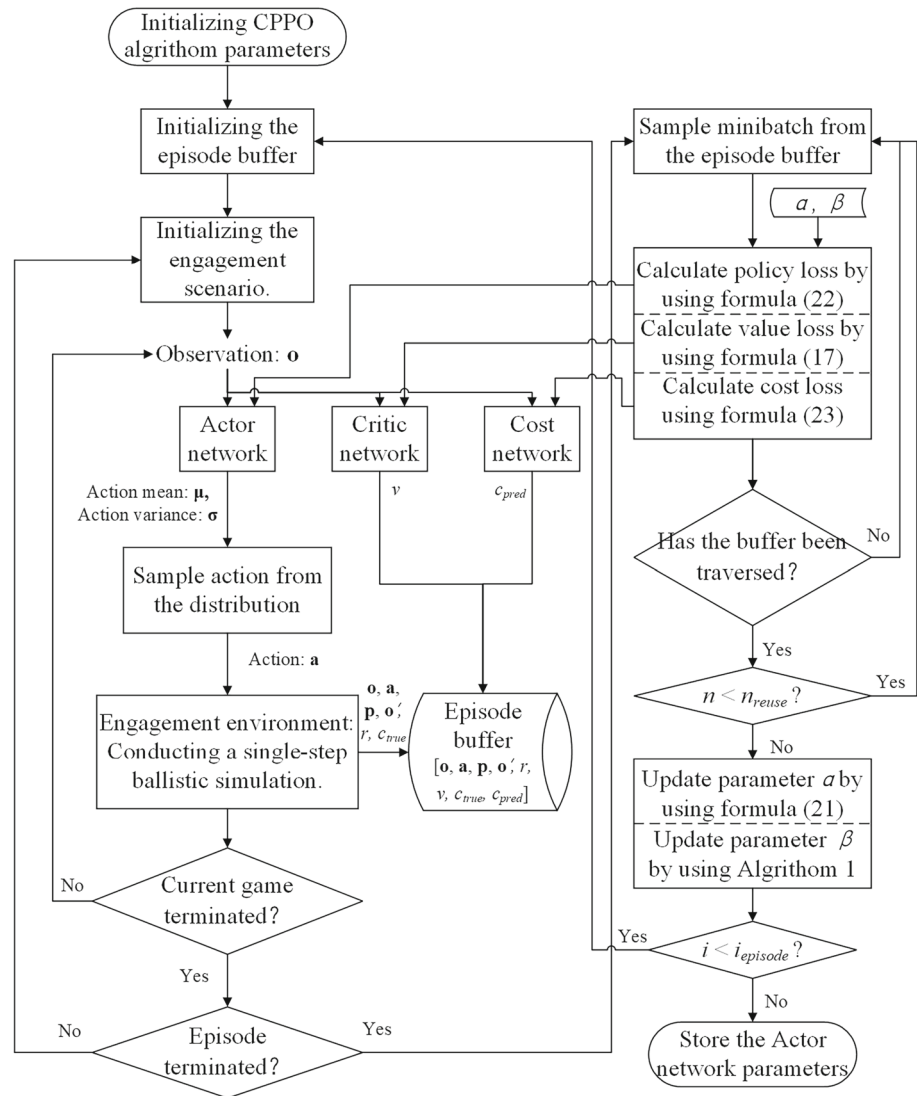
The complete flowchart of the proposed method is shown in Fig. 7. The left side of the flowchart represents the interaction phase, while the right side represents the training phase. The final output is the well-trained actor network parameters, which represent the mapping relationship between observations and actions. These parameters can be used for testing and application purposes.

5 Experiment and Results

5.1 Training Results and Parameter Sensitivity Experiments

In this section of the experiment, we compared the classical PPO algorithm with the CPPO algorithm proposed in this paper. The parameter settings and environment setup for the PPO algorithm are consistent with CPPO. The difference between them is that CPPO describes energy cost using a cost function, while PPO describes energy cost using a reward function, as shown in formula (25).

$$r = -k \left(e^{(a_y/a_{max})^2} + e^{(a_z/a_{max})^2} \right), \tag{25}$$

Fig. 7 Research methodology flowchart

where k is a parameter that needs to be tuned, and its value will affect the weight of energy cost in the learning objective of the PPO agent.

Before each training episode, the situation needs to be initialized. With the evader as the center, the position vectors $[\theta, \varphi, R]$ of the pursuer are randomly initialized in the visual coordinate system. Assuming that the maximum thrust acceleration of the pursuer is greater than that of the evader, and both sides have a decision frequency of 10 Hz. The initial values of the situation parameters are shown in Table 1.

The hyperparameters of the CPPO algorithm have been fine-tuned, and their corresponding values are presented in Table 2.

To demonstrate the superiority of the proposed CPPO algorithm in this paper, we conducted a comparison between multiple groups of PPO and CPPO algorithms with different energy consumption constraints. In the CPPO algorithm,

we set total energy constraints $C = 5, 10, \text{ and } 15$, respectively. As for the PPO algorithm, we set $k = 0.00025, 0.0005, \text{ and } 0.001$, respectively. The training results are shown in Fig. 8. Figure 8a–d illustrate the reward curve, maximum energy consumption curve, policy distribution standard deviation curve, and penalty parameter β curve, respectively. In Fig. 8a, both the CPPO and PPO algorithms exhibit robust convergence. The reward curve of PPO appears relatively stable, whereas the CPPO reward curve shows some fluctuations. This can be attributed to the fact that the primary learning objective of the PPO agent is to maximize the reward function, whereas CPPO may encounter a reduction in rewards due to the influence of constraints throughout the training process. The characteristics of the CPPO algorithm are demonstrated in Fig. 8b. The energy consumption of the CPPO agent can converge with high accuracy to the specified constraint C (often slightly below the C value). In contrast, it is difficult for the PPO algorithm to precisely control

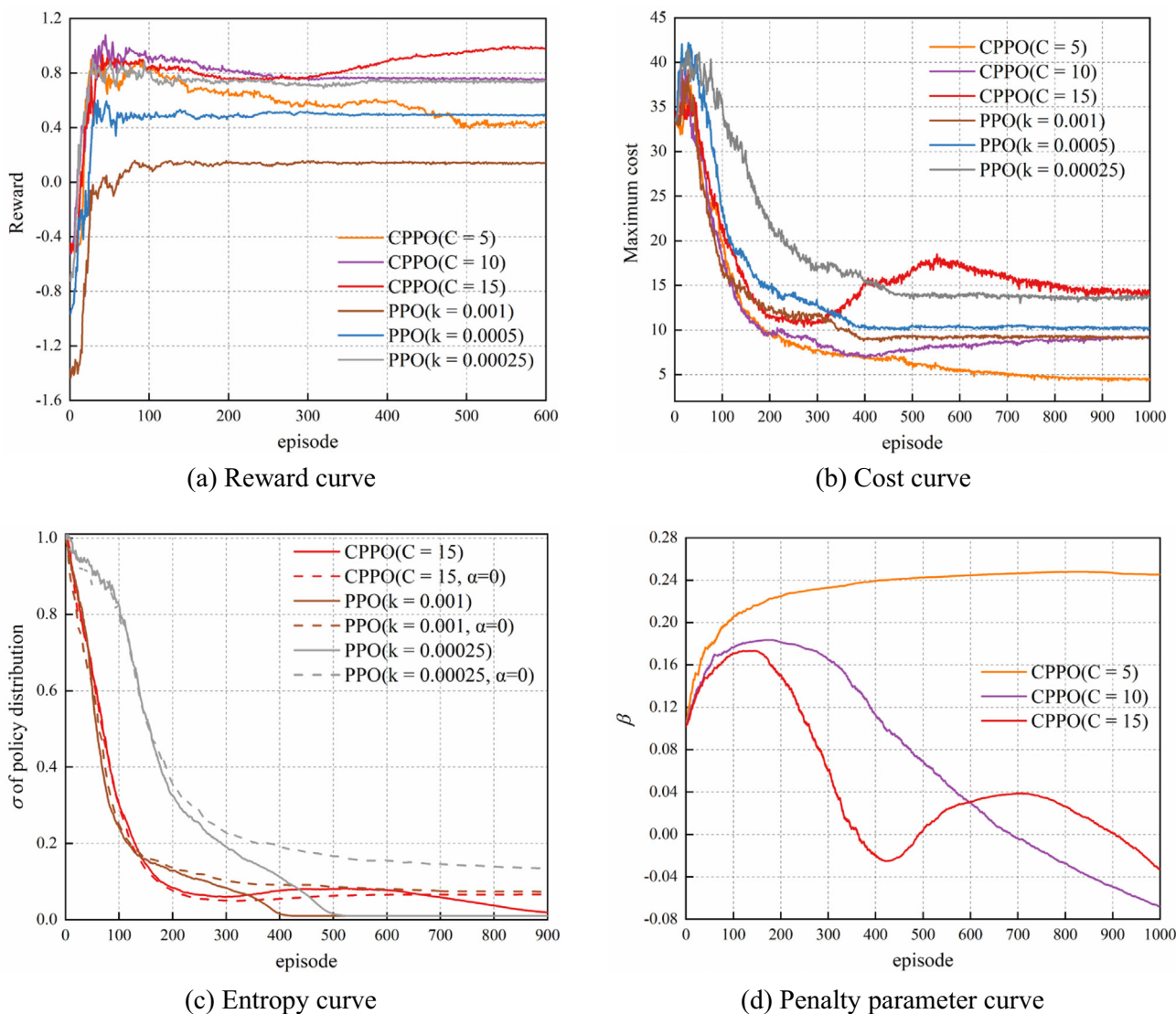


Fig. 8 Training results

the energy consumption of the agent through the parameter k . For instance, when $k = 0.0005$ and 0.001 , despite doubling the parameter values, there is no significant difference in the energy consumption between them. Figure 8 (c) illustrates the effect of the Maximum-Minimum Entropy Learning method. The method enables the standard deviation of the policy distribution to converge to 0.01, which aligns with the target value set in formula (20). However, when α is set to 0, indicating the absence of the Maximum-Minimum Entropy Learning method, the agent’s policy distribution will always maintain a certain level of randomness. Figure 8d illustrates the trend of the penalty parameter β in the CPPO algorithm. When β shows an increasing trend, it indicates that the agent has not yet satisfied the energy consumption constraint, requiring a further increase in the weight of the cost loss compared to the policy loss. Conversely, when the

agent meets the energy consumption constraint, the value of β starts to slowly decrease. Therefore, β is a crucial parameter in the CPPO algorithm.

5.2 Effectiveness Experiment

In this section, we introduced the classic step maneuver as a comparative method and defined two sets of different maneuver parameters. The expressions for two maneuvers are given by formulas (26) and (27) respectively. They are referred to as “step maneuver 1” and “step maneuver 2” in the subsequent text.

$$a_y = a_z = \begin{cases} 0, & t \leq 20s \\ a_{max}, & t > 20s \end{cases}, \tag{26}$$

Table 1 Initial values of the situation parameters

Parameters	Symbol	Value or scale
LOS azimuth angle	φ_m	$[-\frac{\pi}{3}, \frac{\pi}{3}]$
LOS pitch angle	θ_m	$[-\frac{2\pi}{9}, -\frac{\pi}{3}]$
Relative distance	R	[200,000, 220,000]m
Maximum thrust acceleration of the evader	a_{e_max}	20 m/s ²
Maximum thrust acceleration of the pursuer	a_{p_max}	30 m/s ²

Table 2 Hyperparameters of the CPPO algorithm

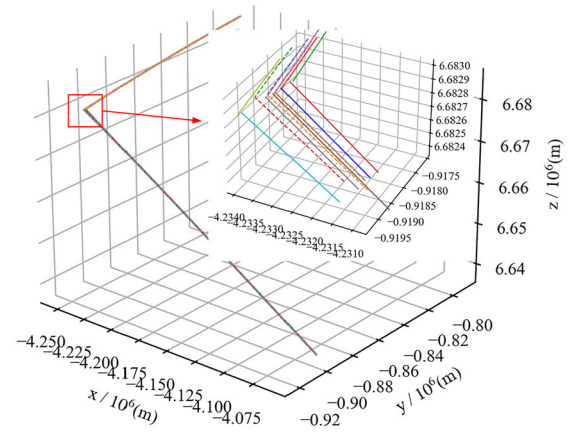
Parameters	Value
<i>Clip</i>	0.2
<i>n_{reuse}</i>	5
<i>Memory size</i>	16,384
<i>Minibatch</i>	512
<i>lr</i>	0.0003
<i>H₀</i>	3
<i>α₀</i>	0.01
<i>lr_α</i>	0.0002
<i>β₀</i>	0.06
<i>lr_β</i>	0.00005

$$a_y = a_z = \begin{cases} 0, & t \leq 15s \\ 0.7a_{max}, & t > 15s \end{cases} \quad (27)$$

5.2.1 Experiment in Typical Situations

In this section of the experiment, we compare the performance of CPPO, PPO, and two types of step maneuvering in specific situations, analyzing the trajectories, acceleration commands and LOS angle rates. The situational parameters of this scene are $\varphi_m = 0.873$, $\theta_m = -0.261$ and $R=220,000$. Figure 9 displays the trajectories of evaders and pursuers using different methods. In this particular scenario, the two step maneuver methods are intercepted, while RL methods successfully evade by performing slight maneuvers.

Figure 10 illustrates the acceleration and LOS angular rate curves of the evader in this scenario. Figure 10a and b shows the accelerations in the y-axis and z-axis directions of the missile body coordinate system, respectively. It is apparent that the results acquired through RL learning also demonstrate an approximation of step maneuvering. In the initial



- CPPO(C=5) evader trajectory
- CPPO(C=5) pursuer trajectory
- CPPO(C=10) evader trajectory
- CPPO(C=10) pursuer trajectory
- CPPO(C=15) evader trajectory
- CPPO(C=15) pursuer trajectory
- PPO(k=0.001) evader trajectory
- PPO(k=0.001) pursuer trajectory
- PPO(k=0.0005) evader trajectory
- PPO(k=0.0005) pursuer trajectory
- PPO(k=0.00025) evader trajectory
- PPO(k=0.00025) pursuer trajectory
- - - Step1 evader trajectory
- - - Step1 pursuer trajectory
- - - Step2 evader trajectory
- - - Step2 pursuer trajectory
- - - Original evader trajectory
- - - Original pursuer trajectory

Fig. 9 Trajectories of evader and pursuer

stage of the game, no maneuvering is performed, and at a specific time in the final stage, the maneuvering starts and gradually increases until reaching maximum acceleration. The distinction among RL methods with different parameters lies in the timing of maneuver initiation and the rate of acceleration variation. Figure 10c and d depicts the LOS angular rate curves. Due to the assumption that the ZEM for pursuers is 0 at the beginning of the game, the LOS angular rate remains constant at 0 until the evader initiates maneuvering. In this state, the pursuer can successfully intercept the evader without any maneuvers. The main difference between RL methods and traditional methods lies in the rate of change of the yaw angle. In RL methods, there is a noticeable variation in the yaw angular rate at the end stage of game, whereas the yaw angular velocity remains almost constant at 0 in traditional maneuvering methods.

The energy consumption and miss distance of various methods are shown in Table 3, indicating a clear advantage of RL methods over traditional maneuvering methods. RL methods achieve a larger miss distance with lower energy consumption. RL methods with larger constraint parameter values have higher energy consumption but also result in safer miss distances. In particular, the CPPO ($C = 5$) method achieves successful evasion with only 441.41 units of energy consumption. On the contrary, step maneuvers 1 and 2 have energy consumptions of 2000 and 2800, respectively, but are eventually intercepted by the pursuer.

Table 3 Results on the typical scenario

Method	Energy cost/(m/s ²)	Miss distance/m
CPPO(C = 5)	441.41	2.163
CPPO(C = 10)	1412.91	22.42
CPPO(C = 15)	2485.94	65.26
PPO(k = 0.001)	1388.29	16.91
PPO(k = 0.0005)	1964.11	30.63
PPO(k = 0.00025)	2625.41	77.77
Step maneuver 1	2000.00	0.12
Step maneuver 2	2800.00	0.14

5.2.2 Experiment on Test Dataset

To comprehensively evaluate the agent’s performance, we generated a test dataset of 100 scenarios randomly, as shown in Fig. 11.

The trained agent and the traditional step maneuvering method were subjected to 100 experiments on the test dataset, and the results are shown in Table 4. According to Table 4, both CPPO and PPO algorithms have a success rate of 100%, while traditional maneuvering methods, despite consuming a large amount of energy, cannot achieve successful escape in all situations. For instance, CPPO (C = 15) and step maneuver 2 have similar energy consumption. However, the

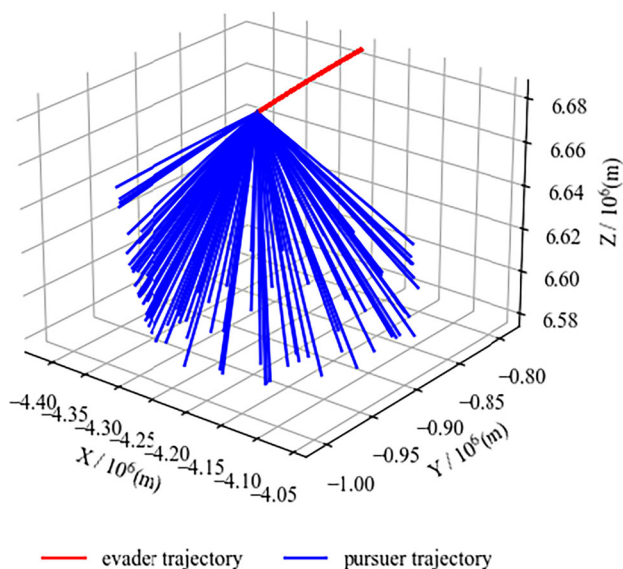
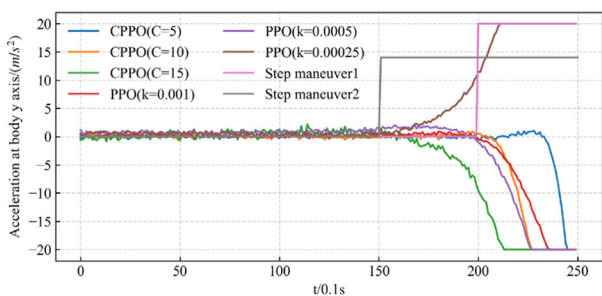
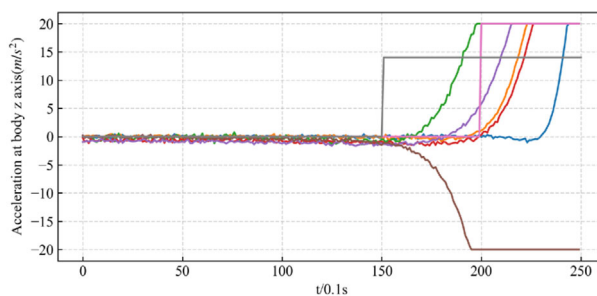


Fig. 11 Test dataset

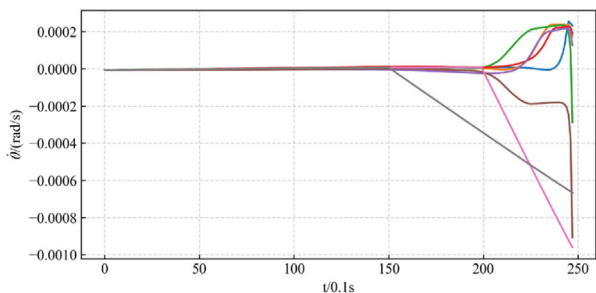
former demonstrates approximately twice the success rate of escape and terminal miss distance compared to the latter. This indicates that intelligent methods can significantly enhance maneuvering efficiency compared to traditional approaches. Both PPO and CPPO are capable of effective maneuvering,



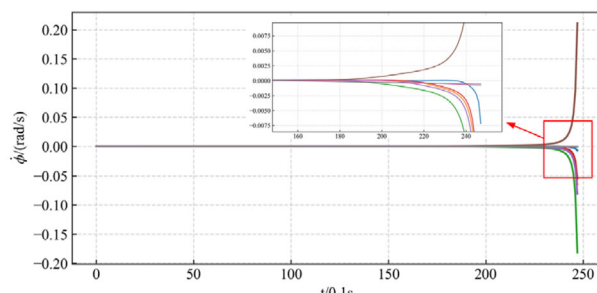
(a) Acceleration at body y axis



(b) Acceleration at body z axis



(c) Change rate of θ_m



(d) Change rate of ϕ_m

Fig. 10 Acceleration and LOS angle curve

and higher energy consumption often leads to a larger terminal miss distance.

Figure 12 shows the scatters of energy consumption and terminal miss distance for CPPO agent and PPO agent on the test dataset, clearly demonstrating the advantages and characteristics of the CPPO algorithm. Figure 12a and c depict scatters of the energy consumption for CPPO and PPO, respectively. It can be observed that the energy constraint parameter C has a significant influence on the agent in the CPPO algorithm. The agent is able to control its energy consumption below the corresponding constraint value in all situations, while obtaining the maximum terminal miss distance. Conversely, accurately controlling the agent's energy consumption is challenging for the PPO algorithm. The difference in miss distance scatters between CPPO and PPO is more evident. Figure 12b demonstrates that CPPO has a more uniform distribution of miss distance, exhibiting noticeable distinctions among CPPO agents with different constraint values.

5.3 Robustness Experiments Under Information Error Conditions

The experiments above were conducted under perfect information conditions, where the observations of the agent were completely accurate. However, in real-life scenarios, the observations of the agent are often inaccurate due to the environmental noise, and filtering algorithm performance. Therefore, the robustness of RL algorithms in information error conditions is also an important evaluation metric.

In the aforementioned assumptions, the LOS angles are directly measured by the evader through an infrared sensor, while the relative distance information is obtained through a data fusion and filtering algorithm. Therefore, we assume that the angle measurements have only minor random errors that follow a normal distribution. As for the relative distance, we assume that errors consist of two components: random errors that follow a normal distribution and systematic errors that

follow a uniform distribution. The error representations of the observations are given by formulas (28) and (29).

$$Err_{\varphi_m}, Err_{\theta_m} \sim N(0, \sigma_a^2), \quad (28)$$

$$Err_R \sim N(0, \sigma_b^2) + U(e_1, e_2) \times \frac{R_{err}}{R_{max}}, \quad (29)$$

where $\sigma_a = 5 \times 10^{-4}$, $\sigma_b = 5 \times 10^{-2}$, $R_{err} = 10^4$ m. We have established 6 levels of error based on different values of e_1 and e_2 , as shown in Table 5.

The comparison between the error values and the accurate values for a specific simulation are illustrated in Fig. 13.

The performance of CPPO agents under different error levels is shown in Table 6.

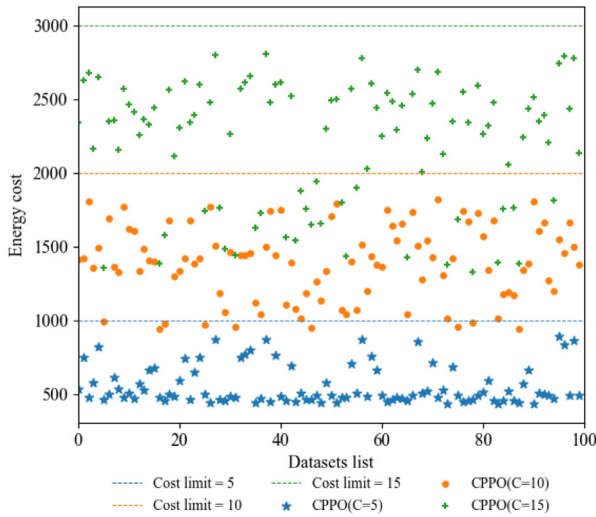
According to Table 6, the CPPO ($C = 5$) agent is highly affected by information errors. Under error level 0, the task success rate decreases to 81%. The performance of CPPO ($C = 5$) under the 4th error level and above is almost unacceptable. On the other hand, the CPPO ($C = 10$) and CPPO ($C = 15$) agents demonstrate remarkable robustness, maintaining a 100% task success rate under all error levels. However, both the maximum energy consumption and terminal miss distance are significantly impacted, as depicted in Fig. 14.

From Fig. 14, it is evident that both energy consumption and miss distance show considerable fluctuations under error conditions. The energy consumption of all three CPPO algorithms slightly surpasses the constraints at error levels 2 and 3. In general, larger constraint values lead to safer strategies for the agents, which in turn enhance their robustness.

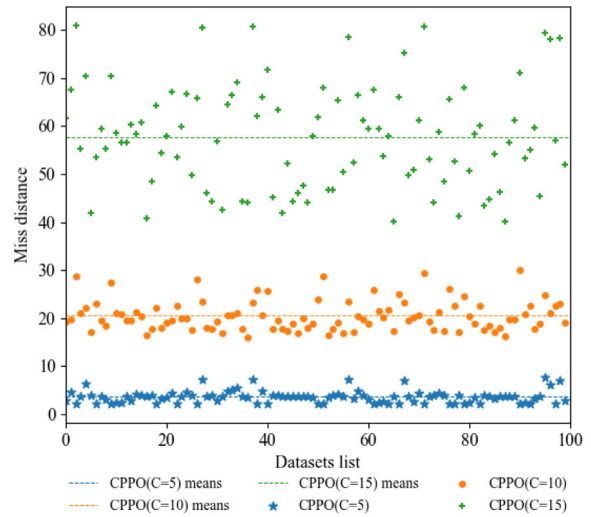
Based on the above analysis, it seems that $C = 10$ is a highly balanced choice. By relying on reasonable energy consumption, it not only ensures a sufficient safety margin for the miss distance but also possesses the capability to counteract noise environments. Therefore, $C = 10$ can be considered as the preferred option in the absence of strict energy constraints. However, in practical scenarios, energy reserves may not be sufficient, implying that energy constitutes a strict

Table 4 Results on test dataset

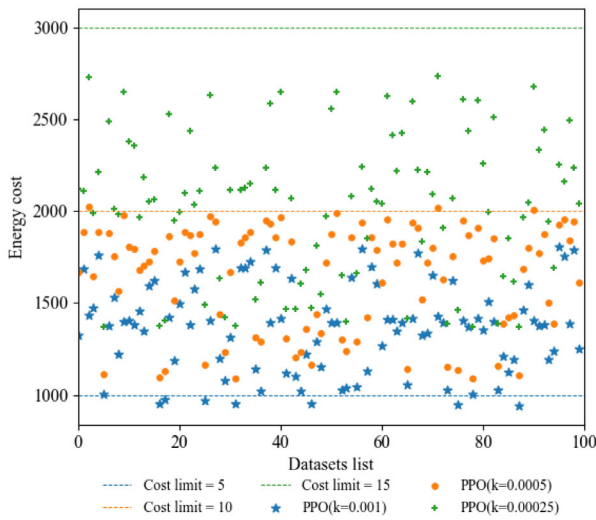
Method	Success rate (%)	Energy cost/(m/s ²)		Average miss distance/m	
		Means	Max	When success	When fail
CPPO($C = 5$)	100	558.08	896.33	3.61	\
CPPO($C = 10$)	100	1387.46	1819.68	20.52	\
CPPO($C = 15$)	100	2211.72	2808.82	57.75	\
PPO($k = 0.001$)	100	1367.23	1813.49	18.72	\
PPO($k = 0.0005$)	100	1647.64	2018.33	26.84	\
PPO($k = 0.00025$)	100	2035.75	2732.56	45.40	\
Step maneuver 1	82	1994.80	2000	33.06	0.27
Step maneuver 2	62	2773.40	2800	26.94	0.23



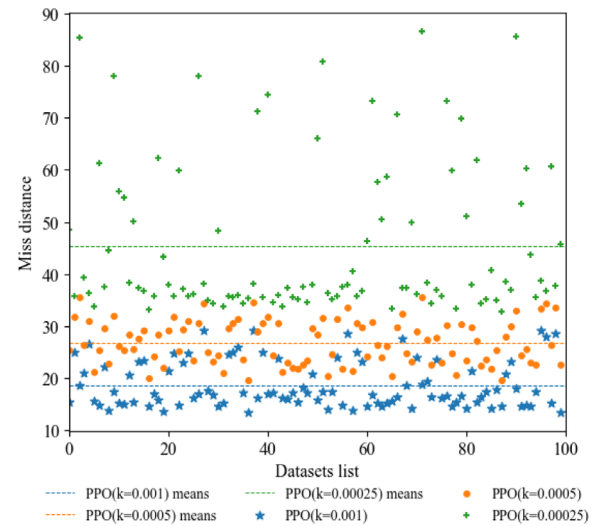
(a) Energy cost of CPPO



(b) Miss distance of CPPO



(c) Energy cost of PPO



(d) Miss distance of PPO

Fig. 12 Test dataset

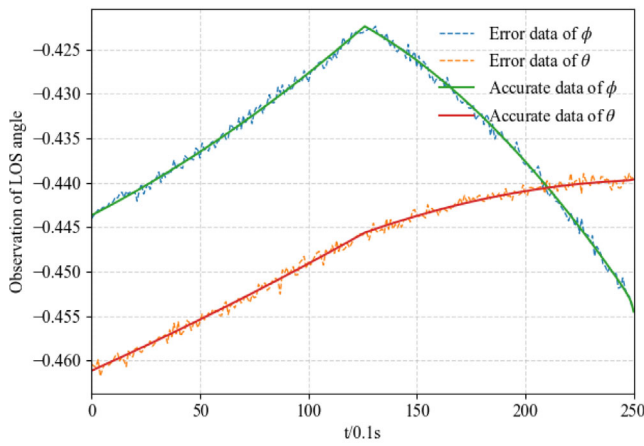
Table 5 Error level

Error level	0	1	2	3	4	5
Parameter values	$e_1=0, e_2=0$	$e_1=-1, e_2=1$	$e_1=-1, e_2=0$	$e_1=-2, e_2=0$	$e_1=0, e_2=1$	$e_1=0, e_2=2$

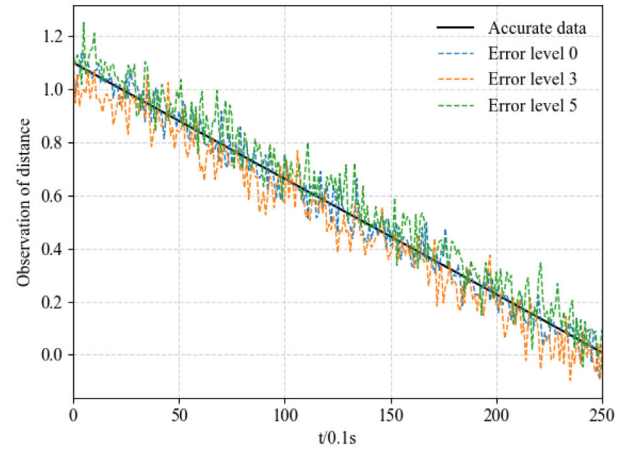
constraint. In this context, we can only determine the corresponding C value based on the total amount of available energy at the time. This necessitates training multiple agents under different C values (e.g., $C = 5, 6, 7, \dots, 10$) during the offline training phase so that we can flexibly invoke different agents based on varying circumstances during the online application phase.

6 Conclusions

- (1) Constrained reinforcement learning is capable of addressing decision-making problems under constraints. Unlike traditional reinforcement learning algorithms, constrained reinforcement learning decouples the constraints from the decision objectives. The intelligent agent trained with constrained reinforcement



(a) LOS angle observation with noise



(b) Relative distance observation with noise

Fig. 13 Observation with error

Table 6 Results on test dataset with information error

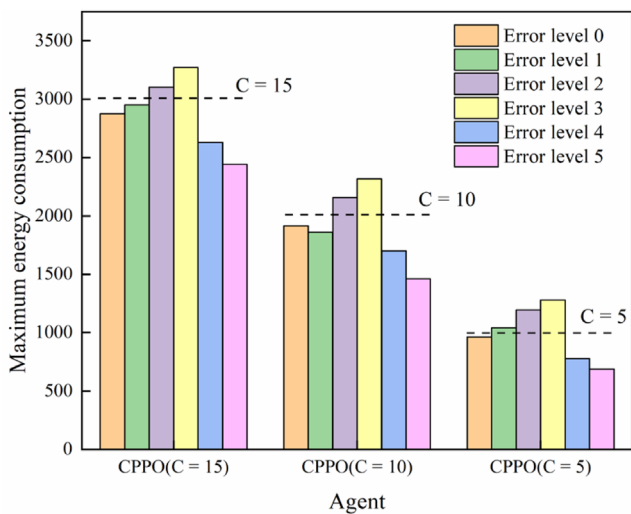
Method	Error level	Success rate (%)	Energy cost/(m/s ²)		Average miss distance/m	
			Means	Max	When success	When fail
CPPO (C = 15)	0	100	2210.79	2874.65	53.14	\
	1	100	2203.85	2949.96	51.78	\
	2	100	2385.70	3102.44	63.73	\
	3	100	2575.18	3272.22	75.35	\
	4	100	2001.25	2628.75	41.43	\
CPPO (C = 10)	0	100	1382.24	1915.54	16.38	\
	1	100	1387.03	1860.88	15.37	\
	2	100	1578.38	2157.35	22.63	\
	3	100	1762.41	2315.65	28.33	\
	4	100	1195.11	1702.19	10.51	\
CPPO (C = 5)	0	81	598.33	963.39	1.53	0.35
	1	68	607.01	1041.26	1.58	0.31
	2	87	768.46	1192.73	2.77	0.29
	3	77	950.90	1279.39	5.26	0.26
	4	55	438.08	777.90	1.02	0.28
	5	16	316.64	688.18	0.69	0.28

learning knows how to find optimal policies while satisfying the constraints. The learned policies can even reflect the relationship between the constraints and decision objectives. In this paper, the agent trained with the CPPO algorithm has learned the correlation between energy consumption and deviation, while the PPO algorithm does not produce such an effect.

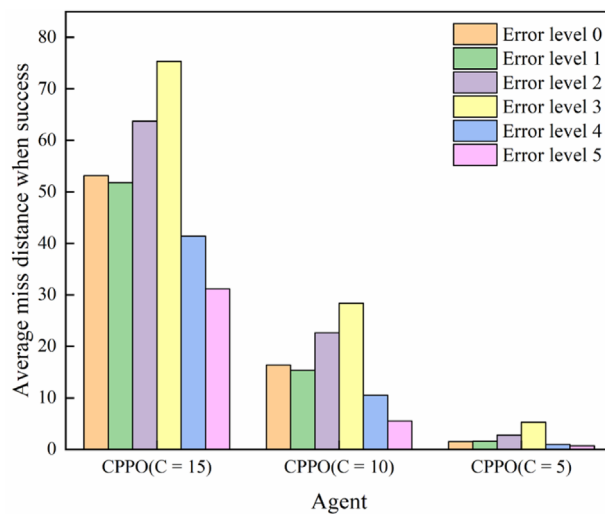
(2) The constrained reinforcement learning method introduced in this paper is characterized as a soft constraint approach. Consequently, when under certain constraint

conditions, the agent will initially prioritize the reward function and may bypass the constraints. Nevertheless, it will gradually converge to the constraints.

(3) The constraint value is correlated with the level of risk-taking by the intelligent agent. Agents with larger constraint values tend to adopt safer strategies, while agents with smaller constraint values are inclined towards more adventurous strategies. Therefore, the robustness of an agent is also influenced by the magnitude of the constraint value. To obtain a more robust intelligent agent,



(a) Maximum energy consumption



(b) Average miss distance when success

Fig. 14 Agent performance under information error conditions

it is necessary to provide the agent with looser constraints (such as enough resources or a broader decision space), enabling the agent to make safer decisions without resorting to risky choices.

- (4) Observation noise is an important factor that affects the performance of agents, especially when the available energy is not sufficient (e.g., $C = 5$), the performance of agents under observation noise may become unacceptable. Therefore, the development of robust reinforcement learning guidance laws under observation noise is a promising research direction for the future.

Acknowledgements This research was funded by Nature Science Foundation of Shannxi Province, China, Grant Number 2021JQ-370 and National Natural Science Foundation of China, Grant Number 62106284.

Declarations

Conflict of Interest The authors declare no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

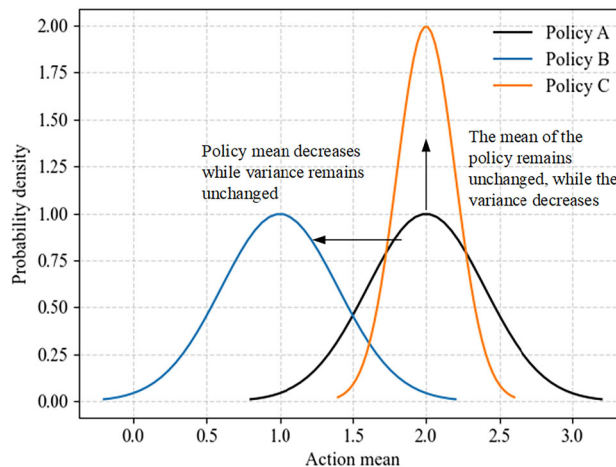


Fig. 15 The initial policy is policy A. Under the constraint of cumulative actions, policy A may evolve towards policy B or policy C. The effect of the two learning directions on the policy loss function may be similar

Appendix

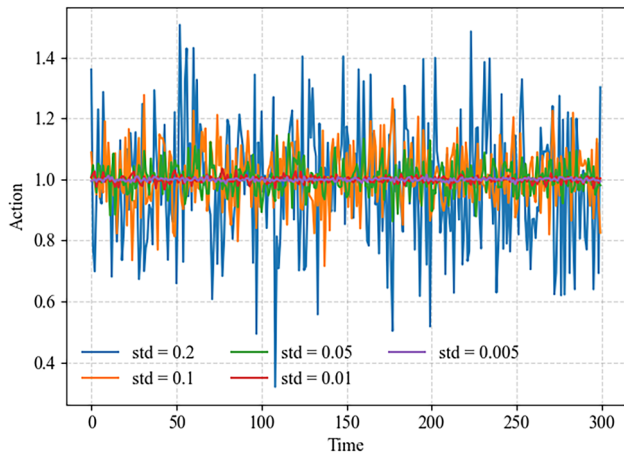
Influence of exploration randomness.

To encourage exploration, the initial variance of the policy distribution is usually set to 1. As the policy converges, the policy variance gradually decreases and eventually stabilizes at a certain value. In the CMDP problem, when we regard the cumulative actions as constraints, like formula $\sum c = \sum \tau |a| \leq C$, the decrease in the mean or variance of the policy distribution achieves similar results during training, as shown in Fig. 15.

To further illustrate this problem, let us consider an example. As shown in Table 7, suppose the mean of policy 1 is 0.4 and the standard deviation is 0.4, while the mean of policy 2

Table 7 An example to illustrate the impact of policy variance on the computation of cumulative constraints

Policy	Action mean	Standard deviation	Accumulated cost ($\tau = 0.1$)
Policy 1	0.4	0.4	14.876
Policy 2	0.5	0.1	14.473

**Fig. 16** Comparison of action outputs from different policy distributions**Table 8** The maximum deviation of cumulative sampled action values

Policy standard deviation	0.2	0.1	0.05	0.01	0.005
Maximum cumulative deviation	11.41	3.49	2.07	0.33	0.20

is 0.5 and the standard deviation is 0.1. We sample both policy distributions for 300 steps (roughly equal to the decision scale of the RL environment in this paper). The cumulative cost generated by both policies during training is approximately 15. When the C value is set to 15, the agent may converge to policy 1. However, policy 2 is clearly the desired result because the mean of actions is the core parameter of the policy.

Next, we explain the choice of H_0 value in formula (20). Suppose the standard deviation of the final policy distribution is 0.2, 0.1, 0.05, 0.01, and 0.005, respectively, with action mean of 1, as shown in Fig. 16. We sample these policy distributions for 300 steps and perform 100 Monte Carlo experiments. The maximum deviation of the cumulative sampled action values is shown in Table 8.

It can be seen that when the standard deviation of the policy distribution is 0.01, the deviation caused by the randomness of the policy can be considered negligible. Therefore, we hope that the standard deviation of the policy

distribution can ultimately converge to around 0.01. Finally, based on the probability density formula, we can calculate the target value H_0 of $\log(p(a))$ in formula (20). We hope that the sampled action of the policy tends towards the mean value, which can be described as,

$$\log(p(a)) \rightarrow \log(p(\mu)). \quad (30)$$

When $a = \mu$, the probability density formula is:

$$p(\mu) = e^{-\frac{1}{2\sigma^2}} / \sqrt{2\pi}\sigma. \quad (31)$$

When $\sigma = 0.01$, we get $p(\mu) \approx 39.89$ and $H_0 = \lfloor \log(p(\mu)) \rfloor = 3$, where $\lfloor x \rfloor$ is the round down symbol.

References

- Chen X, Yu J (2022) Reach-avoid games with two heterogeneous defenders and one attacker. *IEEE Trans Cybern* 16:301–317. <https://doi.org/10.1049/cth2.12226>
- Liu S, Yan B, Zhang X, Liu W, Yan J (2022) Fractional-order sliding mode guidance law for intercepting hypersonic vehicles. *Aerospace* 9:1–16. <https://doi.org/10.3390/aerospace9020053>
- He S, Lee CH (2019) Optimal impact angle guidance for exoatmospheric interception utilizing gravitational effect. *IEEE Trans Aerosp Electron Syst* 55:1382–1392. <https://doi.org/10.1109/TAES.2018.2870456>
- Reisner D, Shima T (2013) Optimal guidance-to-collision law for an accelerating exoatmospheric interceptor missile. *J Guid Control Dyn* 36:1695–1708. <https://doi.org/10.2514/1.61258>
- Liang H, Wang J, Liu J, Liu P (2020) Guidance strategies for interceptor against active defense spacecraft in two-on-two engagement. *Aerosp Sci Technol* 96:105529. <https://doi.org/10.1016/j.ast.2019.105529>
- Liang H, Wang J, Wang Y, Wang L, Liu P (2020) Optimal guidance against active defense ballistic missiles via differential game strategies. *Chin J Aeronaut* 33:978–989. <https://doi.org/10.1016/j.cja.2019.12.009>
- Wang W (2023) Cooperative augmented proportional navigation and guidance for proximity to uncooperative space targets. *Adv Sp Res* 71:1594–1604. <https://doi.org/10.1016/j.asr.2022.09.026>
- Yan X, Lyu S (2020) A two-side cooperative interception guidance law for active air defense with a relative time-to-go deviation. *Aerosp Sci Technol* 100:105787. <https://doi.org/10.1016/j.ast.2020.105787>
- Garcia E, Casbeer DW, Pachter M (2015) Cooperative strategies for optimal aircraft defense from an attacking Missile. *J Guid Control Dyn* 38:1510–1520. <https://doi.org/10.2514/1.G001083>
- Zou X, Zhou D, Du R, Liu J (2016) Adaptive nonsingular terminal sliding mode cooperative guidance law in active defense scenario. *Proc Inst Mech Eng Part G J Aerosp Eng* 230:307–320. <https://doi.org/10.1177/0954410015591613>
- Shaferman V (2021) Near-optimal evasion from pursuers employing modern linear guidance laws. *J Guid Control Dyn* 44:1823–1835. <https://doi.org/10.2514/1.G005725>
- Carr RW, Cobb RG, Pachter M, Pierce S (2018) Solution of a pursuit-evasion game using a near-optimal strategy. *J Guid Control Dyn* 41:841–850. <https://doi.org/10.2514/1.G002911>

13. Fonod R, Shima T (2016) Multiple model adaptive evasion against a homing missile. *J Guid Control Dyn* 39:1578–1592. <https://doi.org/10.2514/1.G000404>
14. Sun Q, Zhang C, Liu N, Zhou W, Qi N (2019) Guidance laws for attacking defended target. *Chin J Aeronaut* 32:2337–2353. <https://doi.org/10.1016/j.cja.2019.05.011>
15. Yue L, Yang R, Zhang Y, Zuo J (2023) Research on reinforcement learning-based safe decision-making methodology for multiple unmanned aerial vehicles. *Front Neurobot*. <https://doi.org/10.3389/fnbot.2022.1105480>
16. Zhou X, Zhang X, Zhao H, Xiong J, Wei J (2022) Constrained soft actor-critic for energy-aware trajectory design in UAV-aided IoT Networks. *IEEE Wirel Commun Lett* 11:1414–1418. <https://doi.org/10.1109/LWC.2022.3172336>
17. Gu S, Grudzien Kuba J, Chen Y, Du Y, Yang L, Knoll A, Yang Y (2023) Safe multi-agent reinforcement learning for multi-robot control. *Artif Intell* 319:103905. <https://doi.org/10.1016/j.artint.2023.103905>
18. He S, Shin HS, Tsourdos A (2021) Computational missile guidance: a deep reinforcement learning approach. *J Aerosp Inf Syst* 18:571–582. <https://doi.org/10.2514/1.I010970>
19. Hong D, Kim M, Park S (2020) Study on reinforcement learning-based missile guidance law. *Appl Sci*. <https://doi.org/10.3390/AP10186567>
20. Gaudet B, Furfaro R (2012) Missile homing-phase guidance law design using reinforcement learning. *AIAA Guid Navig Control Conf*. <https://doi.org/10.2514/6.2012-4470>
21. Gong X, Chen W, Chen Z (2022) All-aspect attack guidance law for agile missiles based on deep reinforcement learning. *Aerosp Sci Technol* 127:107677. <https://doi.org/10.1016/j.ast.2022.107677>
22. Li W, Zhu Y, Zhao D (2022) Missile guidance with assisted deep reinforcement learning for head-on interception of maneuvering target. *Complex Intell Syst* 8:1205–1216. <https://doi.org/10.1007/s40747-021-00577-6>
23. Gong X, Chen W, Chen Z (2023) Intelligent game strategies in target-missile-defender engagement using curriculum-based deep reinforcement learning. *Aerospace*. <https://doi.org/10.3390/aerospace10020133>
24. Shalumov V (2020) Cooperative online Guide-Launch-Guide policy in a target-missile-defender engagement using deep reinforcement learning. *Aerosp Sci Technol* 104:105996. <https://doi.org/10.1016/j.ast.2020.105996>
25. Yang B, Liu P, Feng J, Li S (2021) Two-stage pursuit strategy for incomplete-information impulsive space pursuit-evasion mission using reinforcement learning. *Aerospace*. <https://doi.org/10.3390/aerospace8100299>
26. Brandonsio A, Capra L, Lavagna M (2023) Deep reinforcement learning spacecraft guidance with state uncertainty for autonomous shape reconstruction of uncooperative target. *Adv Sp Res*. <https://doi.org/10.1016/j.asr.2023.07.007>
27. Zhao L, Zhang Y, Dang Z (2023) PRD-MADDPG: an efficient learning-based algorithm for orbital pursuit-evasion game with impulsive maneuvers. *Adv Sp Res* 72:211–230. <https://doi.org/10.1016/j.asr.2023.03.014>
28. Zhang J, Zhang K, Zhang Y, Shi H, Tang L, Li M (2022) Near-optimal interception strategy for orbital pursuit-evasion using deep reinforcement learning. *Acta Astronaut* 198:9–25. <https://doi.org/10.1016/j.actaastro.2022.05.057>
29. Qiu X, Gao C, Jing W (2022) Maneuvering penetration strategies of ballistic missiles based on deep reinforcement learning. *Proc Inst Mech Eng Part G J Aerosp Eng* 16:3494–3504. <https://doi.org/10.1177/09544100221088361>
30. Jiang L, Nan Y, Li ZH (2021) Realizing midcourse penetration with deep reinforcement learning. *IEEE Access* 9:89812–89822. <https://doi.org/10.1109/ACCESS.2021.3091605>
31. Gaudet B, Furfaro R, Linares R (2020) Reinforcement learning for angle-only intercept guidance of maneuvering targets. *Aerosp Sci Technol* 99:105746. <https://doi.org/10.1016/j.ast.2020.105746>
32. Yeh FK (2010) Design of nonlinear terminal guidance/autopilot controller for missiles with pulse type input devices. *Asian J Control* 12:399–412. <https://doi.org/10.1002/asjc.196>
33. Zarchan P (2019) Tactical and strategic missile guidance. *AIAA*, Georgia, pp 41–42
34. Qi N, Sun Q, Zhao J (2017) Evasion and pursuit guidance law against defended target. *Chin J Aeronaut* 30:1958–1973. <https://doi.org/10.1016/j.cja.2017.06.015>
35. Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O (2017) Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*
36. Haarnoja T, Zhou A, Hartikainen K, Tucker G, Ha S, Tan J, Kumar V, Zhu H, Gupta A, Abbeel P, Levine S (2018) Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.