scientific reports

Check for updates

OPEN

A hierarchical reinforcement learning method for missile evasion and guidance

Mengda Yan[1,2]✉, Rennong Yang[1,2], Ying Zhang[1,2], Longfei Yue[1] & Dongyuan Hu[1]

This paper proposes an algorithm for missile manoeuvring based on a hierarchical proximal policy optimization (PPO) reinforcement learning algorithm, which enables a missile to guide to a target and evade an interceptor at the same time. Based on the idea of task hierarchy, the agent has a two-layer structure, in which low-level agents control basic actions and are controlled by a high-level agent. The low level has two agents called a guidance agent and an evasion agent, which are trained in simple scenarios and embedded in the high-level agent. The high level has a policy selector agent, which chooses one of the low-level agents to activate at each decision moment. The reward functions for each agent are different, considering the guidance accuracy, flight time, and energy consumption metrics, as well as a field-of-view constraint. Simulation shows that the PPO algorithm without a hierarchical structure cannot complete the task, while the hierarchical PPO algorithm has a 100% success rate on a test dataset. The agent shows good adaptability and strong robustness to the second-order lag of autopilot and measurement noises. Compared with a traditional guidance law, the reinforcement learning guidance law has satisfactory guidance accuracy and significant advantages in average time and average energy consumption.

Target-missile-defender (TMD) engagement has always been a valuable issue, in which a missile struggles to hit the target and a defender aims to intercept the missile. Traditional studies on the TMD problem focus on how the defender can intercept the attack missile[1–4]. With the development of these studies, interception technology has increasingly advanced in recent years. On the other hand, the survivability of attack missiles has been greatly threatened. Therefore, our research focuses on how an attack missile can evade a defender and hit the target. Most contributions have approached this problem using optimal control and differential game theory. Ryoo et al.[5] studied the evasive manoeuvre strategy of anti-ship missiles against a shipborne close-in weapon system (CIWS) in three-dimensional space. Through the evasive manoeuvre of anti-ship missiles, the CIWS aiming error increases. However, most weapons used in CIWSs are cannons, which are quite different from a homing guidance missile. Yogaswara et al.[6] studied the evasion strategy of attack missiles against the interceptor missiles of an integrated air defense system. The attack missile needs to avoid interception of the interceptor missiles and eventually guide to the target. A synthesis guidance law is proposed in[6], which uses an artificial potential field function, time-to-go polynomial guidance law, and logarithmic barrier function to solve the evasion command, impact angle and acceleration constraints, and field-of-view (FOV) constraints. The proposed synthesis guidance law has good performance, however, there are many parameters involved, which make it difficult to choose the optimal value. Qi et al. and Sun et al.[7,8] derived evasion and pursuit guidance laws for a missile attacking a defended aircraft based on differential game theory. The proposed approach focuses on the miss distance as the outcome of the conflict. Liang et al.[9] investigated the optimal guidance problem for an interceptor against a ballistic missile with active defence. A class of optimal guidance schemes are proposed based on a linear quadratic differential game method and the numerical solution to the Riccati differential equation. In[9], the fuel cost, control saturation and chattering phenomenon were considered. Weiss et al.[10] used a minimum-effort guidance approach to obtain a combined guidance law for the attacker.

Relevantly, the evasive of unmanned aircraft manoeuvring, which is similar to the missile evasion problem, has been studied extensively. Turetsky and Shima[11] used a matrix game method to study the engagement process of aircraft manoeuvres and homing missiles in the plane. Fonod and Shima[12] studied aircraft evasive manoeuvring under incomplete information, and an adaptive evasive framework was proposed for the proportional guidance, augmented proportional guidance, and optimal guidance law that the missile may use. Keong et al.[13] introduced a reinforcement learning method to the issue of avoidance control between aircraft. In[14], the author

[1]School of Air Traffic Control and Navigation, Air Force Engineering University, Xian 710051, China. [2]These authors contributed equally: Mengda Yan, Rennong Yang and Ying Zhang. ✉email: yanmd1@163.com

used reinforcement learning to train an aircraft agent to continuously avoid multiple surface-to-air missiles, and the agent can control four aircraft simultaneously. This research is valuable, but the aircraft's decision-making model cannot be directly applied to missiles, because missile guidance not only requires extremely high accuracy but also needs to satisfy several constraints, such as impact angle and FOV constraints. Wang et al.[15] combined reinforcement learning with a fuzzy method to address a fixed-time pursuit-evasion game. This work shows that artificial intelligence methods, such as, reinforcement learning, can be applied to such pursuit-evasion problems.

As reinforcement learning has recently made remarkable achievements in various fields, its application in missile guidance has gradually become a hotspot. Gaudet, B[16–19] may be the first to apply reinforcement learning to missile guidance law. The research in[16] shows that a reinforcement learning guidance law performs better than a proportional guidance method and an enhanced proportional guidance method considering the noise and time delay of the sensors and actuators. In[17], reinforcement learning combined with meta-learning was applied to the guidance law of exo-atmospheric interceptors. A reinforcement learning algorithm outputs four propulsion instructions for the steering thrusters. The results show that a reinforcement learning guidance law is superior to the traditional zero-effort-miss(ZEM) guidance law in interception rate and energy consumption. Recently, this meta-reinforcement learning method was applied to the hypersonic guidance problem[18,19]. Hong et al. and He et al.[20,21] studied the comparison between a deep deterministic policy gradient (DDPG)[22] reinforcement learning guidance law and the traditional proportional guidance law, and experiments verified that a guidance law based on reinforcement learning can be applied to the missile guidance law. In[23], a double duelling deep Q-network (D3Q)[24] reinforcement learning algorithm was applied to the mid-course penetration of exo-atmospheric ballistic missiles. However, the evasive manoeuvre of air-to-surface missiles is quite different from the mid-course penetration of exo-atmospheric ballistic missiles.

The purpose of this study is to use reinforcement learning to realize the evasive manoeuvre of an attack missile against an interceptor, and finally guide to the target. To this end, this study first modelled the missile guidance process as a Markov process, with the kinematic relationship between the interceptor, missile, and target as the environment, and the acceleration command as the action of the agent. The hierarchical reinforcement learning method introduces the idea of task decomposition into reinforcement learning, which can reduce the complexity of the problem. Hierarchical reinforcement learning has been adopted in some research studying complex decision problems. Pope et al. and Sun et al.[25,26] used hierarchical reinforcement learning to address air combat decision-making. In a StarCraft game environment[27], proposed a hierarchical command and control architecture, consisting of single high-level and multiple low-level reinforcement learning agents operating in a dynamic environment. This hierarchical model enables the low-level unit agents to make individual decisions while taking commands from the high-level commander agent.

The main contributions of this paper can be summarised as follows.

1. A hierarchical reinforcement learning framework for missile evasion guidance is proposed. The entire reinforcement learning framework is divided into two levels. The low level includes two agents, namely, a guidance agent and an evasive agent. The high level is the selection agent, which determines which agent in the low level should be activated in every decision moment.
2. To improve and evaluate the performance of the agents, several metrics are considered, including guidance accuracy, flight time, energy consumption, and an FOV constraint. All agents use the PPO[28] reinforcement learning algorithm, but each has different observations and rewards.
3. Agents was tested in different scenarios. Experiments show that the hierarchical PPO method is significantly better than the PPO algorithm without a hierarchical structure and a traditional method. Additionally, the agent shows good adaptability and strong robustness to the second-order lag of the autopilot and measurement noises.

The paper is organized as follows. In Section "Problem formulation", the problem formulation is presented. In Section "Hierarchical reinforcement learning-based guidance law", the hierarchical reinforcement learning-based guidance law is proposed. In Section "Experiment and result analysis", experiments and results analysis are presented.

Problem formulation

Relative kinematics. The kinematics relationship is shown in Fig. 1. In this paper, we use the term interceptor instead of defender. Therefore, the engagement in this paper can be called the target-missile-interceptor problem. As presented in the geometry, the inertial reference frame is denoted as XOY. The notations v_m and v_i are the velocities of the attack missile and interceptor, respectively. The notations a_m and a_i are the lateral accelerations of the attack missile and interceptor, respectively. The notations φ_m and φ_i denote the flight path angle of the attack missile and interceptor missile, respectively. The notations λ_m and λ_i denote the line-of-sight(LOS) angle of attack missile to target and interceptor missile to attack missile. The notation of d_m and d_i denote the missile-target relative range and interceptor-missile relative range, respectively.

The equation of motion for both the attack and intercept missiles in the inertial reference is generally given by

$$\dot{x} = v\cos\varphi. \tag{1}$$

$$\dot{y} = v\sin\varphi. \tag{2}$$

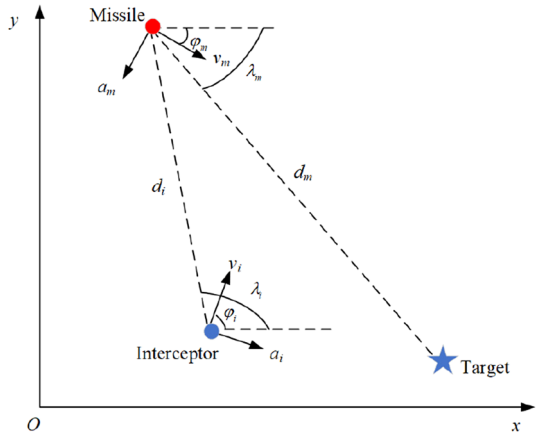


Figure 1. Planar engagement geometry.

$$\dot{\varphi} = \frac{a}{v}. \tag{3}$$

The equations describing the interceptor-missile relative motion kinematics can be formulated as

$$\dot{d}_i = -v_i \cos(\lambda_i - \varphi_i) - v_m \cos(\pi - \lambda_i - \varphi_m). \tag{4}$$

$$\dot{\lambda}_i = \frac{v_i \sin(\lambda_i - \varphi_i) - v_m \sin(\pi - \lambda_i - \varphi_m)}{d_i}. \tag{5}$$

Similarly, the equations describing the missile-target relative motion kinematics can be formulated as

$$\dot{d}_m = -v_m \cos(\lambda_m - \varphi_m). \tag{6}$$

$$\dot{\lambda}_m = \frac{-v_m \sin(\lambda_m - \varphi_m)}{d_m}. \tag{7}$$

In this study, the intercept missile applies typical proportional navigation guidance (PNG) to intercept the attacking missile. The acceleration command for the intercept missile is formulated as

$$a_i = -NV_i\dot{\lambda}_i. \tag{8}$$

where the navigation constant $N = 3$.

The lateral acceleration delays generated by the autopilot are also considered. In this study, the missile autopilot controller is modelled as the differential equation of a second-order system as

$$\ddot{a}_m = -2\xi\omega_n\dot{a}_m - \omega_n^2 a_m + \omega_n^2 a_c. \tag{9}$$

where ξ is the damping ratio, ω_n is the natural frequency, a_m is the derived actuation, and a_c is the acceleration command.

Engagement scenario. The engagement scenario in this study is shown in Fig. 2. The attack missile performs an evasive manoeuvre first and then guide to the target. To simplify the problem, we assume that the target is stationary, which does not affect the validity of the research, since the simulation focuses on the missile and the interceptor. The intercept missile has a capture zone, which is the maximum launch distance of the interceptor. The interceptor can be launched only when the attacking missile enters the capture zone. The parameters of the engagement scenario are shown in Table 1.

The ZEM is an expected miss distance if there is no further manoeuvre from the current location[20,29]. We only consider the ZEM at the terminal state called the ZEM*, which was proposed in[20]. The ZEM* diagram is shown in Fig. 3.

The ZEM* can be calculated by

$$\mathrm{ZEM}^* = |\mathbf{d}_m^{t-1}| \sin\theta. \tag{10}$$

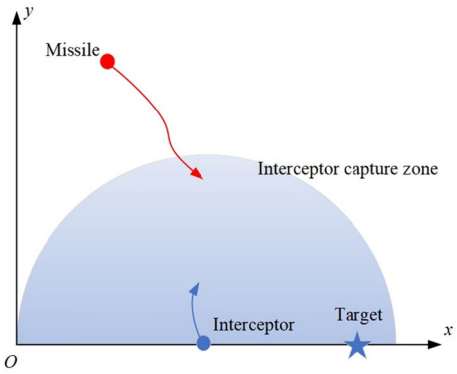


Figure 2. Engagement scenario.

Position of the target	Position of the interceptor launch platform	Range of interceptor capture zone	Flight time of interceptor
(9, 0) km	(5, 0) km	6 km	21 s

Table 1. Parameters of the engagement scenario.

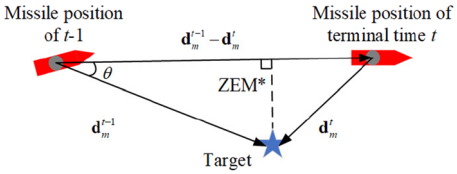


Figure 3. ZEM* Diagram.

$$\theta = \arccos\left(\frac{\left(\mathbf{d}_m^{t-1} - \mathbf{d}_m^t\right) \cdot \mathbf{d}_m^{t-1}}{|\mathbf{d}_m^{t-1} - \mathbf{d}_m^t||\mathbf{d}_m^{t-1}|}\right). \tag{11}$$

It should be noted that the simulation terminal state is defined by the relative distance. The intercepted state and the terminal impact state are defined by judgement terms $d_i \leq 10(m)$ and $d_m \leq 10(m)$, respectively. The error when using the terminal distance as the judgement condition for simulation termination is acceptable. Once the relative distance is less than 10 metres, it can be considered a hit state. However, due to the limitation of simulation accuracy, the terminal distance is often larger than the ZEM, so it cannot really reflect the guidance accuracy. Therefore, the ZEM* is used to describe the guidance accuracy and used in the reward function, which are described in Section "Agent architecture".

Hierarchical reinforcement learning-based guidance law

Reinforcement learning. Reinforcement learning is a learning process using exploration. The agent learns how to make optimal decisions through continuous interaction with the environment. The agent receives the state of the environment and performs an action based on the state. The environment changes to the next state according to the action and returns a reward or penalty to the agent. The loop continues until the environment reaches the terminal state (success or failure). A simplified diagram of reinforcement learning is shown in Fig. 4.

Reinforcement learning usually models the problem as a Markov decision process (MDP), which comprises a tuple $\langle S, A, T, R\rangle$. Given a state $s \in S$, selecting an action $a \in A$ will transform the environment to a new state $s' \in S$ with transition probability $T(s, a, s') \in [0, 1]$ and return a reward $R(s, a)$. A stochastic policy $\pi : S \rightarrow A$ is a mapping from states to probabilities of selecting each possible action. The goal is to determine the optimal policy π^* that provides the highest expected sum of rewards:

$$\pi^* = \arg\max E_\pi\left\{\sum_t^T \gamma^t r_{t+1} | s_0 = s\right\}. \tag{12}$$

where $\gamma \in [0, 1]$ is a discount factor. The objective function, called the Q-function, is defined as:

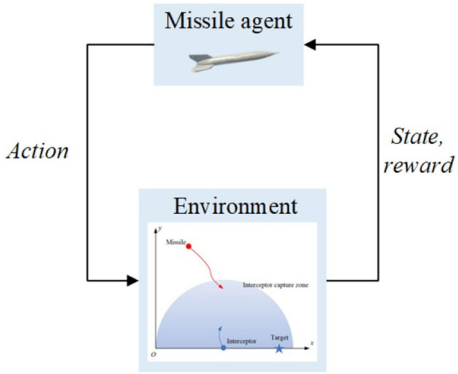


Figure 4. Diagram of reinforcement learning.

$$Q_\pi(s,a) = \mathrm{E}_\pi\left\{\sum_t^T \gamma^t r_{t+1} | s_0 = s, a_0 = a\right\}. \tag{13}$$

PPO algorithm. The PPO algorithm, which is a state-of-the-art reinforcement learning algorithm, is used to train the agent in this study. The PPO algorithm is an on-policy reinforcement learning algorithm based on an actor-critic (AC) framework. The PPO algorithm uses importance sampling to calculate the ratio of the old policy to the new policy to measure the quality of the new policy, which is shown in Formula (14).

$$p_k(\theta) = \frac{\pi_\theta(\mathbf{u}_k|\mathbf{o}_k)}{\pi_{\theta\,old}(\mathbf{u}_k|\mathbf{o}_k)}. \tag{14}$$

The samples obtained through importance sampling can be reused many times, and the number of samples used, which is defined by the notation n_{reuse} in this paper, is an important hyperparameter in the PPO algorithm. After the PPO algorithm was proposed, several versions have been developed. The most used version is the clip version. The gap between the old policy and the new policy is controlled by a clip function. The objective function is shown in Formula (15).

$$J(\theta) = \mathrm{E}_{p(\tau)}\left[\min\left[p_k(\theta), \mathrm{clip}(p_k(\theta), 1-\varepsilon, 1+\varepsilon)\right] A^\pi_{\mathbf{w}}(\mathbf{o}_k, \mathbf{u}_k)\right]. \tag{15}$$

$$A^\pi_{\mathbf{w}}(\mathbf{x}_k, \mathbf{u}_k) = \left[\sum_{l=k}^T \gamma^{l-k} r(\mathbf{o}_l, \mathbf{u}_l)\right] - V^\pi_{\mathbf{w}}(\mathbf{x}_k). \tag{16}$$

Similar to other algorithms with an AC framework, the loss function of the critic network is shown in Formula (17).

$$L(\mathbf{w}) = \sum_{i=1}^M \left(V^\pi_{\mathbf{w}}(\mathbf{o}^i_k) - \left[\sum_{l=k}^T \gamma^{l-k} r(\mathbf{o}^i_l, \mathbf{u}^i_l)\right]\right)^2. \tag{17}$$

Hierarchical reinforcement learning. A hierarchical structure is used in many traditional action planning methods, such as a hierarchical task network (HTN) planning method[30]. The relationship between high-level tasks and low-level tasks has two types: choice and sequence. The application of this idea to reinforcement learning is called a hierarchical reinforcement learning (RL) method[31,32]. The main advantages of using hierarchical RL are transfer learning (using previously learned skills and subtasks in new tasks), scalability (decomposing large problems into smaller problems, avoiding the problem of dimensionality in high-dimensional state spaces) and generalization (combining smaller subtasks allows generating new skills, superspecialization)[25].

The PHANG-MAN dog-fight agent[25] resembles options learning algorithms[33], and it is closely related to the methods presented in[34], in which subpolicies are hierarchically structured to perform a new task. In[34], subpolicies are primitives that are pretrained in similar environments but with different tasks. In[25], PHANG-MAN has a two-layer policy structure. The high level is the policy selector, which selects one of the low-level policies to activate given the current state of the environment. Similarly, the agent in this study is a two-layer structure, which will be described in detail in the next section.

Agent architecture. In this study, the missile agent has a two-layer policy structure, as shown in Fig. 5. On the low level, there are two different agents called a guidance controller and an evasion controller, which have been trained in a particular environment state prior to the high-level agent. It should be noted that, the two

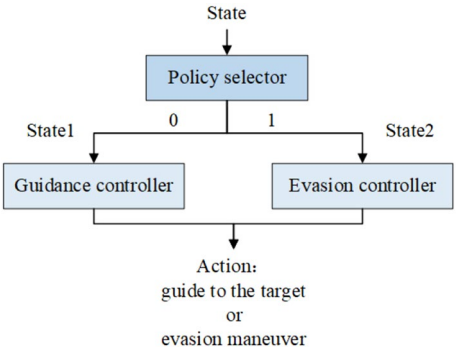


Figure 5. Hierarchical architecture of the agent.

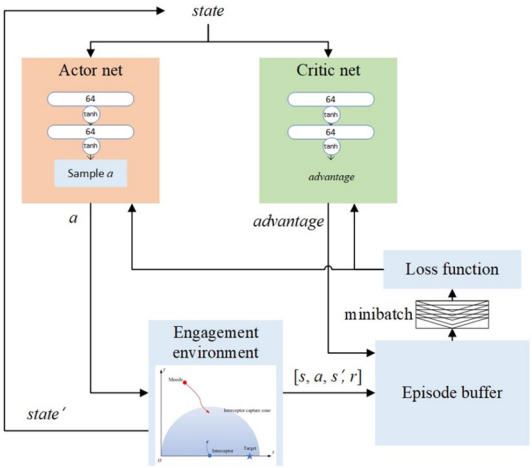


Figure 6. Training structure of the low-level agent.

agents at the low level need different states as inputs. On the high level, there is an agent called a policy selector, whose task is to choose one of the low-level agents to be active at each decision time.

Low-level policies. At the low level, both the guidance agent and the evasive agent are trained using the PPO algorithm with the same neural network structure and action space, but different state space and reward functions. In the PPO algorithm, the actor net and the critic net have the same two-layer network with 64 neurons in each layer. All activation functions are the tanh function. The action space is a continuous interval acceleration command. The state space of the guidance controller agent only considers the relative motion relationship between the missile and the target, that is $state_1 = \left[d_m, \lambda_m, \dot{d}_m, \dot{\lambda}_m\right]$. Similarly, the state space of the evasion controller agent only considers the relative motion relation-ship between the missile and the interceptor, that is $state_2 = \left[d_i, \lambda_i, \dot{d}_i, \dot{\lambda}_i\right]$. The training architecture is shown in Fig. 6.

In the PPO algorithm, parameter sharing between the actor network and critic network is a commonly used tip. Cobbe et al.[35] studied the performance of sharing parameters in the PPO algorithm and declared that sharing parameters can achieve impressive results in a way, but the hyperparameters must be set reasonably. Since the input and output dimensions of the actor and the critic networks of the low-level agent are the same, it is easy to conduct parameter sharing by combining the loss of the actor and critic networks with appropriate hyperparameters. Maximum entropy reinforcement learning[36] has achieved impressive results on continuous issues such as walking robots. Adding a policy entropy item to the loss function can enhance the exploration of the agent so it can converge to the optimal state with robustness. The loss function of the low-level agent is formulated as:

$$J(\theta) = -\mathrm{E}_{p(\tau)}\left[\min\left[p_k(\theta), \mathrm{clip}(p_k(\theta), 1-\varepsilon, 1+\varepsilon)\right] A^{\pi}_{\mathbf{w}}(\mathbf{o}_k, \mathbf{u}_k)\right] + \alpha \sum_{i=1}^{M} \left(V^{\pi}_{\mathbf{w}}(\mathbf{o}^i_k) - \left[\sum_{l=k}^{T} \gamma^{l-k} r(\mathbf{o}^i_l, \mathbf{u}^i_l)\right]\right)^2 + \beta\left(-\sum_{x} p(x)\log p(x)\right). \tag{18}$$

where the third term is the policy entropy, and α and β are hyperparameters.

The reward functions of the guidance controller agent consider guidance efficacy and constraints. The agent needs to trade off guidance accuracy, energy consumption, and flight time to obtain a satisfying guidance performance. Meanwhile, the agent should satisfy the FOV constraint. These reward functions are described as follows.

r_{impact} rewards the agent for mission success when the relative distance between the missile and target satisfies the termination condition. That is

$$r_{\text{impact}} = \begin{cases} 1, & \text{if } d_m \le 10(m) \\ 0, & \text{otherwise} \end{cases} \tag{19}$$

r_{ZEM} penalizes the agent for guidance accuracy if the missile hits the target. That is,

$$r_{ZEM} = \begin{cases} -ZEM^*, & \text{if } d_m \le 10(m) \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

r_{out} penalizes the agent for losing the target when the target is out of the FOV. That is,

$$r_{\text{out}} = \begin{cases} -1, & \text{if } |\lambda_m - \varphi_m| > \frac{\eta_{\max}}{2} \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

where $\eta_{\max}$ is the maximum FOV of the missile.

r_a penalizes the agent for energy consumption. That is,

$$r_a = -\left(\frac{a}{a_{\max}}\right)^2. \tag{22}$$

where a and $a_{\max}$ are the actual acceleration command and the maximum acceleration, respectively.

r_t penalizes the agent for flight time. The agent receives a small penalty for every step, which guides the agent to complete the task with as few steps as possible. That is, $r_t = -0.01$.

Note that the first three reward functions are composed of the terminal reward function, which is only given to the agent at the termination state. In Formula (23), r_{impact} and r_{out} represent termination conditions. The current simulation terminates once the agent hits the target or the target is out of the seeker's FOV. If the agent hits the target, r_{ZEM} is appended to the reward function at the terminal state as an evaluation metric. The last two reward functions, called shaping reward, are given to the agent at each decision time, to reduce the energy consumption and flight time of the agent.

$$r_{\text{terminal}} = r_{impact} + k_z r_{\mathrm{ZEM}} + r_{out}. \tag{23}$$

$$r_{shaping} = k_a r_a + k_t r_t. \tag{24}$$

where k_z, k_a and k_t are hyperparameters.

The reward function for the evasion agent is simpler. The evasion agent will not be rewarded, but will be penalized for being intercepted, losing the target, and energy consumption.

$r_{\text{intercpted}}$ penalizes the agent for being intercepted. That is,

$$r_{\text{intercepted}} = \begin{cases} -1, & \text{if } d_m \le 10(m) \\ 0, & \text{otherwise} \end{cases} \tag{25}$$

The reward function r_{out} defined in Formula (21) is also applied to the evasion agent. The evasion task is a subtask of the whole task, so the agent is expected to complete the evasion task while not losing the target.

The reward function r_a for the evasion agent is the same as r_a defined in Formula (22), which is the shaping reward.

Similarly, the reward functions of the evasion agent are described as r_{terminal} and $r_{shaping}$. Once the agent is intercepted or loses the target, the current simulation terminates. Meanwhile, the energy consumption should be kept as small as possible during the evasion process.

$$r_{\text{terminal}} = r_{\text{intercepted}} + r_{out}. \tag{26}$$

$$r_{shaping} = k_a r_a. \tag{27}$$

The values of reward hyperparameters of the low-level agents are shown in Table 2.

Hyperparameters	k_z	k_a	k_t
Value	0.0001	0.001	0.05

Table 2. Reward hyperparameter values of low-level agent.

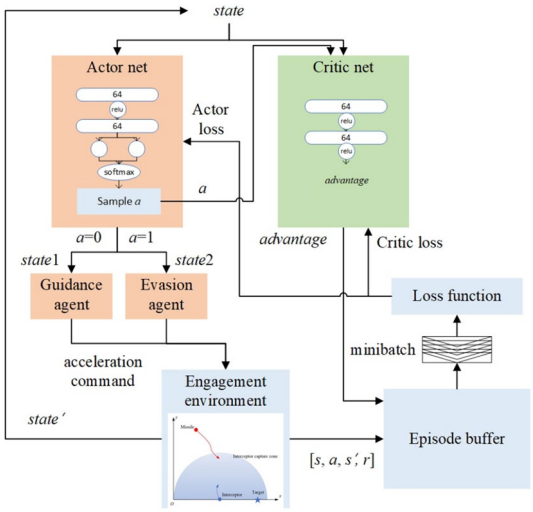


Figure 7. Training structure of the high-level agent.

High-level policy. At the high level, the policy selector agent is also trained using the PPO algorithm. Different from the low-level agent, the action space of the high-level agent is discrete, which can be described as $action\_space = [0, 1]$, where 0 and 1 represent guidance and evasion, respectively. Therefore, the activation function of the last layer of the actor net is the softmax function, which can output the probability of sampling each action. The other hidden layers employ RELU activation functions. The state space is the sum of $state_1$ and the $state_2$, that is, $state = [state_1, state_2] = \left[d_m, \lambda_m, \dot{d_m}, \dot{\lambda_m}, d_i, \lambda_i, \dot{d_i}, \dot{\lambda_i}\right]$. The input to the critic net includes the state space plus the action selected at that moment. The training architecture of the high-level agent is shown in Fig. 7. It should be noted that the input and output dimensions of the actor and critic networks are different. The actor network essentially addresses a classification problem, and the classification result needs to be used as the input to the critic network to output a continuous value. It can be said that the optimization goals of the actor and critic networks are inconsistent. According to[35], the actor and critic networks generally do not share parameters in this case; otherwise, it will have a negative impact on the training process. Therefore, the loss functions of the actor and critic networks are Formulas (15) and (17), respectively. The decision frequency of the agent is 10 Hz. The agent must decide which low-level agent should be activated at each decision moment.

The reward functions r_{impact}, r_{out} and $r_{\text{intercpted}}$, which compose the r_{terminal}, are also applied to the high-level agent.

$$r_{\text{terminal}} = r_{impact} + r_{\text{intercepted}} + r_{out}. \tag{28}$$

To shape the reward, less energy consumption and flight time are also expected. Since these two objects are considered in the low-level agent, the goal of the high-level agent is to select the evasion agent as little as possible while completing the whole task. The $r_{shaping}$ for the high-level agent is described as

$$r_{shaping} = -ka. \tag{29}$$

where k ia s hyperparameter, which is set to 0.001 in the experiment.

Experiment and result analysis

Low-level agent training and test. Notably, the training of the low-level agent and the high-level agent are independent of each other. After the low-level agents are trained, they are embedded in the high-level agent. The simulation step is 0.1 s.

Guidance agent.

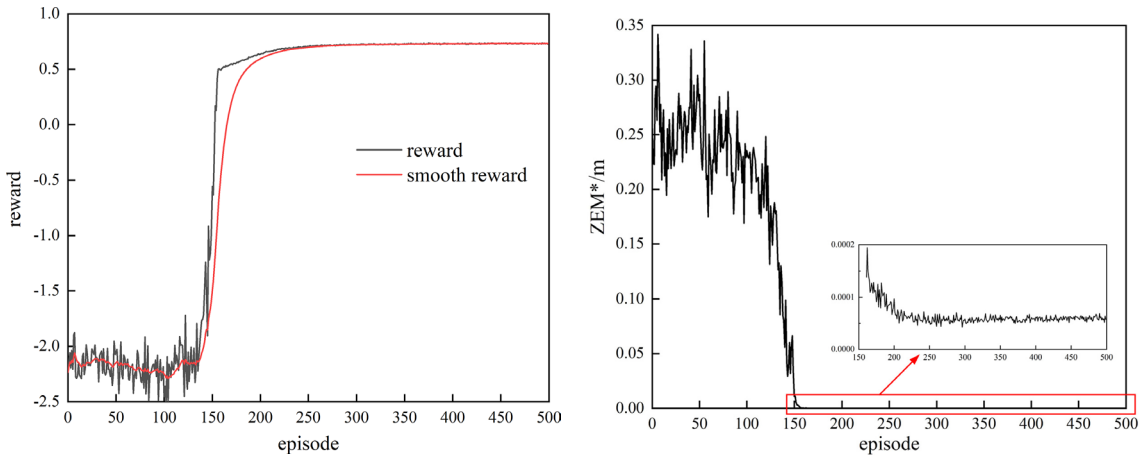
A. Training conditions and hyperparameters of the algorithm.

Parameters	Value
d_t	[9, 11] (km)
λ_m	$[-\pi/2, 0]$ (rad)
φ_m	$[\lambda_m - \pi/6, \lambda_m + \pi/6]$ (rad)
v_m	200 m/s
x_target	9 (km)
y_target	0 (km)

Table 3. Initial conditions in the training process of the guidance agent.

Parameter	Value
$n_{episode}$	500
n_{step}	20480
n_{reuse}	5
clip	0.1
α	0.5
β	0.01
Gamma	0.995
Lamda	0.95
lr	0.0001
Minibatch	640

Table 4. Hyperparameters of the PPO algorithm.



(a) Episode reward curve (b) Episode ZEM* curve

Figure 8. Episode reward and ZEM* of the guidance agent in the training process.

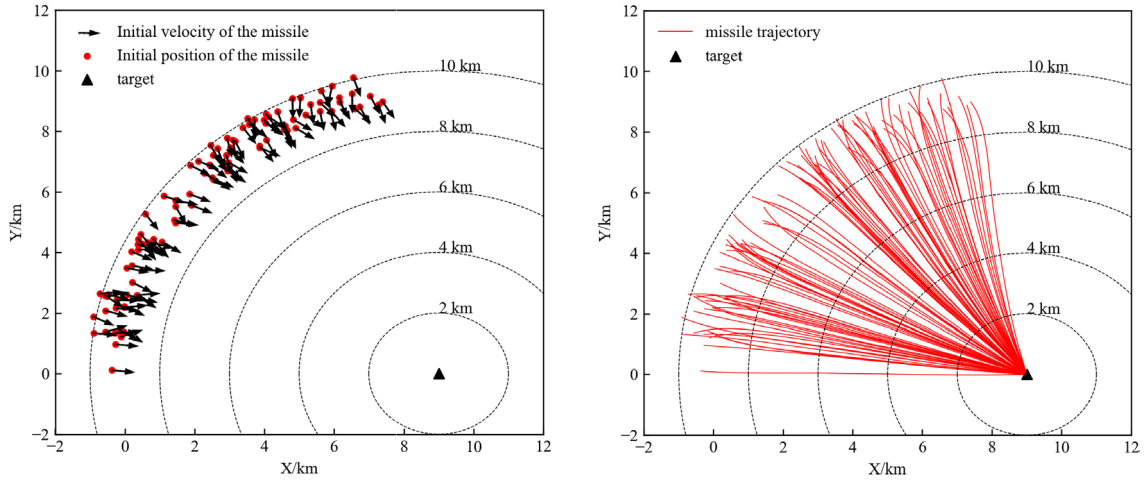
The initialization of the training environment conditions of the guidance agent is shown in Table 3. The relative distance between the missile and the target d_m, the LOS angle λ_m, and the missile heading angle φ_m are randomly selected within a certain range in the initial state of each round of training. The speed of the missile v_m, and the Cartesian coordinate values of the target are constant values.

The values of the PPO algorithm hyperparameters are shown in Table 4.

In Table 4, $n_{episode}$ is the maximum number of training episodes, n_{step} is the maximum number of steps of each episode, and *gamma*, *lamda* and *lr* are the hyperparameters in the Adam optimizer[37].

B. Training result and test.

Figure 8a shows the reward curve for the training process of the guidance agent. The training performance was unsatisfactory before 100 episodes, but it quickly improved and converged in a very stable state after 250 episodes. Figure 8b shows the ZEM* curve for the training process. During the training process, the ZEM*



(a) Test dataset of the guidance agent (b) Test result of the guidance agent

Figure 9. Test dataset and test result of guidance agent.

Success rate	ZEM*/m	Average time	Average energy consumption/$(m\ s^{-2})^2$
100%	0.31	47.53	44.3

Table 5. Metrics of guidance agent.

dropped from 3000m to approximately 1m, and finally converged to approximately 0.5m, which is a satisfactory result for missile guidance.

To test the performance of the guidance agent, we randomly generated 100 initial states to form a test dataset, which is shown in Fig. 9a. Each black arrow is an initial velocity of the missile. Figure 9b shows the performance of the guidance agent on the test dataset. At each initial state, the agent can reach the target with a simple trajectory. This study evaluates the performance of the agent using four metrics: the success rate, ZEM*, average time, and average energy consumption. We define the average energy consumption as:

$$E = \frac{\sum_{i=1}^{N}\left(\sum_{t=1}^{T} {a_{it}}^2/T\right)}{N}. \tag{30}$$

where N and T are the number of samples and total time of each trajectory, respectively. The data in Table 5 show that the guidance agent has impressive performance.

Evasion agent.

A. Training conditions and hyperparameters of the algorithm.

The initialization of the training environment conditions of the evasion agent is shown in Table 6. The relative distance between the missile and the interceptor d_i, the LOS angle of the interceptor λ_i, and the missile heading angle φ_m are randomly selected within a certain range at the initial state. Note that the value of the missile heading angle φ_m is still based on the LOS angle of the target λ_m. The initial velocity vector of the interceptor is pointed toward the missile. The speed of the missile v_m, speed of the interceptor v_i, and Cartesian coordinate values of the target and interceptor platform are constant values.

The hyperparameter values of the PPO algorithm are the same as those in Table 4 shown in Section "Guidance agent".

B. Training result and test.

The training reward curve of the evasive agent is shown in Fig. 10. The curve is close to convergence after training 200 episodes and approximately obtains the optimal state at the 500th episode.

To test the performance of the evasive agent, 100 initial states were randomly generated to form a test dataset, which is shown in Fig. 11a. Figure 11b shows the performance of the evasive agent on the test dataset. In each initial state, the agent can quickly make manoeuvres to prevent interception. Only the average energy consumption is used to evaluate the performance of the evasion agent. As shown in Table 7, the average energy

Parameter	Value
d_i	[4, 6] (km)
λ_i	$[\pi/3, \pi]$ (rad)
φ_m	$[\lambda_m - \pi/6,\ \lambda_m + \pi/6]$ (rad)
v_m	200 m/s
v_i	220 m/s
x_platform	5 (km)
y_platform	0 (km)
x_target	9 (km)
y_target	0 (km)

Table 6. Initial conditions in the training process of the evasion agent.

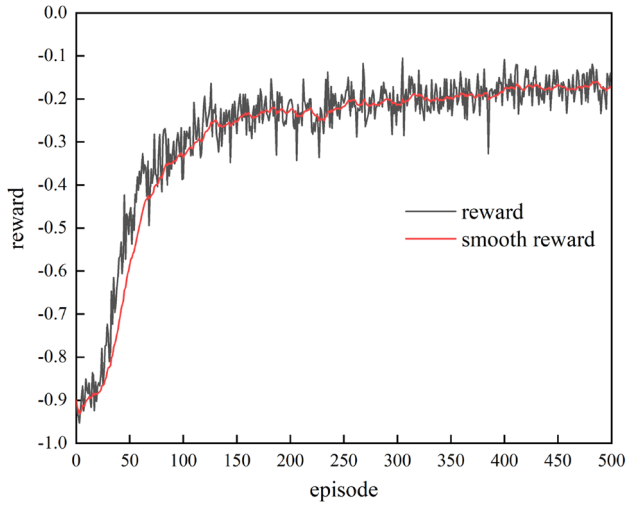
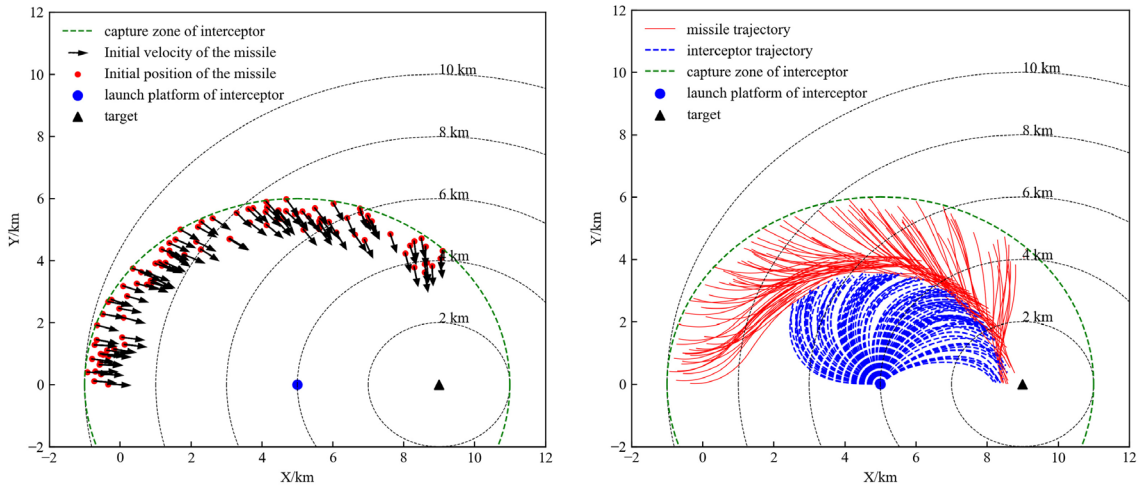


Figure 10. Episode reward in the training of the evasion agent.



(a) Test dataset of the evasion agent (b) Test result of the evasion agent

Figure 11. Test dataset and test result of the evasion agent.

Success rate	Average energy consumption/(m s^{-2})^2
100%	225.67

Table 7. Metric analysis of the evasion agent.

Parameter	Value
d_t	[9, 11] (km)
λ_m	$[-\pi/2, 0]$ (rad)
φ_m	$[\lambda_m - \pi/6, \lambda_m + \pi/6]$ (rad)
v_m	200 m/s
v_i	220 m/s
x_platform	5 (km)
y_platform	0 (km)
x_target	9 (km)
y_target	0 (km)

Table 8. Initial conditions in the training process of the high-level agent.

Parameter	Value
$n_{episode}$	100
n_{step}	10240
n_{reuse}	1
Clip	0.1
Gamma	0.995
Lamda	0.95
lr	0.0001
Minibatch	5120

Table 9. Hyperparameters of PPO algorithm.

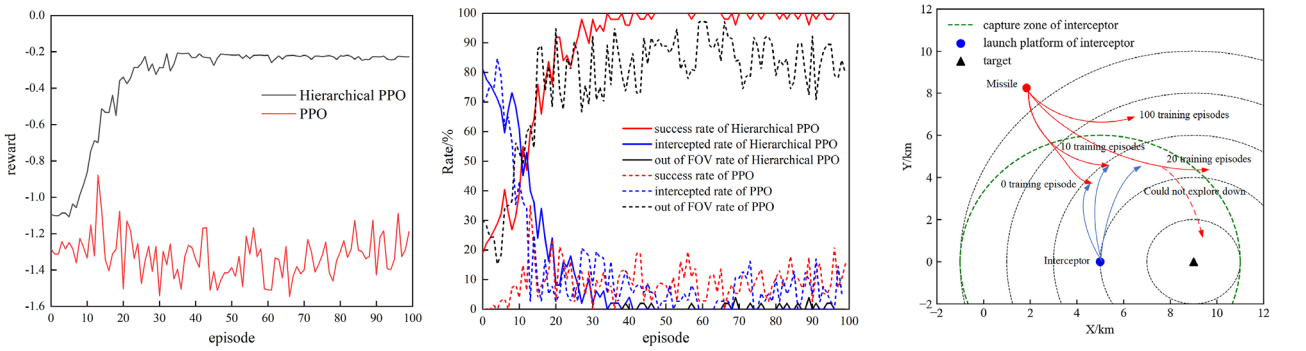
consumption of the evasion agent on the test dataset is 225.67(approximately 1.5 g), which is not a very large acceleration command for an evasive manoeuvre.

High-level agent training and testing. *Training conditions and hyperparameters of the algorithm.* Table 8 shows the initial training conditions for the policy selector at the high level.

The values of the hyperparameters of the PPO algorithm, which are shown in Table 9, are slightly different from those of the low-level agent.

It was found that the reward converges quickly because the action space of the high-level agent is discrete. However, if the convergence is too fast, the training will be unstable. For instance, the reward curve will quickly rise to a peak and then decrease slowly. For this reason, the hyperparameter n_{reuse} is set to a small value, and *minibatch* is set to a large value so the agent can smoothly converge to the optimal state.

Training result and test of the high-level agent. In this section, we compare our method with the PPO algorithm without a hierarchical structure. The comparison of training results is shown in Fig. 12. Figure 12a shows the average reward curve for the training process. The proposed hierarchical PPO algorithm has converged when training approximately 40 episodes, while the PPO algorithm without a hierarchical structure never converged. Figure 12b shows the curve of the probability of each termination state in the training process. The success rate of the hierarchical PPO gradually increased during the training process, and the intercepted rate gradually decreased. After 40 training episodes, the success rate reached 100%. In contrast, although the PPO algorithm without a hierarchical structure can also reduce the intercepted rate, the success rate was not improved. The agent converged to the state out of the FOV. Figure 12c shows the training process of the PPO without a hierarchical structure and explains why it fails to complete the task. During the training process, the agent was intercepted at the beginning and then learned to manoeuvre to evade the interceptor. However, the PPO algorithm could not guide the agent to perform a contrary action to explore down in the later training process. Therefore, the agent could not explore the successful terminal state, but converged to a state outside the FOV as soon as possible to reduce the penalty.

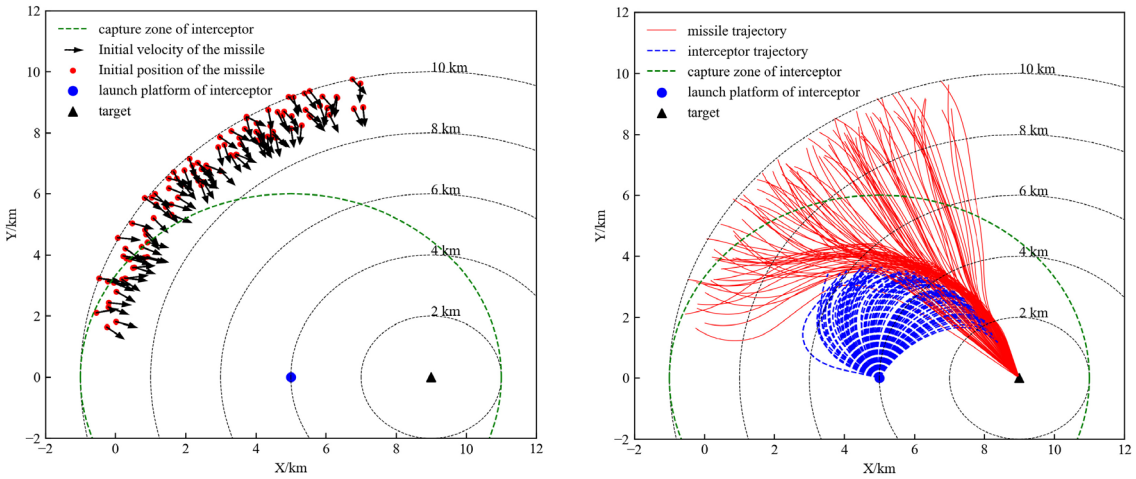


(a) Episode reward curve of the policy selector agent

(b) Curve of the terminal state rate

(c) Training process of the PPO without a hierarchical structure

Figure 12. (**a**) shows the episode reward and (**b**) shows the terminal state comparison of hierarchical PPO and PPO. (**c**) shows the training process of the PPO without a hierarchical structure and explains why it fails to complete the task.



(a) Test dataset of the policy selector agent

(b) Test result of policy selector agent

Figure 13. Test dataset and test result of the high-level agent.

Similarly, 100 initial states were randomly generated as a test dataset, which is shown in Fig. 13a. The performance of the agent on the test dataset is shown in Fig.13b, which indicates that the agent can complete the task of evading the interceptor and then reaching the target. Next, we analyse the performance of the agent with the time delay constant of the controller model.

Typical scenario analysis. In this section, we select six typical scenarios for detailed analysis. In the six typical scenarios, the initial distance of the missile to the target is 10km, the initial velocity vectors all point toward the target, and the azimuth angles relative to the target are $\frac{\pi}{12}$, $\frac{\pi}{6}$, $\frac{\pi}{4}$, $\frac{\pi}{3}$, $\frac{5\pi}{12}$, and $\frac{\pi}{2}$. The flight trajectories of the agents in the six typical scenarios are shown in Fig. 14.

First, we analyse the selection of the policy selector agent. The output of the policy selector agent in the six typical scenarios is shown in Fig. 15a–f. The guidance agent is selected when there is no interceptor in the scenario. The figure on the left shows the output of the policy selector agent in the six typical scenarios. 0 indicates the choice of the guided agent, and 1 indicates the choice of the evasive agent. The figure on the right shows the probability of choosing each agent in the six typical scenarios. The yellow curve shows the probability of choosing the evasion agent, and the blue curve shows the probability of choosing the guidance agent. The probability of choosing the evasion agent shows a downwards trend. The policy selector agent generally chooses the evasion agent to control the missile to conduct evasive manoeuvres immediately after the interceptor is launched. However, the probability of choosing the evasion agent may be greatly reduced at the end of the interceptor's flight time, indicating that the interception capability of the interceptor begins to decline, when the missile can be guided to the target in advance. Representative examples are scenarios 5 and 6. If in a highly threatening scenario, the policy selector agent may choose the evasion agent most of the time; otherwise, it will be intercepted.

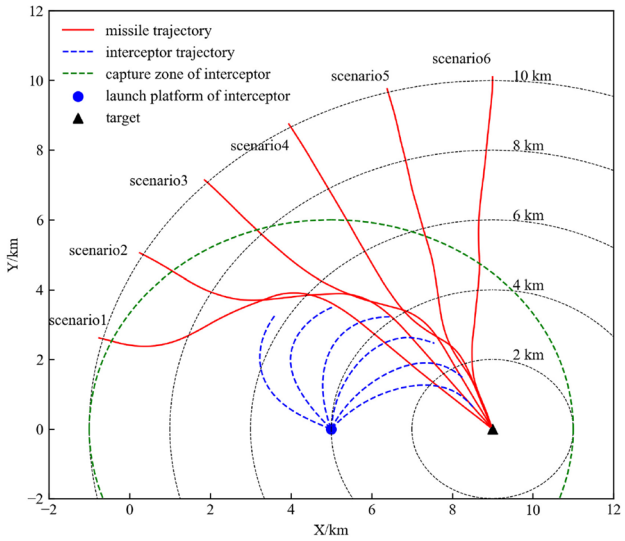


Figure 14. Results of the typical scenarios.

For instance, in scenario 1, the probability of choosing the evasion agent remained above 0.9 during the whole flight time.

Figure 16 shows the acceleration curves of the missile in six typical scenarios. This indicates that the missile rarely uses the acceleration limit value in the evasion process, but the maximum acceleration is often used immediately when the evasion process is completed. The purpose is to complete the task more quickly. The terminal accelerations of the missile are all close to 0 in scenarios 1 to 5, although no terminal acceleration constraints are set in the training algorithm. The reason for scenario 6 having a large terminal acceleration is that the missile is quite close to the target when completing the evasion task. Therefore, a violent manoeuvre must be carried out to guide to the target.

Robustness experiment. *Robustness of the autopilot parameters.* In this section, we study the robustness of the agent with the autopilot parameters. We add a parameter Δ to Formula (9) to simulate the uncertainty caused by aerodynamic parameter errors of the missile model[38,39]. Then the autopilot controller can be modelled as

$$\ddot{a_m} = -2\xi\omega_n\dot{a_m} - \omega_n^2 a_m + \omega_n^2 a_c + \omega_n^2\Delta. \tag{31}$$

The default values of parameters ξ, ω_n and Δ are 0.8, 8 rad/s, and 8sin(t), respectively. The experiment studies the robustness of the agent when ξ changes within the range [0.6, 1], ω_n changes within the range [8, 10], and Δ changes from 6sin(t) to 10sin(t). Table 10 shows the experimental results. In Table 10, the success rates are all 100% except for the state when $\xi = 1$, which indicates that the agent has good robustness. Figure 17 shows the trend of the metrics when the values of the parameters change.

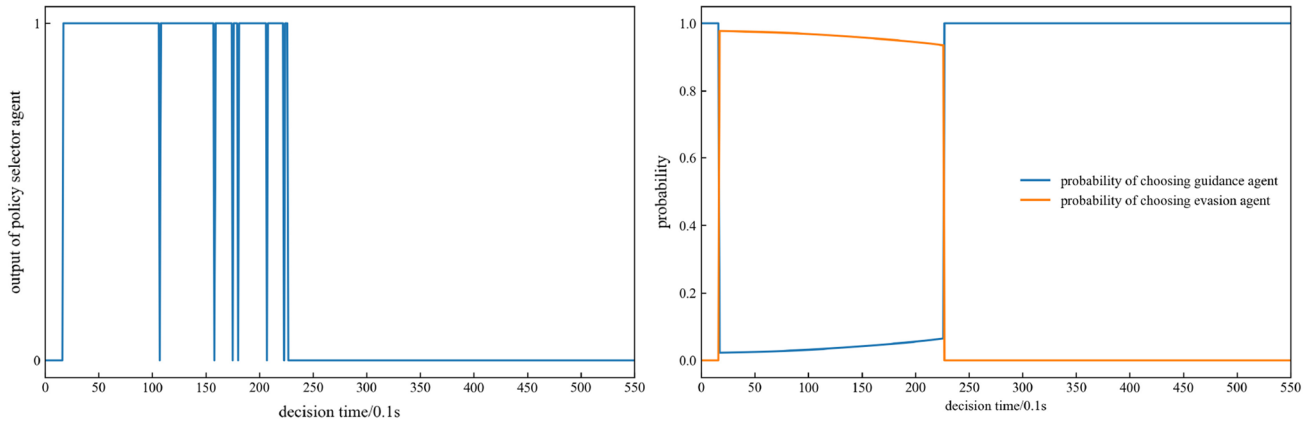
Figure 17a shows that the parameters Δ and ω_n can affect the guidance accuracy, while ξ has little effect. The guidance accuracy becomes worse as Δ increases. In contrast, the guidance accuracy becomes better as ω_n increases. Figure 17b shows that ω_n is the main factor that affects the time consumption. Similar to the ZEM, the time consumption becomes better when ω_n increases, and ξ and Δ have little effect on the time consumption. Figure 17c shows that the energy consumption is affected by all three parameters. The energy consumption becomes worse as Δ and ω_n increase. However, the energy consumption will decrease as ξ increases.

Robustness of the information errors. In a realistic engagement scenario, the information is always combined with errors. We study the robustness of the agent considering information errors in this section. The error of the radar information is usually described by the mean squared error (MSE), which is shown as

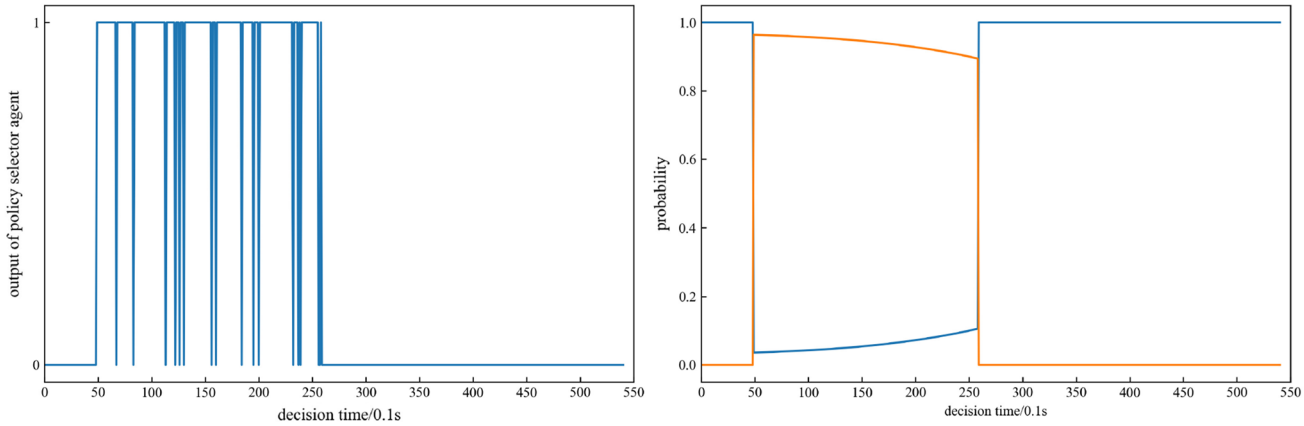
$$MSE = \sum_{i=1}^{n} \left| \frac{observation(i) - observation_{real}(i)}{observation_{real}(i)} \right| \Big/ n\,. \tag{32}$$

Figure 18 shows the experimental results of the performance of the agent with different information MSE. The success rate remains above 99% until the MSE exceeds 10%, and above 98% until the MSE exceeds 25%, which indicates that the agent has impressive robustness to information errors. However, the efficiency of the agent will decrease as the MSE increases, which can be seen in Fig. 18b, c and d.

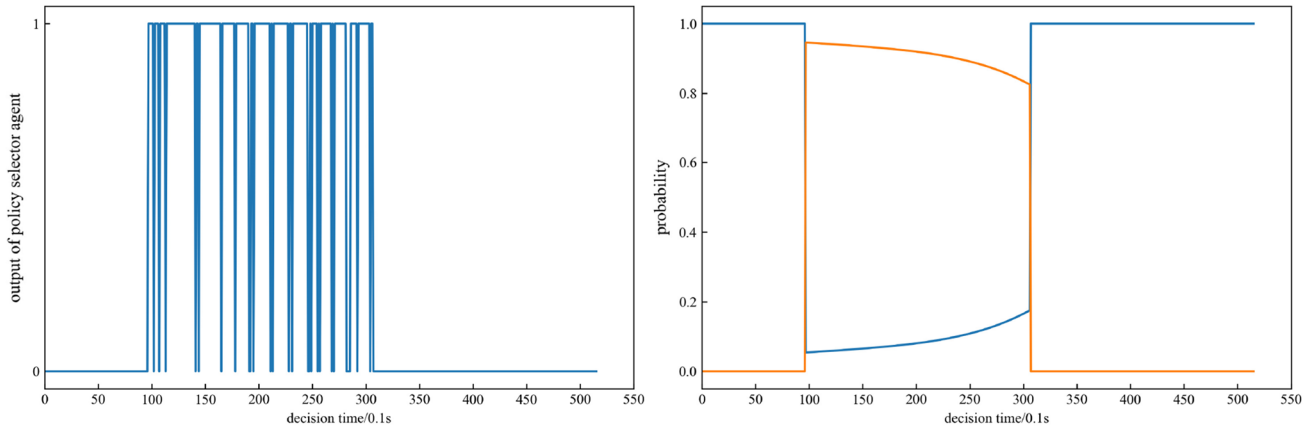
Comparison experiment. Finally, this paper compares the proposed hierarchical PPO guidance method with traditional guidance methods based on[6]. A guidance synthesis was designed for this combat scenario in[6].



(a) Selection result and probability in scenario 1



(b) Selection result and probability in scenario 2

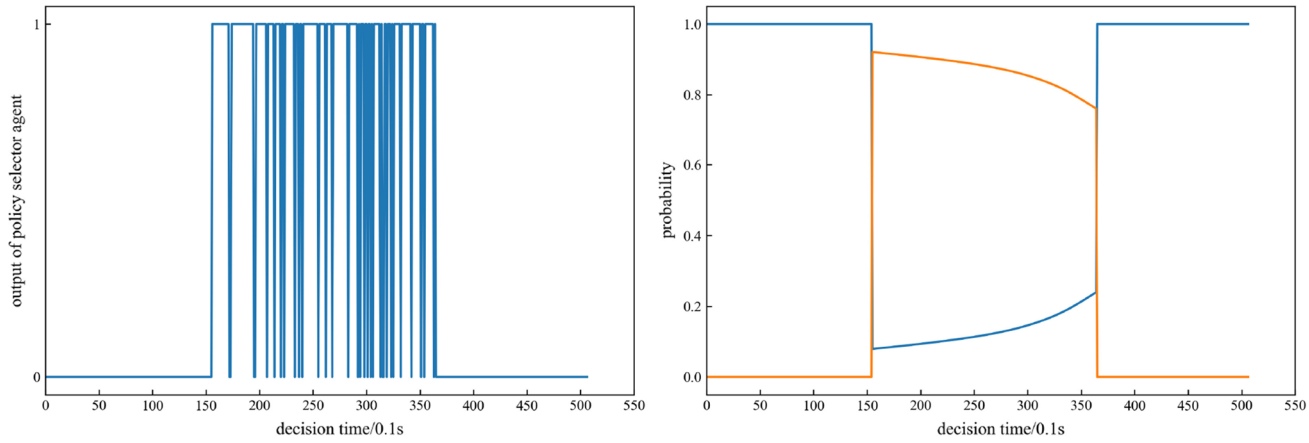


(c) Selection result and probability in scenario 3

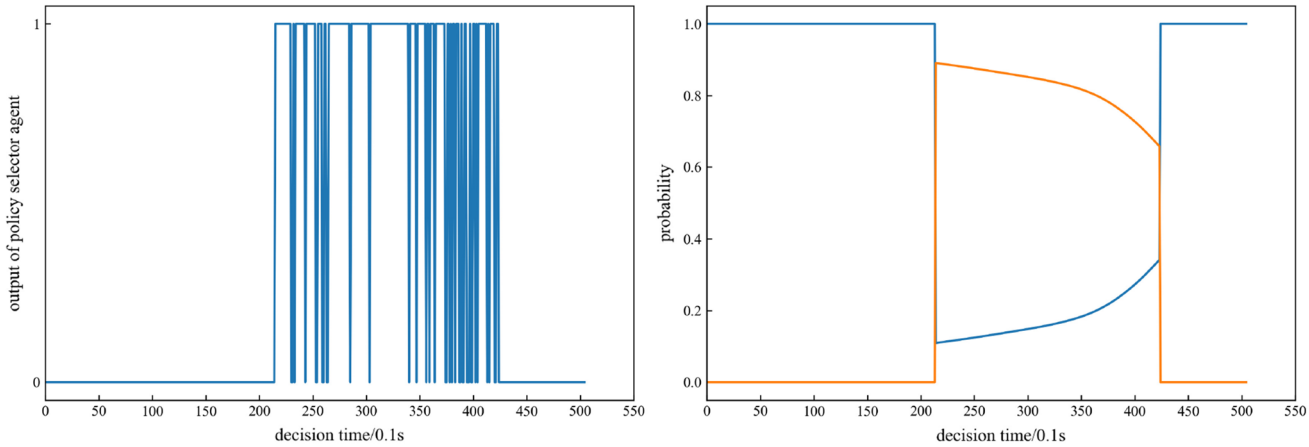
Figure 15. Policy selection analysis for each typical scenario.

The acceleration command of the missile consists of three parts: an artificial potential field(APF), polynomial guidance function, and logarithmic barrier function(LBF). The artificial potential field generates evasive acceleration commands. The polynomial guidance function generates guidance acceleration commands considering the impact angle constraints and terminal acceleration constraints. The logarithmic barrier function controls the missile so that it does not exceed the FOV of the missile. Since the impact angle and terminal acceleration constraints are not considered in this study, the polynomial guidance is replaced by a proportional guidance method with N=3. Therefore, the guidance synthesis for comparison in this paper is composed of an artificial potential field, PNG3, and logarithmic barrier function.

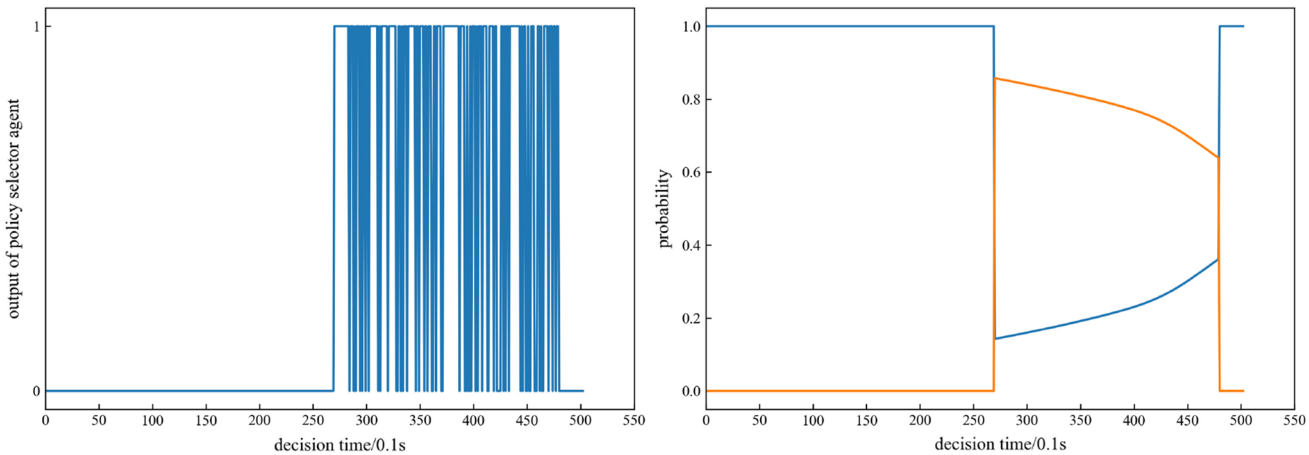
The artificial potential field function is formulated as:



(d) Selection result and probability in scenario 4



(e) Selection result and probability in scenario 5



(f) Selection result and probability in scenario 6

Figure 15. (continued)

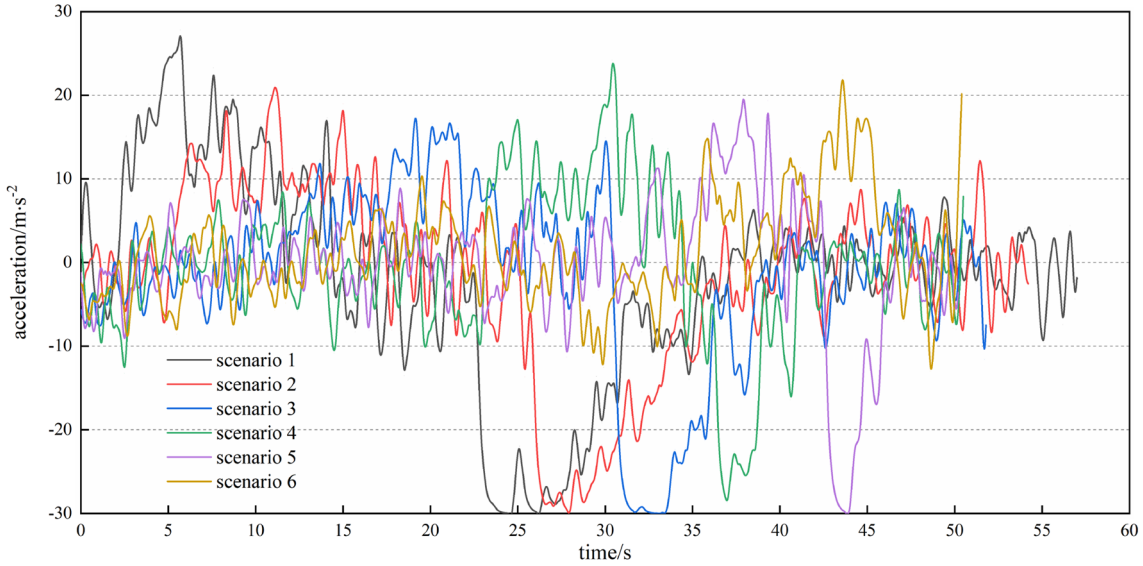
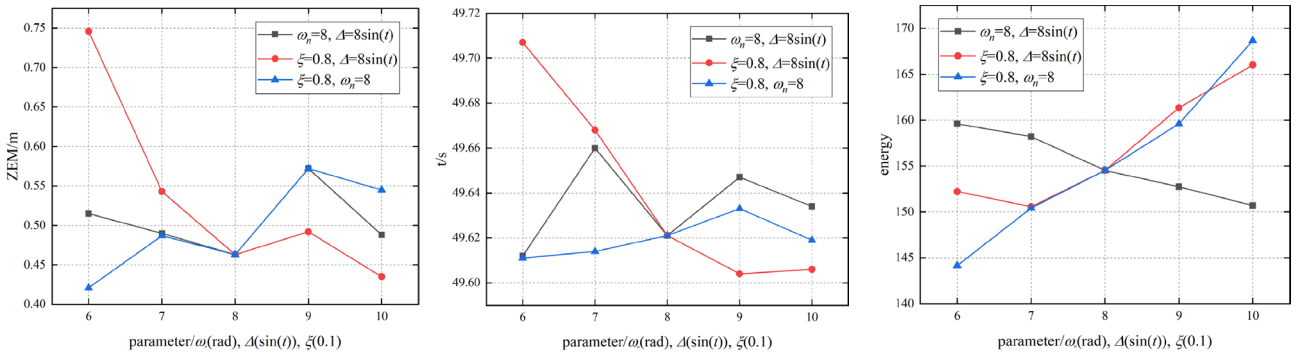


Figure 16. Acceleration analysis for each typical scenario.

Parameter values			Success rate/%	ZEM*/m	Average time/s	Average energy consumption/(m s^{-2})^2
ξ	ω_n	Δ				
0.8	8	8sin(t)	100	0.463	49.621	154.5247
0.6	8	8sin(t)	100	0.515	49.612	159.5974
0.7	8	8sin(t)	100	0.490	49.660	158.2002
0.9	8	8sin(t)	100	0.572	49.647	152.7397
1.0	8	8sin(t)	99	0.488	49.634	150.7082
0.8	6	8sin(t)	100	0.746	49.707	152.2261
0.8	7	8sin(t)	100	0.543	49.668	150.5717
0.8	9	8sin(t)	100	0.492	49.604	161.3431
0.8	10	8sin(t)	100	0.435	49.606	166.0081
0.8	8	9sin(t)	100	0.421	49.611	144.1569
0.8	8	10sin(t)	100	0.487	49.614	150.4246
0.8	8	11sin(t)	100	0.572	49.633	159.5985
0.8	8	12sin(t)	100	0.545	49.619	168.6634

Table 10. Autopilot parameter analysis.

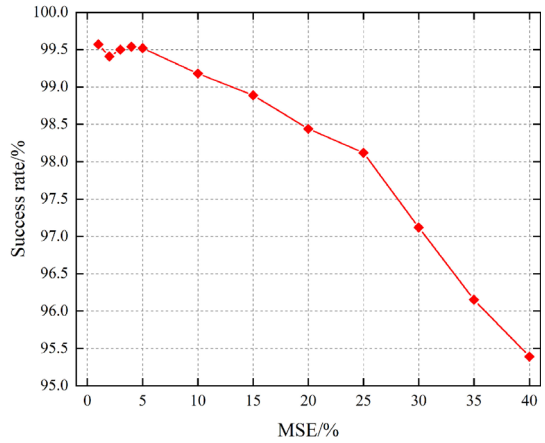


(a) ZEM* with different parameters

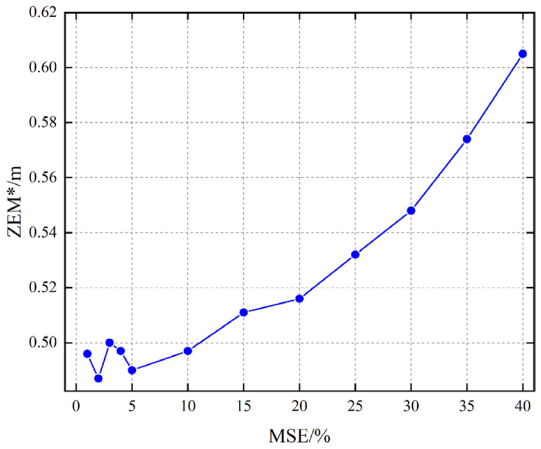
(b) Average time with different parameters

(c) Energy consumption with different parameters

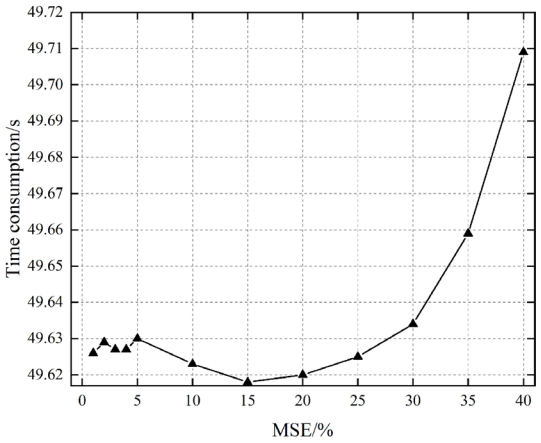
Figure 17. Autopilot parameter analysis.



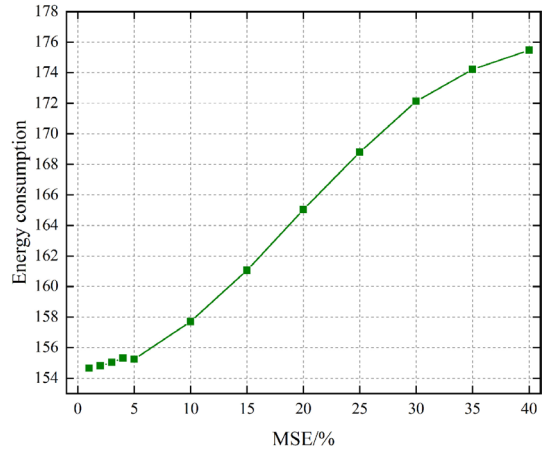
(a) Success rate with different information MSE



(b) ZEM* with different information MSE



(c) Average time with different information MSE



(d) Energy consumption with different information MSE

Figure 18. The performance of the agent with different information MSEs.

$$a_{APF} = \begin{cases} F_{\text{repObst}}\mathbf{n}_{\text{obst}} + F_{\text{repGoal}}\mathbf{n}_{\text{goal}}, & \text{if } d_{\text{obst}} \le d_0 \\ 0, & \text{otherwise} \end{cases} \tag{33}$$

$$F_{\text{repObst}} = \varepsilon\zeta d_{\text{goal}}\, e^{-\zeta d_{\text{obst}}} \tag{34}$$

$$F_{\text{repGoal}} = \varepsilon e^{-\zeta d_{\text{obst}}} \tag{35}$$

where $\mathbf{n}_{\text{obst}}$ and $\mathbf{n}_{\text{goal}}$ are the unit vectors of the interceptor pointing to the missile and the missile pointing to the target, respectively; d_o is the distance of the interceptor capture zone; d_{obst} and d_{goal} are the missile-interceptor relative distance and missile-target relative distance, respectively; ε and ζ are hyperparameters.

The PNG3 is formulated as:

$$a_{PNG} = -NV_m\dot{\lambda_m} \tag{36}$$

where N = 3.

The logarithmic barrier function is formulated as:

$$a_{LBF} = \begin{cases} a_{\text{lim}} & \text{if } \lambda \ge b \\ \mu \log(b - |\lambda|) & \text{if } b > \lambda \ge \left(b - 10^{\circ}\right) \\ 0 & \text{if } \lambda < \left(b - 10^{\circ}\right) \end{cases} \tag{37}$$

where b is the maximum FOV angle, μ is a hyperparameter, and a_{lim} is the maximum acceleration value.

The acceleration synthesis is:

Method	Average ZEM*/m	Average time/s	Average energy consumption/(m s^{-2})^2
Hierarchical PPO	0.441	49.485	153.76
Traditional method	0.428	54.934	161.24

Table 11. Comparison of the hierarchical PPO and Synthesis guidance laws.

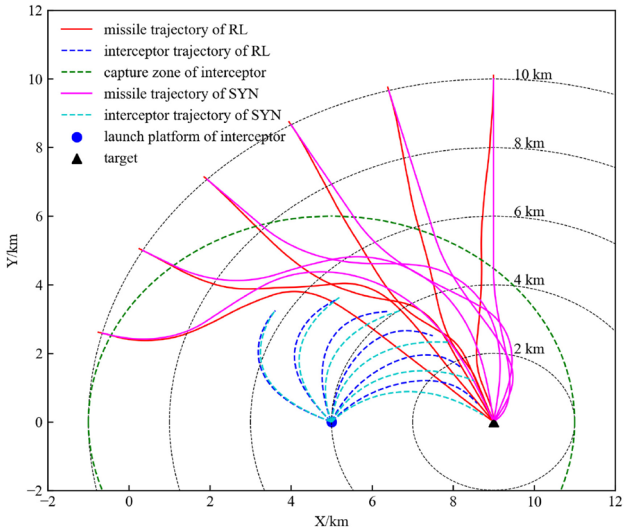


Figure 19. Flightpath comparison of the hierarchical PPO guidance law and synthesis guidance law.

$$a_{SYN} = a_{APF} + a_{PNG} + a_{LBF}. \tag{38}$$

where ε, ζ and μ are hyperparameters in guidance synthesis. The values of the hyperparameters are$\mu = 5, \varepsilon = 40$ and $\zeta = 3e^{-4}$, which ensures that the success rate of the guidance synthesis on the test dataset is 100%.

Since the actions of reinforcement learning agents have a certain randomness, we conducted 100 experiments to compare the performance of the reinforcement learning method and the traditional method. Table 11 shows the results of the comparison experiment. It seems that the traditional method is better for guidance accuracy, but the gap is less than 0.02 m and can be ignored. The guidance accuracy of the reinforcement learning method and the traditional method is good enough. Most importantly, the improvements in flight time and energy consumption of reinforcement learning methods are significant.

Figure 19 shows the comparison of the missile flight trajectories of the two guidance methods in the six typical scenarios described in Section "High-level agent training and testing". In the first four scenarios, the missile flight trajectory generated by the guidance law based on hierarchical reinforcement learning is significantly better than the trajectory generated by the synthesis guidance law. The former can concisely complete evasion tasks and quickly guide to the target, while the latter requires a longer flight path. There is an interesting comparison in scenario 5 and scenario 6. The two guidance methods output acceleration commands in opposite directions when evading the interceptor. The synthesis guidance law controls the missile to move away from the interceptor, which is the typical idea. In contrast, the RL-based guidance law first controls the missile to move towards the interceptor for a period of time, and then conducts violent manoeuvres in the opposite direction. Both methods completed the task of evading the interceptor and reaching the target. The RL-based guidance law requires less time. Although it may seem "risky", this is the advantage of the reinforcement learning guidance law – the agent can find a better solution than experience and knowledge.

Conclusion

We study the application of reinforcement learning in missile maneuvering. We propose a hierarchical structure and analyse its effectiveness and robustness. Therefore, our research presentation has high professionalism. The conclusions include the following three points:

(1) The method based on hierarchical reinforcement learning can complete the task of evading an interceptor and guiding to the target, with a success rate of 100% on a test dataset. The agent performs with great robustness with autopilot parameters and information errors. As the disturbance of the autopilot increases,the metrics of the agent decrease, but it can still almost ensure that the task is completed with a 100% success rate. In an information error experiment, the success rate remained above 99% until the MSE

exceeded 10%, and above 98% until the MSE exceeded 25%, which indicates that the agent has impressive robustness to information errors.

(2) The PPO algorithm without a hierarchical structure cannot complete the task. Although the agent can learn to not be intercepted, it cannot converge to a state of successfully guiding to the target. The agent finally converges to the state outside the FOV. The method based on hierarchical reinforcement learning can solve this problem.

(3) The performance of the proposed reinforcement learning method is better than that of the traditional method, especially considering the average time and average energy consumption metrics. The traditional method has slight advantages in guidance accuracy, while the guidance law based on hierarchical reinforcement learning also has satisfactory guidance accuracy.

In future work, it is necessary to focus on the application of reinforcement learning in the environment of incomplete information and unknown adversaries to improve the robustness and transferability of reinforcement learning agents.

Data availability

The datasets and code used and/or analysed during the current study available from the corresponding author on reasonable request.

Received: 10 June 2022; Accepted: 30 September 2022
Published online: 07 November 2022

References

1. Guo, H., Fu, W., Fu, B., Chen, K. & Yan, J. Smart homing guidance strategy with control saturation against a cooperative target-defender team. *J. Syst. Eng. Electron.* **30**, 366–383 (2019).
2. Shalumov, V. Online launch-time selection using deep learning in a target-missile-defender engagement. *J. Aerosp. Inf. Syst.* **16**, 224–236 (2019).
3. Shi, H., Chen, Z., Zhu, J. & Kuang, M. Model predictive guidance for active aircraft protection from a homing missile. *IET Control Theory Appl.* **16**, 208–218 (2022).
4. Shalumov, V. Cooperative online Guide-Launch-Guide policy in a target-missile-defender engagement using deep reinforcement learning. *Aerosp. Sci. Technol.* **104**, 105996 (2020).
5. Ryoo, C. K., Whang, I. H. & Tahk, M. J. 3-D evasive maneuver policy for anti-ship missiles against close-in weapon systems. In *AIAA Guid. Navig. Control Conf. Exhib.* (2003).
6. Yogaswara, Y. H., Hong, S. M., Tahk, M. J. & Shin, H. S. Impact angle control guidance synthesis for evasive maneuver against intercept missile. *Int. J. Aeronaut. Sp. Sci.* **18**, 719–728 (2017).
7. Qi, N., Sun, Q. & Zhao, J. Evasion and pursuit guidance law against defended target. *Chin. J. Aeronaut.* **30**, 1958–1973 (2017).
8. Sun, Q., Zhang, C., Liu, N., Zhou, W. & Qi, N. Guidance laws for attacking defended target. *Chin. J. Aeronaut.* **32**, 2337–2353 (2019).
9. Liang, H., Wang, J., Wang, Y., Wang, L. & Liu, P. Optimal guidance against active defense ballistic missiles via differential game strategies. *Chin. J. Aeronaut.* **33**, 978–989 (2020).
10. Weiss, M., Shima, T., Castaneda, D. & Rusnak, I. Combined and cooperative minimum-effort guidance algorithms in an active aircraft defense scenario. *J. Guid. Control Dyn.* **40**, 1241–1254 (2017).
11. Turetsky, V. & Shima, T. Target evasion from a missile performing multiple switches in guidance law. *J. Guid. Control Dyn.* **39**, 2364–2373 (2016).
12. Fonod, R. & Shima, T. Multiple model adaptive evasion against a homing missile. *J. Guid. Control Dyn.* **39**, 1578–1592 (2016).
13. Keong, C. W., Shin, H. S. & Tsourdos, A. Reinforcement learning for autonomous aircraft avoidance. In *2019 Int. Work. Res. Educ. Dev. Unmanned Aer. Syst.* 126–131 (2019).
14. Lee, G. T. & Kim, C. O. Autonomous control of combat unmanned aerial vehicles to evade surface-to-air missiles using deep reinforcement learning. *IEEE Access* **8**, 226724–226736 (2020).
15. Wang, X., Shi, P., Schwartz, H. & Zhao, Y. An algorithm of pretrained fuzzy actor-critic learning applying in fixed-time space differential game. *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.* **235**, 2095–2112 (2021).
16. Gaudet, B. & Furfaro, R. Missile homing-phase guidance law design using reinforcement learning. In *AIAA Guid. Navig. Control Conf.* (2012).
17. Gaudet, B., Furfaro, R. & Linares, R. Reinforcement learning for angle-only intercept guidance of maneuvering targets. *Aerosp. Sci. Technol.* **99**, 105746 (2020).
18. Gaudet, B., Drozd, K., Meltzer, R. & Furfaro, R. Adaptive approach phase guidance for a hypersonic glider via reinforcement meta learning. In *AIAA SCITECH 2022 Forum* (2022).
19. Gaudet, B., Way, E. R. & Arizona, T. Terminal adaptive guidance for autonomous hypersonic strike weapons via reinforcement learning. arXiv preprint arXiv:2110.00634 (2021).
20. Hong, D., Kim, M. & Park, S. Study on reinforcement learning-based missile guidance law. *Appl. Sci.* **10**, 6567 (2020).
21. He, S., Shin, H. S. & Tsourdos, A. Computational missile guidance: A deep reinforcement learning approach. *J. Aerosp. Inf. Syst.* **18**, 571–582 (2021).
22. Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D. & Wierstra, D. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971 (2016).
23. Jiang, L., Nan, Y. & Li, Z. H. Realizing midcourse penetration with deep reinforcement learning. *IEEE Access* **9**, 89812–89822 (2021).
24. Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M. & De Frcitas, N. Dueling network architectures for deep reinforcement learning. In *33rd Int. Conf. Mach. Learn. ICML* 2939–2947 (2016).
25. Pope, A. P., Ide, J. S., Micovic, D., Diaz, H., Rosenbluth, D., Ritholtz, L., Twedt, J. C., Walker, T. T., Alcedo, K. & Javorsek, D. Hierarchical reinforcement learning for air-to-air combat. In *2021 Int. Conf. Unmanned Aircr. Syst. ICUAS* 275–284 (2021).
26. Sun, Z. *et al.* Multi-agent hierarchical policy gradient for Air Combat Tactics emergence via self-play. *Eng. Appl. Artif. Intell.* **98**, 104112 (2021).
27. Zhou, W. J., Subagdja, B., Tan, A. H. & Ong, D. W. S. Hierarchical control of multi-agent reinforcement learning team in real-time strategy (RTS) games. *Expert Syst. Appl.* **186**, 115707 (2021).
28. Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017).

29. Li, H., Li, H. & Cai, Y. Efficient and accurate online estimation algorithm for zero-effort-miss and time-to-go based on data driven method. *Chin. J. Aeronaut.* **32**, 2311–2323 (2019).
30. Sirin, E., Parsia, B., Wu, D., Hendler, J. & Nau, D. HTN planning for web service composition using SHOP2. *J. Web Semant.* **1**, 377–396 (2004).
31. Dayan, P. & Hinton, G. E. Feudal reinforcement learning. In *Advances in Neural Information Processing Systems* (eds Hanson, S., Cowan, J. & Giles, C.) (1992).
32. Barto, A. G. & Mahadevan, S. Recent advances in hierarchical reinforcement learning. *Discrete Event Dyn. Syst.* **13**, 41–77 (2003).
33. Comanici, G. & Precup, D. Optimal policy switching algorithms for reinforcement learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems* 709–714 (2010).
34. Frans, K., Ho, J., Chen, X., Abbeel, P. & Schulman, J. Meta learning shared hierarchies. In *6th International Conference on Learning Representations* (2018).
35. Cobbe, K., Hilton, J., Klimov, O. & Schulman, J. Phasic policy gradient. In *International Conference on Machine Learning, Virtual* 2020-2027 (2021).
36. Haarnoja, T., Ha, S., Zhou, A., Tan, J., Tucker, G., Levine, S. Learning to walk via deep reinforcement learning. arXiv preprint arXiv:1812.11103 (2018).
37. Kingma, D. P. & Ba, J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).
38. Chwa, D. Y. & Choi, J. Y. Adaptive nonlinear guidance law considering control loop dynamics. *IEEE Trans. Aerosp. Electron. Syst.* **39**, 1134–1143 (2003).
39. Chwa, D. Y., Choi, J. Y. & Anavatti, S. G. Observer-based adaptive guidance law considering target uncertainties and control loop dynamics. *IEEE Trans. Control Syst. Technol.* **14**, 112–123 (2006).

Acknowledgements

We would like to thank everyone who provided comments on this work.

Author contributions

M.Y. contributed to the investigation, methodology, experiment and writing of the research; R.Y. and Y.Z. contributed to the conceptualization, analysis and revision of the research. L.Y. and D.H. contributed to the experiment and analysis of the research. All authors of this research paper have read and approved the final version submitted.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to M.Y.

Reprints and permissions information is available at www.nature.com/reprints.

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