Journal of Systems Engineering and Electronics
Vol. 35, No. 4, August 2024, pp.1042 – 1052

Computational intelligence interception guidance law using online off-policy integral reinforcement learning

WANG Qi* and LIAO Zhizhong

China Airborne Missile Academy, Luoyang 471009, China

Abstract: Missile interception problem can be regarded as a two-person zero-sum differential games problem, which depends on the solution of Hamilton-Jacobi-Isaacs (HJI) equation. It has been proved impossible to obtain a closed-form solution due to the nonlinearity of HJI equation, and many iterative algorithms are proposed to solve the HJI equation. Simultaneous policy updating algorithm (SPUA) is an effective algorithm for solving HJI equation, but it is an on-policy integral reinforcement learning (IRL). For online implementation of SPUA, the disturbance signals need to be adjustable, which is unrealistic. In this paper, an off-policy IRL algorithm based on SPUA is proposed without making use of any knowledge of the systems dynamics. Then, a neural-network based online adaptive critic implementation scheme of the off-policy IRL algorithm is presented. Based on the online off-policy IRL method, a computational intelligence interception guidance (CIIG) law is developed for intercepting high-maneuvering target. As a model-free method, intercepting targets can be achieved through measuring system data online. The effectiveness of the CIIG is verified through two missile and target engagement scenarios.

Keywords: two-person zero-sum differential games, Hamilton–Jacobi–Isaacs (HJI) equation, off- policy integral reinforcement learning (IRL), online learning, computational intelligence interception guidance (CIIG) law.

DOI: 10.23919/JSEE.2024.000067

1. Introduction

Precision guided weapons have the advantages of high guidance accuracy, great lethality and high combat effectiveness, among which guidance accuracy is the underlying fundamental issue. Since the birth of guided weapons, theoretical exploration and applied research have been carried out on various guidance laws. For example, ideal proportional navigation guidance (IPNG), generalized proportional navigation guidance (GPNG), and realistic true proportional navigation guidance (RTPNG). Such proportional navigation guidance (PNG)-based control laws and their variations are easy to implement in engineering and have been applied extensively in actual missile combat missions [1]. As is known, it is more difficult to intercept maneuvering targets than non-maneuvering targets and stationary targets. In order to improve the performance of PNG-based guidance law in intercepting maneuvering targets, augmented PNG (APNG), linear quadratic optimal guidance (LQOG) [2], adaptive sliding mode guidance (ASMG) [3], backstepping guidance law [4] and adaptive dynamic surface guidance (ADSG) [5,6] were derived, respectively. These guidance laws are all closed-form guidance laws, which are investigated using model-based methods. First, establish the mathematical model of the guidance, then solve the guidance problem through modern control theories and technologies, and consequently obtain the closed-form guidance laws.

In addition to the analytic guidance law mentioned above, some numerical optimization algorithms are also used to solve the guidance problem, such as model predictive static programming (MPSP). Dwivedi et al. used the MPSP algorithm to calculate the suboptimal solution of trajectory optimization in midcourse guidance [7,8], reentry ballistic guidance law of reusable space vehicle [9], impact-angle constraint suboptimal guidance [10], and unmanned aerial vehicle (UAV) autonomous landing [11]. The MPSP algorithm is essentially a Newton iterative algorithm [12]. The guidance laws based on numerical algorithms also need to establish the mathematical model of guidance problem, and then solve optimal guidance problem iteratively.

In actual engineering application, it is usually intractable to obtain the accurate mathematical models of complicated engineering systems, therefore, the model-based methods cannot achieve the optimum performance when the approximate models are used. In recent years, there has been a new and accelerated paradigm shift trend in the communities of aerospace guidance and control, where the amount of computation far exceeds the basic algebraic operations required to evaluate model-based

Manuscript received October 31, 2022.

*Corresponding author.

guidance and control laws, this led to the emerging concept of computational guidance law (CGL) [13]. The distinguishing trademark of CGL is that it relies heavily on real-time on-board data sampling and numerical computing operations to obtain guidance instructions, thereby eliminating the need for system modeling, gain adjustment, important pre-mission planning, or extensive offline design [14]. Generating guidance instructions through data and computation often requires iterations. Therefore, in dynamically evolving scenarios, how to use sampled data to generate online guidance instructions efficiently, reliably, and robustly is not trivial, which is still an open issue.

Different from model-dependent guidance laws, the design of data-driven CGLs always involve machine learning technologies. Data-driven computational guidance algorithm can be implemented through offline training or online learning, then the implicit mapping relationships between the guidance instructions and the training datasets are learned. As an interdisciplinary subject in the communities of artificial intelligence and control, reinforcement learning (RL) which is supposed to get the approximate solution of the control system optimization problem by using sampled data, has been broadly introduced and developed in the field of artificial intelligence and machine learning. The problem-solving framework of RL includes value iteration (VI) algorithm and policy iteration (PI) algorithm, which first appeared in the solution methods of Markov decision problem [15]. The approximate solutions of H_2 and H_∞ control problems can be obtained by using VI or PI algorithm through a random or directed exploration in the control strategy space [16,17]. In recent years, RL-based missile interception guidance laws have attracted increasing interest, but most of the algorithms use offline training or partial systems dynamics is required. In [18–21], RL-based guidance law was proposed for missile homing-phase interception guidance problem, which used observations consisting solely of line-of-sight (LOS) rate, and the above-mentioned guidance laws based on RL algorithm were realized by offline training. The approximate optimal guidance laws and cooperative guidance laws based on online adaptive dynamic programming were introduced in [22–26], where the guidance laws were solved using neural network (NN) based online PI algorithms. But these algorithms are on-policy RL methods, in which the value function is evaluated by employing sampled data generated by evaluating policies [27], and the systems dynamics is required.

Off-policy integral RL (IRL) was first introduced for solving control optimization problem of nonlinear continuous-time dynamic system in [28,29]. Unlike on-policy RL algorithm, the optimal control policy and the value function are learned by employing data generated by off-policy sampling, and the system dynamics is completely unknown. In [30,31], the offline off-policy IRL method was employed to learn the solution of Hamilton-Jacobi-Isaacs (HJI) equation related with the H_∞ optimal control problem of nonlinear system with unknown internal system model.

In order to realize a completely model-free, data-driven guidance technology for intercepting high-maneuvering target, a computational intelligence interception guidance (CIIG) law using online off-policy IRL algorithm is introduced in this paper. A linear-in-parameter NN structure is built to solve the approximate Nash equilibrium solution of the missile-target interception two-person zero-sum differential games problem without using any knowledge of the system dynamics. Interception problem can be solved through online data sampling. The organization of this paper is as follows. The general two-person zero-sum differential games problem statement, simultaneous policy updating algorithm (SPUA) algorithm, and off-policy IRL method are presented in Section 2. In Section 3, the engagement geometry of missile and target is given, and then, the CIIG law for intercepting high-maneuvering target is presented. Two engagement scenarios of missile and target are provided in Section 4. Section 5 concludes the paper.

2. Nonlinear zero-sum differential game and off-policy IRL algorithm

2.1 Problem formulation and preliminary

In this section, the background knowledge review and preliminary results of two-person zero-sum differential games are provided. Consider a class continuous-time nonlinear affine in control input game system defined as

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u} + \boldsymbol{k}(\boldsymbol{x})\boldsymbol{w} \tag{1}$$

where $\boldsymbol{x} \in \mathbf{R}^n$ is the system state vector, $\boldsymbol{u} \in \mathbf{R}^m$ and $\boldsymbol{w} \in \mathbf{R}^q$ represent control input and disturbance input of the two participants in the game, respectively. $\boldsymbol{f}(\boldsymbol{x}) \in \mathbf{R}^n$ is internal dynamics, $\boldsymbol{g}(\boldsymbol{x}) \in \mathbf{R}^{n\times m}$ and $\boldsymbol{k}(\boldsymbol{x}) \in \mathbf{R}^{n\times q}$ are input-to-state matrix and disturbance coefficient matrix, respectively. On a compact set $\boldsymbol{\Omega} \subset \mathbf{R}^n$, $\boldsymbol{f}(\boldsymbol{x})$, $\boldsymbol{g}(\boldsymbol{x})$, and $\boldsymbol{k}(\boldsymbol{x})$ are locally Lipschitz continuous and $\boldsymbol{f}(0) = 0$, that is, $\boldsymbol{x} = 0$ is a system equilibrium point. Assume that $\boldsymbol{u}(t) \in L_2[0,\infty)$ and $\boldsymbol{w}(t) \in L_2[0,\infty)$.

For nonlinear game system in (1), define the following infinite-horizon performance index function.

$$J(\boldsymbol{x}_0,\boldsymbol{u},\boldsymbol{w}) = \int_0^\infty \left(Q(\boldsymbol{x}) + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u} - \gamma^2\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}\right)\mathrm{d}t = \int_0^\infty r(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w})\,\mathrm{d}t \tag{2}$$

where utility function $r(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w}) = Q(\boldsymbol{x}) + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u} - \gamma^2\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}$ with $Q(\boldsymbol{x}) \geqslant 0$, $\boldsymbol{R} > 0$, and $\gamma > 0$, assume $\boldsymbol{R}$ is a diagonal matrix. Game theory addresses the problem of policy interaction between the two participants, each of which has an objective contained in a cost function that the participants try to minimize or maximize. In two-person zero-sum differential games, the objective of participant $\boldsymbol{u}$ is to minimize the performance functional, whereas the objective of player $\boldsymbol{w}$ is to maximize it, that is

$$V^*(\boldsymbol{x}_0) = \min_{\boldsymbol{u}} \max_{\boldsymbol{w}} J(\boldsymbol{x}_0,\boldsymbol{u},\boldsymbol{w}) = \min_{\boldsymbol{u}} \max_{\boldsymbol{w}} \int_0^{\infty} \left(Q(\boldsymbol{x}) + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u} - \gamma^2\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}\right)\mathrm{d}t. \tag{3}$$

It is worth noting that this problem is equivalent to a suboptimal H_∞ control problem for some prescribed $\gamma > 0$, and the system L_2-gain is less than or equal to γ [32]. Define a value or cost function for the policies of the two players as

$$V^{\boldsymbol{u},\boldsymbol{w}}(\boldsymbol{x}(t)) = \int_t^{\infty} \left(Q(\boldsymbol{x}) + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u} - \gamma^2\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}\right)\mathrm{d}t. \tag{4}$$

Then the Hamiltonian function associated with the value function is

$$H(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w},\nabla V) = Q(\boldsymbol{x}) + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u} - \gamma^2\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + \nabla V^{\mathrm{T}}(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u} + \boldsymbol{k}(\boldsymbol{x})\boldsymbol{w}) \tag{5}$$

where $\nabla V = \partial V / \partial \boldsymbol{x}$. Appling the stationarity conditions on (5) with V^*, one gets

$$\begin{cases} \dfrac{\partial H(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w},\nabla V^*)}{\partial \boldsymbol{u}} = 0 \\ \dfrac{\partial H(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w},\nabla V^*)}{\partial \boldsymbol{w}} = 0 \end{cases}. \tag{6}$$

One can get the optimal control inputs of the two participants for the game problem, that are

$$\boldsymbol{u}^* = -\frac{1}{2}\boldsymbol{R}_1^{-1}\boldsymbol{g}^{\mathrm{T}}\frac{\partial V^*}{\partial \boldsymbol{x}}, \tag{7}$$

$$\boldsymbol{w}^* = \frac{1}{2\gamma^2}\boldsymbol{k}^{\mathrm{T}}\frac{\partial V^*}{\partial \boldsymbol{x}}. \tag{8}$$

In the light of the Bellman's principle of optimality, the following optimality condition is obtained.

$$0 = \min_{\boldsymbol{u}} \max_{\boldsymbol{w}} H(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w},\nabla V^*) \tag{9}$$

By substituting (7) and (8) into (9), we get the well-known HJI equation.

$$0 = Q(\boldsymbol{x}) + \left(\frac{\partial V^*}{\partial \boldsymbol{x}}\right)^{\mathrm{T}}\boldsymbol{f}(\boldsymbol{x}) - \frac{1}{4}\left(\frac{\partial V^*}{\partial \boldsymbol{x}}\right)^{\mathrm{T}}\boldsymbol{g}\boldsymbol{R}_1^{-1}\boldsymbol{g}^{\mathrm{T}}\frac{\partial V^*}{\partial \boldsymbol{x}} + \frac{1}{4\gamma^2}\left(\frac{\partial V^*}{\partial \boldsymbol{x}}\right)^{\mathrm{T}}\boldsymbol{k}\boldsymbol{k}^{\mathrm{T}}\frac{\partial V^*}{\partial \boldsymbol{x}} \tag{10}$$

If there exists a minimal non-negative solution $V^*(x) \geqslant 0 : \mathbf{R}^n \to \mathbf{R}$ to the HJI equation (10), and then the action strategy pairs given in (7) and (8) are Nash equilibrium saddle-point solution [33]. The Nash equilibrium strategy is inherently robust, once the equilibrium is reached, no participant can unilaterally deviate from its Nash strategy to improve its payoff. If a player deviates from its optimum behavior and the opponent sticks to its optimum policy, the former cannot get the optimal benefit from the game.

Because HJI equation (10) is a nonlinear partial differential equation, finding the analytic solution of HJI equation is usually intractable. Furthermore, complete knowledge of nonlinear game system dynamics is needed to solve (10). Therefore, a various of offline numerical calculation methods have been investigated, and then the approximate optimal solution of two-person zero-sum differential games problem is obtained.

2.2 On-policy IRL algorithm for solving HJI equation

By differentiating (4), one can get the following nonlinear system Lyapunov equation (LE).

$$\begin{cases} 0 = Q(\boldsymbol{x}) + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u} - \gamma^2\boldsymbol{w}^{\mathrm{T}}\boldsymbol{w} + \dfrac{\partial V}{\partial \boldsymbol{x}}(\boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{u} + \boldsymbol{k}(\boldsymbol{x})\boldsymbol{w}) \\ V(0) = 0 \end{cases} \tag{11}$$

A solution $V(\boldsymbol{x}) \geqslant 0$ is the cost function (4) associated with control policies $\boldsymbol{u}$ and $\boldsymbol{w}$ specified. Noting that the LE (11) is linear with respect to the value function gradient $\partial V / \partial \boldsymbol{x}$, but the HJI equation (10) is nonlinear with respect to the optimal cost function gradient $\partial V^* / \partial \boldsymbol{x}$. Therefore, HJI equation can be iteratively solved by utilizing one of several offline algorithms based on solving the LE iteratively [34].

Inspired by the integral reinforcement learning methods introduced by artificial intelligence scientists, a discretized version of LE for nonlinear continuous-time game system in line with [35] is given as follows:

$$V^{\boldsymbol{u},\boldsymbol{w}}(\boldsymbol{x}(t)) = \int_t^{t+\Delta t} r(\boldsymbol{x},\boldsymbol{u},\boldsymbol{w})\mathrm{d}t + V^{\boldsymbol{u},\boldsymbol{w}}(\boldsymbol{x}(t+\Delta t)),\ V^{\boldsymbol{u},\boldsymbol{w}}(0) = 0 \tag{12}$$

where the integral in (12) could be regarded as a reinforcement term on the time interval $[t, t+\Delta t)$. According to (12), an on-policy IRL algorithm which is termed as SPUA in [27] is introduced as shown in Algorithm 1.

Algorithm 1 SPUA for H_∞ control problem

Step 1 Set $i = 0$, give an initial value function V_i, and initial disturbance and control policies associated $\boldsymbol{u}_i = -\dfrac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^{\mathrm{T}}\dfrac{\partial V_i}{\partial \boldsymbol{x}}, \boldsymbol{w}_i = \dfrac{1}{2\gamma^2}\boldsymbol{k}^{\mathrm{T}}\dfrac{\partial V_i}{\partial \boldsymbol{x}}$.

Step 2 Solve for $V_{i+1}(\boldsymbol{x})$ with $V_{i+1}(0)=0$ by using

$$V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x}(t))=\int_t^{t+\Delta t} r(\boldsymbol{x},\boldsymbol{u}_i,\boldsymbol{w}_i)\,\mathrm{d}t+V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x}(t+\Delta t)) \tag{13}$$

Step 3 Update the disturbance policy and control policy using

$$\boldsymbol{w}_{i+1}=\frac{1}{2\gamma^2}\boldsymbol{k}^{\mathrm{T}}\frac{\partial V_{i+1}}{\partial \boldsymbol{x}}, \tag{14}$$

$$\boldsymbol{u}_{i+1}=-\frac{1}{2}\boldsymbol{R}^{-1}\boldsymbol{g}^{\mathrm{T}}\frac{\partial V_{i+1}}{\partial \boldsymbol{x}}. \tag{15}$$

Step 4 If $\|V_{i+1}-V_i\|_{\Omega}\leqslant\varepsilon$, $\varepsilon>0$ (positive real number and small), stop calculation, else set $i=i+1$, move to Step 2 and go on iteration.

In [35], it has been proved that solving for $V^{\boldsymbol{u},\boldsymbol{w}}(\boldsymbol{x})$ in (11) is equivalent to calculating the solution of (13). Although solving (11) and (13) can obtain the same solution, the knowledge of internal dynamics $\boldsymbol{f}(\boldsymbol{x})$ is not required by solving (13), which is required explicitly in (11). The SPUA for two-person zero-sum differential games also includes policy evaluation step and policy update step, which can be regarded as an IRL algorithm for two participants to learn the optimum strategies in an uncertain scenario.

2.3 Off-policy IRL algorithm and NNs-based online implementation

SPUA is an on-policy learning algorithm. For on-policy learning, the sampled data are generated by using the evaluating control and disturbance strategies. Therefore, when online implementation of SPUA, the disturbance policies need to be adjustable and specified, which is usually unrealistic for practical applications. To overcome this drawback, motivated by [30], an online off-policy integral RL algorithm is proposed to solve the HJI equation without making use of any prior information of the system dynamics. Then, a NN-based implementation is given.

2.3.1 Off-policy IRL algorithm for solving HJI equation

Firstly, the system dynamics in (1) is rewritten as

$$\dot{\boldsymbol{x}}=\boldsymbol{f}(\boldsymbol{x})+\boldsymbol{g}(\boldsymbol{x})\boldsymbol{u}_i+\boldsymbol{k}(\boldsymbol{x})\boldsymbol{w}_i+\boldsymbol{g}(\boldsymbol{x})(\boldsymbol{u}-\boldsymbol{u}_i)+\boldsymbol{k}(\boldsymbol{x})(\boldsymbol{w}-\boldsymbol{w}_i) \tag{16}$$

where $\boldsymbol{u}\in\mathbf{R}^m$ and $\boldsymbol{w}\in\mathbf{R}^q$ are behavior policy and actual disturbance, respectively. $\boldsymbol{u}_i\in\mathbf{R}^m$ and $\boldsymbol{w}_i\in\mathbf{R}^q$ are strategies to be evaluated and updated. Let $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x})$ be the solution of (11), and differentiating $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x})$ along with the system dynamics in (16) yields

$$\dot{V}_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}=(\nabla V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i})^{\mathrm{T}}(\boldsymbol{f}+\boldsymbol{g}\boldsymbol{u}_i+\boldsymbol{k}\boldsymbol{w}_i)+(\nabla V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i})^{\mathrm{T}}\boldsymbol{g}(\boldsymbol{u}-\boldsymbol{u}_i)+(\nabla V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i})^{\mathrm{T}}\boldsymbol{k}(\boldsymbol{w}-\boldsymbol{w}_i). \tag{17}$$

Using (11), (14) and (15) one can obtain:

$$\dot{V}_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}=-r(\boldsymbol{x},\boldsymbol{u}_i,\boldsymbol{w}_i)-2\boldsymbol{u}_{i+1}^{\mathrm{T}}\boldsymbol{R}(\boldsymbol{u}-\boldsymbol{u}_i)+2\gamma^2\boldsymbol{w}_{i+1}^{\mathrm{T}}(\boldsymbol{w}-\boldsymbol{w}_i)=-Q(\boldsymbol{x})-\boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u}_i+\gamma^2\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{w}_i-2\boldsymbol{u}_{i+1}^{\mathrm{T}}\boldsymbol{R}(\boldsymbol{u}-\boldsymbol{u}_i)+2\gamma^2\boldsymbol{w}_{i+1}^{\mathrm{T}}(\boldsymbol{w}-\boldsymbol{w}_i). \tag{18}$$

Calculating integral on both sides of (18) over the time interval $[t,t+\Delta t)$ yields the off-policy integral RL equation as follows:

$$V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x}(t+\Delta t))-V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x}(t))=-\int_t^{t+\Delta t} r(\boldsymbol{x},\boldsymbol{u}_i,\boldsymbol{w}_i)\,\mathrm{d}t-\int_t^{t+\Delta t} 2\boldsymbol{u}_{i+1}^{\mathrm{T}}\boldsymbol{R}(\boldsymbol{u}-\boldsymbol{u}_i)\,\mathrm{d}t+\int_t^{t+\Delta t} 2\gamma^2\boldsymbol{w}_{i+1}^{\mathrm{T}}(\boldsymbol{w}-\boldsymbol{w}_i)\,\mathrm{d}t. \tag{19}$$

It is worth noting that for arbitrary behavior strategy $\boldsymbol{u}$ and practical disturbance $\boldsymbol{w}$ which are acted on the game system, (19) could be used to solve cost function $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}$, evaluated control strategy $\boldsymbol{u}_{i+1}$ and $\boldsymbol{w}_{i+1}$. The off-policy integral RL algorithm for iteratively solving HJI equation (10) using (19) is given as shown in Algorithm 2.

Algorithm 2 Off-policy integral RL for calculating the solution of HJI equation

Step 1 Use the behavior policy $\boldsymbol{u}$ and the actual disturbance $\boldsymbol{w}$ to collect N system data which contain system state, disturbance input and control input at different sampling time interval.

Step 2 Set $i=0$, give an initial cost function V_0, and initial evaluated policy $\boldsymbol{w}_0$ and $\boldsymbol{u}_0$.

Step 3 Reuse the collected data to solve (19) for $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x})$, $\boldsymbol{u}_{i+1}$ and $\boldsymbol{w}_{i+1}$, with $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(0)=0$.

Step 4 If $\|V_{i+1}-V_i\|+\|\boldsymbol{u}_{i+1}-\boldsymbol{u}_i\|+\|\boldsymbol{w}_{i+1}-\boldsymbol{w}_i\|\leqslant\varepsilon$, $\varepsilon>0$ (positive real number and small), stop iteration and output $V_{i+1}(\boldsymbol{x})$ as the approximate optimal solution of HJI equation (10), output $\boldsymbol{u}_{i+1}$ as the approximate optimal control input, i.e., $V^*(\boldsymbol{x})=V_{i+1}(\boldsymbol{x})$ and $\boldsymbol{u}^*=\boldsymbol{u}_{i+1}$, else set $i=i+1$, go to Step 3 and go on iteration.

In [27], Wu et al. proofed that SPUA approach was essentially a Newton iteration algorithm for pursuing a solution of the fixed-point equation in a Banach space, the convergence could be established by the Kantorovich's theorem. The following theorem proves that Algorithm 2 has the same solution and convergence as the SPUA.

Theorem 1 Equation (19) which is an off-policy integral RL equation has the same solution of the cost function $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}$ as the discretized version of LE (13).

Proof From the derivation of (19), first assume that $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x})$ is the solution of (11), then (19) is derived by using (14) and (15). On the other hand, with the boundary condition $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(0)=0$, it can be proved that $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x})$ is the unique solution of (19) by contradiction in Theorem 1 [30]. Therefore, (11) and (19) have the same solution. At the same time, it has been proved that infinitesimal version of LE in (11) and discretized version of Lyapunov equation (13) have the same solution $V^{\boldsymbol{u},\boldsymbol{w}}(\boldsymbol{x})$ in Lemma 1 [35]. Consequently, the off-policy integral RL equation in (19) can get the same solution for the cost function as (13). □

Remark 1 It can be shown that (19) and (13) have the same solution, thus the algorithm convergence of the off-policy IRL method is the same as the SPUA in line with [27], that is, with the increase of iteration step i, the solution of iteration equation (19) will gradually approach the optimal solution of the HJI equation in (10).

Remark 2 Different from [30] in which the off-policy IRL algorithm is presented to learn the approximate optimal solution of HJI equation without system internal dynamic model $\boldsymbol{f}(\boldsymbol{x})$, the Algorithm 2 proposed above can get the approximate optimal solution of HJI equation without using any knowledge of the system dynamics.

Noting that (19) is a scalar equation, the least squares method can be utilized to calculate the solution when the linear-in-parameter NNs are used to approximate the value function and control policies.

2.3.2 NNs based online implementation of Algorithm 2

In order to solve $V_{i+1}^{\boldsymbol{u}_i,\boldsymbol{w}_i}(\boldsymbol{x})$, $\boldsymbol{u}_{i+1}$ and $\boldsymbol{w}_{i+1}$ in (19) by using system sampled data, a NNs-based actor-critic structure is introduced. According to the famous Weierstrass high-order approximation theory [36] which points out that any continuous functions can be fitted using the infinite dimensional set of linear independent basis functions, here three NNs, i.e., one critic NN and two actor NNs, are applied to approximate the value function, disturbance strategy and control strategy, respectively. For the actual implementation, it generally only needs to fit a function using a finite-dimensional functions set on a compact set. Thus, the three NNs are given as

$$\hat{V}_{i+1}(x) = \boldsymbol{W}_c^{\mathrm{T}}\boldsymbol{\rho}(x), \tag{20}$$

$$\hat{\boldsymbol{u}}_{i+1}(x) = \boldsymbol{W}_a^{\mathrm{T}}\boldsymbol{\phi}(x), \tag{21}$$

$$\hat{\boldsymbol{w}}_{i+1}(x) = \boldsymbol{W}_d^{\mathrm{T}}\boldsymbol{\varphi}(x), \tag{22}$$

where $\boldsymbol{\rho}(x) = [\rho_1(x), \rho_2(x), \cdots, \rho_{L_1}(x)]^{\mathrm{T}} \in \mathbf{R}^{L_1}$ is the linearly independent basis function vector for the critic NN, $\boldsymbol{\phi}(x) = [\phi_1(x), \phi_2(x), \cdots, \phi_{L_2}(x)]^{\mathrm{T}} \in \mathbf{R}^{L_2}$ and $\boldsymbol{\varphi}(x) = [\varphi_1(x), \varphi_2(x), \cdots, \varphi_{L_3}(x)]^{\mathrm{T}} \in \mathbf{R}^{L_3}$ are the linearly independent basis function vectors for actor and disturber NNs, respectively, which are all defined on $\boldsymbol{\Omega} \subset \mathbf{R}^n$. L_1, L_2 and L_3 are the numbers of neurons in the hidden layer of the three NNs respectively. $\boldsymbol{W}_c \in \mathbf{R}^{L_1}$, $\boldsymbol{W}_a \in \mathbf{R}^{L_2 \times m}$ and $\boldsymbol{W}_d \in \mathbf{R}^{L_3 \times q}$ are weight vectors which are constant vectors. Define $\boldsymbol{R} = \mathrm{diag}(r_1, r_2, \cdots, r_m)$, substituting (20)−(22) in (19) yields:

$$e(t) = \boldsymbol{W}_c^{\mathrm{T}}(\boldsymbol{\rho}(\boldsymbol{x}(t+\Delta t)) - \boldsymbol{\rho}(\boldsymbol{x}(t))) + \int_t^{t+\Delta t} r(\boldsymbol{x}, \boldsymbol{u}_i, \boldsymbol{w}_i)\mathrm{d}t + 2\sum_{j=1}^{m} r_j \int_t^{t+\Delta t} \boldsymbol{W}_{a,j}^{\mathrm{T}}\boldsymbol{\phi}(x)\nu_j^1 \mathrm{d}t - 2\gamma^2 \sum_{k=1}^{q} \int_t^{t+\Delta t} \boldsymbol{W}_{d,k}^{\mathrm{T}}\boldsymbol{\varphi}(x)\nu_k^2 \mathrm{d}t \tag{23}$$

where $\boldsymbol{\nu}^1 = [\nu_1^1, \nu_2^1, \cdots, \nu_m^1] = \boldsymbol{u} - \boldsymbol{u}_i$, $\boldsymbol{\nu}^2 = \left[\nu_1^2, \nu_2^2, \cdots, \nu_q^2\right] = \boldsymbol{w} - \boldsymbol{w}_i$, $\boldsymbol{W}_{a,j}$ is the jth column of matrix $\boldsymbol{W}_a$, $\boldsymbol{W}_{d,k}$ is the kth column of matrix $\boldsymbol{W}_d$, and $e(t)$ is the fitting error of (19). $e(t)$ can be regarded as a residual error of continuous-time system temporal difference [37]. Note that (23) is linear in the NNs weight vectors. Define

$$\boldsymbol{W} = \left[\boldsymbol{W}_c^{\mathrm{T}}, \boldsymbol{W}_{a,1}^{\mathrm{T}}, \cdots, \boldsymbol{W}_{a,m}^{\mathrm{T}}, \boldsymbol{W}_{d,1}^{\mathrm{T}}, \cdots, \boldsymbol{W}_{d,q}^{\mathrm{T}}\right]^{\mathrm{T}}, \tag{24}$$

$$\boldsymbol{\omega}(t) = \begin{bmatrix} \boldsymbol{\rho}(\boldsymbol{x}(t+\Delta t)) - \boldsymbol{\rho}(\boldsymbol{x}(t)) \\ 2r_1\int_t^{t+\Delta t} \boldsymbol{\phi}(x)\nu_1^1 \mathrm{d}t \\ \vdots \\ 2r_m\int_t^{t+\Delta t} \boldsymbol{\phi}(x)\nu_m^1 \mathrm{d}t \\ -2\gamma^2\int_t^{t+\Delta t} \boldsymbol{\varphi}(x)\nu_1^2 \mathrm{d}t \\ \vdots \\ -2\gamma^2\int_t^{t+\Delta t} \boldsymbol{\varphi}(x)\nu_q^2 \mathrm{d}t \end{bmatrix}, \tag{25}$$

$$\lambda(t) = \int_t^{t+\Delta t} (-r(\boldsymbol{x}, \boldsymbol{u}_i, \boldsymbol{w}_i))\mathrm{d}t = \int_t^{t+\Delta t} \left(-Q(\boldsymbol{x}) - \boldsymbol{u}_i^{\mathrm{T}}\boldsymbol{R}\boldsymbol{u}_i + \gamma^2\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{w}_i\right)\mathrm{d}t. \tag{26}$$

Then, (23) can be rewritten as

$$e(t) + \lambda(t) = \boldsymbol{W}^{\mathrm{T}}\boldsymbol{\omega}(t). \tag{27}$$

Note that (27) is linear in parameter $\boldsymbol{W}$, therefore $\boldsymbol{W}$ can be solved in the sense of least-squares by minimizing the square of error $e(t)$. Because of $\boldsymbol{W} \in \mathbf{R}^{L_1 + m\cdot L_2 + q\cdot L_3}$, therefore one needs to collect $N > L_1 + m\cdot L_2 + q\cdot L_3$ system data about system state, disturbance signal, and control input from t_1 to t_N in the state space. Then, for the given evaluating policies $\hat{\boldsymbol{w}}_i$ and $\hat{\boldsymbol{u}}_i$, using this information to calculate (25) and (26) at N different sampled points, one can get

$$\boldsymbol{\Omega} = [\omega(t_1), \omega(t_2), \cdots, \omega(t_N)], \tag{28}$$

$$\boldsymbol{\Lambda} = [\lambda(t_1), \lambda(t_2), \cdots, \lambda(t_N)]^{\mathrm{T}}. \tag{29}$$

The least-squares solution of (27) is given as

$$\boldsymbol{W} = \left(\boldsymbol{\Omega}\boldsymbol{\Omega}^{\mathrm{T}}\right)^{-1}\boldsymbol{\Omega}\boldsymbol{\Lambda}. \tag{30}$$

Then one can obtain $\hat{V}_{i+1}$, $\hat{\boldsymbol{w}}_{i+1}$ and $\hat{\boldsymbol{u}}_{i+1}$ using (20)−(22). Reusing the collected data to solve (30), one can get the approximate optimal control input pairs $\boldsymbol{u}^*$ and $\boldsymbol{w}^*$.

Remark 3 Equation (30) is a batch least-squares equation, one can also use recursive least-squares method to solve this problem, as shown in [31].

Remark 4 Although (23) contains $\boldsymbol{x}(t+\Delta t)$, the system data can be sampled continuously over the same time interval Δt, i.e., $\boldsymbol{x}(t), \boldsymbol{x}(t+\Delta t), \cdots, \boldsymbol{x}(t+N\Delta t)$. Thus, none of the knowledge of game system dynamics is needed to calculate the future system state $\boldsymbol{x}(t+\Delta t)$ at instant t.

3. CIIG

In this section, for the typical engagement scenario of air-to-air missile intercepting high-maneuvering target, the CIIG algorithm based on online off-policy IRL method is proposed. The guidance system can be modeled in the pitch plane and yaw plane respectively, therefore the interception problem in the yaw plane is taken as an example to present the CIIG law.

In the yaw plane, the engagement geometry of interception scenario is indicated in Fig. 1.

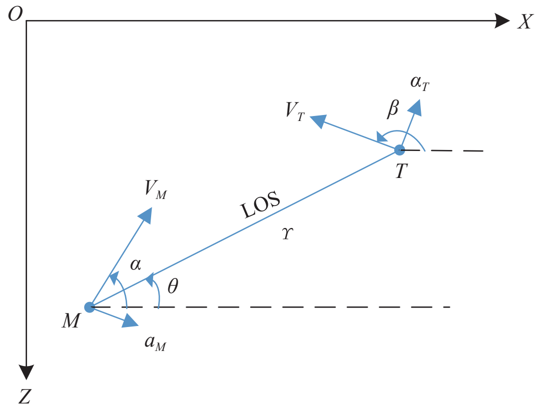


Fig. 1 Missile and target engagement geometry

As shown in Fig. 1, the missile dynamic equations in the yaw plane are given as

$$\begin{cases} \dot{x}_M = V_M \cos\alpha \\ \dot{z}_M = -V_M \sin\alpha \\ \dot{\alpha} = -a_M / V_M \\ \dot{a}_M = (\mu_M - a_M)/\tau_M \end{cases} \tag{31}$$

where (x_M, z_M) are the missile position coordinates in the inertial reference frame, V_M represents missile velocity, α is the flight-path angle (FPA) of missile. Missile acceleration a_M is assumed to be perpendicular to its velocity V_M, τ_M is the autopilot time constant of missile, and μ_M is the acceleration instruction of missile. The equivalent first-order model of missile autopilot is considered here. And, the dynamics of target has the same form.

$$\begin{cases} \dot{x}_T = V_T \cos\beta \\ \dot{z}_T = -V_T \sin\beta \\ \dot{\beta} = -a_T / V_T \\ \dot{a}_T = (\nu_T - a_T)/\tau_T \end{cases} \tag{32}$$

where (x_T, z_T) are the target position coordinates in the inertial reference frame, V_T represents target velocity, β is the FPA of target. Target acceleration a_T is assumed to be perpendicular to its velocity V_T too, τ_T is the autopilot time constant of target, and ν_T is the acceleration instruction of target.

Assume that missile and target both have constant speeds, the relative motion equations of missile and target in the engagement scenario are given as

$$\dot{r} = V_r = V_T\cos(\beta-\theta) - V_M\cos(\alpha-\theta), \tag{33}$$

$$\dot{\theta} = [V_T\sin(\beta-\theta) - V_M\sin(\alpha-\theta)]/r, \tag{34}$$

where θ represents LOS angle, $\dot{\theta}$ is LOS rate, r represents missile-target relative distance, and V_r is closing velocity, i.e., range rate along the LOS.

At an instant t, the quantity of terminal zero effort miss distance (ZEMD) is calculated as the minimum distance between missile and target if neither the missile nor the target maneuvers from this moment [38], i.e., $r_{\text{ZEMD}}(t)$ is calculated as

$$r_{\text{ZEMD}}(t) = \frac{r^2\dot{\theta}}{\sqrt{V_r^2 + r^2\dot{\theta}^2}}. \tag{35}$$

In the terminal guidance phase, from (35), one can see that keeping $V_r < 0$ and regulating $\dot{\theta}$ to zero can ensure that the target is captured effectively. Therefore, the distance rate V_r and the LOS rate $\dot{\theta}$ need to meet the following conditions.

$$V_r < 0, \dot{\theta} \to 0. \tag{36}$$

In the interception problem, the expectation of missile is to minimize $r_{\text{ZEMD}}(t)$, however, the objective of target which is an opponent is to maximize $r_{\text{ZEMD}}(t)$. Therefore, the guidance application can be regarded as solving a two-person zero-sum differential games problem. Select $x_1 = \theta$ and $x_2 = \dot{\theta}$ as the system state, take the derivative of both sides of (34), and substitute (31), (32), and (33) into it, one can obtain:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\dfrac{2V_r}{r}x_2 + \dfrac{\cos(\alpha-\theta)}{r}a_M - \dfrac{\cos(\beta-\theta)}{r}a_T \end{cases}. \tag{37}$$

In the real guidance process, it is important to note that the guidance procedure will end when the minimum detection distance of seeker is greater than the missile-target distance r, i.e., the parameter r appearing in the denominator in (37) will not be zero throughout the guidance process. Therefore, the system dynamics satisfies locally Lipschitz condition. Furthermore, from (37), it can be seen that $|\beta-\theta| = \pi/2, |\alpha-\theta| = \pi/2$ is an uncontrollable and unstable equilibrium. Hence, the guidance law domain of validity meets the constraint of the following conditions.

$$\boldsymbol{\Omega} = \{\boldsymbol{x} : |\beta-\theta| \neq \pi/2, |\alpha-\theta| \neq \pi/2, V_r < 0\}. \tag{38}$$

Next, the CIIG law is proposed by using the online off-policy IRL algorithm, the computation procedure of CIIG is given in Algorithm 3.

Algorithm 3 CIIG law

Step 1 Set $k = 1$, collect N system data using the missile acceleration a_M and the target actual acceleration a_T from the time interval $[(k-1)N\Delta t, kN\Delta t)]$.

Step 2 After the system data are collected, use Algorithm 2 to solve the approximate optimal cost function $V_k^*(\boldsymbol{x})$, missile approximate optimal control input $a_{M,k}^*$ and target approximate optimal disturbance strategy $a_{T,k}^*$ of the kth calculation cycle.

Step 3 Take $a_{M,k}^*$ as the actual behavior control strategy of missile to be used. If the missile target distance is less than a certain value, stop updating $a_{M,k}^*$, else set $k = k+1$, and collect N system data with the target actual acceleration a_T from the time interval $[(k-1)N\Delta t, kN\Delta t)]$, then go to Step 2.

Remark 5 Because r, V_r, α and β are changed over time, (37) is a time-varying differential equation, the missile optimal control policy $a_{M,k}^*$ need to be updated periodically.

Remark 6 The real-time performance of the Algorithm 3 is a challenge when implementing it online. As shown in Algorithm 2, integral and iterative procedure are required. The calculation of integrals is usually time-consuming and increases exponentially with the increase of the system state dimension and the number of basis functions. The amounts of integral which remain invariant during the iteration process can be precalculated and stored, therefore, the integrals only need to be calculated once for all, which can reduce the number of integral evaluations. On the other hand, the appropriate selection of the set of basis functions and its sizes can also be helpful to balance convergence and computational time.

4. Simulation verification

Two computer simulation examples are carried out to illustrate the effectiveness of CIIG, where one is to intercept non-maneuvering target, and the other is to intercept high-maneuvering target.

The hidden layer neurons of critic NN, actor NN and disturber NN on $\boldsymbol{\Omega}$ are set as follows:

$$\boldsymbol{\rho}(x_1,x_2) = \left[x_1^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4, x_1^2, x_1x_2, x_2^2\right]^{\mathrm{T}}, \tag{39}$$

$$\boldsymbol{\phi}(x_1,x_2) = \left[x_2, x_1^2x_2, x_2^3\right]^{\mathrm{T}}, \tag{40}$$

$$\boldsymbol{\varphi}(x_1,x_2) = \left[x_2, x_1^2x_2, x_2^3\right]^{\mathrm{T}}. \tag{41}$$

Thus, $\boldsymbol{W}_c \in \mathbf{R}^8$, $\boldsymbol{W}_a \in \mathbf{R}^{3\times1}$, $\boldsymbol{W}_d \in \mathbf{R}^{3\times1}$, and $\boldsymbol{W} \in \mathbf{R}^{14}$. The numerical simulation calculation step Δt is set to be 0.005 s, and take $N = 100$, i.e., collect 100 data points every 0.5 s along the system state trajectories, therefore Algorithm 2 needs to be used for solving the solution of HJI equation every 0.5 s. The initial values of V_0, $\boldsymbol{u}_0$ and $\boldsymbol{w}_0$ in Algorithm 2 are set to be zero. The approximate optimal control strategy of the missile obtained in the previous cycle will be used as behavior policy of the missile in the next cycle. The behavior policy of target is its actual maneuver strategy, therefore target acceleration does not need to be adjustable.

4.1 Non-maneuvering target

Numerical simulation conditions of Example 1 are shown in Table 1.

Table 1 Simulation conditions of Example 1

Parameter	Symbol	Value
Initial position of missile/m	(x_M, z_M)	(0, 0)
Initial FPA of missile/(°)	α	0
Missile velocity/(m·s^{-1})	V_M	600
Initial position of target/m	(x_T, z_T)	(5 000, 0)
Initial FPA of target/(°)	β	210
Target velocity/(m·s^{-1})	V_T	200
Target acceleration/g	a_T	0
Missile autopilot first-order lag/s	τ_M	0.1
Target autopilot first-order lag/s	τ_T	0.1

The parameters of utility function of example 1 are set as $Q(x_1,x_2) = q_1x_1^2 + q_2x_2^2$, $q_1 = 0$, $q_2 = 4\times10^7$, $R = 1$ and $\gamma = 10$. Numerical simulation results of example 1 are shown in Fig. 2–Fig. 6. The actor NN weights learned during the online learning guidance stage are shown in Fig. 2, the weights of the NN are updated every 0.5 s. Fig. 3 shows the number of iterations per calculation cycle, as can be seen that the maximum iteration number is 40 at $t = 5.0$ s. The learning processes of the actor NN weights at $t = 5.0$ s are shown in Fig. 4. The actor NN weights converge ultimately, and the approximate optimal control strategy of the missile is obtained. The normal acceleration instruction and response histories of missile are demonstrated in Fig. 5, where g is the acceleration of gravity. Fig. 6 shows the missile and target trajectories.

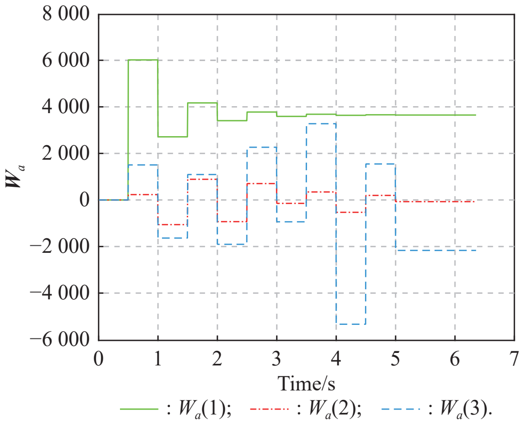


Fig. 2 Actor NN weights learned during the online learning guidance stage of Example 1

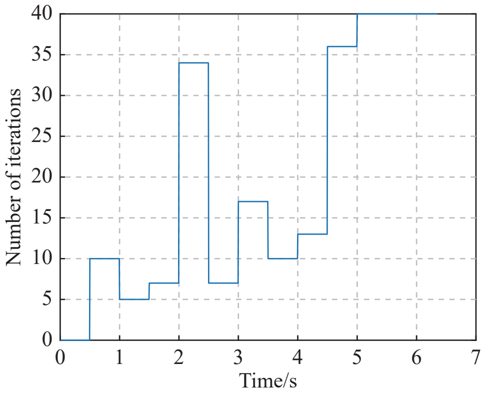


Fig. 3 Number of iterations per calculation cycle of Example 1

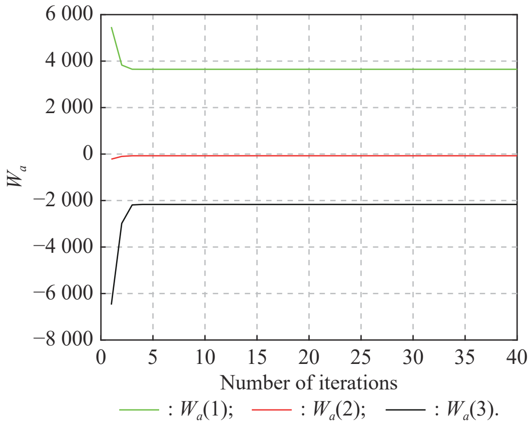


Fig. 4 Actor NN weights learning process at t=5.0 s of Example 1

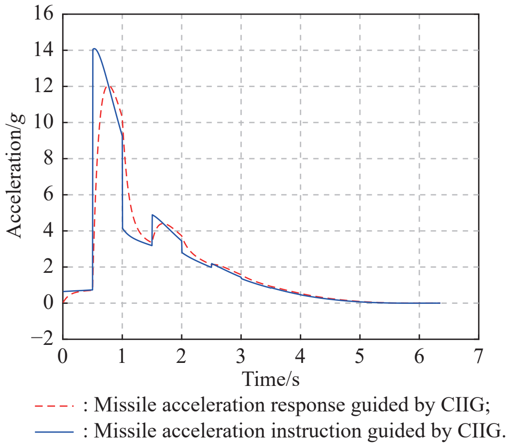


Fig. 5 Missile acceleration instruction and response for intercepting non-maneuvering target

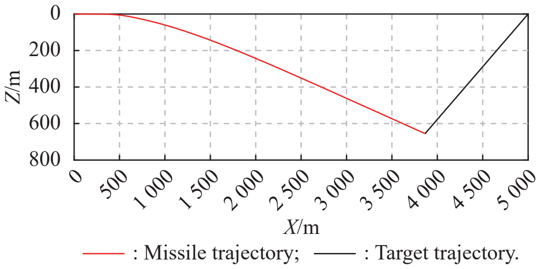


Fig. 6 Missile and target trajectories of Example 1

4.2 High-maneuvering target

Numerical simulation conditions of Example 2 are shown in Table 2.

Table 2 Simulation conditions of Example 2

Parameter	Symbol	Value
Initial position of missile/m	(x_M, z_M)	(0, 0)
Initial FPA of missile/(°)	α	0
Missile velocity/(m·s^{-1})	V_M	600
Initial position of target/m	(x_T, z_T)	(8 000, 0)
Initial FPA of target/(°)	β	190
Target velocity/(m·s^{-1})	V_T	300
Target acceleration/g	a_T	12
Missile autopilot first-order lag/s	τ_M	0.1
Target autopilot first-order lag/s	τ_T	0.1

The parameters of utility function of Example 2 are set as $Q(x_1, x_2) = q_1 x_1^2 + q_2 x_2^2$, $q_1 = 0$, $q_2 = 10^8$, $R = 1$ and $\gamma = 10$. In this scenario, the value of q_2 is greater than that of intercepting non-maneuvering target, to improve the missile acceleration and reduce miss distance. In

Example 2, the CIIG law proposed will be compared with the adaptive dynamic surface guidance (ADSG) law introduced in [5,6] and APNG law. Simulation results of missile-target engagement are shown in Fig. 7–Fig. 11. Fig. 7 shows the missile and target trajectories, the miss distance is 0.111 m and 0.236 m by using CIIG and ADSG respectively, and the guidance accuracy of the two guidance algorithms is comparable. The miss distance of APNG is 5.624 m. The normal acceleration instruction and response histories of the three guidance algorithms are demonstrated in Fig. 8. At the end of the trajectories, the acceleration commands calculated by APNs tend to saturate. In the initial phase, the target is non-maneuvering, and after 1.5 s, the target actuates a circular maneuver with acceleration of 12*g*, i.e., the maneuvering acceleration is 12 times that of gravity. In the initial, the acceleration instruction of missile guided by CIIG is driven by white noise. The actor NN weights learned during the online learning guidance stage are shown in Fig. 9. Fig. 10 shows the number of iterations per calculation cycle, as can be seen that the maximum iteration number is 14 at $t = 3.0$ s and $t = 12.0$ s. The learning processes of the actor NN weights at these two calculation cycles are shown in Fig. 11. The actor NN weights converge ultimately, pointing out that the approximate optimal control strategy for the missile is obtained.

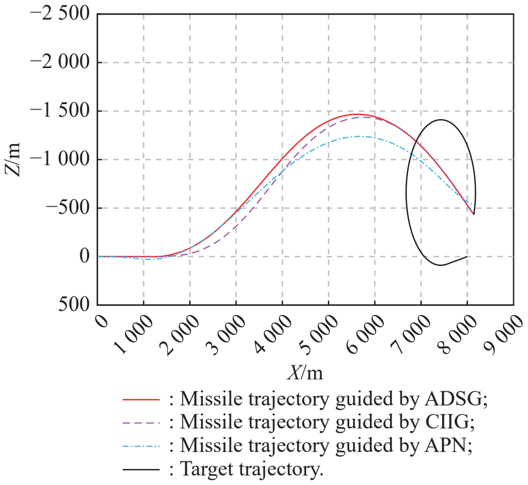


Fig. 7 Missile and target trajectories of Example 2

From the above simulation results, one can see that the approximate Nash equilibrium solution of missile-target two-person zero-sum differential games problem is obtained by using off-policy IRL when the online learning process converges. Because the Nash equilibrium strategy is inherently robust, the CIIG algorithm is effective in intercepting high-maneuvering targets.

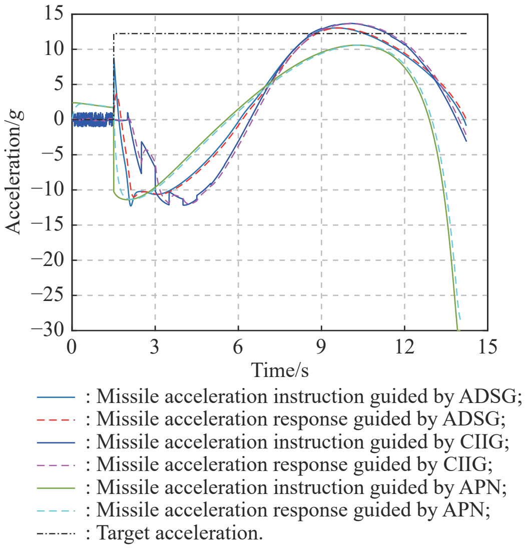


Fig. 8 Missile acceleration instruction and response for intercepting high-maneuvering target

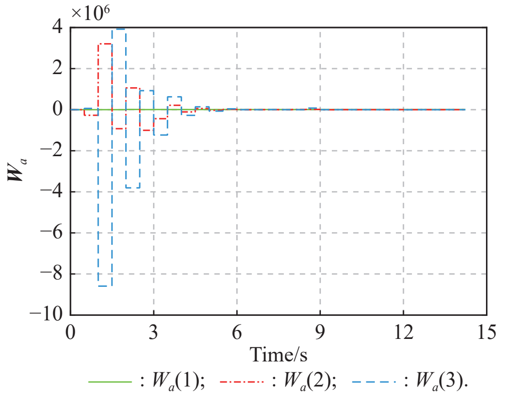


Fig. 9 Actor NN weights learned during the online learning guidance stage of Example 2

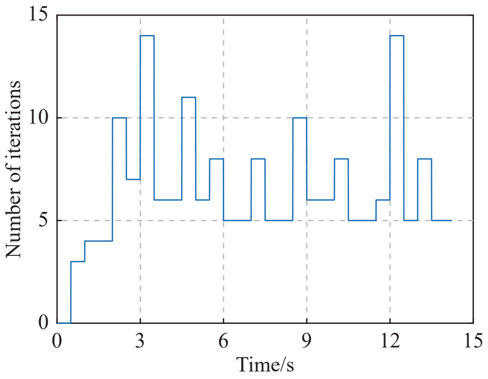
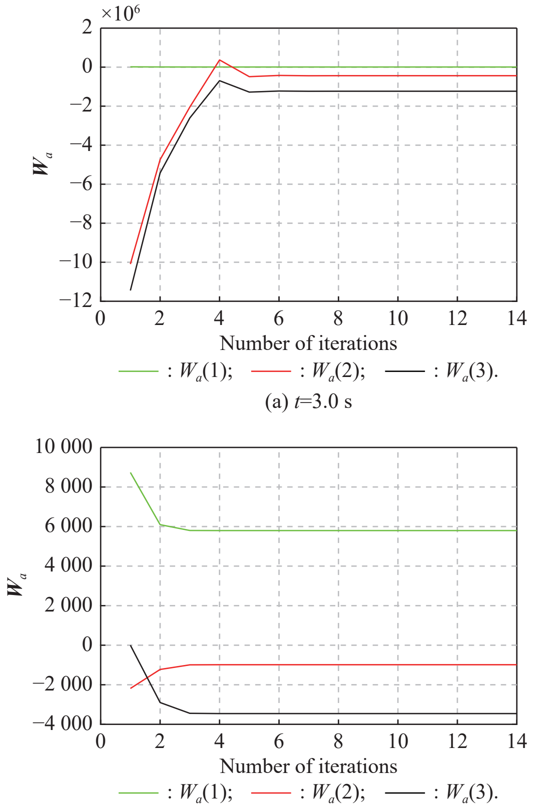


Fig. 10 Number of iterations per calculation cycle of Example 2



(a) t=3.0 s

(b) t=12.0 s

Fig. 11 Actor NN weights learning process of Example 2

5. Conclusions

A model-free, data-driven the CIIG law which can intercept high-maneuvering target is introduced in this paper. Firstly, the missile-target interception problem is regarded as a nonlinear two-person zero-sum differential games problem, in which the missile is a minimizing participant and the target is a maximizing opponent. And then, an online off-policy IRL algorithm is proposed to solve this nonlinear differential game problem, the algorithm proposed has the same convergence as SPUA. The NNs based framework is presented to carry out the off-policy IRL algorithm online, and a batch least squares method is applied to update the weight vectors of NNs. Based on the online off-policy IRL method, the CIIG algorithm for intercepting high-maneuvering target is introduced. Finally, numerical simulation results of air-to-air missile intercepting non-maneuvering target and high-maneuvering target are given to verify the effectuality of the CIIG law proposed.

References

[1] YANG C D, YANG C C. Analytical solution of three-dimensional realistic true proportional navigation. Journal of Guidance Control and Dynamics, 1996, 19(3): 569–577.

[2] PALUMBO N F, BLAUWKAMP R A, LLOYD J M. Modern homing missile guidance theory and techniques. Johns Hopkins APL Technical Digest, 2010, 29(1): 42–59.

[3] ZHOU D, MU C D, XU W L. Adaptive sliding-mode guidance of a homing missile. Journal of Guidance, Control, and Dynamics, 1999, 22(4): 589–594.

[4] WANG X X, HUANG X L, DING S C. Terminal angle constraint finite-time guidance law with input saturation and autopilot dynamics. Journal of the Franklin Institute, 2022, 359(16): 8687–8712.

[5] ZHOU D, XU B. Adaptive dynamic surface guidance law with input saturation constraint and autopilot dynamics. Journal of Guidance, Control, and Dynamics, 2016, 39(5): 1152–1159.

[6] WANG Q, LIAO Z Z. Implementation method of parallel approaching guidance based on adaptive dynamic surface control accounting for autopilot lag. Proc. of the International Conference on Guidance, Navigation and Control, 2022: 309–317.

[7] DWIVEDI P N, BHATTACHARYYA A, PADHI R. Computationally efficient suboptimal mid course guidance using model predictive static programming. Proc. of the 17th World Congress, 2008: 3550–3555.

[8] DWIVEDI P N, BHATTACHARYA A, PADHI R. Suboptimal midcourse guidance of interceptors for high-speed targets with alignment angle constraint. Journal of Guidance, Control, and Dynamics, 2011, 34(3): 860–877.

[9] HALBE O, RAJA R G, PADHI R. Robust reentry guidance of a reusable launch vehicle using model predictive static programming. Journal of Guidance, Control, and Dynamics, 2014, 37(1): 134–148.

[10] OZA H B, PADHI R. Impact-angle-constrained suboptimal model predictive static programming guidance of air-to-ground missiles. Journal of Guidance, Control, and Dynamics, 2012, 35(1): 153–164.

[11] TRIPATHI A K, PADHI R. Autonomous landing for UAVs using T-MPSP guidance and dynamic inversion autopilot. IFAC Papers-OnLine, 2016, 49(1): 18–23.

[12] PAN B F, MA Y Y, YAN R. Newton-type methods in computational guidance. Journal of Guidance, Control, and Dynamics, 2019, 42(2): 377–383.

[13] PING L. Introducing computational guidance and control. Journal of Guidance, Control, and Dynamics, 2017, 40(2): 193.

[14] TSIOTRAS P, MESBAHI M. Toward an algorithmic control theory. Journal of Guidance, Control, and Dynamics, 2017, 40(2): 194–196.

[15] HOWARD R A. Dynamic programming and Markov processes. Massachusetts: MIT Press, 1960.

[16] WEI Q L, WANG F Y, LIU D R, et al. Finite-approximation-error-based discrete-time iterative adaptive dynamic programming. IEEE Trans. on Cybernetics, 2014, 44(12): 2820–2833.

[17] LEWIS F L, VRABIE D. Reinforcement learning and adaptive dynamic programming for feedback control. IEEE Circuits and Systems Magazine, 2009, 9(3): 32–50.

[18] GAUDET B, FURFARO R. Missile homing-phase guidance law design using reinforcement learning. Proc. of the AIAA Guidance, Navigation, and Control Conference, 2012. DOI: 10.2514/6.2012-4470.

[19] GAUDET B, FURFARO R, LINARES R. Reinforcement learning for angle-only intercept guidance of maneuvering targets. Aerospace Science and Technology, 2020, 99: 105746.

[20] HE S, SHIN H S, TSOURDOS A. Computational missile guidance: a deep reinforcement learning approach. Journal of Aerospace Information Systems, 2021, 18(8): 571–582.

[21] LI W F, ZHU Y H, ZHAO D B. Missile guidance with assisted deep reinforcement learning for head-on interception of maneuvering target. Complex & Intelligent Systems, 2021, 8(3): 1205–1216.

[22] WANG Q, LIAO Z Z, YAN F. Computational guidance law based on data-driven online adaptive critic designs. Proc. of the China Automation Congress, 2021: 1289−1294.

[23] SUN J L, LIU C S. Backstepping-based adaptive dynamic programming for missile-target guidance systems with state and input constraints. Journal of the Franklin Institute, 2018, 355(17): 8412–8440.

[24] WANG Q, LIAO Z Z. Computational intelligence game guidance law based on online adaptive dynamic programming. Aerospace Control, 2022, 40(6): 48–54.

[25] YU J L, DONG X W, LI Q D, et al. Task coupling based layered cooperative guidance: theories and applications. Control Engineering Practice, 2022, 121: 105050.

[26] YU J L, DONG X W, LI Q D, et al. Adaptive practical optimal time-varying formation tracking control for disturbed high-order multi-agent systems. IEEE Trans. on Circuits and Systems, 2022, 69(6): 2567–2578.

[27] WU H N, LUO B. Neural network based online simultaneous policy update algorithm for solving the HJI equation in nonlinear H_∞ control. IEEE Trans. on Neural Networks and Learning Systems, 2012, 23(12): 1884–1895.

[28] JIANG Y, JIANG Z P. Robust adaptive dynamic programming and feedback stabilization of nonlinear systems. IEEE Trans. on Neural Networks and Learning Systems, 2014, 25(5): 882–893.

[29] LUO B, WU H N, HUANG T W, et al. Data-based approximate policy iteration for affine nonlinear continuous-time optimal control design. Automatica, 2014, 50(12): 3281–3290.

[30] LUO B, WU H N, HUANG T W. Off-policy reinforcement learning for H_∞ control design. IEEE Trans. on Cybernetics, 2015, 45(1): 65–76.

[31] FU Y, CHAI T Y. Online solution of two-player zero-sum games for continuous-time nonlinear systems with completely unknown dynamics. IEEE Trans. on Neural Networks and Learning Systems, 2016, 27(12): 2577–2587.

[32] BASAR T, BERNHARD P. H_∞ optimal control and related minimax design problems. Massachusetts: Springer Science, 1995.

[33] VAN D S A J. L2-gain analysis of nonlinear systems and nonlinear state-feedback H_∞ control. IEEE Trans. on Automatic Control, 1992, 37(6): 770–784.

[34] ABU-KHALAF M, LEWIS F L, HUANG J. Neurodynamic programming and zero-sum games for constrained control systems. IEEE Trans. on Neural Networks, 2008, 19(7): 1243–1252.

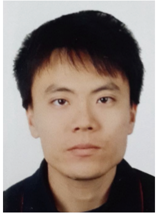
[35] VRABIE D, LEWIS F. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. Neural Networks, 2009, 22(3): 237–246.

[36] HORNIK K, STINCHCOMBE M, WHITE H. Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks. Neural Networks, 1990, 3(5): 551–560.

[37] MODARES H, FRANK L, JIANG Z P. H_∞ tracking control of completely unknown continuous-time systems via off-policy reinforcement learning. IEEE Trans. on Neural Networks and Learning Systems, 2015, 26(10): 2550–2562.

[38] BARDHAN R, GHOSE D. Nonlinear differential games-based impact-angle-constrained guidance law. Journal of Guidance, Control, and Dynamics, 2015, 38(3): 384–402.

Biographies



WANG Qi was born in 1982. He received his bachelor degree in spacecraft design from Beihang University (BUAA), Beijing, in 2005 and Ph.D. degree in aircraft design from Chinese Aeronautical Establishment, Beijing, in 2023. He is currently a senior engineer in China Airborne Missile Academy. His research interests are navigation, guidance and control of tactical missile, machine learning, adaptive dynamic programming, and reinforcement learning.

E-mail: wangqibuaa@126.com

LIAO Zhizhong was born in 1962. He received his bachelor degree and a Master degree in aircraft design from Northwestern Polytechnical University, Xi’an, in 1982 and 1984, respectively. He received his Ph.D. degree in engineering mechanics from Tsinghua University, Beijing, in 2001. He is currently the Deputy Chief Designer of China Airborne Missile Academy and a professor of Northwestern Polytechnical University. His research interests are aircraft design and project management based on systems engineering.

E-mail: lzzcama@139.com