electronics

MDPI

Article

Solution for Pursuit-Evasion Game of Agents by Adaptive Dynamic Programming

Zifeng Gong, Bing He *, Gang Liu and Xiaobo Zhang

Department of Engineering, PLA Rocket Force University of Engineering, Xi'an 710025, China
* Correspondence: celadongong@mau.edu.mk

Abstract: The paper studies a novel method for real-time solutions of the two-player pursuit-evasion game. The min-max principle is adopted to confirm the Nash equilibrium of the game. As agents in the game can form an Internet of Things (IoT) system, the real-time control law of each agent is obtained by taking a linear-quadratic cost function in adaptive dynamic programming. By introducing the Lyapunov function, we consider the scenario when capture occurs. Since most actual systems are continuous, the policy iteration algorithm is used to make the real-time policy converge to the analytical solution of the Nash equilibrium. Furthermore, we employ the value function approximation method to calculate the neural network parameters without directly solving the Hamilton–Jacobi–Isaacs equation. Simulation results depict the method's feasibility in different scenarios of the pursuit-evasion game.

Keywords: pursuit-evasion game; adaptive dynamic programming; Lyapunov function

1. Introduction

In recent years, the pursuit-evasion (PE) problem has attracted great attention because of its widespread application background in competitive games, optimization of IoT resources, and military attacks [1–4]. However, due to the real-time confrontation between the pursuit and evasion sides, the traditional unilateral control theory cannot solve the problem accurately [5]. Although the existing algorithms can solve the differential game problem in many scenarios, an offline algorithm cannot make real-time responses to the information of agents of the PE game with strong real-time performance. Thus, this paper focuses on the online PE game problem and realizes the solution of the agent policy according to the concept of adaptive dynamic programming.

The core of solving the PE game problem is to obtain the control policy of each agent on both sides of the game. Isaacs [6] introduced the modern control theory into the game theory and established the differential game theory. Thereafter, as a branch of the differential game, the PE game of agents has attracted much attention. With the continuous development of aerospace technology and the launch of man-made satellites, the game problems of continuous confrontation between both sides and even multiple players need to be solved urgently [7,8]. Friedman [9] proved the existence of saddle points in differential games, thus enabling them to optimize the strategies of all agents in the PE problem. For the control problem in a linear differential game system [10,11] discussed the control method for the cost function of a quadratic form. In contrast, [12] discussed the uniqueness of the Nash equilibrium point, so that the analytical solution can be obtained for the classical differential game problem.

However, a general system may be more complex, and it could be difficult to obtain its analytical solution. Therefore, compared with analytic methods, scholars usually prefer numerical methods for solving the problem with more complex agents [13], such as the pursuit-evasion problem of aircraft. Qiuhua et al. [14] and Pontani and Conway [15] studied the optimal control strategies and solution methods for two spacecraft pursuit-evasion

Citation: Gong, Z.; He, B.; Liu, G.; Zhang, X. Solution for Pursuit-Evasion Game of Agents by Adaptive Dynamic Programming. *Electronics* **2023**, *12*, 2595. https://doi.org/10.3390/electronics12122595

Academic Editors: Shengqing Li, Srikanta Patnaik and Jianqi Li

Received: 4 April 2023
Revised: 29 May 2023
Accepted: 5 June 2023
Published: 8 June 2023

problems via a multiple shooting method. Xu and Cai [16] used a genetic algorithm in a game problem to find the Nash equilibrium and obtained the control of two aircraft. There are also applications in multi-agent pursuit-evasion systems in recent research [17–21]. Thus, the offline methods for the PE game problem are becoming increasingly sophisticated. However, the policy obtained offline cannot deal with online emergencies, such as temporarily changing agents' goals.

Solving PE game problems online is increasingly becoming a focus. Werbos et al. [22,23] designed actor-critic structures for implementing algorithms in real time, where the learning mechanisms of the structures are composed of policy evaluation and policy improvement. Bertsekas and Tsitsiklis [24] introduced RL methods of different forms and the policy iteration (PI) and value iteration (VI) methods for discrete-time (DT) dynamic systems are compared, which initially apply the idea of RL to the problem of a control system. Werbos [25,26] developed an RL approach based on VI for feedback control of DT dynamic systems using value function approximation (VFA). It is proven that the VFA method is suitable for finding the optimal control online for DT control problems. However, in the actual scenario, the pursuit and evasion problem mostly formulate the control and game of a continuous system. Vrabie [27] presented a method of adaptive dynamic programming (ADP) that is useful to circumvent differential games and establish PI algorithms for continuous-time (CT) control problems. Noting that the information of the system might need to be completed, Vrabie and Lewis [28] considered different forms of systems to obtain online learning methods via optimum control with incomplete information of various systems. The concept of adaptive dynamic programming was further extended to the field of differential games by Vrabie [29], and the synchronous tuning algorithm was used to achieve the Nash equilibrium. However, the system information about both sides of the game must be obtained completely. Kartal et al. [30] used the synchronous tuning algorithm in the pursuit-evasion game of the first-order system to obtain the capture conditions of agents in the game and reached the Nash equilibrium. Zhang et al. [31] and Li et al. [32] determined the scheme's feasibility in distributed systems. However, in the general differential game problems, the states of agents are usually not used as direct control variables, and hence a system becomes more complex. Furthermore, solving the pursuit-evasion game of the actual scenario in real time without using the whole information of game systems has been a hot research field.

This paper proposes a novel ADP method for online solving the Nash equilibrium policies of two-player pursuit-evasion differential games. The min-max principle is adopted to confirm the Nash equilibrium of the game. As the agents in the game can form an Internet of Things (IoT) system, the real-time control law of each agent is obtained by taking a linear-quadratic cost function in adaptive dynamic programming. To consider the scene when capture happens, we introduce the Lyapunov function. Since most actual systems are continuous, we use the policy iteration algorithm to make the real-time policy converge to the analytical solution of the Nash equilibrium. Moreover, we employ the value function approximation method to calculate the neural network parameters without solving the Hamilton–Jacobi–Isaacs equation directly. The feasibility of the proposed method is demonstrated through simulation results from different scenarios of the pursuit-evasion game. This paper is inspired by recent research in various fields, such as motion coordination in wafer scanners [33], soil-structure interaction [34,35], driving fatigue feature detection [36], H∞ consensus for multiagent-based supply chain systems [37], and reliable and secure communications in wireless-powered NOMA systems [38]. These studies have contributed significantly to advancing real-time control and optimization methods in various applications.

The contributions of the paper are shown as follows:

1. The min-max principle is used to find the analytical solution of Nash equilibrium, and the method's stability is proven by establishing a Lyapunov function for obtaining the capture conditions of the game.

2. By constructing a form of adaptive dynamic programming, the policies of agents in each cycle are obtained through the PI method, and we prove that it converges to the Nash equilibrium.
3. To avoid the inconvenience of solving the HJI equation, we establish a set of functions to approximate the value function. As the neural network parameters converge, the agent's solution in policy iteration is obtained.

The rest of the paper is organized as follows. The dynamic model of the PE game is established in Section 2. We discuss the features of Nash equilibrium in Section 3, and the capture conditions of agents are concerned as different parameters are set. Section 4 executes the adaptive dynamic programming method, which consists of the PI method and the VFA algorithm. The agents' policies are obtained without directly solving the Riccati equations of the PE game. Section 5 presents the simulations of some actual problems. Section 6 concludes the paper and discusses the limitations of the research.

2. Formulation of the Game

Consider a system containing two objects and composing a pursuer-evader couple. The pursuer tries to capture the evader, while the evader tries to escape from being captured.

The pursuit-evasion game in real-time is a typical differential game problem. Here, the motion equation of each participant can be expressed as a couple of differential equations defined in a fixed coordinate system. The game with one pursuer and one evader is a typical zero-sum differential game as the benefits of both sides are mutually exclusive.

$$\dot{x}_p = Ax_p + Bu_p \tag{1}$$

$$\dot{x}_e = Ax_e + Bu_e \tag{2}$$

where x_p, u_p, x_e, and u_e are the state variables and control variables of the two players. Among them, the state variable contains the state information of the players, and there may be various physical quantities representing the operation of the players according to different game systems. To facilitate the subsequent operations in this paper, the state variables here must contain the location information of the agents in each dimension. The control variables contain the elements which realized to control the agents in each dimension.

In the PE game problem, the relative motion state of agents is very important. So, we let δ be the difference in the states between the two agents:

$$\delta = x_p - x_e \tag{3}$$

The pursuer tries to reduce the distance of two agents, which is embedded in δ, while the evader tries to enlarge it. Substituting Equations (1) and (2) into Equation (3) and calculating its derivative with respect to time, we have:

$$\dot{\delta} = A\delta + B(u_p - u_e) \tag{4}$$

For formulating a zero-sum pursuer-evader (PE) game, we construct a performance function with integral form as:

$$J(\delta, u_p, u_e) = \int_0^\infty (\delta^\top Q\delta + u_p^\top R_p u_p - u_e^\top R_e u_e)d\tau \tag{5}$$

where Q is a non-negative definite coefficient matrix. R_p and R_e are both positive definite matrices. In the integral function, $\delta^\top Q\delta$ is the term that measures the relative state of the system (4) and is used to give limits to the distance between agents. $u_p^\top R_p u_p$ and $u_e^\top R_e u_e$ stand for the scales in consumption corresponding to the two agents, which are used for realizing the limitations of the controls.

The value function is given as follows when the agents execute certain policies:

$$V(\delta) = \int_0^\infty (\delta^\top Q\delta + u_p^\top R_p u_p - u_e^\top R_e u_e)d\tau \tag{6}$$

If both pursuer and evader employ their optimal policies along the optimal paths, then the optimal value of the game can be obtained as:

$$V^*(\delta) = \min_{u_p}\max_{u_e} J = \min_{u_p}\max_{u_e}\int_0^\infty (\delta^\top Q\delta + u_p^\top R_p u_p - u_e^\top R_e u_e)d\tau \tag{7}$$

In this paper, the goal is to find out the control policy of each agent. The difficulty of the work lies in finding the numerical solution to each agent's policy, in which the steps of policy iteration and the selection of the appropriate value function approximation are very important. In reinforcement learning, the policy needs some iterative steps. In a continuous system, we adopt adaptive dynamic programming for solving the agents' policies. It makes the value function of this paper different from the end value performance index constructed by Jagat and Sinclair [2] because the end value performance index cannot solve the optimal strategy iteratively. Moreover, the performance index of this paper is selected as a quadratic structure, which pays more attention to the intermediate process of the game, improves the real-time competitiveness of both sides of the game, and facilitates the development of the strategy iteration algorithm as stated in Section 4. The distance between two agents is regarded as the tracking error, which means that both pursuer and evader optimize their policies throughout the game process. It is not only in line with the actual situation but also convenient for solving this problem.

3. Solution of the Pursuit-Evasion Game

In this section, we substitute the dynamic model of the PE game problem to the minimax principle and obtain the analytic Nash equilibrium of the PE game. The cases when capture occur are studied and proven by adopting the Lyapunov function approach.

The PE game of agents is regarded as a kind of differential game, which is settled based on the bilateral optimal control theory. The optimal policies of agents are obtained by using the min-max principle. The differential game refers to a continuous game with a couple of players in continuous-time systems. Each agent tries to achieve its goal and magnify its benefit. The game will end up with every participant achieving the Nash equilibrium policy. Using the minimax theorem, we can ensure that agents' policy is their corresponding optimal policies. When each agent adopts its optimal policy, the Nash equilibrium is achieved. Currently, the condition when optimal policies are adopted is called the saddle point.

In a 2-player PE game problem, the optimal policy of the pursuer tries to minimize the Hamilton function whereas that of the evader tries to maximize it. Therefore, there exists a couple of policies (u_p^*, u_e^*). When the pursuer adopts u_p^* and the evader adopts u_e^*, the game reaches the Nash equilibrium. We call (u_p^*, u_e^*) the saddle point of the game.

The expressions in Equation (6) are the same as the Bellman equation of a zero-sum game. From Equations (1) and (2) and Leibniz's formula, we have:

$$\begin{aligned} H(\delta(t), \nabla V, u_p, u_e) &= \delta^\top Q\delta + u_p^\top R_p u_p - u_e^\top R_e u_e + \dot{V} \\ &= \delta^\top Q\delta + u_p^\top R_p u_p - u_e^\top R_e u_e + \nabla V^\top \dot{\delta} \\ &= \delta^\top Q\delta + u_p^\top R_p u_p - u_e^\top R_e u_e + \nabla V^\top (A(x_p - x_e) + B(u_p - u_e)) \end{aligned} \tag{8}$$

where $H(\delta, \nabla V, u_p, u_e)$ is the Hamiltonian, u_p and u_e are admissible control policies of the pursuer and evader, respectively. ∇V denotes $\frac{\partial V}{\partial \delta}$.

We can obtain the optimal control of each agent according to the stationary condition:

$$\frac{\partial H}{\partial u_p} = 0 \tag{9}$$

$$\frac{\partial H}{\partial u_e} = 0 \tag{10}$$

.

Additionally, the second derivative of the Hamiltonian to u_p and u_e should satisfy:

$$\frac{\partial^2 H}{\partial u_p^2} = 2R_p > 0 \tag{11}$$

$$\frac{\partial^2 H}{\partial u_e^2} = -2R_e < 0 \tag{12}$$

The optimal controls of the agents are obtained as:

$$u_p^* = -\frac{1}{2} R_p^{-1} B^\top \nabla V^* \tag{13}$$

$$u_e^* = -\frac{1}{2} R_e^{-1} B^\top \nabla V^* \tag{14}$$

As the system is invariant for infinite time, the solution of the problem is defined by Equations (13) and (14), in which the value V can solve the following equation analytically:

$$\delta^\top Q \delta + u_p^{*\top} R_p u_p^* - u_e^{*\top} R_e u_e^* + \nabla V^{*\top} (A\delta + B u_p^* - B u_e^*) = 0 \tag{15}$$

Since the pursuit-evasion behavior between two agents becomes a zero-sum game when both agents adopt their optimal policies, which is called the game theoretic saddle point policy, the game will reach the Nash equilibrium at that condition.

Before proving that the game can achieve the Nash equilibrium as per policies Equations (13) and (14), we need to use the properties of the Hamiltonian function of the system, which is demonstrated in Lemma 1.

Lemma 1. *Suppose V^* satisfies the HJI Equation (15), which makes the Hamiltonian $H(\delta(t), \nabla V^*, u_p^*, u_e^*)$ comes to 0. Then, (8) transforms to:*

$$\begin{aligned} &H(\delta(t), \nabla V^*, u_p, u_e) = \\ &\nabla V^{*\top} B((u_p - u_p^*) + (u_e - u_e^*)) + u_p^\top R_p u - u_p^{*\top} R_p u_p^* + u_e^{*\top} R_e u_e^* - u_e^\top R_e u_e \end{aligned} \tag{16}$$

Proof of Lemma 1. Suppose V^* satisfies the HJI Equation (15), which makes the Hamiltonian $H(\delta(t), \nabla V^*, u_p^*, u_e^*)$ comes to 0. Then, (8) transforms to:

$$\begin{aligned} H(\delta, \nabla V, u_p, u_e) \quad &= \delta^\top Q \delta + \nabla V^\top B(u_p^* - u_e^*) + u_p^{*\top} R_p u_p^* - u_e^{*\top} R_e u_e^* + \nabla V^\top B((u_p - u_p^*) + (u_e^* - u_e)) \\ &+ u_p^\top R_p u_p - u_p^{*\top} R_p u_p^* + u_e^{*\top} R_e u_e^* - u_e^\top R_e u_e \\ &= H(\delta, \nabla V, u_p^*, u_e^*) + \nabla V^\top B((u_p - u_p^*) + (u_e^* - u_e)) \\ &+ u_p^\top R_p u_p - u_p^{*\top} R_p u_p^* + u_e^{*\top} R_e u_e^* - u_e^\top R_e u_e \end{aligned} \tag{17}$$

If the value function V comes to the optimal value, we have:

$$\begin{aligned} H(\delta, \nabla V^*, u_p, u_e) \quad &= H(\delta, \nabla V^*, u_p^*, u_e^*) + \nabla V^\top B((u_p - u_p^*) + (u_e^* - u_e)) \\ &+ u_p^\top R_p u_p - u_p^{*\top} R_p u_p^* + u_e^{*\top} R_e u_e^* - u_e^\top R_e u_e \end{aligned} \tag{18}$$

According to the HJI function Equation (15), the Hamiltonian comes to 0 as the value function reaches the optimal value, and the proof is completed. □

We can transform the Hamiltonian in the way as demonstrated in Lemma 1 to support the proof of the Nash equilibrium as shown in the following theorem.

Theorem 1. *Consider the dynamics of the agents Equations (1) and (2) with the value function (6). Define V^* as a positive definite solution of the HJI Equation (15). Then, u_p^* and u_e^* in Equations (13) and (14) are the Nash equilibrium policies of agents, and V^* is the optimal value of the PE game.*

Proof of Theorem 1. Suppose V^* satisfies the HJI Equation (15), which makes the Hamiltonian $H(\delta(t), \nabla V^*, u_p^*, u_e^*)$ comes to 0. Then, (8) transforms to:

To prove that u_p^* and u_e^* are the Nash equilibrium solution, we have to confirm that the value function is maximized when the evader executes u_e^* in (13). Similarly, the value function is maximized when the pursuer executes u_p^* in (14), which can be expressed as:

$$u_p^* = \text{argmin} V_{u_p, u_e^*}(\delta(t)) \tag{19}$$

$$u_e^* = \text{argmax} V_{u_p^*, u_e}(\delta(t)) \tag{20}$$

Moreover, let $V_{u_p^*, u_e^*}(\delta(t))$ be the value when the pursuer executes u_p^* and the evader executes u_e^*, we can turn Equations (19) and (20) into inequalities as:

$$V_{u_p^*, u_e^*}(\delta(t)) \leq V_{u_p, u_e^*}(\delta(t)) \tag{21}$$

$$V_{u_p^*, u_e^*}(\delta(t)) \geq V_{u_p^*, u_e}(\delta(t)) \tag{22}$$

where $V_{u_p, u_e}(\delta(t))$ is the solution of the Hamilton function (16). Let $V(\delta(t_0))$ is the initial state of value function. Here, we assume that the capture will happen within the period $t \in [t_0, \infty)$. This indicates $\lim_{x \to +\infty} V_{u_p, u_e}(\delta(t)) = 0$. To verify the establishment of inequalities (21) and (22), we add this term into Equation (8) and have:

$$V(\delta) = \int_{t_0}^{\infty} (\delta^\top Q \delta + u_p^\top R_p u_p - u_e^\top R_e u_e) d\tau + \int_{t_0}^{\infty} \dot{V}_{u_p, u_e} d\tau + V(\delta(t_0)) \tag{23}$$

From Equation (23), obviously we have $V_{u_p^*, u_e^*}(\delta(t)) = V^*(\delta(t_0))$. Upon using lemma 1, (23) becomes:

$$\begin{array}{ll} V_{u_p, u_e}(\delta(t_0)) & = \int_{t_0}^{\infty} (\nabla V^{*\top} B((u_p - u_p^*) + (u_e^* - u_e)) + u_p^\top R_p u_p \\ & -u_p^{*\top} R_p u_p^* + u_e^{*\top} R_e u_e^* - u_e^\top R_e u_e) d\tau + V(\delta(t_0)) \end{array} \tag{24}$$

Let $\varepsilon(V)$ be the integral in Equation (24). We just need to verify that $\varepsilon(V_{u_p, u_e^*}) \geq 0$ and $\varepsilon(V_{u_p^*, u_e}) \leq 0$ to prove (21) and (22). Using (24) we get

$$\begin{array}{ll} \varepsilon(V_{u_p^*, u_e}) & = \int_t^{\infty} (\nabla V^{*\top} B(u_e^* - u_e) + u_e^{*\top} R_e u_e^* - u_e^\top R_e u_e) d\tau \\ & = \int_t^{\infty} -(u_e - u_e^*)^\top R_e (u_e - u_e^*) d\tau \leq 0 \end{array} \tag{25}$$

$$\begin{array}{ll} \varepsilon(V_{u_p, u_e^*}) & = \int_t^{\infty} (\nabla V^{*\top} B(u_p - u_p^*) - u_p^{*\top} R_p u_p^* + u_p^\top R_p u_p) d\tau \\ & = \int_t^{\infty} (u_p^* - u_p)^\top R_p (u_p^* - u_p) d\tau \geq 0 \end{array} \tag{26}$$

which accomplishes the proof. □

Remark 1. *It can be seen from Theorem 2 that the value function does not continue to decrease when it reaches the Nash equilibrium, regardless of how the pursuer unilaterally changes its policy.*

Similarly, no matter how the evader unilaterally changes the policy, the value function will not continue to increase. When (u_p, u_e) reaches the game theoretic saddle point, if one agent changes its policy unilaterally, which is contrary to its benefit, then the other one will reap the benefit from the change. As the game comes to Nash equilibrium, if the pursuer unilaterally alters the strategy, the evader will be harder to capture. On the contrary, if the evader unilaterally changes its policy, it will be easier for the pursuer to realize capture.

In the PE game problem, it is noteworthy whether the pursuer can capture the evader. If so, the problem changes to a finite-time game. Such issues are common in the interception field. Then, we will figure out the conditions which lead to the capture in the game.

The following theorem gives the necessary condition for the occurrence of the capture.

Theorem 2. *Let the pursuer and evader meet the same dynamic model as Equations (1) and (2). Further, let Equations (13) and (14) are the controls of the agents in the game, in which $V(\delta)$ is the analytical solution of the HJI Equation (15). Then, the capture scenario happens only if dynamic (6) is asymptotically stable.*

Proof of Theorem 2. Because $V(\delta)$ solves the HJI Equation (15) analytically, it's obvious that $V(\delta)$ is positive and $V(\delta(t_0)) = 0$. Select function $V(\delta)$ as a candidate of the Lyapunov function. The derivative of $V(\delta)$ is given by:

$$\begin{aligned} \dot{V} &= \nabla V^\top \dot{\delta} = \nabla V^\top (A\delta + B(u_p - u_e)) = -\delta^\top Q\delta - u_p^\top R_p u_p + u_e^\top R_e u_e \\ &= -\delta^\top Q\delta - \tfrac{1}{4}\nabla V^\top B(R_p^{-1} - R_e^{-1})B^\top \nabla V \end{aligned} \tag{27}$$

As we can see, the derivative of the value $\dot{V}$ can be negative under the condition of $R_p^{-1} - R_e^{-1} \geq 0$. That means, if system dynamics (4) is stabilizable and observable, with $R_e - R_p \geq 0$ holds, then dynamic (6) is asymptotically stable, and the capture occurs. On the other hand, if $R_p^{-1} - R_e^{-1} \leq 0$, which fails to meet the Lyapunov stability condition, then the states of the PE game (4) are likely to diverge. Therefore, this will cause the distance between the two agents to enlarge, making the occurrence of the capture impossible. At this moment, the pursuer cannot capture the evader. □

Remark 2. *It can be predicted that when the dynamic of δ is stable, the distance between two agents in the game will approach 0 as time $t \to \infty$. Conversely, if $R_p^{-1} - R_e^{-1}$ is non-positive, the pursuer probably cannot capture the evader. If the capture takes place, as the distance of the two agents is embedded in state variables, the divergence of the positive matrix $R_p^{-1} - R_e^{-1}$ will change the capture time and pattern of the PE game. In value function (6), $u_p^\top R_p u_p$ and $u_e^\top R_e u_e$ stand for the summation of control energy consumption for two players. For the pursuer and evader, R_p and R_e represents dynamic constraints to their control or performance [13], which is known as control penalty. In this way, larger R_e or smaller R_p tends to facilitate the capture scenario occurs.*

Remark 3. *In [39,40], a non-quadratic form Lyapunov function is proposed to verify the system convergence, and the tracking performance is better than that of the quadratic form Lyapunov function. However, the model construction of the pursuit-evasion game involved in this paper focuses more on the physical meaning of the object. The quadratic form of the state variable can represent the relative error, including the relative distance and the relative speed difference. The quadratic form of the control can represent the power of the object after integration. For other systems with relative order 1, the Lyapunov function can be used in the form of $V(\delta) = \delta^{1+\alpha}, 0 < \alpha < 1$ to improve convergence efficiency and make the tracking error approach 0 faster.*

4. Numeric Solution of PE Game by ADP Method

We attain the expressions of the policies adopted by each agent in Section 3 to let the game reach the Nash equilibrium. To obtain the policy employing numerical methods,

relevant researchers [15] introduced genetic algorithms in calculating the controls of each player in the game offline. So far, the offline strategy cannot track the temporary changes of the states of all agents for the online game with continuous-time systems. Therefore, to solve the policies of agents online efficiently, the ADP method is used in this section. As is hard to obtain the gradient terms of the value function, we introduce policy iteration method to form an iteration, and the value function is fitted in the process of VFA.

4.1. Policy Iteration

Now that the value function is an integral, the whole time period is segmented by the concept of ADP, to realize the PI method.

For convenience, the value function of the PE game is simplified as:

$$V(\delta(t)) = \int_t^{\infty} r(\delta(\tau), u_p(\tau), u_e(\tau))d\tau \tag{28}$$

for $r(\delta(\tau), u_p(\tau), u_e(\tau)) = \delta^\top Q\delta + u_p^\top R_p u_p - u_e^\top R_e u_e$.

Let T is an interval, and Equation (6) is expanded as:

$$V(\delta(t)) = \int_t^{t+T} r(\delta, u_p, u_e)d\tau + \int_{t+T}^{\infty} r(\delta, u_p, u_e)d\tau = \int_t^{t+T} r(\delta, u_p, u_e)d\tau + V(\delta(t+T)) \tag{29}$$

It should be noted that T is neither state nor control of the game. Usually, T is regarded as a hyper parameter of ADP. The variation of T may affect the performance and efficiency of the method. Here, we separate the entire period into some segments of intervals and assume $[t, t+T]$ to be the ith interval of the PE game. Moreover, the policies executed by the two agents in $[t, t+T]$ are $u_p^{(i)}$ and $u_e^{(i)}$. Then, we have:

$$V^{(i)}(\delta(t)) = \int_t^{t+T} r(\delta, u_p^{(i)}, u_e^{(i)})d\tau + V^{(i)}(\delta(t+T)) \tag{30}$$

We can obtain the controls of pursuer and evader which should be adopted in the next interval as (30):

$$\begin{aligned} u_p^{(i+1)} &= -\tfrac{1}{2}R_p^{-1}B^\top \nabla V^{(i)} \\ u_e^{(i+1)} &= -\tfrac{1}{2}R_e^{-1}B^\top \nabla V^{(i)} \end{aligned} \tag{31}$$

From (31), the policies are obtained based on the states and controls of the two agents. Matrix A in Equation (4) are absent in the process illustrated. In actual scenarios, there may be unknown parameters in the modeling of various agents. Therefore, the method could be applied to solving online games efficiently.

Equations (30) and (31) formulate a cycle of the PI method. As the iterations execute, the PE game will converge to the Nash equilibrium gradually. We will show the feasibility of this property in Theorem 3.

Theorem 3. *For a PE game, let u_p^0 and u_e^0 as the admissible initial controls of the pursuer and evader. The function $V(\delta)$, and controls u_p and u_e will converge to $V^*(\delta)$, u_p^*, and u_e^*, respectively. The game reaches Nash equilibrium as the controls converge.*

Proof of Theorem 3. Let $V_{u_p^{(i)}, u_e^{(i)}}$ be the value function when pursuer and evader execute the policies $(u_p^{(i)}, u_e^{(i)})$. Set i as the iteration counter. Subtract value function from (24) as the agents execute $(u_p^{(i+1)}, u_e^*)$ and $(u_p^{(i)}, u_e^*)$, respectively, and have:

$$V_{u_p^{(i+1)}, u_e^*}(\delta(t)) - V_{u_p^{(i)}, u_e^*}(\delta(t)) = -\int_t^{\infty} (u_p^{(i+1)} - u_p^{(i)})^\top R_p (u_p^{(i+1)} - u_p^{(i)}) \leq 0 \tag{32}$$

which indicates the function set $V_{u_p^{(i)},u_e^*}(\delta(t))$, $i = 1, \ldots, N$ decreases monotonously. Meanwhile, the set $V_{u_p^*,u_e^{(i)}}(\delta(t))$, $i = 1, \ldots, N$ increases monotonously. As value function (6) is unique for any adopted controls for agents, according to the Dini's Theorem, the value function of any controls $V_{u_p^{(i)},u_e^{(i)}}$ converges uniformly at $V_{u_p^*,u_e^*}$.

Since $V_{u_p^{(i)},u_e^{(i)}}(\delta(t))$ is continuous and differentiable, $\nabla V_{u_p^{(i)},u_e^{(i)}}$ will converge to $\nabla V_{u_p^*,u_e^*}$ as iteration goes. Thus, the policies of both players $(u_p^{(i)}, u_e^{(i)})$ converges to Nash equilibrium policies (u_p^*, u_e^*) as $V_{u_p^{(i)},u_e^{(i)}}(\delta(t))$ converges, which complete the proof. □

Remark 4. *For continuous-time PE games, the whole period is divided into intervals by PI method. As the iteration goes, it can converge to the Nash equilibrium, and the control policy of the agents can also converge to the saddle point. In particular, the method is still available in time-varying PE game problems. If system matrices vary to other admissible values, when the new system is still stable, then it will converge to a new saddle point, which is the Nash equilibrium for the new problem.*

Remark 5. *The PI method does not need to know the system matrix A in obtaining the policies of two players, which indicates that for systems with structures of incomplete information, the method still fulfilled the process of convergence to the Nash equilibrium. However, the state of agents $\delta(t)$ and $\delta(t+T)$ along with their controls need to be known at each step.*

4.2. Value Function Approximation

For most PE games, the HJI equation is difficult to be solved analytically, or it might not have any analytical solution. Therefore, we use an approximation process to obtain the solution of HJI equation. The method focuses on approximating the value function, which is named the VFA algorithm.

Assume that a linearly independent integration of a set of basis functions $\phi_j(\delta)$ is competent to approximate the value function V, which is expressed as:

$$V(\delta(t)) = \sum_{j=1}^{L} w_j \phi_j(\delta) = w_L^\top \varphi_L(\delta) \tag{33}$$

where L stands for the amount of retained functions, and $\varphi_L(\delta)$ forms a L-dimension vector of basic functions. w_L stands for the neural network parameters to be determined, which is composed of each element $w_j, (j = 1, \ldots, L)$.

Using the above value function approximation (VFA) for the cost function, the HJI equation of policy iteration in Algorithm 1 can be expressed as:

$$w_L^\top \cdot \varphi_L(\delta(t)) = \int_t^{t+T} r(\delta, u_p, u_e) d\tau + w_L^\top \cdot \varphi_L(\delta(t+T)) \tag{34}$$

Since the weight parameter is unknown, its initial value is guessed at the beginning of the iteration process, which produces residual error before the neural network parameters reach the convergence to the optimal value. From (34), the residual error is expressed as:

$$\xi(\delta, T) = \int_t^{t+T} r(\delta, u_p, u_e) d\tau + w_L^\top (\varphi_L(\delta_{t+T}) - \varphi_L(\delta_t)). \tag{35}$$

where $\varphi_L(\delta_{t+T}) = \varphi_L(\delta(t+T))$, $\varphi_L(\delta_t) = \varphi_L(\delta(t))$. The above residual error stands for a temporal difference residual error.

To find the neural network vector $w_L^{(i)}$ that approximates the cost function $V^{(i)}$, The least-square method is used at every iteration. Hence, the weight parameters is adapted to minimize:

$$S = \int \xi^2(\delta, T) d\delta \tag{36}$$

The quadratic integral residual S comes to the minimum as the partial derivative to $w_L^{(i)}$ becomes 0:

$$\int \frac{d\xi}{dw_L^{(i)}} \cdot \xi d\delta = 0 \tag{37}$$

Substituting (35) into (37) and assuming $\rho = \int_t^{t+T} r(\delta, u_p, u_e) d\tau$, we obtain:

$$w_L^{(i)} = \Phi^{-1}\Theta, \tag{38}$$

where

$$\Phi = \int (\varphi_L(\delta_{t+T}) - \varphi_L(\delta_t)) \cdot (\varphi_L(\delta_{t+T}) - \varphi_L(\delta_t)^\top d\delta,$$

and

$$\Theta = \int (\varphi_L(\delta_{t+T}) - \varphi_L(\delta_t))\ \rho\ d\delta.$$

Though the classical ADP method can also solve such game problems, though in the process of policy iteration, the basis function signal composed of state variables is required to meet a persistence excitation condition. Thus, there exists a positive constant that holds the following condition:

$$\int_t^{t+T} \dot{\varphi}_L \dot{\varphi}_L^\top d\tau \succeq cI. \tag{39}$$

Equation (39) is the general definition of maintaining persistence excitation for input signals, where φ_L should be a continuously differentiable bounded function.

However, as the times of iterations increase, although we require that each basis function in the basis function vector is linearly independent if the state input is minimal, the basis function φ_L may approach 0. In addition, the base function φ_L tends to become weaker with the convergence of the state, and ultimately cannot form an effective excitation. Currently, it is necessary to introduce an additional excitation signal δ_d to promote the iteration and prevent the divergence of the states. Here, the excitation signal δ_d should hold Equation (40):

$$\int_t^{t+T} \dot{\varphi}_L(\delta_d) \dot{\varphi}_L^\top(\delta_d) d\tau \succeq cI. \tag{40}$$

In this case, Φ^{-1} will not be incomputable as the states converge, making the updates of the weight parameter w_L more credible.

We summarize the policy iteration method by using VFA in Algorithm 1.

Algorithm 1 Policy Iteration Algorithm Using Value Function Approximation

Step 1 Select 0 and 0 as policies for pursuer and evader, respectively.

Step 2 Approximate the value function in each interval using a set of admissible basis functions as follows:

$V(\delta(t)) = \sum_{j=1}^{L} w_j \phi_j(\delta) = w_L^\top \varphi_L(\delta)$

Step 3 Define an additional excitation signal to guarantee that the input holds the persistence of excitation.

Step 4 Determine the weight parameter $w_L^{(i)}$ by minimizing the quadratic integral residual of the HJI equation as

$w_L^{(i)\top} \varphi_L(\delta(t)) = \int_t^{t+T} r\left(\delta, u_p^{(i)}, u_e^{(i)}\right) d\tau + w_L^{(i)\top} \varphi_L(\delta(t+T))$

Step 5 Update the control variable using the following:

$u_p^{(i+1)} = -\frac{1}{2} R_p^{-1} B^\top \nabla \varphi_L^\top w_L^{(i)}$

$u_e^{(i+1)} = -\frac{1}{2} R_e^{-1} B^\top \nabla \varphi_L^\top w_L^{(i)}$

Step 6 Stop if the weight parameter converges; else turn to step 3.

Algorithm 1 introduces VFA based on the PI method, which adopts a numerical way of solving the HJI equation. Neural network parameters are approximated at every iteration

step using the patch least-square method. The policies of the players are obtained as the weight parameters are determined. According to Theorem 3, if the basis function is competent to approximate the value function, then the weight parameters will converge as soon as the games reach the Nash equilibrium.

Remark 6. *The VFA algorithm is embedded in the PI method in solving solutions of the agents. In the batch least squares algorithm, n has a minimum value in every iteration step, which should be no less than L (the scale of neural network parameters). Otherwise, the solution of the PE game is probably divergent, and the value function V would be impossible to be approximated, resulting in fatal errors in the VFA algorithm [27].*

5. Numerical Simulation

In this section, the pursuit-evasion game is numerically simulated. Based on the general motion model, the pursuit and escape the problem of the second-order system is studied, which considers the acceleration of both players along all the dimensions as controls. The position and velocity of the agents are monitored online as the state variables.

Consider the PE game problem in a two-dimensional space whose dynamic model would be:

$$\begin{cases} \dot{s}_{px} = v_{px} \\ \dot{s}_{py} = v_{py} \\ \dot{v}_{px} = a_{px} \\ \dot{v}_{py} = a_{py} \end{cases} \tag{41}$$

$$\begin{cases} \dot{s}_{ex} = v_{ex} \\ \dot{s}_{ey} = v_{ey} \\ \dot{v}_{ex} = a_{ex} \\ \dot{v}_{ey} = a_{ey} \end{cases} \tag{42}$$

where s_{px}, s_{py}, v_{px}, and v_{py} are the coordinates and velocities of the pursuer in x and y directions, respectively. Similarly, s_{ex}, s_{ey}, v_{ex}, and v_{ey} are the coordinates and velocities of the evader in x and y directions, respectively. As for the controls, $(a_{px},\ a_{py})$, and $(a_{ex},\ a_{ey})$ are the accelerator couples of the two agents, which stand for the policies of the two agents, respectively.

Here, we subtract model (41) from (42), and obtain the system of difference (43), whose state variables are $\delta = [l_x, \Delta v_x, l_y, \Delta v_y]$. Among them, l_x and l_y stand for the distance in x and y direction, respectively. The complete system of difference model is:

$$\begin{cases} \dot{l}_x = \Delta v_x \\ \Delta\dot{v}_x = a_{px} - a_{ex} \\ \dot{l}_y = \Delta v_y \\ \Delta\dot{v}_y = a_{py} - a_{ey} \end{cases} \tag{43}$$

The distance between the two agents can be regarded as the capture condition of the PE game problem, which is given as follows:

$$l = \sqrt{(s_{px} - s_{ex})^2 + (s_{py} - s_{ey})^2} \tag{44}$$

To determine whether the pursuer can catch up with the evader, set d as the capture radius. When the distance between the two agents is lower than d, we can call it an effective capture, which terminates the PE game.

In this process, the velocity of agents is unconstrained and effect-less in the benefits of agents, so matrix Q in value function (6) can be $Q = diag([1, 1, 0, 0])$.

Generally, the basic functions in the VFA algorithm are made up of the Kronecker product of the quadratic polynomial terms $\{\delta_i * \delta_j\}_{i,j=1,\ldots,4}$. However, for game problems with more states, this definition will make the calculation inefficient. To improve the

operation efficiency and obtain the policies of the agents, we construct a single-layer neural network as follows:

$$V = \sum_{k=1}^{6} w_k \phi_k = w_1 l_x^2 + w_2 l_x \Delta v_x + w_3 \Delta v_x^2 + w_4 l_y^2 + w_5 l_y \Delta v_y + w_6 \Delta v_y^2 \tag{45}$$

Then, the parameters w_k are updated online through the algorithm introduced in Chapter 4. The initial value of the parameter is selected as $w_L^{(0)} = [1.6; 1.2; 1.6; 1.4; 1.4; 1.2]$. As the input changes in real time, the residual error can be calculated according to Equation (35). Then, as we minimize the quadratic integral residual, we can obtain the updated parameter vector $w_L^{(i+1)}$ by Equation (38).

To keep the base function vector in a persistence excitation condition, the excitation function δ_d is defined as $\delta_d = 0.1\left[\sin(t) + \sin(0.5t) \quad \sin(2t) \quad \sin(t) + \sin(0.5t) \quad \sin(2t)\right]^\top$. The selection of the excitation function δ_d is set to hold Equation (40).

Other initial states are set in the following simulation as $x_{p0} = [3; 1; -2; -1]$, $x_{e0} = [13; 12; 0; 4]$, $R_p = 0.3$, $R_e = 1$, $a_{e0} = a_{p0} = 0$.

Set the capture radius to $d = 0.2$ m and begin to simulate the PE game problem. The locations of each agent vary as shown in Figure 1.

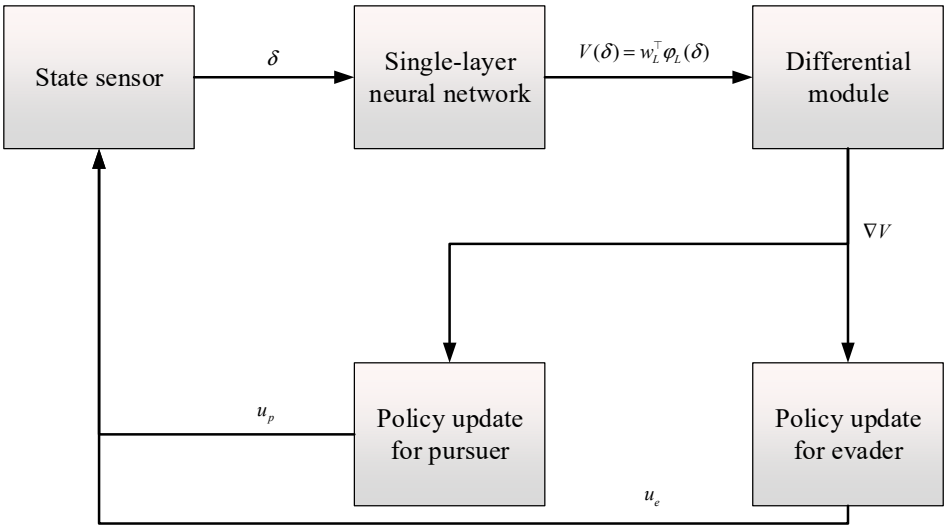

Figure 1. Information transmission of Algorithm 1.

In this problem, the policies of the agents are their accelerations. After the game starts, even if the pursuer and the evader initially move in different directions concerning each other, the pursuer can still adjust its policy as soon as possible and accelerate its velocity in the direction of the evader. Meanwhile, the evader can adjust its policy in time to escape from being caught. However, the evader is still captured for more stringent constraints of the control effort. The capture occurs at $t_c = 2.95$ s and the coordinates of the pursuer and evader are $(19.6235,\ 34.7588)$ and $(19.7570,\ 34.8126)$ respectively.

The distance between the two agents is shown in Figure 2.

As the iteration continues, the parameters of the neural network convergence as the game reaches the Nash equilibrium, which indicates that the policies of both agents have converged to the optimized values. The distance of the two agents when capture scenario happens is 0.1440 m. In this simulation, the policies of both agents are updated at the end of each iteration cycle. The interval of PI method is set as $T = 0.05$ s. It can be seen in Figure 2 that the policy obtained by the PI method can almost converge to the analytical solution of the Nash equilibrium, and the capture is made in the nick of the Nash equilibrium by using the PI method.

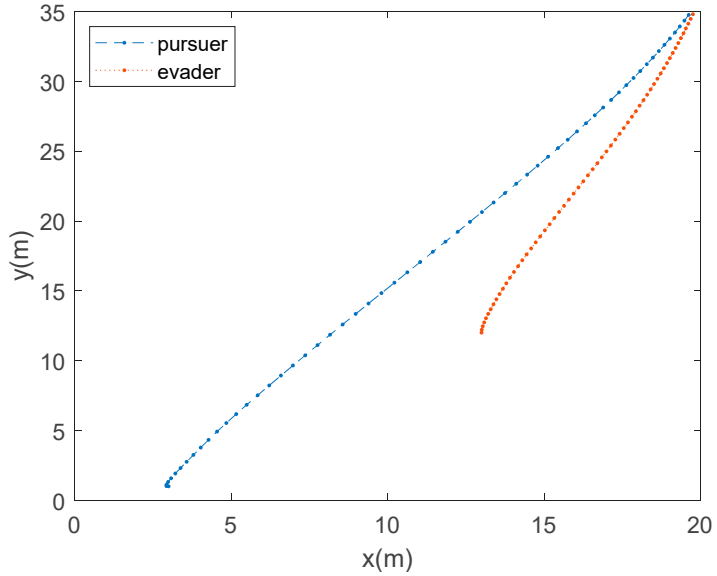

Figure 2. Location of the agents in the game.

Compared with the online PSO algorithm, the policies obtained by the ADP algorithm are closer to the analytical Nash equilibrium, reflecting its better convergence performance compared to the online PSO algorithm. This can be better reflected as the number of iterations increases over time.

According to Algorithm 1, the initial values of the network parameters are given arbitrarily. During the whole process of the PE game, six complete cycles of the PI method are executed. The neural network parameters of the VFA algorithm converge gradually, stabilizing to a set of fixed amounts from the third cycle to the termination. Moreover, to verify that the converged neural network parameters are not locally optimal, we compare the analytical value to our solution, shown in Figure 3. Each parameter converges to its analytical solution, which reflects the stability of the algorithm in solving the problem.

As mentioned above, matrix R is the soft constraint of the agents, which is determined by the actual structure parameters. Different values may lead to various endings for the PE game problem. Now we change the value of R_p to repeat the simulation above and keep the initial states unchanged. The values of R_p are taken as 0.3, 0.5, and 0.8. Then, using Algorithm 1, the obtained distance of the two agents in the PE game is shown in Figure 4.

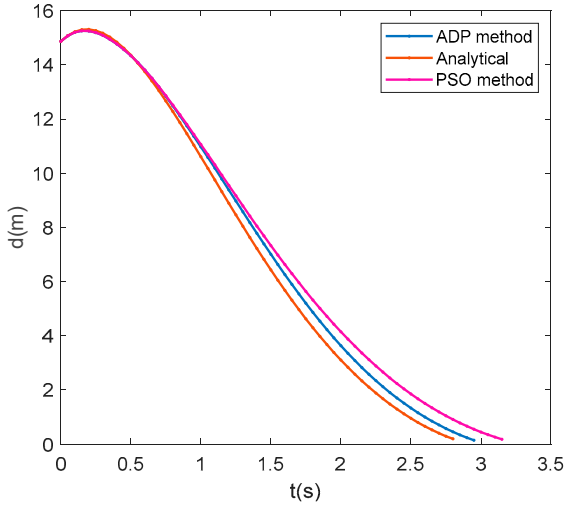

Figure 3. Distance of the agents in the game.

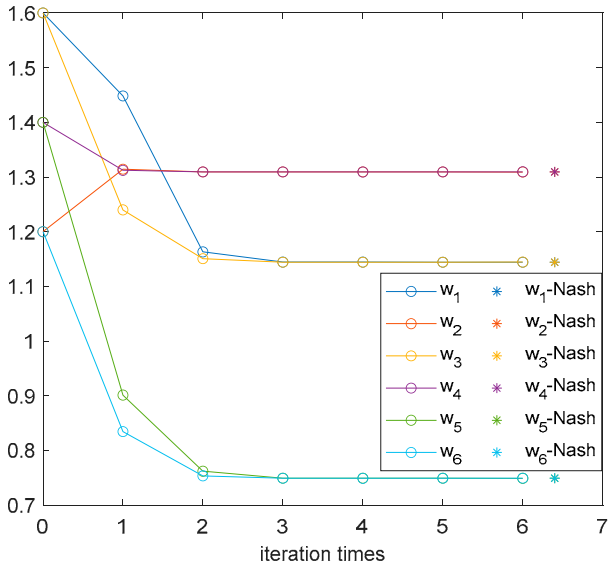

Figure 4. The convergence of neural network parameters.

When R_p is less than R_e (when the PE game system remains stable), the closer the two values, the longer the time required for the pursuer to catch the evader. Therefore, R matrix can be regarded as the motion performance limitation index of agents. The smaller the R value, the better its motion performance, and the wider the value of its control.

Moreover, the interval T in the PI method at each step can influence the performance of the solution we obtain. The computing ability plays an important role in setting an appropriate T. Here, we set different PI method intervals $T = 1$ s, 0.5 s and 0.025 s to recompute the game problem. Note that choosing $R_p = 0.5$, $R_e = 1$, and remaining other initial states and parameters unchanged, we recompute the PE game to obtain the distance of the two agents, shown in Figure 5. The parameters w_i, $i = 1, \ldots, 6$ are shown in Figure 6.

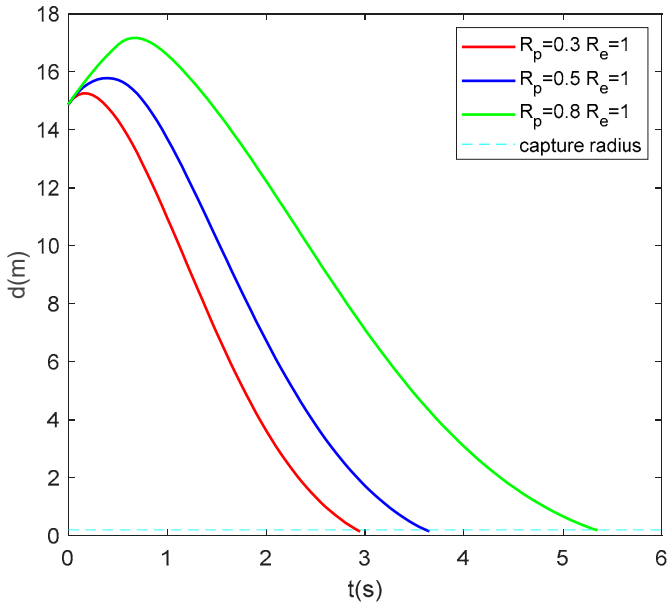

Figure 5. Distance between agents for different R_p.

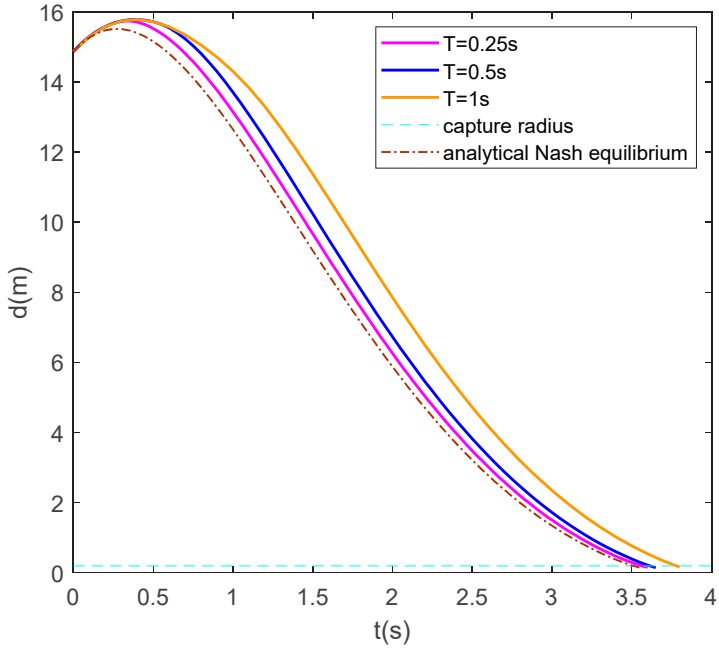

Figure 6. Distance of pursuer and evader for different PI method interval T.

It can be seen from Figure 6 that a shorter interval of the PI method can boost the efficiency in the convergence of the Nash equilibrium. However, a shorter interval led to more iteration cycles to terminate the game, which indicates that the calculation cost grows as the amount of iteration cycles increases. Figure 7 shows that the neural network parameters converge to the analytical solution corresponding to the Nash equilibrium for the intervals of $T = 0.1$ s and $T = 0.5$ s, respectively. The parameter w_L occasionally deviates from the analytical solution in the simulation when $T = 0.25$ s, which means that a too small iteration interval may cause the parameter to diverge. It is beneficial to select a moderate iterative interval according to the conditions of the agents.

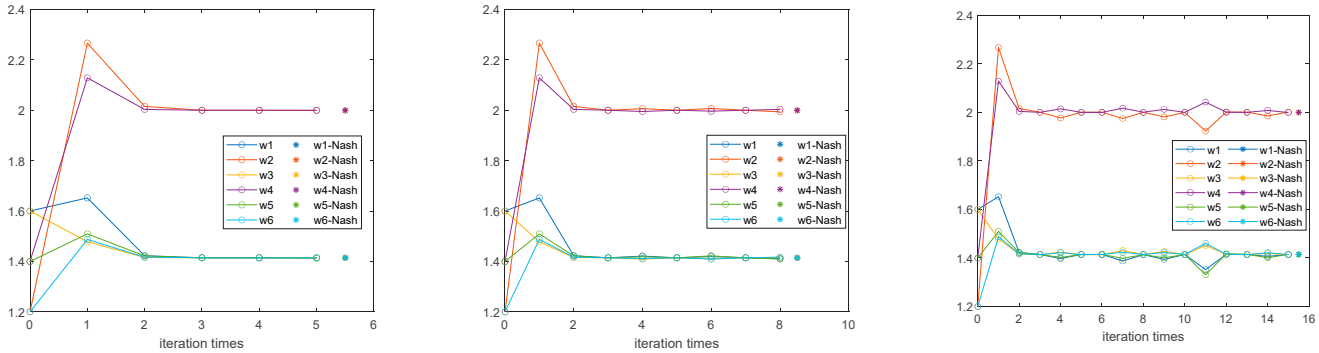

Figure 7. Parameters when $T = 1$ s, 0.5 s, 0.25 s.

Now we consider that the mobility of the pursuer is less than that of the evader, i.e., there is no capture in the game process. Currently, the solution to the game problem does not meet the condition of the system stability. Therefore, the policies of both agents, i.e., the scale of control, may diverge. Let $R_p = 0.3$, and $R_e = 0.24$. We impose hard constraints to the control on both sides, which is $\left\|a_{px}, a_{py}, a_{ex}, a_{ey}\right\|_1 \leq 100$. The motion trajectories of both sides of the game are shown in Figure 8.

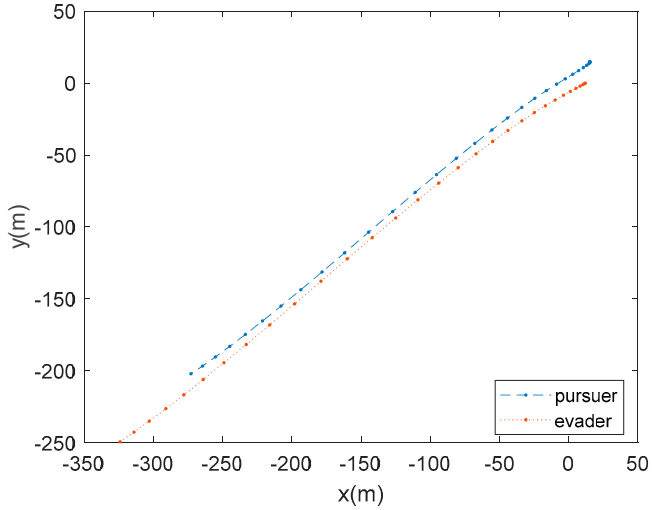

Figure 8. Distance when the system diverges.

Figure 8 shows that the distance between two agents increases further with time elapse. Therefore, it is also proved in Theorem 2 that when $R_e - R_p \leq 0$, then the capture may not exist. At this moment, the state variables of both agents diverge, and the pursuer cannot catch the evader.

6. Conclusions

In this paper, the solution of a two-player pursuit-evasion game is discussed. Through the minimax principle, the analytic solution of Nash equilibrium is obtained, and the necessary condition which causes the occurrence of capture is discussed. The PI method is adopted in solving the PE game online, and the VFA algorithm is adopted to prevent possible inconvenience in dealing with the HJI equation. There is no need to know the system matrix to obtain the policies, and the game approaches the analytical Nash equilibrium solution, which is verified in the simulation.

In the future, we will study more complex PE game problems with more agents. The case when coupling exists between state variables or control variables also deserves further study.

Author Contributions: Conceptualization, B.H.; Methodology, Z.G.; Software, X.Z.; Investigation, G.L.; Resources, B.H.; Writing—original draft, Z.G.; Visualization, G.L.; Supervision, X.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gong, H.; Gong, S.; Li, J. Pursuit–evasion game for satellites based on continuous thrust reachable domain. *IEEE Trans. Aerosp. Electron. Syst.* **2020**, *56*, 4626–4637. [CrossRef]
2. Jagat, A.; Sinclair, A.J. Nonlinear control for spacecraft pursuit-evasion game using the state-dependent riccati equation method. *IEEE Trans. Aerosp. Electron. Syst.* **2017**, *53*, 3032–3042. [CrossRef]
3. Zhang, K.; Wang, P.; Zhou, J. Distributed nash equilibrium seeking for noncooperative games in nonlinear multi-agent systems: An event-triggered neuro-adaptive approach. *Asian J. Control* **2021**, *24*, 605–613. [CrossRef]
4. Li, C.; Li, S.; Zhang, A.; He, Q.; Liao, Z.; Hu, J. Meta-learning for few-shot bearing fault diagnosis under complex working conditions. *Neurocomputing* **2021**, *439*, 197–211. [CrossRef]
5. Ye, D.; Shi, M.; Sun, Z. Satellite proximate pursuit-evasion game with different thrust configurations. *Aerosp. Sci. Technol.* **2020**, *99*, 105715. [CrossRef]
6. Isaacs, R. *Games of Pursuit*; Rand: Santa Monica, CA, USA, 1951.

7. Shima, T. Optimal cooperative pursuit and evasion strategies against a homing missile. *J. Guid. Control Dyn.* **2011**, *34*, 414–425. [CrossRef]
8. Turetsky, V.; Glizer, V.Y. Open-loop solution of a defender–attacker–target game: Penalty function approach. *J. Control Decis.* **2019**, *6*, 166–190. [CrossRef]
9. Friedman, A. *Differential Games*; Courier Corporation: Phoenix, AZ, USA, 2013.
10. Lukes, D.L. Equilibrium feedback control in linear games with quadratic costs. *SIAM J. Control Optim.* **1971**, *9*, 234–252. [CrossRef]
11. Lopez, V.G.; Lewis, F.L.; Wan, Y.; Sanchez, E.N.; Fan, L. Solutions for multiagent pursuit-evasion games on communication graphs: Finite-time capture and asymptotic behaviors. *IEEE Trans. Autom. Control* **2019**, *65*, 1911–1923. [CrossRef]
12. Yu, Z. Linear–quadratic optimal control and nonzero-sum differential game of forward–backward stochastic system. *Asian J. Control* **2011**, *14*, 173–185. [CrossRef]
13. Faruqi, F.A. *Differential Game Theory with Applications to Missiles and Autonomous Systems Guidance*; John Wiley & Sons: Hoboken, NJ, USA, 2017. [CrossRef]
14. Zhang, Q.; Sun, S.; Chen, Y. Strategy and numerical solution of pursuit-evasion with fixed duration for two spacecraft. *J. Astronaut.* **2014**, *35*, 537–544.
15. Pontani, M.; Conway, B.A. Optimal interception of evasive missile warheads: Numerical solution of the differential game. *J. Guid. Control Dyn.* **2008**, *31*, 1111–1122. [CrossRef]
16. Xu, X.; Cai, Y. Design and numerical simulation of a differential game guidance law. In Proceedings of the 2016 IEEE International Conference on Information and Automation (ICIA), Ningbo, China, 1–3 August 2016; pp. 314–318.
17. Talebi, S.P.; Werner, S. Distributed kalman filtering and control through embedded average consensus information fusion. *IEEE Trans. Autom. Control* **2019**, *64*, 4396–4403. [CrossRef]
18. Zhang, Z.; Yan, W.; Li, H. Distributed optimal control for linear multiagent systems on general digraphs. *IEEE Trans. Autom. Control* **2020**, *66*, 322–328. [CrossRef]
19. Li, C.; Li, S.; Zhang, A.; Yang, L.; Zio, E.; Pecht, M.; Gryllias, K. A Siamese hybrid neural network framework for few-shot fault diagnosis of fixed-wing unmanned aerial vehicles. *J. Comput. Des. Eng.* **2022**, *9*, 1511–1524. [CrossRef]
20. Zhou, J.; Wu, X.; Lv, Y.; Li, X.; Liu, Z. Recent progress on the study of multi-vehicle coordination in cooperative attack and defense: An overview. *Asian J. Control* **2021**, *24*, 794–809. [CrossRef]
21. Li, Z.; Zhu, H.; Luo, Y. An escape strategy in orbital pursuit-evasion games with incomplete information. *Sci. China Technol. Sci.* **2021**, *64*, 559–570. [CrossRef]
22. Werbos, P.J.; Miller, W.; Sutton, R. A menu of designs for reinforcement learning over time. In *Neural Networks for Control*; MIT Press: Cambridge, MA, USA, 1990; Volume 3, pp. 67–95.
23. Werbos, P.J. Intelligence in the brain: A theory of how it works and how to build it. *Neural Netw.* **2009**, *22*, 200–212. [CrossRef]
24. Bertsekas, D.P.; Tsitsiklis, J.N. Neuro-dynamic programming: An overview. In Proceedings of the 1995 34th IEEE Conference on Decision and Control, New Orleans, LA, USA, 13–15 December 1995; Volume 1, pp. 560–564.
25. Werbos, P. Approximate dynamic programming for realtime control and neural modelling. In *Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches*; Van Nostrand Reinhold: New York, NY, USA, 1992; pp. 493–525.
26. Werbos, P.J. Neural networks for control and system identification. In Proceedings of the 28th IEEE Conference on Decision and Control, Tampa, FL, USA, 13–15 December 1989; pp. 260–265.
27. Vrabie, D. Online Adaptive Optimal Control for Continuous-Time Systems. 2010. Available online: https://rc.library.uta.edu/uta-ir/handle/10106/2083 (accessed on 3 April 2023).
28. Vrabie, D.; Lewis, F. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neural Netw.* **2009**, *22*, 237–246. [CrossRef]
29. Vrabie, D.; Lewis, F. Adaptive dynamic programming algorithm for finding online the equilibrium solution of the two-player zero-sum differential game. In Proceedings of the 2010 International Joint Conference on Neural Networks (IJCNN), Barcelona, Spain, 18–23 July 2010; pp. 1–8. [CrossRef]
30. Kartal, Y.; Subbarao, K.; Dogan, A.; Lewis, F. Optimal game theoretic solution of the pursuit-evasion intercept problem using on-policy reinforcement learning. *Int. J. Robust Nonlinear Control* **2021**, *31*, 7886–7903. [CrossRef]
31. Zhang, K.; Yang, Z.; Liu, H.; Zhang, T.; Basar, T. Fully decentralized multi-agent reinforcement learning with networked agents. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 5872–5881.
32. Li, Y.; Tang, Y.; Zhang, R.; Li, N. Distributed reinforcement learning for decentralized linear quadratic control: A derivative-free policy optimization approach. *IEEE Trans. Autom. Control* **2021**, *67*, 6429–6444. [CrossRef]
33. Song, F.; Liu, Y.; Shen, D.; Li, L.; Tan, J. Learning Control for Motion Coordination in Wafer Scanners: Toward Gain Adaptation. *IEEE Trans. Ind. Electron.* **2022**, *69*, 13428–13438. [CrossRef]
34. Li, J.; Chen, M.; Li, Z. Improved soil–structure interaction model considering time-lag effect. *Comput. Geotech.* **2022**, *148*, 104835. [CrossRef]
35. Hong, Y.; Yao, M.; Wang, L. A multi-axial bounding surface p-y model with application in analyzing pile responses under multi-directional lateral cycling. *Comput. Geotech.* **2023**, *157*, 105301. [CrossRef]
36. Wang, F.; Wang, H.; Zhou, X.; Fu, R. A Driving Fatigue Feature Detection Method Based on Multifractal Theory. *IEEE Sens. J.* **2022**, *22*, 19046–19059. [CrossRef]

37. Li, Q.-K.; Lin, H.; Tan, X.; Du, S. H_∞ Consensus for Multiagent-Based Supply Chain Systems Under Switching Topology and Uncertain Demands. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *50*, 4905–4918. [CrossRef]
38. Cao, K.; Wang, B.; Ding, H.; Lv, L.; Tian, J.; Hu, H.; Gong, F. Achieving Reliable and Secure Communications in Wireless-Powered NOMA Systems. *IEEE Trans. Veh. Technol.* **2021**, *70*, 1978–1983. [CrossRef]
39. Hosseinzadeh, M.; Yazdanpanah, M.J. Performance enhanced model reference adaptive control through switching non-quadratic Lyapunov functions. *Syst. Control Lett.* **2015**, *76*, 47–55. [CrossRef]
40. Tao, G. Model reference adaptive control with L tracking. *Int. J. Control* **1996**, *64*, 859–870. [CrossRef]