An algorithm of pre-trained fuzzy Actor-Critic learning applying in fixed-time space differential game ⋆

Xiao Wang[1]

Beihang University, Beijing, 100191, P. R. China

Peng Shi[2,*]

Beihang University, Beijing, 100191, P. R. China

Howard Schwartz[3]

Carleton University, Ottawa, K1S 5B6, Canada

Yushan Zhao[4]

Beihang University, Beijing, 100191, P. R. China

Abstract

Solving space differential game in an unknown environment, remains a challenging problem. This paper proposes a pre-trained fuzzy Actor-Critic learning algorithm for dealing with the space pursuit-evasion game in fixed time. It is supposed that the research objects are two agents including one pursuer and one evader in space. A virtual environment, which is defined as the known part of the real environment, is utilized for deriving optimal strategies of the pursuer and the evader, respectively. Through employing the fuzzy inference system, a pre-trained process, which is based on the genetic algorithm, is designed to obtain the initial consequent set of the pursuer and the evader. Besides, an Actor-Critic framework is applied to finely learn the suitable consequent set of the pursuer and evader in the real environment. Numerical experimental results validate the effectiveness of the proposed algorithms on improving the ability of the agents to adapt to the real environment.

Keywords: differential game, reinforcement learning, Actor-Critic, fuzzy system

⋆This work is supported by National Natural Science Foundation of China (11572019), Shanghai Innovation Foundation of Spaceflight Technology (SAST2019084) and Key laboratory of Opto-electronic Information Technology, Ministry of Education (Tianjin University), Tianjin, 300072, P. R. China.

*Corresponding author

Email address: shipeng@buaa.edu.cn (Peng Shi)

[1]Ph.D Student, School of Astronautics.

[2]Associate professor, School of Astronautics.

[3]Professor, Department of Systems and Computer Engineering.

[4]Professor, School of Astronautics.

1. Introduction

Since the game theory had been proposed, many scholars had joined the research of the solution methods[1]. Due to the research demand of continuous systems, a kind of game, which was named as differential game, was developed[2, 3]. Nowadays, differential games are usually applied to describe the competitive scenarios of grid walking or ground territory guarding. However, for the vehicles in space, the space differential game is still a challenging problem because of more complex dynamics.

When the differential game came to the field of aircraft combat, the proportional navigation method was attempted[4, 5]. This kind of technique made it possible to deduce a superior strategy of the pursuer in one game. However, it was not suitable for scenarios where both the pursuer and the evader need to derive their advantage strategies, and it was not suitable for the scenario of spacecraft game. In the past, for solving the space differential game, the two-side optimal theory, which was an extension of the traditional optimal theory, was found[6, 7]. In the aerospace pursuit-evasion field, the semi-direct collocation method was studied based on the two-side extremum principle[8]. Further, the semi-direct collocation method was extended to three-dimensional space for solving the spacecraft differential game, where the genetic algorithm was applied to find the initial values of the co-states[9–13]. To give the shortest space interception time, a two-step interception strategy and an open-loop control method was proposed[14]. As for qualitative spacecraft pursuit-evasion problem, the pursuit-evasion barrier was found, and the results of the space differential game were analyzed[15, 16]. However, the optimal strategy, which is strictly depended on the system information, can be only found when the system is totally known. Therefore, generally, the optimal strategy does not have the ability to deal with the uncertainties of the man-made model. In addition, the optimal strategy is open-looped, which makes the pursuer or the evader can not interact with the real environment to adjust the control policy.

Reinforcement learning is one of the machine learning types, which is closely related to the dynamic programming theory and the optimal control theory[17]. The basic idea of reinforcement learning is to map states to actions so as to maximize a numerical reward[18]. Since the classical Q-learning algorithm, which was based on a lookup table, had been widely studied[19, 20], some discrete games can be solved based on the technique of Q-learning and its branches. However, for a real agent which may have continuous states and actions, it will be hard to discrete all actions and states. After introducing neural networks or fuzzy systems to generalize the states and actions, the curse of dimensionality was solved[18]. Based on the technique of space generalization, the reinforcement learning can be introduced to solve many problems in continuous space, such as obstacle avoidance[21], autonomous control[22] and etc[23, 24]. In addition, the reinforcement learning had also been found effective in multi-agent systems[25], which means that it can also been employed in competitive scenarios with continuous multi-agent systems[26, 27]. The Actor-Critic algorithm, which is one of the most active branches of reinforcement learning, plays an important role when applying the learning process into continuous systems because of its ability for dealing with the large-scale space of states and actions. Recent years, the Actor-Critic algorithm has been attempted to solve some typical differential games under the unknown environment[28–31]. One of the typical games is the problem of territory guarding, which is a type of grid walking game on the ground[32]. In addition, the differential game between the pursuer and the evader with the

single control input separately has been considered in [33, 34].

Due to the shortcomings of the optimal strategies which are totally depended on system information, it seems to be potential to re-solve the problem by reinforcement learning, as such a learning method can help the agent optimize its control strategy in an unknown environment. However, unlike the scenarios of grid walking, or pursuit-evasion problems in a limited square of ground, which are relatively easy to solve through reinforcement learning, the situation of state changing with action in outer space is more complicated. Therefore, in space differential game, it will be extremely hard to find the optimal strategies for the agents without any prior information. However, in the real world, we usually know some part of the environment. In order to solve the space differential game with an unknown environment, it is reasonable to find a compromise reinforcement learning method which can make use of the known part to the learning process. In this paper, we propose an innovative algorithm of pre-trained fuzzy Actor-Critic learning(PTFACL), which is based on the Actor-Critic framework. A virtual environment, which is defined as the known part of the unknown environment, can be taken advantage to the solution of the game. Compared with the previous Actor-Critic algorithms, we add a pre-trained process. The pre-trained process is conducted through the genetic algorithm, where the optimal strategies obtained from the virtual environment are set as the training data. The proposed algorithm covers two agents, the pursuer and the evader, and each agent has its x channel and y channel for learning separately. Under the help of the pre-trained process, it makes the learning process easier because of the utilization of known part of the environment information.

The main contributions of this paper are as follows: (1) It is the first time to introduce fuzzy Actor-Critic learning (FACL) into space differential game. In the previous research, the FACL is designed for ground objects with single control output; (2) It is the first time to add a pre-trained process to reinforcement learning for solving space differential game. Because of the difficulty to directly apply reinforcement learning for solving the space differential game, it will be helpful to make use of the known part of the environment.

The structure of this paper is as follows: Section 2 presents the dynamics and the environment statement; Section 3 discusses the fuzzy inference system and its combination with the reinforcement learning for continuous systems; Section 4 applies the pre-trained fuzzy Actor-Critic learning algorithm to the space differential game; Section 5 simulates the proposed algorithm under three scenarios respectively; Finally, Section 6 draws the conclusions.

2. Environment statement

2.1. Dynamics of space differential game

To facilitate the description of the problem, the following coordinate systems are established: (a) Earth centered inertial ($OXYZ$); (b) orbital coordinate system of the spacecraft ($Ox_oy_oz_o$); (c) orbital coordinate system of the virtual host spacecraft ($Ox_ry_rz_r$). In $Ox_ry_rz_r$, the state vector $\boldsymbol{x}_i = [x_i, y_i, z_i, v_i^x, v_i^y, v_i^z]^{\mathrm{T}}$ is denoted for the agent, where $i = P, E$.

It is supposed that the research objects in this paper are one pursuing satellite and one evading satellite, which are also named as space pursuer and evader. Let P and E denote the pursuer and the evader, respectively, where the satellite P aims to track the satellite E, and the satellite E aims to escape from the satellite P. The reference orbit frame, F_o,

is established, where the origin point o is located near the two satellites. The position relationship among the pursuer, the evader, and the virtual host point o is drawn in figure 1.

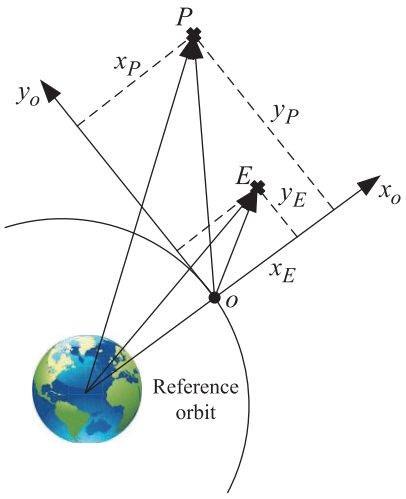


Figure 1: The location of the pursuer and the evader

In this space differential game, the pursuer and the evader can be abstracted as the agents, which have the ability of interacting with the environment. By inputting the current states and the maneuvering strategy, the agent is able to obtain the reward for preparing the correction of its control policy.

This pursuit-evasion game is supposed to occur in the neighborhood of a near circular reference orbit. In addition, it is supposed that there may exist an external disturbance force acting on the pursuer and the evader. Denote the position of satellite P as $\boldsymbol{x}_P = [x_P, y_P, z_P]^{\mathrm{T}}$, while the position of satellite E as $\boldsymbol{x}_E = [x_E, y_E, z_E]^{\mathrm{T}}$.Therefore, the dynamics of the agent is expressed as below $(i = P, E)$[35]:

$$\begin{cases} \dot{x}_i(t) = v_i^x(t) \\ \dot{y}_i(t) = v_i^y(t) \\ \dot{z}_i(t) = v_i^z(t) \\ \dot{v}_i^x(t) = 2\dfrac{\mu}{r^3(t)} x_i(t) + 2\omega(t) v_i^y(t) + \dot{\omega}(t) y_i(t) + \omega^2(t) x_i(t) + T_P u_i^x(t) + d_i^x \\ \dot{v}_i^y(t) = -\dfrac{\mu}{r^3(t)} y_i(t) - 2\omega(t) v_i^x(t) - \dot{\omega}(t) x_i(t) + \omega^2(t) y_i(t) + T_i u_i^y(t) + d_i^y \\ \dot{v}_i^z(t) = -\omega^2(t) z_i(t) + T_i u_i^z(t) + d_i^z \end{cases} \tag{1}$$

where μ represents the Earth's gravitational constant, $\omega(t)$ represents the instantaneous angular velocity of the reference orbit, $r(t)$ represents the instantaneous radius of the orbit, $u_i^j (j = x, y, z)$ represents the force in the corresponding channel and $T_i (i = P, E)$ represents the maximum unit mass thrust of the agent.

2.2. Statement of optimal strategies and the reinforcement learning

As the pursuer aims to track the evader while the evader aims to escape from the pursuer, which is a typical zero-sum two-player differential game, the relative position is always

focused. Therefore, the symbol D is defined to represent the objective function which the pursuer and the evader fight for.

$$D(t) = \frac{1}{2}\left((x_p(t) - x_e(t))^2 + (y_p(t) - y_e(t))^2 + (z_p(t) - z_e(t))^2\right) \tag{2}$$

Further, if we want to describe the specific missions of the pursuer and the evader in details, the cost functions should be introduced. In our case, the initial time is set as t_0, and the terminal time is fixed as t_n. Therefore, the cost function for the pursuer is designed as

$$J = D(t_n) + \int_{t_0}^{t_n} \dot{D}dt = \Phi + \int_{t_0}^{t_n} \ell dt \tag{3}$$

where $\Phi = D(t_n)$ represents the terminal cost and ℓ represents the accumulating cost. Denote the symbols g_x, g_y and g_z as

$$\begin{aligned} g_x &= (x_p(t) - x_e(t)) \\ g_y &= (y_p(t) - y_e(t)) \\ g_z &= (z_p(t) - z_e(t)) \end{aligned} \tag{4}$$

then, the expression of ℓ is shown below.

$$\begin{aligned} \ell &= g_x(\dot{x}_p(t) - \dot{x}_e(t)) + g_y(\dot{y}_p(t) - \dot{y}_e(t)) + g_z(\dot{z}_p(t) - \dot{z}_e(t)) \\ &= g_x\left(v_p^x(t) - v_e^x(t)\right) + g_y\left(v_p^y(t) - v_e^y(t)\right) + g_z\left(v_p^z(t) - v_e^z(t)\right) \end{aligned} \tag{5}$$

Because of the characteristics of the zero-sum game, it is noticed that under the cost function, J, the pursuer aims to minimize the function, and the evader aims to maximize the function at the same time.

$$\boldsymbol{u}_P^* = \min J(\boldsymbol{x}_P, \boldsymbol{x}_E, \boldsymbol{u}_P, \boldsymbol{u}_E, t_0, t_f) \tag{6}$$

$$\boldsymbol{u}_E^* = \max J(\boldsymbol{x}_P, \boldsymbol{x}_E, \boldsymbol{u}_P, \boldsymbol{u}_E, t_0, t_f) \tag{7}$$

When the pursuer and the evader find their optimal strategies, the following condition should be satisfied

$$J(\boldsymbol{u}_P^*, \boldsymbol{u}_E) \leq J(\boldsymbol{u}_P^*, \boldsymbol{u}_E^*) \leq J(\boldsymbol{u}_P, \boldsymbol{u}_E^*) \tag{8}$$

where $\boldsymbol{u}_P^*$ and $\boldsymbol{u}_E^*$ represent the optimal strategy of the pursuer and that of the evader, respectively.

Reinforcement learning is a type of algorithm that interacts with the environment. The agent optimizes its behaviour through the rewards obtained from the environment for maximizing the total benefits. In Markov process, the value function of reinforcement learning can be expressed as

$$\begin{aligned} V &= E\left\{r_{m+1} + \gamma r_{m+2} + \cdots + \gamma^\tau r_{m+\tau+1}\right\} \\ &= E\left\{\sum_{k=0}^{\tau} \gamma^k r_{m+k+1}\right\} \end{aligned} \tag{9}$$

where $\gamma \in [0, 1)$ is the discount factor, m represents the current time, and r_m is the immediate reward which is obtained from the environment. It is seen that reinforcement learning is an algorithm that accumulates the rewards during the learning process. An agent under such a learning framework is expected to get the maximum accumulating discounted rewards.

For the pursuer, to make the agent judge its policy of actions during the learning process, the reward is defined as follows.

$$\begin{aligned} r_{m+1}|_P &= D(m) - D(m+1) \\ r_{m+\tau+1}|_P &= -D(t_n) \end{aligned} \tag{10}$$

It is seen that under the above reward functions, the agent, P, will get positive reward if it decreases the relative distance from the evader E at the next time step. Besides, the agent will get a negative reward if it fails to make the terminal relative distance zero, and the amplitude of the reward depends on the value of the terminal distance. As for the evader, the reward functions are designed below

$$\begin{aligned} r_{m+1}|_E &= -D(m) + D(m+1) \\ r_{m+\tau+1}|_E &= D(t_n) \end{aligned} \tag{11}$$

where the functions are opposite from the functions of the pursuer.

2.3. Optimal strategy solving under the virtual environment

Recall the cost function in the space differential game as below.

$$J = D(t_n) + \int_{t_0}^{t_n} \dot{D} dt = \Phi + \int_{t_0}^{t_n} \ell dt \tag{12}$$

Now it is supposed that we know some part of the real environment because one can build a mathematical model according to the orbital dynamics. The known part of the real environment is defined as a virtual environment, which is used for deriving the optimal strategies for the agents. Define $\hat{\boldsymbol{x}} = [\hat{\boldsymbol{x}}_P, \hat{\boldsymbol{x}}_E]^{\mathrm{T}}$ as the state variable in the virtual environment, where $\hat{\boldsymbol{x}}_P = \left[x_p, y_p, z_p, v_p^x, v_p^y, v_p^z\right]^{\mathrm{T}}$ and $\hat{\boldsymbol{x}}_E = [x_e, y_e, z_e, v_e^x, v_e^y, v_e^z]^{\mathrm{T}}$. By denoting the estimated ω as $\hat{\omega}$, the dynamics of the pursuer and the evader in the virtual environment can be expressed as

$$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{A}\hat{\boldsymbol{x}} + T_P \boldsymbol{B}_P \boldsymbol{u}_P + T_E \boldsymbol{B}_E \boldsymbol{u}_E \tag{13}$$

where

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_P(t) & 0_{6\times 6} \\ 0_{6\times 6} & \boldsymbol{A}_E(t) \end{bmatrix} \tag{14}$$

$$\boldsymbol{A}_P = \boldsymbol{A}_E(t) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\hat{\omega}^2 & 0 & 0 & 0 & 2\hat{\omega} & 0 \\ 0 & 0 & 0 & -2\hat{\omega} & 0 & 0 \\ 0 & 0 & -\hat{\omega}^2 & 0 & 0 & 0 \end{bmatrix} \tag{15}$$

$$\boldsymbol{B}_P = \begin{bmatrix} 0_{3\times 3} \\ I_{3\times 3} \\ 0_{6\times 3} \end{bmatrix} \quad \boldsymbol{B}_E = \begin{bmatrix} 0_{6\times 3} \\ 0_{3\times 3} \\ I_{3\times 3} \end{bmatrix} \tag{16}$$

According to the optimal theory, the Hamiltonian function is defined as

$$\begin{aligned} H &= \boldsymbol{\lambda}^{\mathrm{T}}\dot{\hat{\boldsymbol{x}}} + \ell \\ &= {\boldsymbol{\lambda}_P}^{\mathrm{T}}\left(\boldsymbol{A}_P\hat{\boldsymbol{x}}_P + T_P\boldsymbol{u}_P\right) + {\boldsymbol{\lambda}_E}^{\mathrm{T}}\left(\boldsymbol{A}_E\hat{\boldsymbol{x}}_E + T_E\boldsymbol{u}_E\right) \\ &+ g_x\left(v_p^x(t) - v_e^x(t)\right) + g_y\left(v_p^y(t) - v_e^y(t)\right) + g_z\left(v_p^z(t) - v_e^z(t)\right) \end{aligned} \tag{17}$$

where $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_P, \boldsymbol{\lambda}_E]^{\mathrm{T}}$ represents the co-states, and it also has the following relationship.

$$\boldsymbol{\lambda}_P = \left[\lambda_x^P, \lambda_y^P, \lambda_z^P, \lambda_{v_x}^P, \lambda_{v_y}^P, \lambda_{v_z}^P\right]^{\mathrm{T}} \tag{18}$$

$$\boldsymbol{\lambda}_E = \left[\lambda_x^E, \lambda_y^E, \lambda_z^E, \lambda_{v_x}^E, \lambda_{v_y}^E, \lambda_{v_z}^E\right]^{\mathrm{T}} \tag{19}$$

To find the optimal strategy of the pursuer and that of the evader, it should satisfies that

$$\boldsymbol{u}_P^* = \underset{\|\boldsymbol{u}_P\|\leq 1}{\operatorname{argmin}} H \quad \boldsymbol{u}_E^* = \underset{\|\boldsymbol{u}_E\|\leq 1}{\operatorname{argmax}} H \tag{20}$$

Therefore, the optimal strategies of the pursuer and the evader are expressed below.

$$\begin{cases} u_P^x = -\dfrac{\lambda_{\dot{x}P}}{\sqrt{\lambda^2{}_{\dot{x}P} + \lambda^2{}_{\dot{y}P} + \lambda^2{}_{\dot{z}P}}} \\ u_P^y = -\dfrac{\lambda_{\dot{y}P}}{\sqrt{\lambda^2{}_{\dot{x}P} + \lambda^2{}_{\dot{y}P} + \lambda^2{}_{\dot{z}P}}} \\ u_P^z = -\dfrac{\lambda_{\dot{z}P}}{\sqrt{\lambda^2{}_{\dot{x}P} + \lambda^2{}_{\dot{y}P} + \lambda^2{}_{\dot{z}P}}} \end{cases} \tag{21}$$

$$\begin{cases} u_E^x = \dfrac{\lambda_{\dot{x}E}}{\sqrt{\lambda^2{}_{\dot{x}E} + \lambda^2{}_{\dot{y}E} + \lambda^2{}_{\dot{z}E}}} \\ u_E^y = \dfrac{\lambda_{\dot{y}E}}{\sqrt{\lambda^2{}_{\dot{x}E} + \lambda^2{}_{\dot{y}E} + \lambda^2{}_{\dot{z}E}}} \\ u_E^z = \dfrac{\lambda_{\dot{z}E}}{\sqrt{\lambda^2{}_{\dot{x}E} + \lambda^2{}_{\dot{y}E} + \lambda^2{}_{\dot{z}E}}} \end{cases} \tag{22}$$

The differential equation of the co-states is shown as follows

$$\dot{\boldsymbol{\lambda}} = -\left(\frac{\partial H}{\partial \boldsymbol{x}}\right)^{\mathrm{T}} \tag{23}$$

which can also be written as below.

$$\begin{cases} \dot{\lambda}_x^P = -3\omega^2\lambda_{v_x}^P \\ \dot{\lambda}_y^P = 0 \\ \dot{\lambda}_z^P = \omega^2\lambda_{v_z}^P \\ \dot{\lambda}_{v_x}^P = -\lambda_x^P + 2\omega\lambda_{v_y}^P - g_x \\ \dot{\lambda}_{v_y}^P = -\lambda_y^P - 2\omega\lambda_{v_x}^P - g_y \\ \dot{\lambda}_{v_z}^P = -\lambda_z^P - g_z \end{cases} \quad \begin{cases} \dot{\lambda}_x^E = -3\omega^2\lambda_{v_x}^E \\ \dot{\lambda}_y^E = 0 \\ \dot{\lambda}_z^E = \omega^2\lambda_{v_z}^E \\ \dot{\lambda}_{v_x}^E = -\lambda_x^E + 2\omega\lambda_{v_y}^E + g_x \\ \dot{\lambda}_{v_y}^E = -\lambda_y^E - 2\omega\lambda_{v_x}^E + g_y \\ \dot{\lambda}_{v_z}^E = -\lambda_z^E + g_z \end{cases} \tag{24}$$

The terminal condition of the co-states is expressed as

$$\boldsymbol{\lambda}\left(t_f\right)=\left.\left(\frac{\partial \Phi}{\partial \boldsymbol{x}}\right)\right|_{t_f} \tag{25}$$

and the details of the expression are also shown.

$$\begin{cases}\lambda_x^P\left(t_f\right)=x_p|_{t_f}-x_e|_{t_f}=g_x|_{t_f} \\ \lambda_y^P\left(t_f\right)=y_p|_{t_f}-y_e|_{t_f}=g_y|_{t_f} \\ \lambda_z^P\left(t_f\right)=z_p|_{t_f}-z_e|_{t_f}=g_z|_{t_f} \\ \lambda_{v_x}^P\left(t_f\right)=0 \\ \lambda_{v_y}^P\left(t_f\right)=0 \\ \lambda_{v_z}^P\left(t_f\right)=0\end{cases} \quad \begin{cases}\lambda_x^E\left(t_f\right)=-g_x|_{t_f} \\ \lambda_y^E\left(t_f\right)=-g_y|_{t_f} \\ \lambda_z^E\left(t_f\right)=-g_z|_{t_f} \\ \lambda_{v_x}^P\left(t_f\right)=0 \\ \lambda_{v_y}^P\left(t_f\right)=0 \\ \lambda_{v_z}^P\left(t_f\right)=0\end{cases} \tag{26}$$

Therefore, the original problem can be described as a two-point boundary value problem(TPBVP).

$$\begin{aligned}\dot{\hat{\boldsymbol{x}}} &= A\left(t\right)\hat{\boldsymbol{x}}+T_P\boldsymbol{u}_P^*+T_E\boldsymbol{u}_E^* \\ \dot{\boldsymbol{\lambda}} &= -\left(\frac{\partial H}{\partial \hat{\boldsymbol{x}}}\right)^{\mathrm{T}}\end{aligned} \tag{27}$$

For this kind of problem, there are mainly two kinds of methods for solving. One is to apply shooting/multiple-shooting method to find the optimal strategies, $\boldsymbol{u}_P^*$ and $\boldsymbol{u}_E^*$. The other method is to transfer the problem into a single-side optimal problem. Then, the Gauss-Lobatto collation method can be applied, which transfers the original problem into a mathematical programming problem. Through these two ways, the TPBVP can be solved.

2.4. Consistency between the optimal strategy and the reinforcement learning

When the pursuer or the evader applies the optimal control, it will utilize the system information to minimize or maximize the cost function J. The cost function, J, will be calculated according to the terminal cost and the accumulating process costs. However, in reinforcement learning, the agent will recognize the current reward as the highest priority, and the future rewards will be discounted. In this way, the agent will discounts the terminal reward most times, which makes an ambiguity between the optimal strategy and the reinforcement learning.

However, in a scene of fixed time, we can set the discount parameter, γ, as the value of one. Therefore, we can make the environment under the optimal strategy and that under the reinforcement learning consistent.

The symbol R_t is defined as the return at time step t in reinforcement learning, which is the goal for an agent to maximize.

$$R_t=r_{t+1}+r_{t+2}+\cdots+r_{t+\tau+1}=\sum_{k=0}^{\tau}r_{t+k+1} \tag{28}$$

Take reward functions of the pursuer as an example, the return can be rewritten as

$$\begin{aligned}
R_t|_P &= r_{t+1}|_P + r_{t+2}|_P + r_{t+3}|_P + \cdots + r_{t+\tau+1}|_P \\
&= r_{t+\tau+1}|_P + [r_{t+1}|_P + r_{t+2}|_P + \cdots + r_{t+\tau}|_P] \\
&= -D(t_n) + [D(t) - D(t+1) + D(t+1) - D(t+2) + \cdots + D(t+\tau-1) - D(t+\tau)] \\
&= -D(t_n) - [D(t+1) - D(t) + D(t+2) - D(t+1) + \cdots + D(t+\tau) - D(t+\tau-1)] \\
&= -D(t_n) - \left[\dot{D}\Big|_t^{t+1} + \dot{D}\Big|_{t+1}^{t+2} + \cdots \dot{D}\Big|_{t+\tau-1}^{t+\tau}\right] \\
&= -D(t_n) - \int_t^{t_n} \dot{D} dt
\end{aligned} \tag{29}$$

If it is set that when the first r_{t+1} is obtained, the initial time is t_0, then, the following relationship is satisfied.

$$\boldsymbol{u}_P^* = \max\{R_t|_P\} = \min\{J\} \tag{30}$$

It is seen that, from the optimal strategy view, the goal of the pursuer is to minimize the cost J, and from the reinforcement learning view, the goal is to maximize the return R_t. For the same reason, we also have the following relationship.

$$\boldsymbol{u}_E^* = \max\{R_t|_E\} = \max\{J\} \tag{31}$$

In this way, it is proved that the environment under the optimal strategy and that under the reinforcement learning are consistent, which makes it possible to utilize the information from the optimal strategy for helping the agent in reinforcement learning.

3. Reinforcement learning in continuous systems

3.1. The fuzzy inference system

In order to apply the reinforcement learning in large-scale continuous state space and avoid the curse of dimensionality, a generalization technique, the zero order Takagi-Sugeno(T-S) fuzzy system, is employed as the approximator. It is assumed that the fuzzy system has L rules and n input variables. The fuzzy inference rule is

$$Rule\ l: \text{IF}\ s_1\ is\ F_1^l, \cdots,\ and\ s_n\ is\ F_n^l \quad THEN\ z_l = \phi_l \tag{32}$$

where $s_i\,(i = 1, \cdots, n)$ represents the ith input of the fuzzy system, F_i^l represents the fuzzy set of the ith input variable, z_l represents the output of the lth rule and ϕ_l represents the consequent parameter. It is noticed that all of the consequent parameters form the consequent set, which is important for an agent (pursuer or evader) to generate its control variable. With h membership functions of each s_i, the output of the fuzzy system is expressed as

$$Z(\boldsymbol{s}) = \frac{\sum_{l=1}^{L}\left[\left(\prod_{i=1}^{n}\mu^{F_i^l}(s_i)\right)\phi^l\right]}{\sum_{l=1}^{L}\left(\prod_{i=1}^{n}\mu^{F_i^l}(s_i)\right)} = \sum_{l=1}^{L}\Psi_l(\boldsymbol{s})\,\phi_l \tag{33}$$

where $\boldsymbol{s} = [s_1, \cdots, s_n]^{\mathrm{T}}$ is the state vector, and $\mu^{F_i^l}$ is the membership function of s_i under the lth rule. In addition, the expression of $\Psi_l(\boldsymbol{s})$ is as follows.

$$\Psi_l(\boldsymbol{s}) = \frac{\prod_{i=1}^{n} \mu^{F_i^l}(s_i)}{\sum_{l=1}^{L}\left(\prod_{i=1}^{n} \mu^{F_i^l}(s_i)\right)} = \frac{\omega_l(\boldsymbol{s})}{\sum_{l=1}^{L} \omega_l(\boldsymbol{s})} \tag{34}$$

However, when the number of membership functions is arising, the burden for calculating will be heavy. Therefore, the applied membership functions here are triangular membership functions for saving the computing cost, which are shown in figure 2.

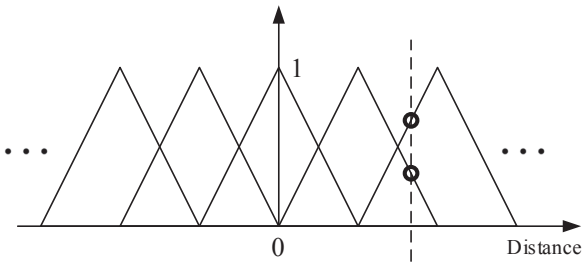


Figure 2: The membership functions for one input

From figure 2, it is seen that for each input, the input will only active two membership functions at one time, which will be beneficial for the case with a large number of membership functions. In this way, the computing cost will be saved. If it is supposed that there are two inputs, then, the inputs will active four membership functions at one time, which is shown in figure 3.

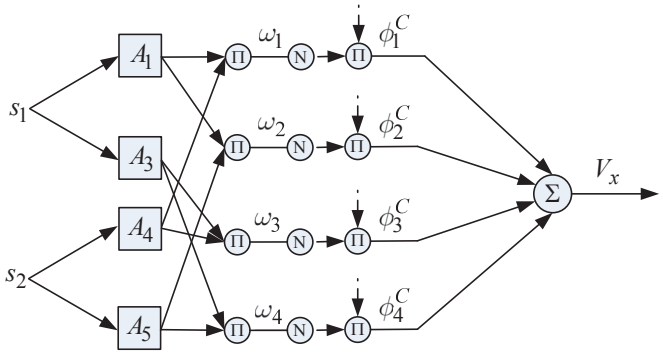


Figure 3: The fuzzy inference system for two inputs

3.2. The fuzzy Actor-Critic learning algorithm

In order to solve the Markov decision problem in continuous space, a type of reinforcement learning algorithm called Adaptive Heuristic Critic(AHC) has been widely studied and applied. In AHC algorithm, the value function and the policy function are approximated respectively. In this way, the learning structure is called Actor-Critic framework. In such a learning algorithm, the Critic part is used to estimate the value function, while the Actor

part is used to generate the action. To generalize the state space and the action space, the Critic part and the Actor part are both composed of T-S systems. To apply temporal difference(TD) learning method, we need two Critic parts for estimating the current value function $\hat{V}_t(\boldsymbol{s}_t)$ and the next value function $\hat{V}_t(\boldsymbol{s}_{t+1})$. The temporal difference can be expressed as below.

$$\Delta_t = r_t + \gamma \hat{V}_t(\boldsymbol{s}_{t+1}) - \hat{V}_t(\boldsymbol{s}_t) \tag{35}$$

Denote Ξ as the variance of the difference signal, which is shown as

$$\Xi = \frac{1}{2}\Delta_t^2 \tag{36}$$

and the adaptive update rule of the parameters in the Critic is expressed as

$$\phi^C(t+1) = \phi^C(t) - \alpha \frac{\partial \Xi}{\partial \phi^C} \tag{37}$$

where α is the learning rate of the Critic.

Furthermore, according to the gradient descent method, it is shown that

$$\frac{\partial \Xi}{\partial \phi^C} = \Delta_t \left[\gamma \frac{\partial V_t(\boldsymbol{s}_{t+1})}{\partial \phi^C} - \frac{\partial V_t(\boldsymbol{s}_t)}{\partial \phi^C} \right] \tag{38}$$

To sum up, we have

$$\phi^C(t+1) = \phi^C(t) - \alpha \left[R_t + \gamma V_t(\boldsymbol{s}_{t+1}) - V_t(\boldsymbol{s}_t) \right] \left[\gamma \frac{\partial V_t(\boldsymbol{s}_{t+1})}{\partial \phi^C} - \frac{\partial V_t(\boldsymbol{s}_t)}{\partial \phi^C} \right] \tag{39}$$

$$\frac{\partial V_t(\boldsymbol{s}_t)}{\partial \phi^C} = \left[\Psi_1(\boldsymbol{s}_t), \Psi_2(\boldsymbol{s}_t), \cdots, \Psi_L(\boldsymbol{s}_t) \right] \tag{40}$$

$$\frac{\partial V_t(\boldsymbol{s}_{t+1})}{\partial \phi^C} = \left[\Psi_1(\boldsymbol{s}_{t+1}), \Psi_2(\boldsymbol{s}_{t+1}), \cdots, \Psi_L(\boldsymbol{s}_{t+1}) \right] \tag{41}$$

Combining with eq.(34), eq.(39) can be solved.

To explore the potential better rewards, a rand noise, σ, will be added to the output of the Actor, u_t.

$$u_c = u_t + \sigma \tag{42}$$

Therefore, the update rule of the output parameter, ϕ^A, is expressed as

$$\phi^A(t+1) = \phi^A(t) + \beta \Delta_t \frac{\partial u_t}{\partial \phi^A}(u_c - u_t) \tag{43}$$

where β is the learning rate of the Actor. The partial derivative of u_t is expressed as follows.

$$\frac{\partial u_t}{\partial \phi^A} = \left[\Psi_1(\boldsymbol{s}_t), \Psi_2(\boldsymbol{s}_t), \cdots, \Psi_L(\boldsymbol{s}_t) \right] \tag{44}$$

4. Pre-trained fuzzy Actor-Critic learning for space differential game

In Section 2.1, the dynamic model employed in our scene, which has been introduced in eq.(1), is selected as the real environment for the agent in reinforcement learning. The target of the agent is to optimize its control strategy under such an environment.

In reinforcement learning, the agent will interact with the environment to adjust its action set, which means that the algorithm is totally model-free. However, unlike some game cases on the ground, the relative state between the pursuer and the evader in the space differential game is unlimited due to the effect of the gravity force from the Earth. Besides, the relative state will be changed with the actions of the agents sensitively, which may also cause unlimited relative states. Therefore, it will be extremely hard to directly solve the space differential game by reinforcement learning without any prior information.

However, in the real world, for space differential game, we actually know some information about the environment. Therefore, in this paper, we propose a pre-trained fuzzy Actor-Critic learning(PTFACL) to make the learning more solvable by making use of the known information.

4.1. Pre-trained process based on the genetic algorithm

To utilize the optimal strategies under the virtual environment, a pre-trained process is needed to generate the set of consequent parameter based on these strategies.

In our case, there are two channels and each channel has two inputs. The two channels are x-channel and y channel, with the inputs $\{x, v_x\}$ and $\{y, v_y\}$, respectively. It is seen that, for each channel, there are two inputs, where the first one is the relative distance and the other is the relative velocity from the evader to the pursuer. It is noticed that from our research, when it is supposed that the pursuer and the evader are on the same orbital plane, the motions of the agents in the orbital plane are always playing important roles, but the motions out of the orbital plane have very weak effect on the results. Therefore, z channel is not recommended in this paper.

Through the two inputs, the corresponding membership functions will be activated. For the relative distance, there exist 13 membership functions, and for the relative velocity, there exist 7 membership functions, which are shown in figure 4 and figure 5, respectively.

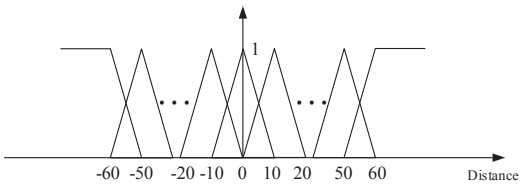


Figure 4: The membership functions for the relative distance

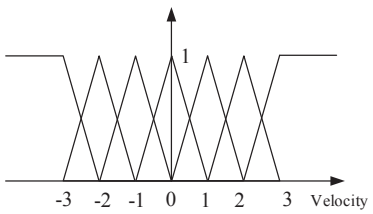


Figure 5: The membership functions for the relative velocity

From figure 4 and figure 5, it is seen that the boundary of the membership functions set is equal to one in case that the value of the input is over the normal covered range. Denote the symbols, ϕ_x^P and ϕ_y^P, to represent the consequent set in x channel and in y channel of the pursuer, and the symbols, ϕ_x^E and ϕ_y^E, to represent the consequent set in x channel and y channel, respectively. The structure of ϕ_x^P, ϕ_y^P, ϕ_x^E and ϕ_y^E is a two-dimension matrix. The row number of the matrix is depended on the number of membership functions of the first input, while the column number is depended on that of the second input. Under the membership functions described in figure 4 and 5, it is clear that those consequent set are 13×7 matrices. Besides, the goal of the pre-trained process is to find the initial values of the set, ϕ_x^P, ϕ_y^P, ϕ_x^E and ϕ_y^E.

Based on the optimal strategies of the pursuer and the evader, it is supposed that we can obtain N pairs of training data. To approximate the training pairs through fuzzy inference system, the genetic algorithm (GA) is applied here to conduct the pre-trained process.

The fitness function during the pre-trained learning is the type of mean square error, which is expressed as below

$$M = \frac{1}{2} \sum_{i=1}^{N} \left(u_A - u_{tr}\left(i\right) \right)^2 \tag{45}$$

where u_A is the output of the fuzzy inference system, and $u_{tr}\left(i\right)$ is the control value of the ith training pair. The GA will be employed in the x, y channel of the pursuer and the x, y channel of the evader, respectively, but it is the same process in different channels or different agents. Therefore, we take the x channel of the pursuer as an example.

The diagram of GA process is described in figure 6. The inputs for GA in x channel are the x and v_x, which will be input to the fuzzy inference system with the membership functions described in figure 4 and figure 5. From figure 6, it is seen that the " chromosome" is a consequent set which is composed of the "genes ". The " genes" are also shown as the consequent parameters. The mean square error, M, is calculated according to the values of u_{tr} from the training data and the values of u_A obtained from the fuzzy inference system. Sorted by the fitness error, the current chromosome will be updated through making the crossover and the mutation on genes. According to the GA technique, which can be found in [36], the consequent set will be optimized to approximate the training data better.

4.2. Re-adapt to the real environment by fuzzy Actor-Critic learning

The proposed learning framework in this paper is single-looped, and covers two agents, the pursuer P and the evader E. In addition, each agent has two channels, x channel and y channel. Each channel has two inputs, the relative distance and the relative velocity of the current channel.

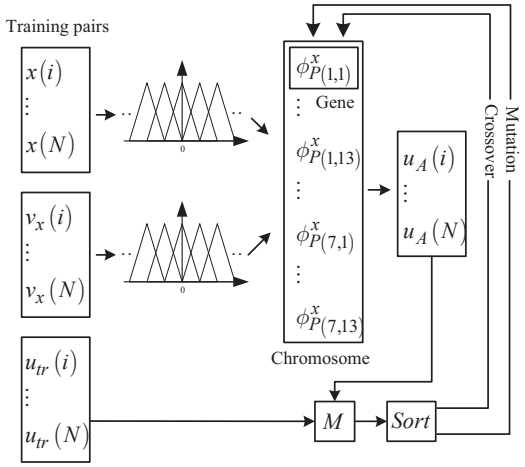


Figure 6: The diagram of the pre-trained process

Take the x channel of the pursuer as an example, its inputs for fuzzy systems are expressed as

$$s_1 = x \qquad s_2 = v_x \tag{46}$$

Denote φ_l as the consequent parameter in the consequent set φ_P^x of the Critic, then, the inference rule is shown as

$$R_l : \text{IF } s_1 \; is \; A_1^l \; and \; s_2 \; is \; A_2^l \;\; THEN \; Z_l = \varphi_l \tag{47}$$

Further, the output can be calculated.

$$\Psi_l(\boldsymbol{s}) = \frac{\prod_{i=1}^{2} \mu^{F_i^l}(s_i)}{\sum_{l=1}^{4} \left(\prod_{i=1}^{2} \mu^{F_i^l}(s_i) \right)} = \frac{\omega_l(\boldsymbol{s})}{\sum_{l=1}^{4} \omega_l(\boldsymbol{s})} \tag{48}$$

$$\hat{V}_P^x = \sum_{l=1}^{4} (\Psi_l) \cdot (\varphi_l) \tag{49}$$

The fuzzy inference process of the Actor part is similar to that of the Critic part and the difference lies in the consequent parameter to each membership degree. Denote ϕ_l as the consequent parameter of the consequent set u_t of the Actor, then the output of the Actor is shown as below.

$$u_t = \sum_{l=1}^{4} (\Psi_l) \cdot (\phi_l) \tag{50}$$

To add a noise σ for exploring, the control variable is expressed as follows.

$$u_P^x = u_t + \sigma \tag{51}$$

The designed reward function, r_t, is expressed as

$$\begin{aligned}
r_t|_P^x &= D_x(t-1) - D_x(t) \\
r_{t_n}|_P^x &= -D_x(t_n) \\
r_t|_P^y &= D_y(t-1) - D_y(t) \\
r_{t_n}|_P^y &= -D_y(t_n) \\
r_t|_E^x &= -D_x(t-1) + D_x(t) \\
r_{t_n}|_P^x &= D_x(t_n) \\
r_t|_P^y &= -D_y(t-1) + D_y(t) \\
r_{t_n}|_P^y &= D_y(t_n)
\end{aligned} \tag{52}$$

where $D_x(t)$ and $D_y(t)$ are the components of $D_t(t)$.

$$\begin{aligned}
D_x(t) &= \frac{1}{2}(x_p(t) - x_e(t))^2 \\
D_y(t) &= \frac{1}{2}(y_p(t) - y_e(t))^2
\end{aligned} \tag{53}$$

The whole diagram of learning logic is illustrated in figure 7. In figure 7, it is shown that

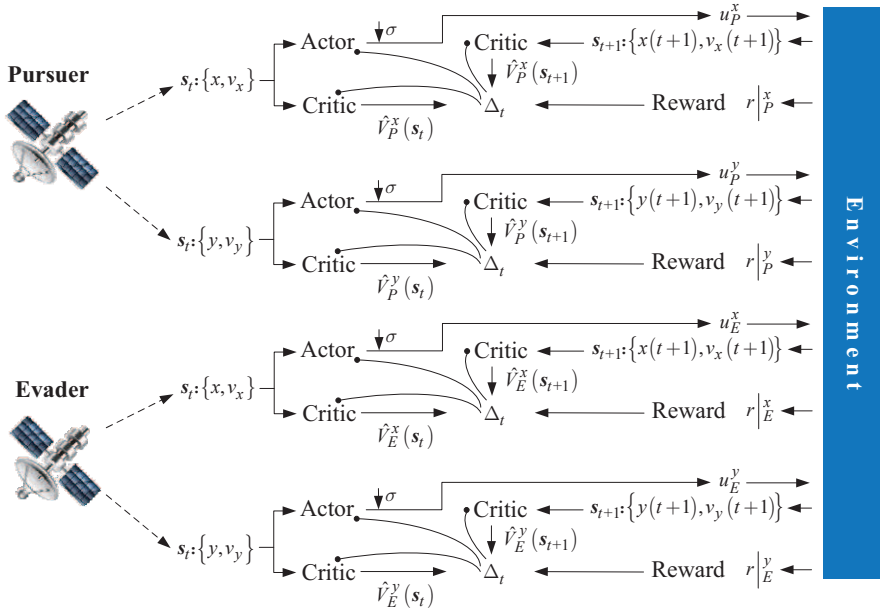


Figure 7: The diagram of learning logic

there are two agents, the pursuer and the evader. Each agent has two channels, x channel and y channel, and each channel has two Critic parts and one Actor part. The two Critic parts are applied to estimate the value of current time, $\hat{V}(t)$, and the value of next time, $\hat{V}(t+1)$. Take the x channel of the agent P as an example. The state of the current time is the combination of x and v_x, which will be input to the Critic part and the Actor part to generate the estimated value $\hat{V}_P^x(s_t)$ and the control variable u_P^x respectively. Acting with

the control $\boldsymbol{u}_P$, the agent will interact with the environment, which means that the next state $\boldsymbol{s}_{t+1}$ and the reward $r|_P^x$ are expected to be obtained. According to $r|_P^x$, $\hat{V}_P^x(s_t)$ and $\hat{V}_P^x(s_{t+1})$, the time difference, Δ_t, is calculated. The consequent parameters of the Critic part and the Actor part can be adjusted according to Δ_t.

To sum up, the proposed learning algorithm of PTFACL is as follows.

Algorithm 1 Pre-trained fuzzy Actor-Critic learning

Maintain the virtual environment
Derive the optimal strategies $\boldsymbol{u}_P^*$ and $\boldsymbol{u}_E^*$ of the pursuer and the evader respectively
Solve the TPBVP problem
for each agent (P and E) **do**
    for each channel (x and y) **do**
        Obtain N pairs of training data
    end for
end for
Initialize the membership functions
Initialize the consequent set of the Actor $\phi_P^x = 0_{13\times7}$, $\phi_P^y = 0_{13\times7}$, $\phi_E^x = 0_{13\times7}$, $\phi_E^y = 0_{13\times7}$
for each agent (P and E) **do**
    for each channel (x and y) **do**
        Initialize a set of chromosomes
        for each iteration **do**
            Calculate the fitness values of all the chromosomes by eq.(45)
            Sort the entire chromosomes according to their fitness values
            Obtain the new generation by crossover and mutation
        end for
    end for
end for
Obtain the initial consequent set ϕ_P^x, ϕ_P^y, ϕ_E^x and ϕ_E^y after the pre-trained process
Initialize the consequent set of the Critic $\varphi_P^x = 0_{13\times7}$, $\varphi_P^y = 0_{13\times7}$, $\varphi_E^x = 0_{13\times7}$, $\varphi_E^y = 0_{13\times7}$
for each episode **do**
    Initialize states of the pursuer and the evader
    for all Time step **do**
        Calculate the output of the Critic $\hat{V}_P^x(\boldsymbol{s}_t)$, $\hat{V}_P^y(\boldsymbol{s}_t)$, $\hat{V}_E^x(\boldsymbol{s}_t)$, $\hat{V}_E^y(\boldsymbol{s}_t)$ from eq.(49)
        Calculate the output of the Actor u_P^x, u_P^y, u_E^x, u_E^y through eq.(50) and eq.(51)
        Interact with the environment
        Obtain the reward $r_t|_P^x$, $r_t|_P^y$, $r_t|_E^x$, $r_t|_E^y$
        Calculate the output of the Critic $\hat{V}_P^x(\boldsymbol{s}_{t+1})$, $\hat{V}_P^y(\boldsymbol{s}_{t+1})$, $\hat{V}_E^x(\boldsymbol{s}_{t+1})$, $\hat{V}_E^y(\boldsymbol{s}_{t+1})$
        Calculate the time difference Δ_t from eq.(35)
        Update φ_P^x, φ_P^y, φ_E^x, φ_E^y and ϕ_P^x, ϕ_P^y, ϕ_E^x, ϕ_E^y according to eq.(37) and eq.(43), respectively
    end for
end for

5. Simulation

A pursuer and an evader in space, which are denoted as P and E are simulated in this paper. It is supposed that the reference orbit is a circular orbit with the radius with 6.9×10^3 km. Denote the symbols, $\boldsymbol{x}_{P0}$ and $\boldsymbol{x}_{E0}$ as the initial states of the pursuer and the evader respectively, where the first three items of the vectors represent the position in m and the last three items represent the velocity in m/s of the agent.

Table 1: Initial states of the pursuer and the evader

State	Value
$\boldsymbol{x}_{P0}$	$\left[-0.4220; -24.0804; 0; 2.678 \times 10^{-2}; -4.715 \times 10^{-5}; 0\right]^{\mathrm{T}}$
$\boldsymbol{x}_{E0}$	$\left[9.91774; 24.1154; 0; -2.678 \times 10^{-2}; -5.608 \times 10^{-3}; 0\right]^{\mathrm{T}}$

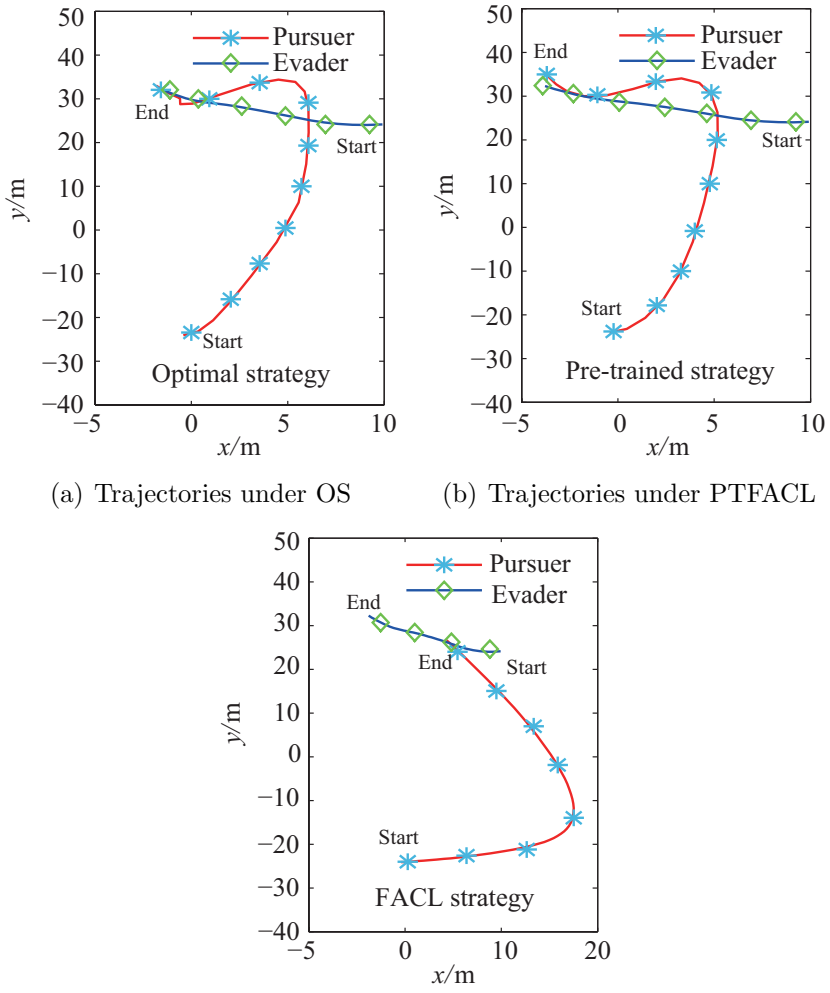
Besides, it is assumed that $T_P = 0.03 \times 9.8 \times 10^{-3}$ and $T_E = 0.01 \times 9.8 \times 10^{-3}$, which means that the pursuer is supposed to have greater maneuver ability than the evader. Because in such a situation that the pursuer has more mobility than the evader, we can judge whether the proposed algorithm is effective by whether the pursuer can change the situation that is not good for itself.

To conduct the simulation, the software Matlab R2013 and an Intel Core i7 computer with a 2.4 GHz clock frequency and 4.0GB of RAM are employed.

5.1. Scenario with 600s : the pre-trained process

To illustrate that how the pre-trained process works, a scenario with 600s is selected where the real environment is set as the same as the virtual environment. The initial positions of the pursuer and the evader are shown in table 1. In this scenario, we want to show the difference between the algorithm with the pre-trained process and the algorithm without this process. Therefore, the trajectory of the evader is fixed. In this way, the control strategy of the pursuer derived from the optimal strategy (OS), and the strategy obtained after pre-trained process (PTFACL), as well as the strategy learned from FACL are drawn and compared, respectively.

The trajectories of the pursuer and the evader under the optimal strategy are shown in figure 8(a) and the trajectories under the pre-trained strategy are shown in figure 8(b). Besides, the control curves of the purser under the three methods are drawn in figure 9. It is seen that the trajectories of P and E under OS and those under PTFACL are similar, which demonstrates that the pre-train process performs well. Therefore, it is effective to transfer the optimal strategies into the the consequent set of the fuzzy inputs of the relative position and the relative velocity. If we do not consider the optimal strategy and the GA process for pre-training, the consequent set are $\phi_P^x = \phi_P^y = \phi_E^x = \phi_E^y = 0_{13\times 7}$. With the learning rate of the Critic, $\alpha = 0.01$, the learning rate of the Actor $\beta = 0.001$, and the random noise $\sigma = 0.05$, the algorithm comes to the FACL, which is applied in figure 8(c). From this figure, we can see that the trajectory trend the pursuer is somehow different from the result under the optimal strategy. From one point, the FACL is a single-looped algorithm which ignores



(a) Trajectories under OS

(b) Trajectories under PTFACL

(c) Trajectories under FACL

Figure 8: Trajectories of the pursuer and the evader in the scenario with 500s

coupling effects between the x and y channel. And from the other point, there will exist error when the fuzzy inference system is applied to deal with the totally unknown environment, because both of the number of the membership functions and the number of inputs will affect on the learning results. Besides, the relative positions between the pursuer and the evader under OS, PTFACL and FACL are compared in figure 10. It is shown that the moving trend under PTFACL is very similar to that under OS because of the effectiveness of the pre-trained process. But in the the terminal error graph, it is also seen that the terminal tracking error under PTFACL is larger than that under OS due to reasons of the GA accuracy and the fuzzy inference accuracy. In addition, the moving trend of the relative position under FACL is also drawn in figure 10, which further expresses the difference between FACL and PTFACL. However, the algorithm still works because it definitely get the moving trend of the evader.

In order to indicate the cost time of the PTFACL and FACL, the values of cost seconds are shown in table 2. From the table, we can see that the PTFACL has taken 230.24s in total, and it has two processes to compose its cost time. One is the OS, which takes 12.14s and the other is the GA process, which takes 218.10s. As for the FACL, it does not have the

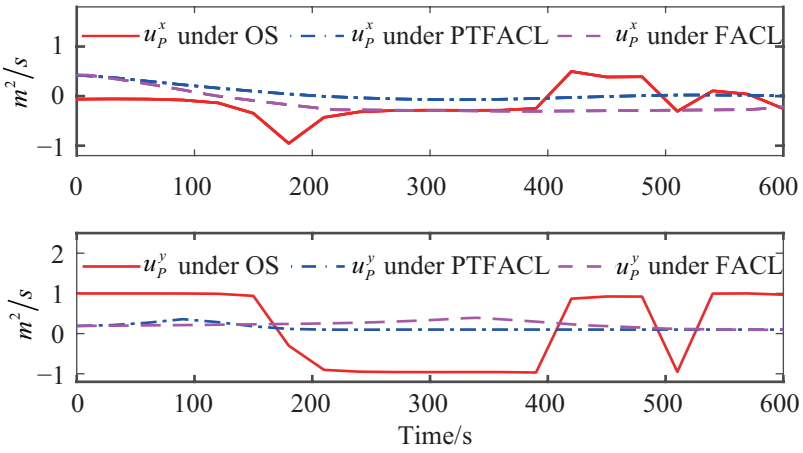


Figure 9: Control curves of the pursuer under OS, PTFACL and FACL

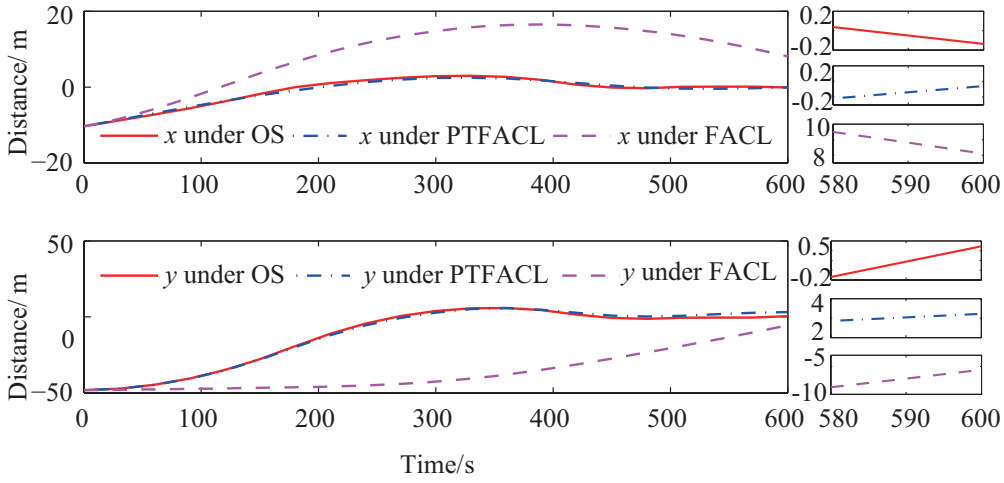


Figure 10: Variations of the relative positions between the pursuer and the evader under OS, PTFACL and FACL

OS process and GA process, and it has taken 426.90s to complete the learning process from the unknown environment without any prior information. Therefore, it is seen that the time cost of PTFACL has saved by 46% compared with FACL because of the utilization of the virtual environment.

If we accumulate all the rewards collected during the flight, the values under the ideal condition, OS, PTFACL and FACL are shown in table 3 respectively. If we suppose that both the pursuer and the evader are smart enough, as the pursuer is given more powerful maneuver ability, there exist the ideal values of the pursuer and the evaders.

From table 3, it is seen that the values of total rewards under OS is nearly the same as the ideal values. The small value difference is due to the calculating accuracy. Compared with the values under the OS, the values of the pursuer under PTFACL are decreased a little because of the accuracy of the GA process and the error of the fuzzy inference systems. Besides, under the FACL, it is seen that the total rewards of the pursuer are much lower than the ideal values, which means that the pursuer cannot track the evader as well as ideal

Table 2: Cost time of PTFACL and FACL in the scenario with 600s

Algorithm	OS Time cost(s)	GA Time Cost(s)	Total Time cost(s)
PTFACL	12.14	218.10	230.24
FACL	—	—	426.90

Table 3: Total rewards under ideal condition, OS, PTFACL and FACL

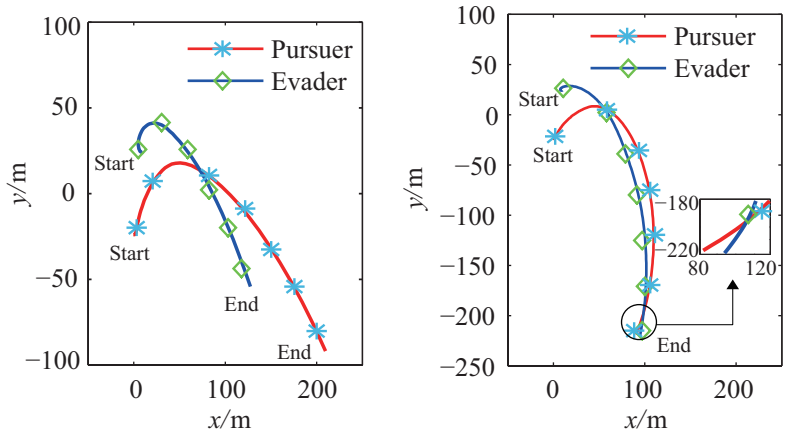
Value	P in x channel	P in y channel	E in x channel	E in y channel
Ideal	53.45	1161.42	-53.45	-1161.42
OS	53.44	1161.41	-53.44	-1161.41
PTFACL	53.43	1151.09	-53.43	-1151.09
FACL	-12.04	1129.89	12.04	-1129.89

expectation, and such a result is also indicated in figure 8(c).

5.2. Scenario with 1500s : the environment has perturbed reference orbit

In this scenario, we suppose that there is a deviation between the real reference orbit and the estimated reference orbit in the virtual environment, where the condition $\omega - \hat{\omega} = 8 \times 10^{-4} rad/s$ exists. In figure 11, the trajectories of the pursuer and the evader with the consequent set obtained from the pre-trained process are shown. From figure 11(a), it is seen that, under such a perturbation, the pursuer still keeps the ability to follow the moving trend of the evader, but the tracking error has been generated because the unsuitable consequent set does no suit well in the real environment. With the learning rate of the Critic, $\alpha = 0.01$, the learning rate of the Actor $\beta = 0.001$, and the random noise $\sigma = 0.1$, the proposed PTFACL is processed. The trajectories of the pursuer and the evader are shown in figure 11(b) after 1640 iterations. In this figure, it shows that the pursuer can track the evader better because of more suitable consequent set. In this process, the pursuer will seek a better consequent parameter for different relative states. In this way, the consequent set is updated, which makes the pursuer tends to get closer and closer to the evader. At the same time, the evader also seeks a better consequent set for getting far away from the pursuer. However, as the pursuer has more powerful control capability, the evader is finally approached by the pursuer.

To illustrate the learning process, the variations of the total accumulating rewards of the pursuer and the evader in x and y channel along with learning times are shown in figure 12 and figure 13. From this figure, we can see that at the initial condition, the total rewards of the pursuer in x channel and in y channel are negative, and the values are relative large. Combined with the trajectories shown in figure 12, it shows that the reason is because that the pursuer can not track the evader well at the initial condition. Besides, the values of the total rewards of the evader in x channel and in y channel are opposite to the values of the pursuer in the corresponding channel, which proves that this differential game is the type



(a) Trajectories before learning (b) Trajectories after learning

Figure 11: Trajectories of the pursuer and the evader under perturbed reference orbit

of zero-sum. These values are relatively large, because the evader is not tracked well by the pursuer, which means the condition is relatively good for the evader. The values of the total rewards have been varied along with the learning times. When it comes to the last episodes, the terminal values are denoted in figure 12. It is seen that the total reward of the pursuer in x channel and y channel have been increased to 53.45 and 1115.35, respectively.

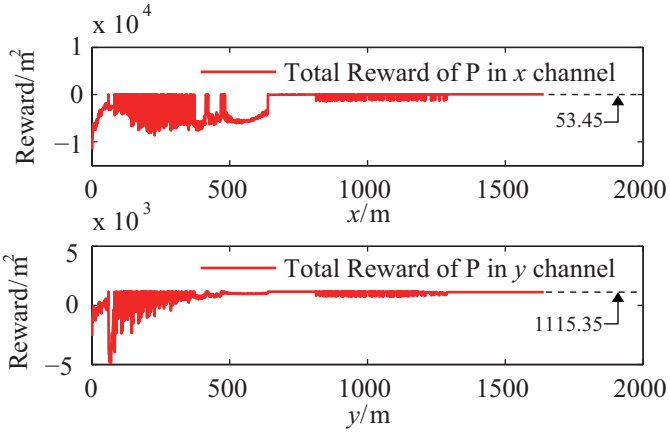


Figure 12: Variations of the total rewards of the pursuer

To show the total rewards in details, the values in different conditions are shown in table 4, where the value under the initial condition means the total rewards collected by the pursuer and the evader in the real environment with the consequent set from the pre-trained process.

In table 4, it is seen that, compared with the initial values of the pursuer, the values under PTFACL has been attenuated to the ideal values a lot, which also means that the pursuer can track the evader better after the learning process, and the evader has been decreased its total rewards due to the principle of zero-sum. If we set the ideal values as a baseline, it also comes to conclude that the value of the pursuer has increased by 99.99% in x channel while 96.09% in y channel.

The variations of time differences of the pursuer and the evader in x channel and y channel

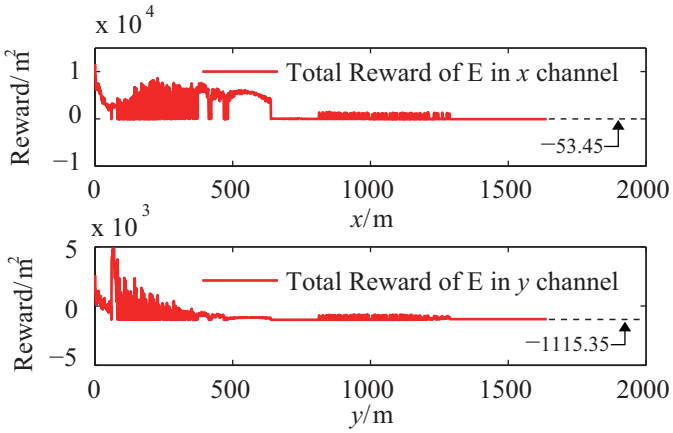


Figure 13: Variations of the total rewards of the evader

Table 4: Total rewards of the pursuer and the evader under the initial condition, ideal condition and the PTFACL

Value	P in x channel	P in y channel	E in x channel	E in y channel
Initial	-6703.03	-249.78	6703.03	249.78
Ideal	53.45	1161.42	-53.45	-1161.42
PTFACL	53.44	1115.36	-53.44	-1115.36

are shown in figure 14 and figure 15. It is shown that the time differences of the different agent in different channel have been decreased along with the learning times. It is noticed that as the time difference decreases, the updating process will be slow down, which will waste a lot of time to make the time difference attenuate to zero but contribute less to the update of the consequent set. Therefore, it is supposed to be acceptable to end the learning process when the norm of the time difference is less than 20.

5.3. Scenario with 1500s : the environment has perturbed external force

In this scene, we consider another kind of difference between the virtual environment and the real environment. The real environment in this scene is supposed to have the external disturbance forces as $d_t^x = 0.8 \times 10^{-5}$ and $d_t^y = 0.8 \times 10^{-5}$, but the virtual environment does not have. Therefore, the consequent set obtained from the pre-trained process is not suitable enough when we put the agents in the real environment. The trajectories of the pursuer and the evader before the learning process are shown in figure 16(a), where it is shown that the pursuer can not track the evader well because of unsuitable consequent set. With the learning rate of the Critic, $\alpha = 0.01$, the learning rate of the Actor $\beta = 0.001$, and the random noise $\sigma = 0.1$, in figure 16(b), the trajectories of the pursuer and the evader after the learning process are drawn, and it is seen that the pursuer has improved the ability to track the evader. From the small graph of terminal condition in figure 16(b), it indicates that there still exists some tracking error when the learning process is finished due to the fuzzy inference accuracy and the channel coupling effect, which are the same as the phenomenon

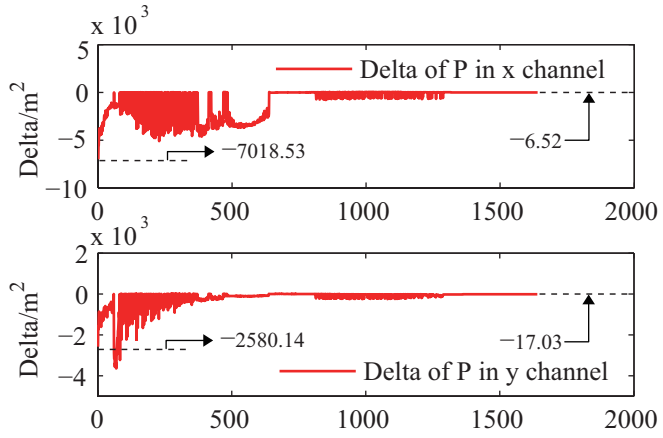


Figure 14: Variations of the time difference of the pursuer

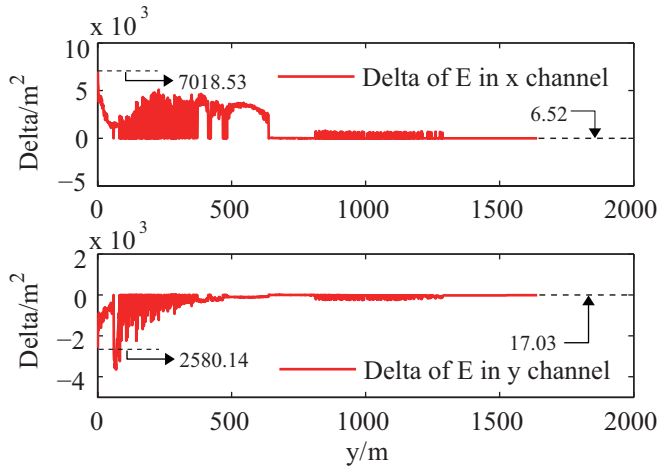
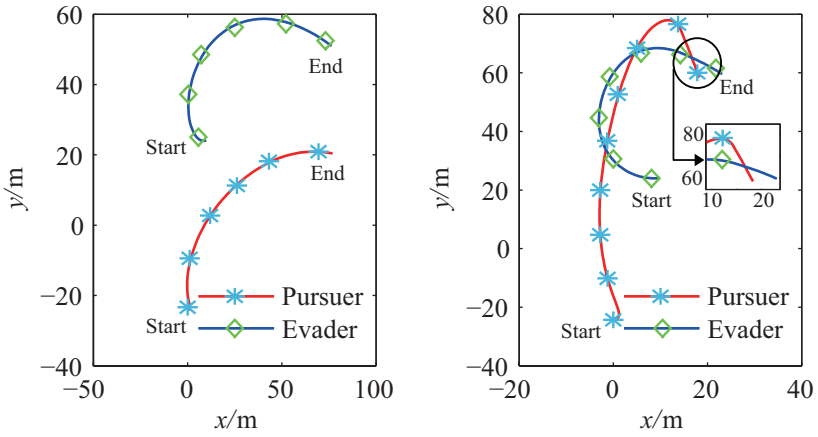


Figure 15: Variations of the time difference of the evader

shown in the small graph of figure 11(b).

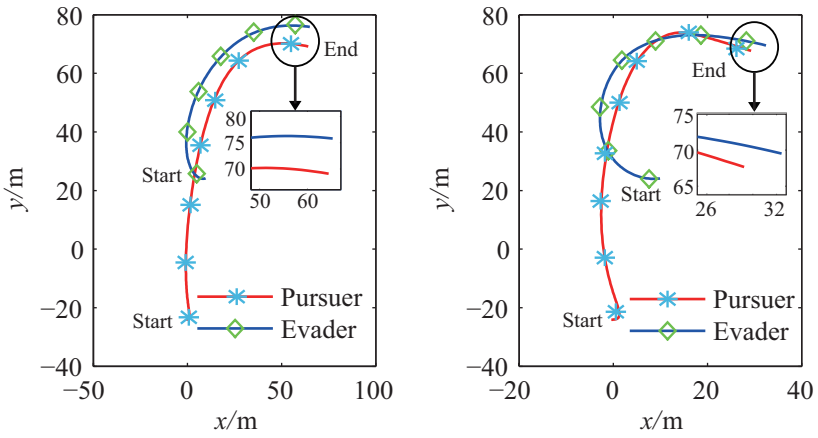
In figure 17(a), it shows the trajectories when the learning times comes to 20. Compared with the figure 16(a), it is seen that the moving trend of the pursuer has been changed to approach the evader while the evader is also changing its moving trend to keep away from the pursuer. Compared with the results in figure 17(a), the evader in figure 17(b) tries to avoid the pursuer by turning. But at the same time, the pursuer also changed its strategy to adapt to the new trajectory of the evader.

To further show the effectiveness of the learning process, we now change the control strategy of the evader. It is supposed that the control strategy of the evader comes to $u_E^x = u_E^y = 0.5m/s^2$. In figure 18(a), the trajectories of the pursuer and the evader before the learning process are shown. It is seen that the pursuer cannot track the evader well. At the same time, we apply the consequent set after the learning process, which is also applied in figure 16(b), the trajectories of the pursuer and the evader are shown in figure 18(b). In addition, when the pursuer takes a PD controller, the trajectories are drawn in figure18(c). Compare with the figures from 18(a) to 18(c), it is clear that the pursuer has the most powerful tracking ability in figure 18(b), because it is after learning. To give an



(a) Trajectories before learning (b) Trajectories ater learning

Figure 16: Trajectories of the pursuer and the evader under perturbed external force

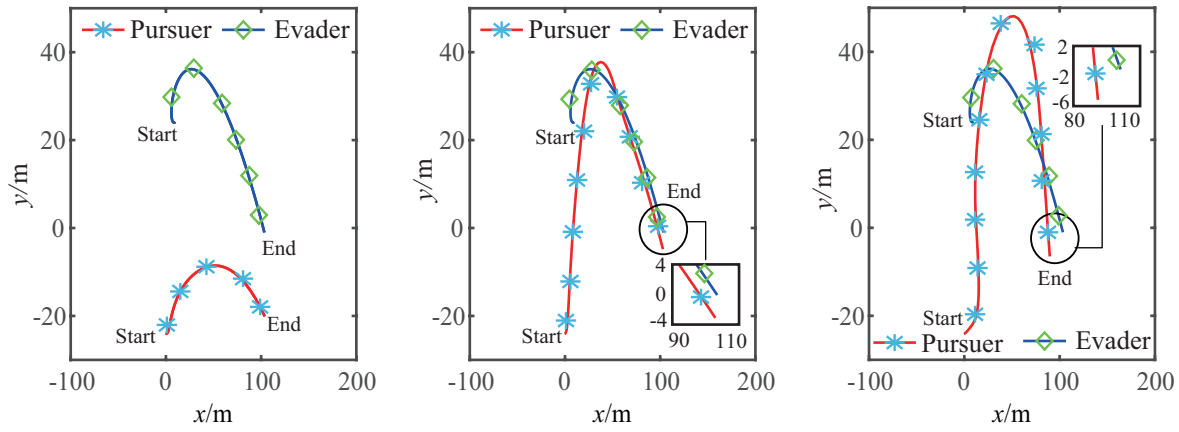


(a) Trajectories after 20 iterations (b) Trajectories after 50 iterations

Figure 17: Trajectories of the pursuer and the evader under perturbed external force during the learning process

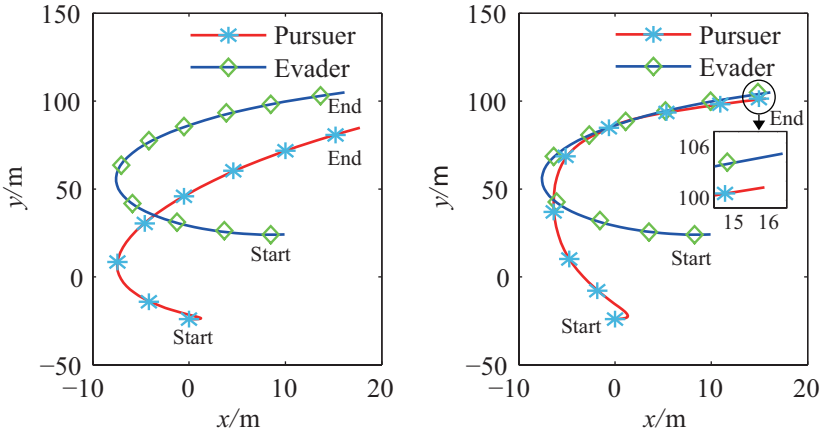
additional experiment, we reset the control strategy of the evader as $u_E^x = -0.5m/s^2$ and $u_E^y = 0.5m/s^2$. Then, the trajectories of the pursuer and evader before the learning process are shown in figure 19(a) while the trajectories after the learning process are shown in figure 19(b). This case also proves the effectiveness of the learning process, because the pursuer is able to approach the evader better with more suitable consequent set. It is noticed that the pursuer in figure 18(b) and 19(b) seems to perform better than in figure 16(b), because the evader has not applied an optimal strategy and learned during the learning process.

To show the total rewards under the PTFACL in this scenario and the total rewards of the scenes which we have changed the strategies of the evader, we denote " case : a " to represent the case under the PTFACL, where both the evader and the pursuer are learning, and we denote "case : b" to represent the case where $u_E^x = u_E^y = 0.5m/s^2$, as well as "case : c" to represent the case where $u_E^x = -0.5m/s^2$ and $u_E^y = 0.5m/s^2$. Therefore, the total rewards of different cases are shown in table 5.



(a) Trajectories before learning (b) Trajectories after learning (c) Trajectories under PD control

Figure 18: Trajectories of the pursuer and the evader with $u_E^x = u_E^y = 0.5m/s^2$



(a) Trajectories before learning (b) Trajectories after learning

Figure 19: Trajectories of the pursuer and the evader with $u_E^x = -0.5m/s^2$ and $u_E^y = 0.5/s^2$

From table 5, it shows that the values of the pursuer in case (a) performs well in x channel but bad in y channel initially, which means that the consequent set obtained from the pre-trained process is suitable for the pursuer in x channel but not as well in y channel. After the learning process, the total rewards of the pursuer increased by 99.85% in y channel, but decreased a little in x channel. This means that the pursuer has learned to adapt to the new environment in y channel well, but as the evader is escaping, it has lost a little rewards in x channel due to the accuracy of pre-trained process, accuracy of fuzzy inference system and accuracy of reinforcement learning. In case (b), it also shows that the pursuer does not perform in y channel at initial condition, but it has learned to perform well after the learning, which means that the value in y channel has been attenuated to the ideal value and increased by 96.06% compared with the initial value. As for case (c), compared with the initial values, it shows that the pursuer has improved its performance by 91.30% in the x channel and by 96.90% in the y channel, which also notices that the more difference between the initial value to the ideal value, the smaller effect of the algorithm accuracy will happen.

Table 5: Total rewards of the pursuer and the evader in case (a), case (b) and case (c)

Value	P in x channel	P in y channel	E in x channel	E in y channel
Ideal	53.45	1161.42	-53.45	-1161.42
Initial:(a)	53.23	225.55	-53.23	-225.55
Case:(a)	30.31	1160.05	-30.31	-1160.05
Initial:(b)	53.59	796.86	-53.59	-796.86
Case:(b)	53.12	1147.08	-53.12	-1147.08
Initial:(c)	50.92	757.75	-50.92	-757.75
Case:(c)	53.23	1148.93	-53.23	-1148.93

6. Conclusion

To provide a method to space differential game in unknown environment, a pre-trained fuzzy Actor-Critic learning (PTFACL) algorithm, which is based on reinforcement learning, is proposed in this paper. To utilize the the known part of the environment, a virtual environment is defined to find the optimal strategies for the pursuer and the evader. By introducing the fuzzy inference systems, the game can be separated into the motions in x channel and y channel with the inputs of the relative position and the relative velocity in each channel. With the help of the genetic algorithm, the optimal strategies can be reloaded into the consequent set of the pursuer and the evader, and this part is seen as a pre-trained process. An Actor-Critic framework is selected to refine the consequent set of the pursuer and the evader in the real environment. Through comparing the PTFACL and the FACL in the scenario with 600s, it shows that the pursuer tracks the evader better under the PTFACL, and saved the time cost by 46% compared with the FACL. When there is a difference between the reference orbit of the real environment and that of the virtual environment, the experimental results indicates that the value of total rewards of the pursuer increased by 99.99% in the x channel, by 96.06% in the y channel compared with the initial values. In addition, when there exist an external disturbance force in the real environment compared with the virtual environment, the pursuer increases its total rewards 99.85% in y channel but decreases a little in x channel due to the accuracy of pre-trained process and the coupling effect. When the control strategy of the evader is fixed to a specific policy, it shows that the pursuer increases its total reward by 96.06% in y channel under $u_E^x = u_E^y = 0.5m/s^2$, besides, 91.30% in x channel and 96.90% in y channel under $u_E^x = -0.5m/s^2$ and $u_E^y = 0.5m/s^2$.

References

[1] John Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):286–295.

[2] J. B. Cruz and C. I. Chen. Series nash solution of two-person, nonzero-sum, linear-quadratic differential games. *Journal of Optimization Theory & Applications*, 7(4):240–257, 1971.

[3] Y. C. Ho and A. W. Starr. Further properties of nonzero-sum differential games. *Journal of Optimization Theory & Applications*, 3(4), 1969.

[4] Guelman and M. Proportional navigation with a maneuvering target. *IEEE Transactions on Aerospace & Electronic Systems*, AES-8(3):364–371.

[5] Becker and K. Closed-form solution of pure proportional navigation. *IEEE Transactions on Aerospace & Electronic Systems*, 26(3):526–533.

[6] Leonard D. Berkovitz. The existence of value and saddle point in games of fixed duration. *SIAM Journal on Control and Optimization*, 23(2):172–196, 1985.

[7] Breitner M, H, H. Pesch, and W. Grimm. Complex differential games of pursuit-evasion type with state constraints, part 2: Numerical computation of optimal open-loop strategies. *Journal of Optimization Theory & Applications*, 78(3):443–463, 1993.

[8] Kazuhiro Horie and Bruce A. Conway. Optimal fighter pursuit-evasion maneuvers found via two-sided optimization. *Journal of Guidance Control and Dynamics*, 29(1):105–112, Jan. 2006.

[9] Mauro Pontani and Bruce A. Conway. Numerical solution of the three-dimensional orbital pursuit-evasion game. *Journal of Guidance Control and Dynamics*, 32(32):474–487, 2009.

[10] Mauro Pontani, Bruce Conway, and Joanie Stupik. Optimal pursuit/evasion spacecraft trajectories in the hill reference frame. In *Aas/aiaa Astrodynamics Specialist Conference*, 2013.

[11] Kazuhiro Horie and Bruce Conway. Genetic algorithm preprocessing for numerical solution of differential games problems. *Journal of Guidance Control Dynamics*, 27:1075–1078, 11 2004.

[12] Ashish Jagat and Andrew J. Sinclair. Optimization of spacecraft pursuit-evasion game trajectories in the euler-hill reference frame. In *Aiaa/aas Astrodynamics Specialist Conference*, 2013.

[13] Ashish Jagat and Andrew J. Sinclair. Nonlinear control for spacecraft pursuit-evasion game using state-dependent riccati equation method. *IEEE Transactions on Aerospace and Electronic Systems*, 53(99):1–1, 2017.

[14] Shen Dan, Khanh Pham, Erik Blasch, Huimin Chen, and Genshe Chen. Pursuit-evasion orbital game for satellite interception and collision avoidance. *Proceedings of SPIE - The International Society for Optical Engineering*, 8044(7):284–287, 2011.

[15] Qiuhua Zhang, Jian Chen, Songtao Sun, and Yi Sun. The differential game barrier model for spacecraft under target maneuvering based on minimum error. *SPIE Proceedings*, 6555:655518, Apr. 2007.

[16] Songtao Sun, Qiuhua Zhang, and Chen Yi. Numerical solution for a class of pursuit-evasion problem in low earth orbit. In *2013 9th Asian Control Conference (ASCC)*, pages 1–6, June 2013.

[17] Andrew G Barto. Reinforcement learning. *A Bradford Book*, volume 15(7):665–685, 1998.

[18] R. S. Sutton, A. G. Barto, and R. J. Williams. Reinforcement learning is direct adaptive optimal control. *IEEE Control Syst. Mag.*, 12(2):19–22, Feb. 1992.

[19] Christopher J. C. H. Watkins and Peter Dayan. Technical note: Q-learning. *Machine Learning*, 8(3-4):279–292, 1992.

[20] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

[21] Joo Er Meng and Chang Deng. Obstacle avoidance of a mobile robot using hybrid learning approach. *IEEE Transactions on Industrial Electronics*, 52(3):898–905.

[22] Xiaohui Dai, Chi-Kwong Li, and A. B. Rad. An approach to tune fuzzy controllers based on reinforcement learning for autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 6(3):285–293, Sep. 2005.

[23] H. Xiao, Li Liao, and F. Zhou. Mobile robot path planning based on q-ann. In *2007 IEEE International Conference on Automation and Logistics*, pages 2650–2654, Aug 2007.

[24] Shao-Ming Hung and Sidney Givigi. A q-learning approach to flocking with uavs in a stochastic environment. *IEEE Transactions on Cybernetics*, 47:1–12, 2016.

[25] Dongbing Gu and Erfu Yang. Fuzzy policy reinforcement learning in cooperative multi-robot systems. *Journal of Intelligent and Robotic Systems*, 48(1):7–22, 2007.

[26] Howard M Schwartz. *Multi-agent machine learning: a reinforcement approach.* 2014.

[27] Dario Izzo, Marcus Mrtens, and Binfeng Pan. A survey on artificial intelligence trends in spacecraft guidance dynamics and control. *Astrodynamics*, 4:287–299, 3 2019.

[28] A. T. Bilgin and E. Kadioglu-Urtis. An approach to multi-agent pursuit evasion games using reinforcement learning. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 164–169, July 2015.

[29] X. Wang, P. Shi, C. Wen, and Y. Zhao. Design of parameter-self-tuning controller based on reinforcement learning for tracking non-cooperative targets in space. *IEEE Transactions on Aerospace and Electronic Systems*, pages 1–1, 2020.

[30] Xiao Wang, Peng Shi, Yushan Zhao, and Yue Sun. A pre-trained fuzzy reinforcement learning method for the pursuing satellite in a one-to-one game in space. *Sensors (Switzerland)*, 20(8), 2020.

[31] Xiao Wang, Peng Shi, Changxuan Wen, and Yushan Zhao. An algorithm of reinforcement learning for maneuvering parameter self-tuning applying in satellite cluster. *Mathematical Problems in Engineering*, 2020, 2020.

[32] C. V. Analikwu and H. M. Schwartz. Reinforcement learning in the guarding a territory game. In *2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1007–1014, July 2016.

[33] Mostafa D. Awheda and Howard M. Schwartz. A residual gradient fuzzy reinforcement learning algorithm for differential games. *International Journal of Fuzzy Systems*, 19(4):1058–1076, Aug. 2017.

[34] Sameh Desouky and Howard Schwartz. Self-learning fuzzy logic controllers for pursuit-evasion differential games. *Robotics and Autonomous Systems*, 59:22–33, 01 2011.

[35] W. H. Clohessy and R. S. Wiltshire. Terminal guidance system for satellite rendezvous. *Journal of the Aerospace Sciences*, 27(9):653–658, 1960.

[36] D. Goldberg. Genetic algorithm in search, optimization, and machine learning. *Addison-Wesley, Reading, Massachusetts*, xiii, 01 1989.