sensors

MDPI

Article

A Pre-Trained Fuzzy Reinforcement Learning Method for the Pursuing Satellite in a One-to-One Game in Space

Xiao Wang [1], Peng Shi [1,*], Yushan Zhao [1] and Yue Sun [2]

[1] School of Astronautics, Beihang University, Beijing 100191, China; w_xiao@buaa.edu.cn (X.W.); yszhao@buaa.edu.cn (Y.Z.)

[2] Shanghai Key Laboratory of Aerospace intelligent Control Technology, Shanghai Aerospace Control Technology Institute, Shanghai 201109, China; yoyo_326@163.com

* Correspondence: shipeng@buaa.edu.cn

Received: 22 March 2020; Accepted: 13 April 2020; Published: 16 April 2020

Abstract: In order to help the pursuer find its advantaged control policy in a one-to-one game in space, this paper proposes an innovative pre-trained fuzzy reinforcement learning algorithm, which is conducted in the x, y, and z channels separately. Compared with the previous algorithms applied in ground games, this is the first time reinforcement learning has been introduced to help the pursuer in space optimize its control policy. The known part of the environment is utilized to help the pursuer pre-train its consequent set before learning. An actor-critic framework is built in each moving channel of the pursuer. The consequent set of the pursuer is updated through the gradient descent method in fuzzy inference systems. The numerical experimental results validate the effectiveness of the proposed algorithm in improving the game ability of the pursuer.

Keywords: differential game; reinforcement learning; actor-critic; fuzzy system

1. Introduction

Tracking space targets is beneficial for orbital garbage removal, recovery of important components, and early warning of space threats [1]. However, with the continuous development of space techniques, the targets in space have been expanded from non-maneuverable ones to maneuverable ones. Tracking a target that has maneuverability is still a challenging problem because the target is non-cooperative and the environment is usually partially unknown.

In order to track a non-cooperative target in space, one can apply control theory to design a control law. By establishing an attitude-position coupling model, an adaptive control law that considers the unknown mass and inertia was proposed [2]. Besides considering the system static errors and disturbances, some adaptive control laws were designed [3,4]. With the development of the research, many mature control methods were introduced to the field of tracking targets in space [5–7]. In addition, a back-stepping adaptive control law with the consideration of a variety of model uncertainties, as well as the input constraints and an optimal inverse controller with external disturbances were attempted in [8,9], respectively. It was proven that the closed-loop system was still stable in the presence of external disturbances and uncertain parameters. However, these proposed control laws are basically used for the targets, which do not have the ability to maneuver.

For tracking a target that can move, there is the potential to describe the problem as a pursuit-evasion problem, which is also known as the space differential game. The differential game, which was introduced in [10,11], is usually applied to continuous systems. To find a superior strategy of the pursuer in aircraft combat, scholars proposed the proportional navigation method [12]. In addition,

when the differential game is applied to the field of space, the so-called two-sided optimal theory, which was an extension of the traditional optimal theory, is found in [13,14]. Besides, in order to transform the two-sided optimal problem into a traditional one-sided optimal problem, the semi-direct collocation method was studied based on the two-sided extremum principle [15]. In order to reduce the difficulty of the solution, the genetic algorithm was employed to help find the initial values of the co-states [16–18]. However, based on the two-sided optimal theory, the optimal strategy of the pursuer can only be found when the system information is totally known, and it will be unable to deal with system uncertainties and external disturbances. Therefore, it is reasonable to find a way to make the pursuer able to adjust its control policy according to the environment. One of the potential methods is to apply reinforcement learning because of its capacity to optimize the control policy under an unknown environment.

Reinforcement learning, which aims to map states to actions so as to maximize a numerical reward, is one of the machine learning methods [19]. At first, reinforcement learning was used for solving problems of discrete systems after the classical Q-learning was proposed [20], and this branch has been developed [21]. Since the technique of space generalization was introduced to avoid the curse of dimensionality, the learning algorithms can be applied to solve the problems in continuous space [22–25]. As for solving a differential game, in recent years, scholars have also found that it was effective to use the reinforcement learning algorithm [26,27]. With the single control input, the ground pursuit-evasion problem was considered in [28,29].

There seems to be potential in using the technique of reinforcement learning because such a learning method can help the pursuer optimize its control policy in an unknown environment. However, the differential game in space has more complex dynamics; therefore, it will be extremely hard to solve without any prior information. To overcome this shortcoming, in this paper, we propose an innovative pre-trained fuzzy reinforcement learning (PTFRL) algorithm to help the pursuer optimize its control policy through a pre-training process. The pre-training process utilizes the known part of the environment and helps the pursuer initialize its consequent set before reinforcement learning. The algorithm is based on the actor-critic framework, which is one of the most active reinforcement learning branches. The learning framework is divided into x, y, and z channels, and each channel learns separately. The man-made model is defined as an estimated environment, which can be used to derive the estimated optimal strategy for the pursuer. With the help of the genetic technique, the pre-training process will be conducted to help the pursuer initialize its consequent set. Then, through the fuzzy inference systems, the control policy of the pursuer will be updated from the fuzzy actor-critic learning.

In general, this paper applies a pre-trained fuzzy reinforcement learning algorithm to optimize the control policy of a pursuer, which is used for a one-to-one game in outer space. The main improvements of this paper are as follows: (1) Unlike the previous control laws, which were designed based on the adaptive control theory, for the first time, we utilize the technique of reinforcement learning to help the pursuer track a moving non-cooperative target in space. Compared with the adaptive control laws, which contain massive derivations and computing costs to deal with the uncertainties of the environment, the proposed algorithm takes advantage of artificial intelligence, avoiding the mathematical complexity. It is a new approach to optimize the control policy of the pursuer by interacting with the space environment. (2) Different from the reinforcement learning algorithms applied in ground games, the game in space has more complex states and actions. Without any prior information, it will be extremely hard for the pursuer to find its control policy because of the complex environment. To reduce the difficulty of solving the game, the proposed algorithm innovatively adds a pre-training process utilizing the known part of the environment.

The structure of this paper is as follows: Section 2 presents the dynamics of the pursuer and the evader; Section 3 discusses the fuzzy inference system and its combination with reinforcement learning for continuous systems; Section 4 applies the pre-trained fuzzy reinforcement learning algorithm for the pursuer; Section 5 simulates the proposed algorithm; Section 6 discusses the experimental results; finally, Section 7 draws the conclusions.

2. Dynamics of the Space Differential Game

To describe the space differential game, the following coordinate systems are established: (a) Earth-centered inertial ($OXYZ$); (b) the orbital coordinate system of the spacecraft ($Ox_oy_oz_o$); (c) the orbital coordinate system of the virtual host spacecraft ($Ox_ry_rz_r$).

In this game, there are one pursuer and one evader, where the pursuer P aims to track the evader E and the evader E aims to escape from the pursuer P. The position relationship among the pursuer, the evader, and the virtual host point o is drawn in Figure 1.

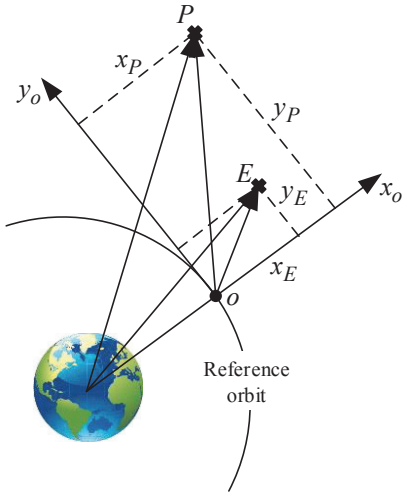


Figure 1. The location of the pursuer and the evader.

The virtual host point o is located near the two satellites. The pursuer and the evader can be abstracted as the agents, which have the ability of interacting with the environment. In this paper, we will focus on the control strategy of the pursuer to make it have an advantage in this game. The pursuer is expected to update its control policy according to its interaction with the environment through reinforcement learning. Therefore, for a simulated experiment in this paper, it is necessary to build an environment that includes the dynamics of the agents in it.

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 pursuer, P, is expressed as below [30]:

$$\begin{cases} \dot{x}_P(t) = v_P^x(t) \\ \dot{y}_P(t) = v_P^y(t) \\ \dot{z}_P(t) = v_P^z(t) \\ \dot{v}_P^x(t) = 2\dfrac{\mu}{r^3(t)} x_P(t) + 2\omega(t) v_P^y(t) + \dot{\omega}(t) y_P(t) + \omega^2(t) x_P(t) + T_P u_P^x(t) + d_t^x \\ \dot{v}_P^y(t) = -\dfrac{\mu}{r^3(t)} y_P(t) - 2\omega(t) v_P^x(t) - \dot{\omega}(t) x_P(t) + \omega^2(t) y_P(t) + T_P u_P^y(t) + d_t^y \\ \dot{v}_P^z(t) = -\omega^2(t) z_P(t) + T_P u_P^z(t) + d_t^z \end{cases} \tag{1}$$

where μ represents the Earth's gravitational constant, $\omega(t)$ represents the instantaneous angular velocity of the reference orbit, and $r(t)$ represents the instantaneous radius of the orbit. Besides, the dynamics of the evader E is expressed as follows.

$$\begin{cases} \dot{x}_E(t) = v_E^x(t) \\ \dot{y}_E(t) = v_E^y(t) \\ \dot{z}_E(t) = v_E^z(t) \\ \dot{v}_E^x(t) = 2\dfrac{\mu}{r^3(t)} x_E(t) + 2\omega(t) v_E^y(t) + \dot{\omega}(t) y_E(t) + \omega^2(t) x_E(t) + T_E u_E^x(t) \\ \dot{v}_E^y(t) = -\dfrac{\mu}{r^3(t)} y_E(t) - 2\omega(t) v_E^x(t) - \dot{\omega}(t) x_E(t) + \omega^2(t) y_E(t) + T_E u_E^y(t) \\ \dot{v}_E^z(t) = -\omega^2(t) z_E(t) + T_E u_E^z(t) \end{cases} \tag{2}$$

where $u_i^j (j = x, y, z)$ represents the force in the corresponding channel and $T_i (i = P, E)$ represents the maximum unit mass thrust of the satellite. It is noted that the external disturbance force is only added to the pursuer, because we always consider the relative states between the pursuer and the evader.

Through Equations (1) and (2), the environment for the learning algorithm is built, and it is seen as the real environment, which is differentiated from the estimated environment referred to in Section 4.2.

3. Reinforcement Learning in Continuous Systems

To avoid the curse of dimensionality, the technique of generalization should be addressed. Besides, the problem regarding satellite motion requires the inputs of the learning system to have clear physical meaning. Therefore, the zero-order Takagi–Sugeno (T-S) fuzzy system, which provides a more meaningful inference rule compared with neural networks, is employed as the approximator. In this way, the fuzzy actor-critic learning framework will be built. Through the gradient descent method, the consequent parameters of the actor and the critic will be updated.

3.1. The Fuzzy Inference System

The fuzzy inference rule of the employed Takagi–Sugeno (T-S) fuzzy system is expressed as below [31].

$$Rule\ l: \text{IF}\ s_1\ is\ F_1^l, \cdots,\ and\ s_n\ is\ F_n^l \quad THEN\ z_l = \phi_l \tag{3}$$

If we assume that the fuzzy system has L rules, n input variables, and each input has h membership functions, the output of the fuzzy system can be expressed as:

$$Z(\boldsymbol{s}) = \frac{\sum_{l=1}^{L}\left[\left(\prod_{i=1}^{n}\mu^{F_i^l}(s_i)\right)\phi^l\right]}{\sum_{l=1}^{L}\left(\prod_{i=1}^{n}\mu^{F_i^l}(s_i)\right)} = \sum_{l=1}^{L}\Psi_l(\boldsymbol{s})\phi_l \tag{4}$$

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

$$\Psi_l(\boldsymbol{s}) = \frac{\prod_{i=1}^{n}\mu^{F_i^l}(s_i)}{\sum_{l=1}^{L}\left(\prod_{i=1}^{n}\mu^{F_i^l}(s_i)\right)} = \frac{\omega_l(\boldsymbol{s})}{\sum_{l=1}^{L}\omega_l(\boldsymbol{s})} \tag{5}$$

The applied membership functions here are triangular membership functions, which are shown in Figure 2. This shows that the input will only activate two membership functions at one time for one input, which will save computing cost when the number of membership functions rises.

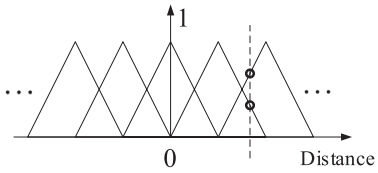


Figure 2. The membership functions for one input.

3.2. The Fuzzy Actor-Critic Learning Algorithm

In the actor-critic learning algorithm, the value function and the policy function are approximated through T-S systems, respectively. The critic part is used to estimate the value function, while the actor part is used to generate the action. To apply the actor-critic learning framework into a continuous system, we need two critic parts to estimate the current value function $\hat{V}_t(\boldsymbol{s}_t)$ and the next value function $\hat{V}_t(\boldsymbol{s}_{t+1})$ and one actor part to generate the current control variable. In this way, the temporal difference can be expressed as below.

$$\Delta_t = r_t + \gamma \hat{V}_t(\boldsymbol{s}_{t+1}) - \hat{V}_t(\boldsymbol{s}_t) \tag{6}$$

Denote $\Xi = \frac{1}{2}\Delta_t^2$ as the variance of the difference signal; therefore, the adaptive update rule of the parameters in the critic is expressed as:

$$\begin{aligned} \phi^C(t+1) &= \phi^C(t) - \alpha \frac{\partial \Xi}{\partial \phi^C} \\ &= \phi^C(t) - \alpha \Delta_t \left[\gamma \frac{\partial V_t(\boldsymbol{s}_{t+1})}{\partial \phi^C} - \frac{\partial V_t(\boldsymbol{s}_t)}{\partial \phi^C} \right] \end{aligned} \tag{7}$$

where ϕ^C represents the consequent parameter of the critic and α represents the learning rate of the critic.

In addition, we have:

$$\frac{\partial V_t(\boldsymbol{s}_t)}{\partial \phi^C} = [\Psi_1(\boldsymbol{s}_t), \Psi_2(\boldsymbol{s}_t), \cdots, \Psi_L(\boldsymbol{s}_t)] \tag{8}$$

$$\frac{\partial V_t(\boldsymbol{s}_{t+1})}{\partial \phi^C} = [\Psi_1(\boldsymbol{s}_{t+1}), \Psi_2(\boldsymbol{s}_{t+1}), \cdots, \Psi_L(\boldsymbol{s}_{t+1})] \tag{9}$$

which can be combined with Equation (5). In this way, Equation (7) can be solved.

Denoting the output of the actor as u_t, a rand noise, σ, will be added to u_t to explore better rewards. Therefore, the real output is $u_c = u_t + \sigma$.

Further, the adaptive update rule of the parameters of the actor is expressed as:

$$\phi^A(t+1) = \phi^A(t) + \beta \Delta_t \frac{\partial u_t}{\partial \phi^A}(u_c - u_t) \tag{10}$$

where ϕ^A represents the consequent parameter of the actor and β represents the learning rate of the actor.

4. Pre-Trained Fuzzy Reinforcement Learning for the Pursuing Satellite in a One-to-One Game in Space

The proposed algorithm is single-looped, which means that for the motions of the pursuing satellite P, each agent has to be divided into three channels, the x, y, and z channels. In each channel, there exists two inputs, the relative distance and the relative velocity of the current channel. With the help of the genetic algorithm, the consequent sets of actors in each channel will be initialized.

4.1. Fuzzy Reinforcement Learning Algorithm

Take the x channel as an example. The inputs are $s_1 = x$ and $s_2 = v_x$; therefore, the inference rule is expressed as:

$$R_l : \text{IF } s_1 \text{ is } A_1^l \text{ and } s_2 \text{ is } A_2^l \text{ THEN } Z_l = \varphi_l \tag{11}$$

where φ_l represents the consequent parameter in the consequent set φ_P^x of critics.

In addition, the following relationship is shown.

$$\Psi_l(\boldsymbol{s}) = \frac{\prod_{i=1}^{2} \mu^{F_i^l}(s_i)}{\sum_{l=1}^{4} \left(\prod_{i=1}^{2} \mu^{F_i^l}(s_i) \right)} = \frac{\omega_l(\boldsymbol{s})}{\sum_{l=1}^{4} \omega_l(\boldsymbol{s})} \tag{12}$$

$$\hat{V}_P^x = \sum_{l=1}^{4} (\Psi_l) \cdot (\varphi_l) \tag{13}$$

Similarly, the output of the actor is shown as below.

$$u_t = \sum_{l=1}^{4} (\Psi_l) \cdot (\phi_l) \tag{14}$$

where ϕ_l represents the consequent parameter in the consequent set ϕ_P^x of actors. To add a noise σ for exploring, the final control variable is expressed as follows.

$$u_P^x = u_t + \sigma \tag{15}$$

The designed reward function, r_t, is expressed as:

$$\begin{aligned} r_t|_P^x &= D_x(t-1) - D_x(t) \\ r_{t_n}|_P^x &= -D_x(t_n) \\ r_t|_P^y &= D_y(t-1) - D_y(t) \\ r_{t_n}|_P^y &= -D_y(t_n) \\ r_t|_P^z &= D_z(t-1) - D_z(t) \\ r_{t_n}|_P^z &= -D_y(t_n) \end{aligned} \tag{16}$$

The expressions of $D_x(t)$, $D_y(t)$ and $D_z(t)$ are as follows.

$$\begin{aligned} D_x(t) &= \frac{1}{2}\big(x_p(t) - x_e(t)\big)^2 \\ D_y(t) &= \frac{1}{2}\big(y_p(t) - y_e(t)\big)^2 \\ D_z(t) &= \frac{1}{2}\big(z_p(t) - z_e(t)\big)^2 \end{aligned} \tag{17}$$

In Figure 3, the learning logic is illustrated. From this figure, it is seen that the learning framework is divided into x, y, and z channels, and each channel has two critic parts and one actor part.

It is noticed that the two critic parts are applied to estimate the value of the current time, $\hat{V}(t)$, and the value of the next time, $\hat{V}(t+1)$. It shows that in the x channel, the combination of x and v_x is input into 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. Combining u_P^x, u_P^y, and u_P^z, the control vector of the pursuing satellite, $\boldsymbol{u}_P$, can be generated. Under such a control policy, the pursuer will interact with the environment, which already contains the motions of the evader. Then, the next state $\boldsymbol{s}_{t+1}$ and the rewards for all the channels are expected to be obtained. Take the x channel as an example; the time time difference,

Δ_t can be calculated according to $r|_P^x$, $\hat{V}_P^x(s_t)$ and $\hat{V}_P^x(s_{t+1})$, and the consequent parameters of the critic part and the actor part can be adaptively tuned through (7) and (10).

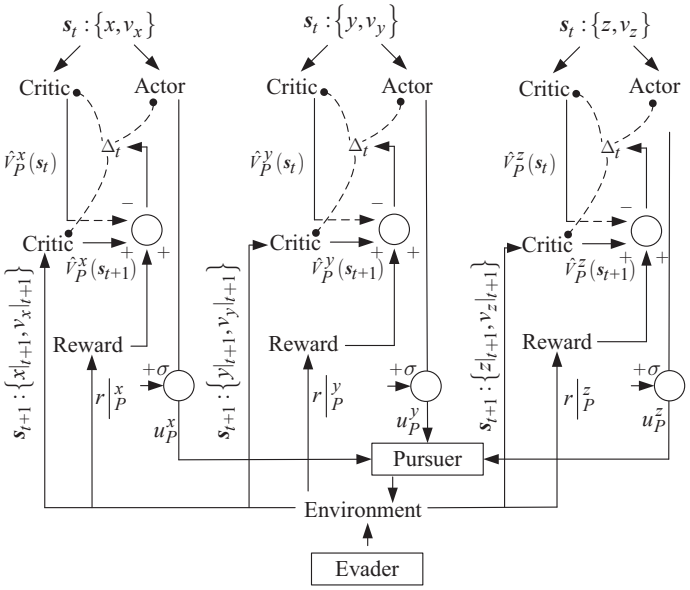


Figure 3. The diagram of the learning logic.

4.2. Pre-Training Process Based on the Genetic Algorithm

Denote the symbols ϕ_x^P, ϕ_y^P, and ϕ_z^P as representing the consequent sets of the actor parts in the x, y, and z channels of the pursuer, respectively. The structure of ϕ_x^P, ϕ_y^P, and ϕ_z^P is defined as a two-dimensional matrix, where the row number depends on the number of membership functions of the first input, and the column number depends on that of the second input. It is supposed that there exist 13 membership functions for the relative distance and 7 membership functions for the relative velocity in each learning channel. Therefore, it is clear that those consequent sets are 13×7 matrices.

Conventionally, the reinforcement learning algorithm is conducted on a totally unknown environment, because the agent is expected to interact with the environment without any external help. However, according to the the human study of orbital dynamics, one can build a mathematical model for the pursuer and the evader in space. Therefore, actually, a part of the real environment seems to be known. To utilize this known part to help find the initial values of the consequent sets, ϕ_x^P, ϕ_y^P, and ϕ_z^P will be helpful for the learning. Training these consequent sets based on the estimated environment is seen as a pre-training process before the learning.

The known part is defined as an estimated environment, which can obtain the estimated optimal strategy for the pursuer. Denote $\hat{\boldsymbol{x}} = [\hat{\boldsymbol{x}}_P, \hat{\boldsymbol{x}}_E]^{\mathrm{T}}$ as the state variable in the estimated 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 $\hat{\boldsymbol{x}}_E = \left[x_e, y_e, z_e, v_e^x, v_e^y, v_e^z\right]^{\mathrm{T}}$. In addition, denote the estimated ω as $\hat{\omega}$; therefore, the dynamics of the pursuer and the evader in the estimated environment can be expressed as:

$$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{A}\hat{\boldsymbol{x}} + T_P \boldsymbol{B}_P \boldsymbol{u}_P + T_E \boldsymbol{B}_E \boldsymbol{u}_E \tag{18}$$

where:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_P(t) & 0_{6\times 6} \\ 0_{6\times 6} & \boldsymbol{A}_E(t) \end{bmatrix} \tag{19}$$

$$\boldsymbol{A}_P = \boldsymbol{A}_E(t) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\hat{\omega}^2 & 0 & 0 & 0 & 2\hat{\omega} & 0 \\ 0 & 0 & 0 & -2\hat{\omega} & 0 & 0 \\ 0 & 0 & -\hat{\omega}^2 & 0 & 0 & 0 \end{bmatrix} \tag{20}$$

$$\boldsymbol{B}_P = \begin{bmatrix} 0_{3\times3} \\ I_{3\times3} \\ 0_{6\times3} \end{bmatrix} \quad \boldsymbol{B}_E = \begin{bmatrix} 0_{6\times3} \\ 0_{3\times3} \\ I_{3\times3} \end{bmatrix} \tag{21}$$

With the cost function, which is shown as follows:

$$J_i = D_i(t_n) + \int_{t_0}^{t_n} \dot{D}_i dt \tag{22}$$

where $i = x, y, z$, the estimated optimal strategy for the pursuer will be obtained. In this way, the training pairs will be generated, which can be used to train ϕ_x^P, ϕ_y^P, and ϕ_z^P.

To approximate the training pairs through the fuzzy inference system, the genetic algorithm (GA) is applied here to conduct the pre-training process. Take the x channel as an example. If it is supposed that we can obtain N pairs of training data, then the diagram of the GA process is described as in Figure 4.

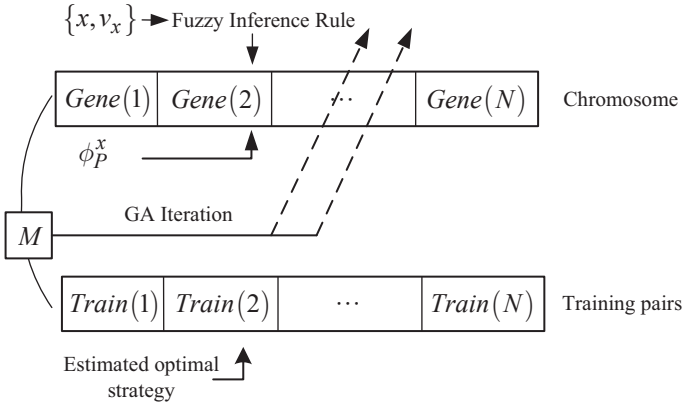


Figure 4. The diagram of the pre-training process.

From the figure, it is seen that the inputs for GA in the x channel are x and v_x, which will be input into the fuzzy inference system. The " chromosome" is a consequent set that is composed of the "genes". The "genes" are also shown as the consequent parameters. The symbol M, which represents the fitness function during the pre-training learning, can be calculated according to the values of u_{tr} from the training data and the values of u_A obtained from the fuzzy inference system. The expression of M is as below:

$$M = \frac{1}{2}\sum_{i=1}^{N}(u_A - u_{tr}(i))^2 \tag{23}$$

where u_A is the output of the fuzzy inference system and $u_{tr}(i)$ is the control value of the ith training pair.

Sorted by the fitness error, the current chromosome will be updated by performing crossover and mutation on the genes. With the help of the GA technique [32], ϕ_x^P, ϕ_y^P, and ϕ_z^P will be trained to approximate the training data better.

It is noted that the proposed algorithm will make use of the estimated optimal strategy; therefore, the reward function shown in Equation (16) should be consistent with the cost function shown in Equation (22).

5. Simulation

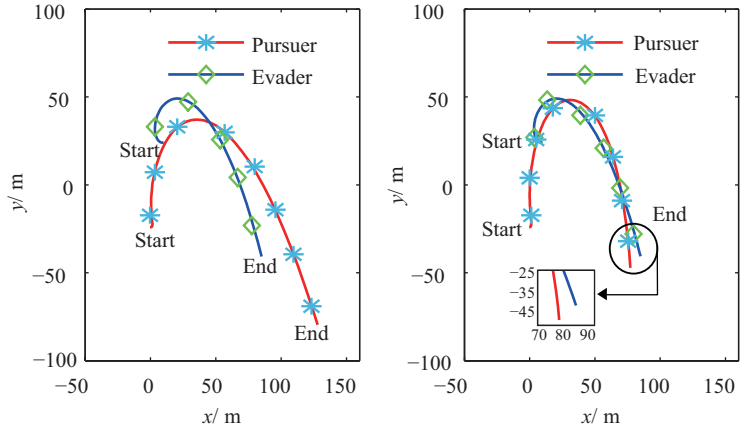
A one-to-one space differential game was simulated in this paper. The scenario contained a pursuing satellite P and an evading satellite E. The reference orbit was a circular orbit with a radius of 6.9×10^3 km. Table 1 denote the symbols $\boldsymbol{x}_{P0}$ and $\boldsymbol{x}_{E0}$ as the initial 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 the velocity in m/s of the agent.

Table 1. Initial states of the pursuer and the evader.

State	Value
$\boldsymbol{x}_{P0}$	$\left[-0.422\ \text{m}; 24.080\ \text{m}; 20.159\ \text{m}; 2.678 \times 10^{-2}\ \text{m/s}; -4.715 \times 10^{-5}\ \text{m/s}; 0\ \text{m/s}\right]^{\text{T}}$
$\boldsymbol{x}_{E0}$	$\left[9.918\ \text{m}; 24.115\ \text{m}; -5.462\ \text{m}; -2.678 \times 10^{-2}\ \text{m/s}; -5.608 \times 10^{-3}\ \text{m/s}; 0\ \text{m/s}\right]^{\text{T}}$

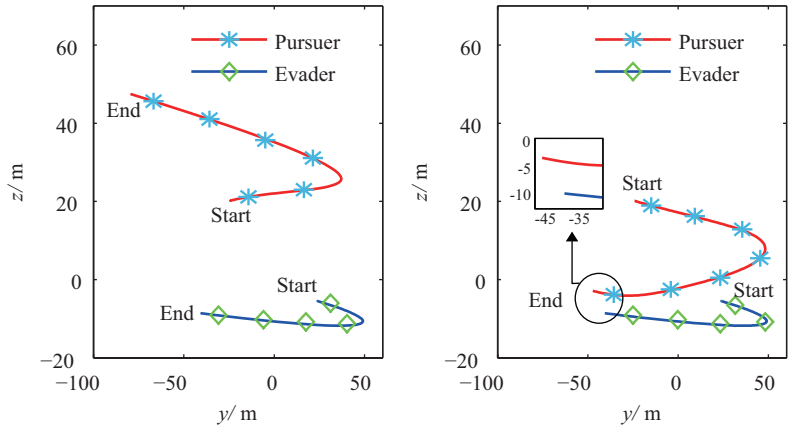
In this scenario, it was supposed that there were some deviations between the real environment and the estimated environment, where the condition $\omega - \hat{\omega} = 8 \times 10^{-4}$ rad/s existed. In addition, the real environment in this scene was supposed to have the external disturbance item as $\boldsymbol{d}_t = \left[1.5 \times 10^{-5}, 1.5 \times 10^{-3}, 2.0 \times 10^{-3}\right]$ m/s^2. With the learning rate of the critic, $\alpha = 0.01$, the learning rate of the actor $\beta = 0.001$, the random noise $\sigma = 0.1$ for exploring, $T_P = 0.03 \times 9.8 \times 10^{-3}$ and $T_E = 0.01 \times 9.8 \times 10^{-3}$, the proposed PTFRL was processed. As the pursuer and the evader moved in the x, y, and z planes at the same time, the simulation results were drawn in the X–Y plane and Y–Z plane, respectively. The total learning process cost 1560 iterations with 3496.98 seconds for learning.

Figure 5a shows the trajectories of the pursuer and the evader after the pre-training process in the X–Y plane. In this figure, the evader has its optimal strategy, and it is seen that there are some tracking errors from the pursuer to the evader because of the deviations between the estimated environment and the real environment. However, it is seen that the pursuer still has the ability to track the moving trend of the evader because it was pre-trained, and it utilized the information of the estimated environment. Compared with Figure 5a, Figure 5b draws the trajectories of the pursuer and the evader after the proposed PTFRL. It clearly shows that the pursuer could track the evader better after the learning. In the Y–Z plane, the trajectories before learning and the ones after learning are illustrated in Figure 6a,b, respectively. Due to the largest external disturbance in the z channel, Figure 6a shows that the pursuer tracked the evader badly; therefore, there was a big tracking error. In Figure 6b, the pursuer improved its control policy for tracking the evader in the z channel. Overall, from Figures 5 and 6, it is shown that, after the proposed learning algorithm, the pursuer could track the evader better because of more suitable consequent set. During the learning process, the pursuer would seek better consequent parameters for different relative states. In this way, the consequent set was updated, which made the pursuer tend to get much closer to the evader.



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

Figure 5. Trajectories of the pursuer and the evader in the X–Y plane.



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

Figure 6. Trajectories of the pursuer and the evader in the Y–Z plane.

The whole learning process could be divided into three periods: before pre-training, after pre-training, and after PTFRL. Before pre-training, the pursuer was in free flight without any control policy. When the pursuer finished the pre-training, it took the estimated optimal control policy based on the estimated environment. Finally, when the pursuer took the control policy after PTFRL, this meant that the pursuer finished the learning. The tracking errors under these three periods of the pursuer in the x, y, and z channels are shown in Figure 7. From this figure, it is seen that compared with the tracking error before pre-training, the one after pre-training effectively decreased, and that after PTFRL further approached zero. The max errors under different periods of all channels are drawn in Figure 8. It is clearly seen that, compared with the max error before pre-training, it decreased after pre-training and was further cut down after PTFRL. If all the rewards during the flight were accumulated, the total reward would be obtained. Therefore, there existed the real total reward under the real flight, and the ideal total reward if the pursuer could track the evader perfectly. The ideal total rewards and the real ones in the x, y, and z channels are shown in Figure 9. It shows that the total reward of each channel after pre-training rose compared with that before pre-trained. In addition, the total rewards attempted to approach the ideal values after PTFRL in all channels.

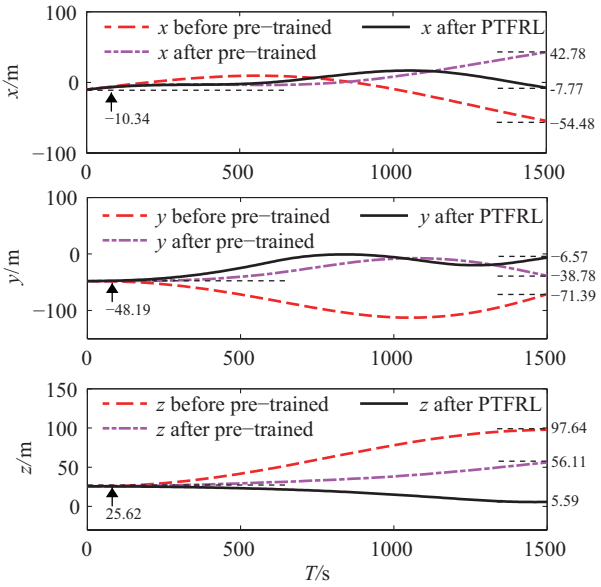


Figure 7. Variations of tracking errors in the x, y, and z channels.

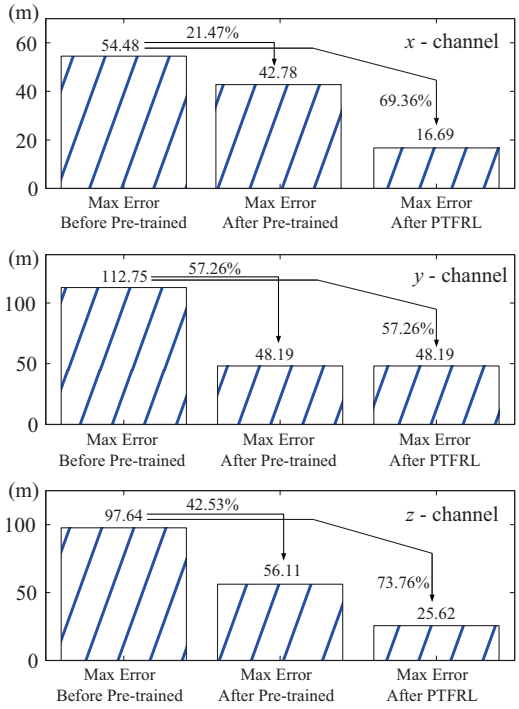


Figure 8. Comparisons of the max tracking errors in different periods.

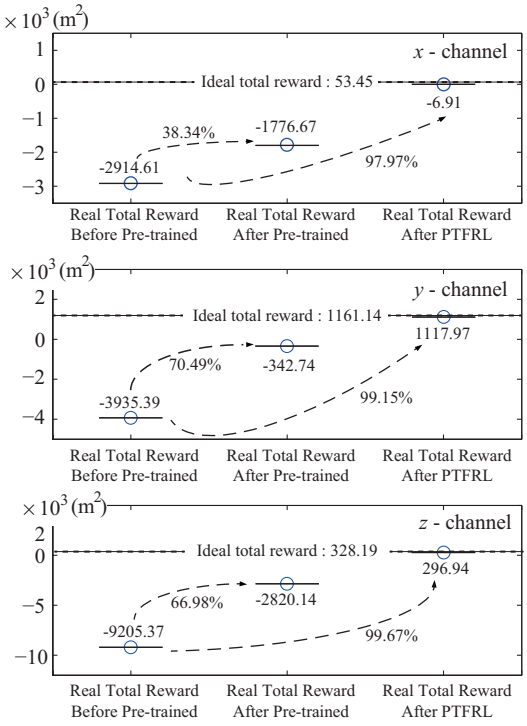


Figure 9. Comparisons of total rewards in different periods.

6. Discussion

Based on numerical experimental results in Section 5, the following discussions are shown below.

(a) From Figure 7, it can be concluded that in the x channel, compared with the terminal tracking error before pre-training, the errors decreased by 21.47% and by 85.74% after pre-training and after PTFRL, respectively. Similarly, the terminal tracking errors decreased by 45.68% and 90.80% after pre-training and after PTFRL in the y channel, while the errors decreased by 42.53% and 94.27% after pre-training and after PTFRL in the z channel.

(b) In Figure 8, it is seen that, compared with the condition before pre-training, the max tracking error decreased by 21.47% after pre-training, as well as 69.36% after PTFRL in the x channel. In the y channel, compared with the max tracking error before pre-training, it decreased by 57.26% after pre-training and after PTFRL, because the max error equaled the initial error. Besides, the max error in the z channel decreased by 42.53% and by 73.76% after pre-training and after PTFRL, respectively.

(c) Figure 9 shows that if the ideal total reward was set as the target value, the real total reward in the x channel improved by 38.34% and by 97.97% after pre-training and after PTFRL, compared with that before pre-training. In addition, the reward improved by 70.49% and 99.15% after pre-training and after PTFRL in the y channel. As for the z channel, compared with the real total reward before pre-training, the reward improved by 66.98% and 99.67% after pre-training and after PTFRL, respectively.

7. Conclusions

To help a pursuer find its advantaged control policy in a one-to-one game in space, an algorithm of pre-trained fuzzy reinforcement learning (PTFRL) was proposed in this paper. To reduce the difficulty of solving without prior information, the man-made model was defined as an estimated environment. By employing the fuzzy inference systems, an actor-critic learning framework, which could be divided into x, y, and z channels, was established. To make use of the estimated optimal strategy, a pre-training process was conducted through initializing the consequent set of the pursuer. With the inputs of the relative position and the relative velocity in each channel, the proposed algorithm controlled the pursuer optimally. By comparing the simulation results before pre-training, after pre-training, and after

PTFRL, it was seen that the tracking errors were effectively decreased after the pre-training process and further approached zero after the proposed PTFRL.

Author Contributions: Conceptualization, Y.S.; Data curation, X.W.; Formal analysis, X.W.; Funding acquisition, Y.S.; Investigation, P.S.; Methodology, X.W.; Project administration, P.S.; Resources, Y.Z.; Supervision, Y.Z.; Writing—original draft, X.W.; Writing—review & editing, P.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (grant number: 11572019) and the Shanghai Academy of Spaceflight Technology (grant number: SAST2019084).

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

References

1. Izzo, D.; Märtens, M.; Pan, B. A survey on artificial intelligence trends in spacecraft guidance dynamics and control. *Astrodynamics* **2019**, *3*, 287–299. [CrossRef]
2. Pan, H.; Kapila, V. Adaptive nonlinear control for spacecraft formation flying with coupled translational and attitude dynamics. In Proceedings of the IEEE Conference on Decision and Control, Orlando, FL, USA, 4–7 December 2001; Volume 3, pp. 2057–2062.
3. Pan, H.; Wong, H.; Kapila, V. Output feedback control for spacecraft with coupled translation and attitude dynamics. In Proceedings of the American Control Conference, Nassau, Bahamas, 14–17 December 2004; Volume 4, pp. 2419–2426.
4. Xin, M.; Pan, H. Nonlinear optimal control of spacecraft approaching a tumbling target. *Aerosp. Sci. Technol.* **2011**, *15*, 79–89. [CrossRef]
5. Singla, P.; Subbarao, K.; Junkins, J.L. Adaptive Output Feedback Control for Spacecraft Rendezvous and Docking Under Measurement Uncertainty. *J. Guid. Control Dyn.* **2006**, *29*, 892–902. [CrossRef]
6. Chen, B.; Geng, Y. Super twisting controller for on-orbit servicing to non-cooperative target. *Chin. J. Aeronaut.* **2015**, *28*, 285–293. [CrossRef]
7. Huang, Y.; Jia, Y. Robust adaptive fixed-time tracking control of 6-DOF spacecraft fly-around mission for noncooperative target. *Int. J. Robust Nonlinear Control* **2018**, *28*, 2598–2618. [CrossRef]
8. Sun, L.; Huo, W.; Jiao, Z. Adaptive Backstepping Control of Spacecraft Rendezvous and Proximity Operations with Input Saturation and Full-state Constraint. *IEEE Trans. Ind. Electron.* **2017**, *64*, 480–492. [CrossRef]
9. Pukdeboon, C. Inverse optimal sliding mode control of spacecraft with coupled translation and attitude dynamics. *Int. J. Syst. Sci.* **2015**, *46*, 2421–2438. [CrossRef]
10. Cruz, J.B.; Chen, C.I. Series Nash solution of two-person, nonzero-sum, linear-quadratic differential games. *J. Optim. Theory Appl.* **1971**, *7*, 240–257. [CrossRef]
11. Ho, Y.C.; Starr, A.W. Further Properties of Nonzero-Sum Differential Games. *J. Optim. Theory Appl.* **1969**, *3*, 207–219.
12. Guelman, M. Proportional Navigation with a Maneuvering Target. *IEEE Trans. Aerosp. Electron. Syst.* **1972**, *AES-8*, 364–371. [CrossRef]
13. Berkovitz, L.D. The Existence of Value and Saddle Point in Games of Fixed Duration. *SIAM J. Control Optim.* **1985**, *23*, 172–196. [CrossRef]
14. Breitner, M.H.; Pesch, H.J.; Grimm, W. Complex differential games of pursuit-evasion type with state constraints, part 2: Numerical computation of optimal open-loop strategies. *J. Optim. Theory Appl.* **1993**, *78*, 443–463. [CrossRef]
15. Horie, K.; Conway, B.A. Optimal Fighter Pursuit-Evasion Maneuvers Found Via Two-Sided Optimization. *J. Guid. Control Dyn.* **2006**, *29*, 105–112. [CrossRef]
16. Stupik, J.; Pontani, M.; Conway, B. Optimal Pursuit/Evasion Spacecraft Trajectories in the Hill Reference Frame. In Proceedings of the AIAA/AAS Astrodynamics Specialist Conference, Hilton Head, SC, USA, 11–15 August 2013.
17. Jagat, A.; Sinclair, A.J. Optimization of Spacecraft Pursuit-Evasion Game Trajectories in the Euler-Hill Reference Frame. In Proceedings of the AIAA/AAS Astrodynamics Specialist Conference, San Diego, CA, USA, 4–7 August 2014.
18. Jagat, A.; Sinclair, A.J. Nonlinear Control for Spacecraft Pursuit-Evasion Game Using State-Dependent Riccati Equation Method. *IEEE Trans. Aerosp. Electron. Syst.* **2017**, *53*, 3032–3042. [CrossRef]

19. Sutton, R.S.; Barto, A.G.; Williams, R.J. Reinforcement learning is direct adaptive optimal control. *IEEE Control Syst. Mag.* **1992**, *12*, 19–22.
20. Watkins, C.J.C.H.; Dayan, P. Technical Note: Q-Learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
21. Van Hasselt, H.; Guez, A.; Silver, D. Deep reinforcement learning with double q-learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016.
22. Er, M.J.; Deng, C. Obstacle avoidance of a mobile robot using hybrid learning approach. *IEEE Trans. Ind. Electron.* **2005**, *52*, 898–905. [CrossRef]
23. Dai, X.; Li, C.K.; Rad, A.B. An approach to tune fuzzy controllers based on reinforcement learning for autonomous vehicle control. *IEEE Trans. Intell. Transp. Syst.* **2005**, *6*, 285–293. [CrossRef]
24. Xiao, H.; Li, L.; Zhou, F. Mobile Robot Path Planning Based on Q-ANN. In Proceedings of the 2007 IEEE International Conference on Automation and Logistics, Jinan, China, 18–21 August 2007; pp. 2650–2654.
25. Hung, S.M.; Givigi, S.N. A Q-Learning Approach to Flocking With UAVs in a Stochastic Environment. *IEEE Trans. Cybern.* **2017**, *47*, 186–197. [CrossRef]
26. Bilgin, A.T.; Kadioglu-Urtis, E. An approach to multi-agent pursuit evasion games using reinforcement learning. In Proceedings of the 2015 International Conference on Advanced Robotics (ICAR), Istanbul, Turkey, 27–31 July 2015; pp. 164–169. [CrossRef]
27. Analikwu, C.V.; Schwartz, H.M. Reinforcement learning in the guarding a territory game. In Proceedings of the 2016 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Vancouver, BC, Canada, 24–29 July 2016; pp. 1007–1014. [CrossRef]
28. Awheda, M.D.; Schwartz, H.M. A Residual Gradient Fuzzy Reinforcement Learning Algorithm for Differential Games. *Int. J. Fuzzy Syst.* **2017**, *19*, 1058–1076. [CrossRef]
29. Desouky, S.; Schwartz, H. Self-learning fuzzy logic controllers for pursuit–evasion differential games. *Robot. Auton. Syst.* **2011**, *59*, 22–33. [CrossRef]
30. Clohessy, W.H.; Wiltshire, R.S. Terminal Guidance System for Satellite Rendezvous. *J. Aerosp. Sci.* **1960**, *27*, 653–658. [CrossRef]
31. Takagi, T.; Sugeno, M. Fuzzy Identification of Systems and Its Applications to Modeling and Control. *Readings Fuzzy Sets Intell. Syst.* **1993**, *15*, 387–403.
32. Goldberg, D.; Richardson, J. Genetic Algorithms with Sharing for Multi-modal Function Optimization. In Proceedings of the International Conference on Genetic Algorithms, Cambridge, MA, USA, 28–31 July 1987; pp. 41–49.