Journal of Systems Engineering and Electronics
Vol. 32, No. 4, August 2021, pp.927 – 938

A guidance method for coplanar orbital interception based on reinforcement learning

ZENG Xin, ZHU Yanwei*, YANG Leping, and ZHANG Chengming

College of Aeronautics and Astronautics, National University of Defense Technology, Changsha 410073, China

Abstract: This paper investigates the guidance method based on reinforcement learning (RL) for the coplanar orbital interception in a continuous low-thrust scenario. The problem is formulated into a Markov decision process (MDP) model, then a well-designed RL algorithm, experience based deep deterministic policy gradient (EBDDPG), is proposed to solve it. By taking the advantage of prior information generated through the optimal control model, the proposed algorithm not only resolves the convergence problem of the common RL algorithm, but also successfully trains an efficient deep neural network (DNN) controller for the chaser spacecraft to generate the control sequence. Numerical simulation results show that the proposed algorithm is feasible and the trained DNN controller significantly improves the efficiency over traditional optimization methods by roughly two orders of magnitude.

Keywords: orbital interception, reinforcement learning (RL), Markov decision process (MDP), deep neural network (DNN).

DOI: 10.23919/JSEE.2021.000079

1. Introduction

Recently, a growing number of countries and private sectors are getting involved in space, resulting in more congestion and competition in space. The technology of orbital rendezvous and interception has already become an important symbol to measure the development level of space technology [1]. Generally, the optimal control sequence for interception is obtained by solving the corresponding optimal control problem, which requires much time and is not efficient enough to apply online. Hence, the technique to improve the efficiency should be further investigated. Fortunately, with the technology of artificial intelligence becoming mature, increasing works address the use of the reinforcement learning (RL) in space such as trajectory guidance [2−6], radar control [7,8], motion control [9], attitude control [10], maneuver detection [11] and landing problems [12]. The deep neural network (DNN) is expected to be an ideal way to improve the efficiency of obtaining the control sequence. This paper concerns the coplanar orbital interception problem and presents a well-designed algorithm, experience based deep deterministic policy gradient (EBDDPG). It can successfully train a DNN controller which improves efficiency by roughly two orders of magnitude compared with traditional optimization methods.

On the one hand, as a special orbital pursuit-evasion game, the orbital interception problem is attracting increased attention from scholars in fields [13−16]. Zeng et al. [17] concerned the long-distance orbital pursuit-evasion games and introduced a mixed global-local optimization strategy to improve the efficiency. Ye et al. [18] investigated the proximity satellite pursuit-evasion game where the pursuer carried three orthogonal thrusters, while the evader had a single thruster, and the heuristic searching method and the Newton method were introduced to solve the open-loop control. Then, the incomplete-information game was addressed in [19], and a currently optimal escape strategy was proposed based on estimation for the evader. Moreover, instead of the near-circle orbit, the elliptical orbital proximity operations differential game was investigated in [20].

On the other hand, as the most common way for orbital interception, the coplanar orbital interception is essentially a problem of trajectory planning involving guidance and control, and the original problem is usually converted into a two-point boundary value problem (TPBVP) that employs an optimization algorithm to achieve an optimal solution [21−23]. In [24], the hp-adaptive pseudo-spectral method was introduced to investigate the optimal rendezvous and interception with elliptic initial and target orbits, which reduced the time consumption. Moreover, in [25], the multiple targets were considered, and the design method of orbit interception based on traversing

Manuscript received November 12, 2020.
*Corresponding author.
This work was supported by the National Defense Science and Technology Innovation (18-163-15-LZ-001-004-13).

points simplified the interception and saved time and fuel. In [26], a homotopic strategy was applied based on the theory of optimal control to gain a speed advantage over traditional methods. By solving a one-sided optimal control problem to rapidly estimate the costates, Carr et al. [27] removed global optimization from the traditional method to significantly improve the efficiency. However, optimal guidance with time-critical autonomous performance which involves planning the trajectory online is hardly developed. This limitation is due to the fact that it takes a long time to obtain the iteration solution for a given initial state through traditional optimization methods.

The RL involves learning from environment in which the learner is not told which action to take but instead must discover which action yields the highest reward by attempting these actions [28]. Recently, an increasing number of work around the RL is getting involved in space. The biggest advantage of the RL is that the trained DNN can significantly improve the efficiency of complex iterative processes of traditional optimization methods. A DNN was used to improve the efficiency of identification in [29]. And, the actor-critic RL algorithm was used to create a closed-loop guidance algorithm for planetary landing in [30,31]. It improves the classical feedback guidance in terms of time and can even be implemented directly on-line. Similarly, the optimal integrated guidance for pinpoint Mars landing based on the RL was investigated in [32]. By converting complex three-dimensional terrains into a reward matrix, the path-planning for hopping rovers on asteroid surface is investigated by using the RL, and a DNN controller is developed to determine the optimal actions. Besides guidance, the RL is also used to design a flexible heuristic method to produce trajectory initial guesses for cis-lunar transportation missions [33]. However, little attention is given to the orbital interception domain.

This paper explores the application of the RL algorithm to obtain a DNN controller for the coplanar orbital interception. The rest of this paper is divided into four sections. In Section 2, the coplanar orbital interception is formulated into a Markov decision process (MDP) model, and an optimal control model is also introduced to generate the training set. Then, the EBDDPG algorithm consisting of three phases is presented in Section 3. Moreover, the numerical simulation is given in Section 4. Finally, the conclusion is drawn in Section 5.

2. Problem formulation

In this section, to apply the RL algorithm to this problem, the coplanar orbital interception is formulated into an MDP model. Moreover, to take advantage of the prior information, the optimal control model is also introduced to generate the training set.

The thrust to mass ratio is treated as a constant for the chaser spacecraft, whereas there is no maneuver capability for the target spacecraft. Both spacecraft are initially in a circular orbit. The chaser spacecraft tries to approach the target spacecraft as quickly as possible. Hence, the objective function is simply

$$\phi = t_f \tag{1}$$

where t_f denotes the terminal time.

The Earth-centred inertial coplanar coordinate is shown in Fig. 1, in which the X axis points to the initial position of the chaser spacecraft and the Y axis is in the coplanar plane making the gradient of the azimuth of the target spacecraft positive. The state variables for the chaser spacecraft are radius r_c, velocity v_c, azimuth θ_c, and velocity azimuth A_c, while the state variables for the target spacecraft are radius r_t and azimuth θ_t. The control variable for the chaser spacecraft is the thrust direction angle φ_c. The thrust magnitude and mass for the chaser spacecraft are T and m, respectively. The subscript c denotes the chaser spacecraft while t denotes the target spacecraft.

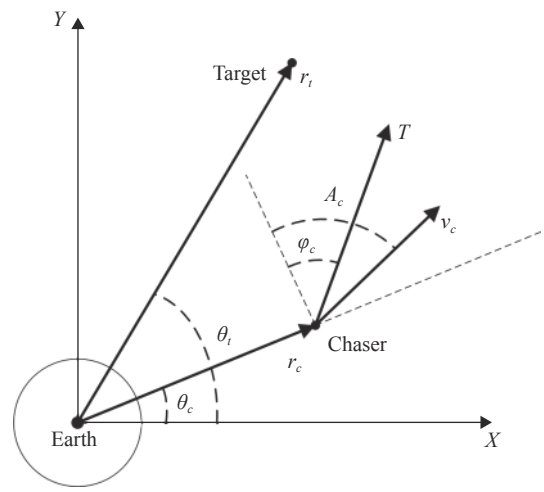

Fig. 1 Coplanar coordinates, state variables, and control variables

The chaser spacecraft drives the state $\boldsymbol{x}$ of the dynamic system with control variable $\boldsymbol{u}$, and its objective is to arrive in the vicinity of the target spacecraft as quickly as possible. Hence, the dynamic system can be described as follows:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) \tag{2}$$

where

$$\boldsymbol{x} = [r_c, v_c, \theta_c, A_c, r_t, \theta_t]^{\mathrm{T}},$$
$$\boldsymbol{u} = [\varphi_c].$$

Then, the equations of motion for the dynamic system

can be derived as follows:

$$\begin{aligned}\dot{r}_c &= v_c \sin A_c,\\ \dot{v}_c &= \frac{T}{m}\cos(\varphi_c - A_c) - \frac{\sin A_c}{r_c^2},\\ \dot{\theta}_c &= \frac{v_c \cos A_c}{r_c},\\ v_c\dot{A}_c &= \frac{T}{m}\sin(\varphi_c - A_c) - \left(\frac{1}{r_c^2} - \frac{v_c^2}{r_c}\right)\cos A_c,\\ \dot{r}_t &= 0,\\ \dot{\theta}_t &= r_t^{-\frac{3}{2}}.\end{aligned}$$

The boundary condition $\boldsymbol{\Psi}$ should be

$$\boldsymbol{\Psi} = \boldsymbol{r}_{cf} - \boldsymbol{r}_{tf} = 0 \tag{3}$$

where $\boldsymbol{r}_{cf}$ and $\boldsymbol{r}_{tf}$ denote the terminal radius vectors of the chaser spacecraft and the target spacecraft, respectively. It means that the mission ultimately ends up with $\boldsymbol{r}_{cf} = \boldsymbol{r}_{tf}$. Hence, the boundary condition can be further written as

$$\boldsymbol{\Psi} = [r_{cf} - r_{tf}, \theta_{cf} - \theta_{tf}]^{\mathrm{T}}. \tag{4}$$

2.1 MDP model

Consider an MDP $(\boldsymbol{X}, \boldsymbol{U}, P, R)$, where $\boldsymbol{X}$ is a set of states, $\boldsymbol{U}$ is a set of actions, P represents the transition probabilities, and R is the reward function. The set of states $\boldsymbol{X}$ matches the space of the state vector $\boldsymbol{x}$, while the set of actions $\boldsymbol{U}$ matches the space of the control variable $\boldsymbol{u}$.

Since the coplanar orbital interception follows the dynamic system, by introducing the time interval τ, the transition probabilities can be written as follows:

$$P_{\boldsymbol{x},\boldsymbol{x}'}^{\boldsymbol{u}} = 1 \tag{5}$$

where

$$\boldsymbol{x}' = \int_t^{t+\tau} f(\boldsymbol{x}, \boldsymbol{u}, t)\mathrm{d}t.$$

$\boldsymbol{x} \in \boldsymbol{X}$ and $\boldsymbol{x}' \in \boldsymbol{X}$ denote the state at time t and $t+\tau$, respectively. $\boldsymbol{u} \in \boldsymbol{U}$ is the control variable. $P_{\boldsymbol{x},\boldsymbol{x}'}^{\boldsymbol{u}}$ gives the conditional probability of transition from state $\boldsymbol{x}$ to state $\boldsymbol{x}'$ by taking action $\boldsymbol{u}$.

Given the initial state $\boldsymbol{x}_0$, an episode of the coplanar orbital interception occurs when the distance between the chaser spacecraft and the target spacecraft d exceeds the presupposed maximum distance $d_{\max}$ or the sum of the transitions exceeds the presupposed value of m. An episode consisting of n transitions is shown as follows:

$$\overbrace{\boldsymbol{x}_0, \boldsymbol{u}_0, r_0, \underbrace{\boldsymbol{x}_1, \boldsymbol{u}_1, r_1, \boldsymbol{x}_2}_{\text{transition 2}}}^{\text{transition 1}}, \cdots$$

The problem of the coplanar orbital interception decision-making process involves finding a strategy mapping from state $\boldsymbol{x}$ to action $\boldsymbol{u}$ that minimizes the sum of future rewards J as follows:

$$J = \sum_{i=0} \gamma^i r_i \tag{6}$$

where $0 \leqslant \gamma \leqslant 1$ is a discount factor that reduces the weight of the reward incurred further in the future. r_i denotes the reward of transition i.

Since the chaser spacecraft aims to approach the target spacecraft in minimum time, the reward function is defined as follows:

$$R_{\boldsymbol{x},\boldsymbol{x}'}^{\boldsymbol{u}} = \begin{cases} -100, & d \leqslant d_{\max} \\ 1, & \text{otherwise} \end{cases} \tag{7}$$

where $R_{\boldsymbol{x},\boldsymbol{x}'}^{\boldsymbol{u}}$ gives the expected immediate reward r after the transition from state $\boldsymbol{x}$ to state $\boldsymbol{x}'$ by taking action $\boldsymbol{u}$.

2.2 Optimal control model

The Hamiltonian H and the function of the terminal conditions Φ are introduced as follows:

$$H = \boldsymbol{\lambda}^{\mathrm{T}} \boldsymbol{f}, \tag{8}$$

$$\Phi = \phi + \boldsymbol{\nu}^{\mathrm{T}} \boldsymbol{\Psi}, \tag{9}$$

where $\boldsymbol{\lambda}$ is the costate vector conjugate to the state function and $\boldsymbol{\nu}$ is the costate vector conjugate to the boundary conditions.

For an open-loop representation of an optimal feedback strategy, the necessary conditions for the costate variables can be written as follows:

$$\dot{\boldsymbol{\lambda}} = -\left[\frac{\partial H}{\partial \boldsymbol{x}}\right]^{\mathrm{T}} = -\left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right]^{\mathrm{T}} \boldsymbol{\lambda} \tag{10}$$

subject to

$$\lambda_{if} - \frac{\partial \Phi}{\partial x_{if}} = 0 \tag{11}$$

where $i = 1, 2, \cdots, 6$.

The costate function and the corresponding boundary of the costate variables for this mission can be derived as follows:

$$\begin{aligned}\dot{\lambda}_1 = &-\frac{2\sin A_c}{r_c^3}\lambda_2 + \frac{v_c \cos A_c}{r_c^2}\lambda_3 + \\ &\frac{\cos A_c}{v_c}\left(\frac{v_c^2}{r_c^2} - \frac{2}{r_c^3}\right)\lambda_4,\\ \dot{\lambda}_2 = &-\sin A_c \lambda_1 - \frac{\cos A_c}{r_c}\lambda_3 + \\ &\left(\frac{\cos A_c}{v_c^2}\left(\frac{v_c^2}{r_c} - \frac{1}{r_c^2}\right) - \right.\\ &\left.\frac{\frac{T}{m}\sin(A_c - \varphi_c)}{v_c^2} - \frac{2\cos A_c}{r_c}\right)\lambda_4,\end{aligned}$$

$$\dot{\lambda}_3 = 0,$$

$$\dot{\lambda}_4 = -v_c \cos A_c \lambda_1 + \left(\frac{T}{m}\sin(A_c - \varphi_c) + \frac{\cos A_c}{r_c^2}\right)\lambda_2 + \frac{v_c \sin A_c}{r_c}\lambda_3 + \left(\frac{\frac{T}{m}\cos(A_c - \varphi_c)}{v_c} + \frac{\sin A_c}{v_c}\left(\frac{v_c^2}{r_c} - \frac{1}{r_c^2}\right)\right)\lambda_4,$$

$$\dot{\lambda}_5 = \frac{3}{2} r_t^{-\frac{5}{2}} \lambda_6,$$

$$\dot{\lambda}_6 = 0,$$

subject to

$$\lambda_{1f} + \lambda_{5f} = 0,$$
$$\lambda_{3f} + \lambda_{6f} = 0,$$
$$\lambda_{2f} = 0,$$
$$\lambda_{4f} = 0.$$

Moreover, the necessary boundary condition for the Hamiltonian H can be written as follows:

$$\frac{\partial \Phi}{\partial t_f} + H_f = 0. \tag{12}$$

Furthermore,

$$1 + H_f = 0. \tag{13}$$

For the chaser spacecraft, the optimal control variable $\boldsymbol{u}^*$ must satisfy the following equations:

$$\frac{\partial H}{\partial \boldsymbol{u}^*} = 0, \tag{14}$$

$$\frac{\partial^2 H}{\partial \boldsymbol{u}^{*2}} \geqslant 0. \tag{15}$$

Hence, for the chaser spacecraft, the optimal control variable φ_c^* satisfies the following equations:

$$\lambda_2 \sin(A_c - \varphi_c^*) + \frac{\lambda_4}{v_c}\cos(A_c - \varphi_c^*) = 0, \tag{16}$$

$$\frac{\lambda_4}{v_c}\sin(A_c - \varphi_c^*) - \lambda_2 \cos(A_c - \varphi_c^*) \geqslant 0. \tag{17}$$

Therefore,

$$\varphi_c^* = A_c + \arctan\left(\frac{\lambda_4}{\lambda_2 v_c}\right), \tag{18}$$

or

$$\varphi_c^* = A_c + \arctan\left(\frac{\lambda_4}{\lambda_2 v_c}\right) + \pi, \tag{19}$$

where (15) can be used to choose the correct control variable from (18) and (19).

Finally, (2), (3), (10), (11), (8), (13), (18) and (19) describe the coplanar orbital interception and constitute a TPBVP.

In conclusion, unlike the traditional optimal control model, there are no costate variables and Hamiltonian in the MDP model, which makes it more simple. Moreover, the coplanar orbital interception will not be transformed into a TPBVP requiring time-consuming iterative optimization algorithm in the MDP model.

3. Algorithm

Generally, there is no requirement of prior information in the RL. However, the environment of the coplanar orbital interception is more complex than traditional environments for the RL (for example, the cartpole, pendulum or atari). It will result in divergence applying common RL algorithms such as the deep deterministic policy gradient (DDPG) [34], advantage actor critic (A2C) [35] or trust region policy optimization (TRPO) [36] to this environment directly.

To resolve the convergence problem, the EBDDPG designed based on the DDPG algorithm [34] is introduced in this section. There are three phases in the EBDDPG: the preparation phase, the experience initialization phase and the gradient descent phase.

3.1 Preparation phase

The purpose of the preparation phase is to prepare the training set T and the validation set V for the following phases. Both sets consist of series of record sequence labeled with an initial state. A record consists of a state, an action and the sum of future rewards as follows:

$$\overbrace{\boldsymbol{x}_0, \boldsymbol{u}_0, J_0}^{\text{record 1}}, \underbrace{\boldsymbol{x}_1, \boldsymbol{u}_1, J_1}_{\text{record 2}}, \overbrace{\boldsymbol{x}_2, \boldsymbol{u}_2, J_2}^{\text{record 3}}, \cdots$$

where the record sequence is generated by solving the TPBVP of the coplanar orbital interception, and the sum of future rewards is calculated according to (6).

Consider a general nonlinear optimization problem of the following form:

$$h = \min_{\boldsymbol{p} \in \mathbb{R}^n} h(\boldsymbol{p}) \tag{20}$$

where h is the objective function and $\boldsymbol{p}$ represents the optimization parameters. The initial costate and terminal time are treated as optimization parameters as follows:

$$\boldsymbol{p} = [\lambda_{10}, \lambda_{20}, \lambda_{30}, \lambda_{40}, \lambda_{50}, \lambda_{60}, t_f].$$

By introducing the weight coefficient vector $\boldsymbol{k}$, the objective function $h(\boldsymbol{p})$ can be transformed from the extended boundary conditions $\boldsymbol{\Psi}_{\text{ext}}$ as follows:

$$h(\boldsymbol{p}) = \boldsymbol{k}^{\mathrm{T}} |\boldsymbol{\Psi}_{\text{ext}}| \tag{21}$$

where

$$\boldsymbol{\Psi}_{\text{ext}} = [r_{cf} - r_{tf}, \theta_{cf} - \theta_{tf}, \cdots, \lambda_{1f} + \lambda_{5f}, \lambda_{3f} + \lambda_{6f}, \lambda_{2f}, \lambda_{4f}, \cdots, 1 + H_f]^{\mathrm{T}}.$$

To solve the optimization problem stated above, the improved stochastic ranking evolution strategy (ISRES) [37] and the principal-axis method (PAM) [38] are introduced as the global optimization algorithm and the local optimization algorithm of the traditional optimization method, respectively, in this paper.

The termination conditions for the global optimization and the local optimization can be bounds on function evaluations, function value tolerance, and parameter tolerance. Hence, by specifying the fractional tolerance δh_r, δp_r and absolute tolerance δh_a, δp_a on the value h and parameters $\boldsymbol{p}$, respectively, the global optimization and the local optimization will halt when the number of function evaluations reaches maxeval N_e or the optimum is found to be within the desired tolerance. The maximum number of function evaluations N_e, fractional tolerance δh_r, δp_r and absolute tolerance δh_a, δp_a are set as follows:

$$N_e = 2 \times 10^4,$$
$$\delta h_a = \delta p_a = 1 \times 10^{-4},$$
$$\delta h_r = \delta p_r = 1 \times 10^{-6}.$$

3.2 Experience initialization phase

The purpose of the experience initialization phase is to train and validate the DNNs with training set T and validation set V, respectively. Like the common DDPG algorithm, there are four DNNs, policy critic network Q, policy actor network μ, target critic network Q' and target actor network μ'. The policy critic network and the target critic network are used to approximate the Q-function which outputs the sum of future rewards by taking an action in a given state, while the policy actor network and the target actor network are used to approximate the strategy μ which is a function of the current state and outputs the action. By initializing the networks above with training set T and validation set V in the experience initialization phase, the convergence of the algorithm will be improved effectively.

Actually, the idea of the experience initialization phase is taken from the fuzzy inference system of fuzzy control [39], which provides the agent with prior information by fuzzy rules before training. Similarly, the training set T is treated as the rules to initialize the networks, while the validation set V is used to validate the initialization.

In this paper, the structures of the actor and the critic networks are shown in Fig. 2. The actor network consists of seven linear layers applying the Relu function and one linear head layer applying the Tanh function. In contrast, the critic network consists of five linear layers applying the Relu function and one linear head layer without any activation function.

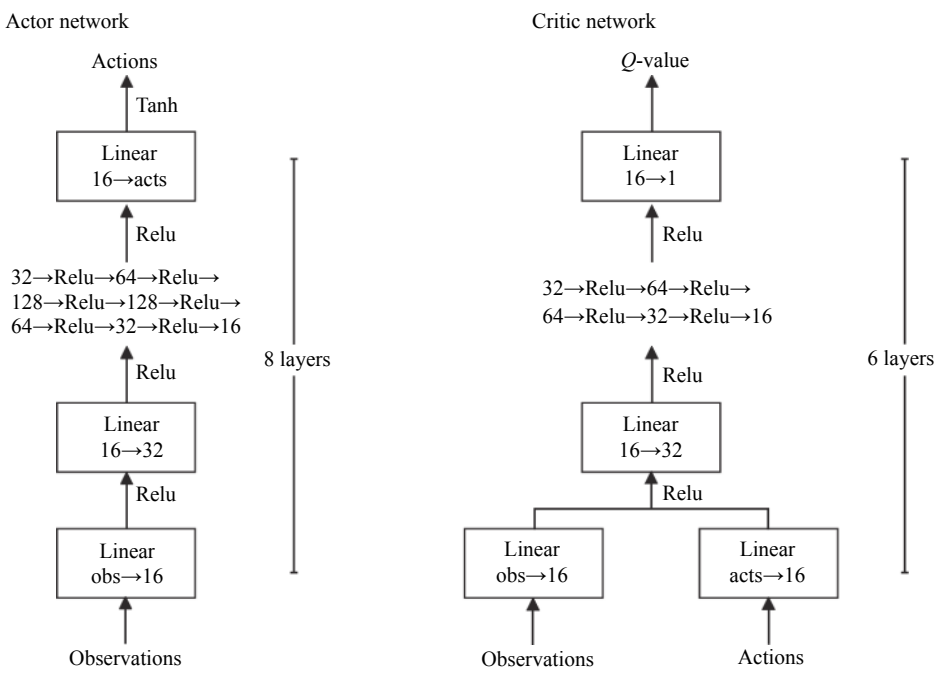

Fig. 2 Structure of the actor and the critic networks

Besides the DNNs, as shown in Fig. 3, replay memory M is also an important part in the common DDPG algorithm. Similarly, there is a record replay memory M' introduced in this phase. Unlike the replay memory M,

the record replay memory M' is used to store records rather than transitions. Similarly, the record replay memory M' can also provide records with "push" and "sample" interfaces. By randomly sampling records from the memory, the policy critic network can be initialized by the expected sum of future rewards, and the policy actor network can be initialized by the expected action.

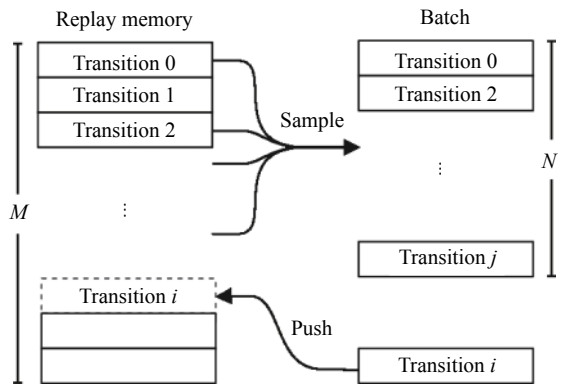

Fig. 3 Replay memory

During this phase, to evaluate the initialization, the loss of the policy critic network δ_c and the loss of the policy actor network δ_a are calculated frequently for all records in the validation set as follows:

$$\delta_c = J_i - Q(\boldsymbol{x}_i, \boldsymbol{u}_i|\theta^Q), \tag{22}$$

$$\delta_a = \boldsymbol{u}_i - \mu(\boldsymbol{x}_i|\theta^\mu). \tag{23}$$

At the end of the experience initialization phase, the parameters of the policy critic network and the policy actor network are copied to the target critic network and the target actor network, respectively.

3.3 Gradient descent phase

The purpose of the gradient descent phase is to train the DNNs. To stabilize and improve the training procedure, as shown in Fig. 3, a replay memory is introduced to store the transitions for reuse. By sampling from the replay memory randomly, a batch of transitions will be decorrelated, which will greatly stabilize and improve the training procedure [34].

The replay memory, in essence, is a cyclic buffer of bounded size M, which holds the transitions. The replay memory can also provide a random batch of N transitions for training.

The parameters of the DNNs are all randomly initialized, and the soft-replacement rate is fixed in the common DDPG algorithm, whereas in the gradient descent phase, the parameters of the networks have been initialized in the experience initialization phase and the soft-replacement rate is updated according to the success rate.

During the gradient descent phase, a noise process $\mathcal{N}$ is introduced to construct an exploration policy $\hat{\mu}$:

$$\hat{\mu}(\boldsymbol{x}_t) = \mu(\boldsymbol{x}_t|\theta^\mu) + \mathcal{N}_\epsilon \tag{24}$$

where

$$\mathcal{N}_\epsilon = \begin{cases} 0, & \epsilon > \epsilon_t \\ \mathcal{N}, & \epsilon \leqslant \epsilon_t \end{cases}.$$

$\epsilon \in (0,1)$ is a random value, and ϵ_t is the threshold that decays from $\epsilon_{\max}$ to $\epsilon_{\min}$ during this phase as follows:

$$\epsilon_t = \epsilon_{\min} + (\epsilon_{\max} - \epsilon_{\min})\mathrm{e}^{-\frac{j}{\epsilon_d}} \tag{25}$$

where j represents the steps elapsed and ϵ_d is the decay rate. As a result of the ϵ-greedy strategy, the influence of the noise decays as the steps proceed.

Since any Q-function obeys the Bellman equation for any strategy, define the temporal difference error δ as follows:

$$\delta = Q(\boldsymbol{x}_t, \boldsymbol{u}_t|\theta^Q) - r + \gamma Q'(\boldsymbol{x}_{t+1}, \boldsymbol{u}_{t+1}|\theta^{Q'}) \tag{26}$$

where

$$\boldsymbol{u}_t = \mu(\boldsymbol{x}_t|\theta^\mu),$$
$$\boldsymbol{u}_{t+1} = \mu'(\boldsymbol{x}_{t+1}|\theta^{\mu'}).$$

Define the Huber loss $\mathcal{L}$ as follows:

$$\mathcal{L} = \frac{1}{N}\sum \mathcal{L}(\delta) \tag{27}$$

where

$$\mathcal{L}(\delta) = \begin{cases} \frac{1}{2}\delta^2, & |\delta| \leqslant 1 \\ |\delta| - \frac{1}{2}, & \text{otherwise} \end{cases}.$$

The Huber loss acts like the mean squared error when the error is small but like the mean absolute error when the error is large. It is used to train the policy critic networks.

For the policy actor networks, the output of the policy critic network is treated as the loss to be minimized, so the sampled gradient ∇, which is used to update the policy actor network, is defined as follows:

$$\nabla = \frac{1}{N}\sum \nabla_{\boldsymbol{u}} Q(\boldsymbol{x}, \boldsymbol{u}|\theta^Q)\nabla_{\theta^\mu}\mu(\boldsymbol{x}|\theta^\mu). \tag{28}$$

For the target critic network and the target actor network, their parameters are updated during the training by soft replacement rather than gradient descent as follows:

$$\theta^{Q'} \leftarrow \alpha\theta^Q + (1-\alpha)\theta^{Q'}, \tag{29}$$

$$\theta^{\mu'} \leftarrow \alpha\theta^\mu + (1-\alpha)\theta^{\mu'}, \tag{30}$$

where α is the replacement rate.

The success rate can be calculated according to the final distance between the chaser spacecraft and the target spacecraft for each episode. By introducing the success

rate, the replacement rate α will be assigned by the success rate frequently during this phase.

4. Numerical simulation

The distance unit is $U_d = R_e$ and the time unit is $U_t = \sqrt{\frac{DU^3}{\mu_e}}$ where R_e is the Earth radius and μ_e is the Earth planetary constant.

The initial radius of the chaser spacecraft r_{c0} and target spacecraft r_{t0} are as follows:

$$r_{c0} = 1.04U_d,$$
$$r_{t0} = 1.08U_d.$$

Their azimuths are θ_{c0} and θ_{t0} as follows:

$$0 \leqslant \theta_{c0} < 2\pi,$$
$$0 \leqslant \theta_{t0} < 2\pi.$$

The constant thrust-to-mass ratio for the chaser spacecraft is set as follows:

$$\frac{T}{m} = 0.01g$$

where g is the gravitational acceleration on ground.

4.1 Training by EBDDPG

4.1.1 Preparation phase

Consider the following three cases, in which the initial states are all the same except for the initial azimuths, which are equidifferent, i.e., $\frac{\pi}{2}$ from each other. The initial difference of the azimuth $\Delta\theta_0 = \theta_{t0} - \theta_{c0}$ is the same, i.e., $\frac{\pi}{18}$.

$$\text{Case 1: } \theta_{c0} = 0,\ \theta_{t0} = \frac{1}{18}\pi;$$
$$\text{Case 2: } \theta_{c0} = \frac{1}{2}\pi,\ \theta_{t0} = \frac{5}{9}\pi;$$
$$\text{Case 3: } \theta_{c0} = \pi,\ \theta_{t0} = \frac{19}{18}\pi.$$

Solve the three cases above by the algorithm stated in Section 2.2. The trajectory results are shown in Fig. 4, and a plot of the control variable versus time is shown in Fig. 5.

The results show that the shapes of the trajectory results and the values of the control variables are almost the same. Hence, the result exhibits azimuth independence, which means that the control result only depends on the initial difference of the azimuth between the chaser spacecraft and the target spacecraft rather than the azimuth itself.

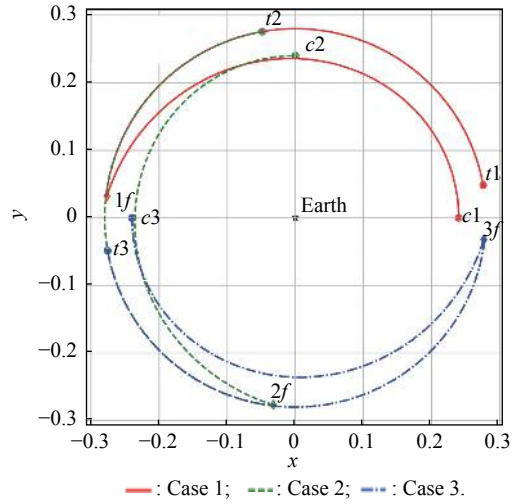

Fig. 4 Trajectory results of three cases with equidifferent azimuths

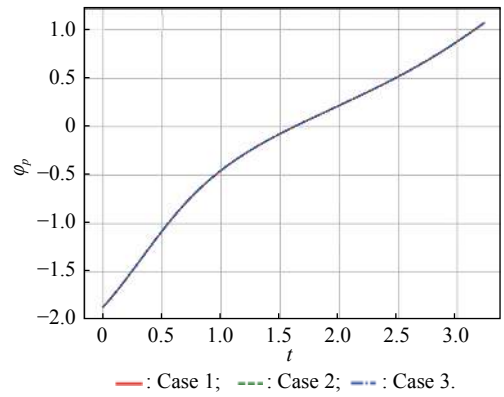

Fig. 5 Control variable vs. time for three cases with equidifferent azimuths

As shown in Fig. 6, the optimized initial difference of the azimuth $\Delta\theta_0$ is approximately $\frac{\pi}{30}$; the farther away the initial difference of the azimuth is from the optimized initial difference of the azimuth, the greater the time required for the chaser spacecraft to approach the target spacecraft.

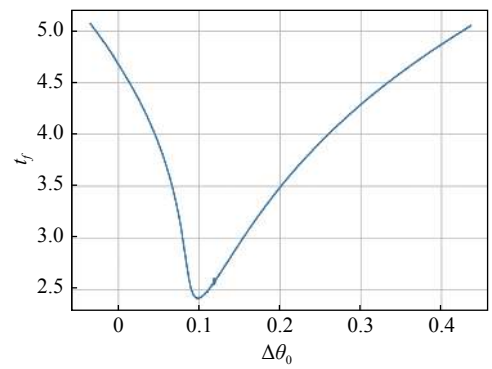

Fig. 6 Normalized terminal time vs. initial difference of the azimuth

As a result, since the chaser spacecraft aims to approach the target spacecraft as quickly as possible in the coplanar orbital interception, by constraining the initial difference of the azimuth as follows:

$$-\frac{1}{90}\pi \leqslant \Delta\theta_0 \leqslant \frac{5}{36}\pi,$$

the terminal time can be guaranteed as follows:

$$t_f \leqslant 5U_t.$$

Apply the algorithm described in Section 3.1. By generating 2 701 points linearly spaced including $-\frac{1}{90}\pi$ and $\frac{5}{36}\pi$, set V_i is obtained by sampling every other 100 points from the 2 701 points and assigning the initial azimuth as follows:

$$\theta_{c0} = 0,$$
$$\theta_{t0} = \Delta\theta_0.$$

Similarly, set T_i consists of the remaining 2 673 points. Then, the invalidation set and the training set can be obtained by setting the discount factor as follows:

$$\gamma = 0.99.$$

4.1.2 Experience initialization phase

The size of the memory and mini-batch is set as follows:

$$M = 10\,000,$$
$$N = 128,$$

and the validation frequency is set to 200.

Apply the algorithm described in Section 3.2. The training process is shown in Fig. 7 and Fig. 8. As shown in the figures, the loss of the critic network and the loss of the actor network decay gradually as the episode elapses and nearly converges to zero (the actor loss is not that smooth at the end, but it is enough for the following training), which means the prior information has been injected into the DNNs successfully.

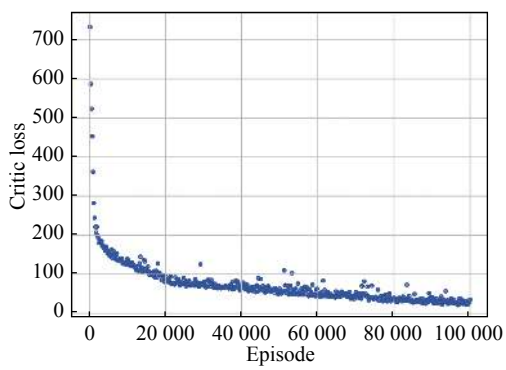

Fig. 7 Critic loss vs. episode

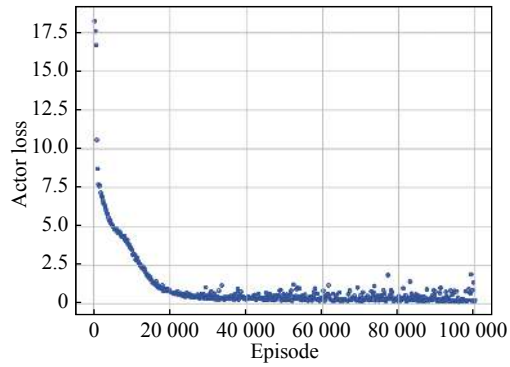

Fig. 8 Actor loss vs. episode

4.1.3 Gradient descent phase

The noise process $\mathcal{N}$ is defined as follows:

$$\mathcal{N} \sim \frac{\pi}{24}\mathcal{N}(-1, 1)$$

where $\mathcal{N}$ represents a uniform distribution.

The parameters of the ϵ-strategy are set as follows:

$$\epsilon_{\min} = 0.05,$$

$$\epsilon_{\max} = 1,$$

$$\epsilon_d = 5\,000.$$

The maximum distance between the chaser spacecraft and the target spacecraft is set as follows:

$$d_{\max} = 0.001U_d,$$

and the game is successful when $d \leqslant d_{\max}$.

For the environment, the state of the environment will be set as "done", when $d \leqslant d_{\max}$ or the number of steps exceeds 250. And, for $d \leqslant d_{\max}$, the state of the environment will also be treated as "successful".

4.2 Comparison with traditional methods

4.2.1 Time comparison

To compare the efficiency of the trained DNN controller and the traditional optimization methods, 100 cases are randomly generated and solved by each one. In this simulation, the traditional optimization method is as stated in Subection 3.2.

As shown in Fig. 9 and Table 1, on the time consumption, the mean value μ and standard deviation σ from the trained DNN controller are both smaller than those from the traditional optimization method, which means that the trained DNN controller is indeed much efficient than the traditional optimization method. More precisely, the trained DNN controller improves the efficiency over the traditional optimization method by roughly two orders of magnitude in solving the coplanar orbital interception.

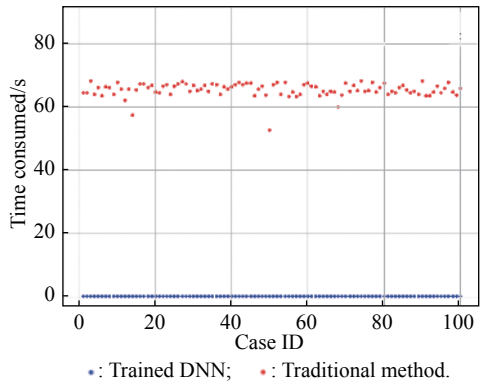

Fig. 9 Comparison on time consumption

Table 1 Comparison on time consumption

Method	μ/s	σ/s
Trained DNN	0.108 8	0.004 0
Traditional method	65.348 1	2.161 0

4.2.2 Comparison in specific case

Consider the following case:

$$\theta_{c0} = 0,\ \ \theta_{t0} = 1.364\,9 \times 10^{-1}.$$

Solve this case by the trained DNN controller and the traditional optimization method. The trajectory result and control variable results are shown in Fig. 10 and Fig. 11, respectively.

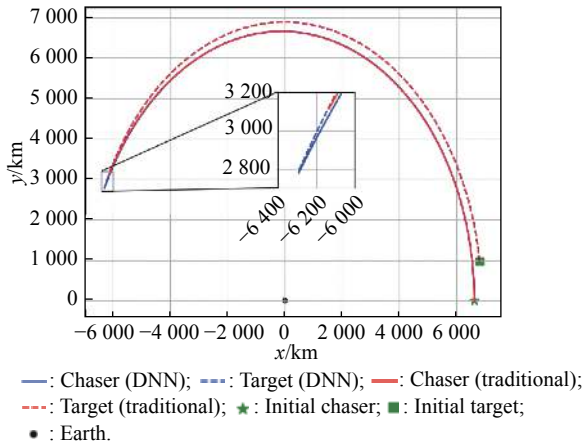

Fig. 10 Trajectory result

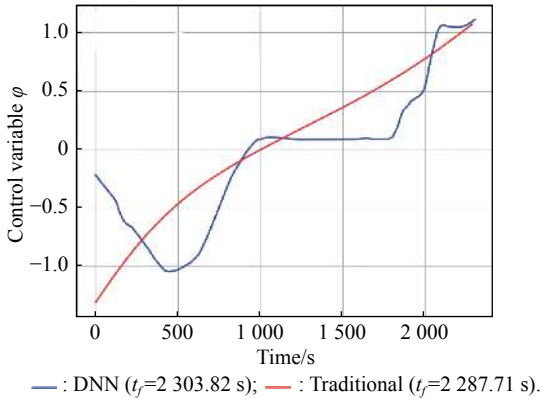

Fig. 11 Control variable vs. time

As shown in Fig. 10, the trajectory result from the trained DNN controller is nearly the same as that from the traditional method. However, as shown in Fig. 11, the control from the traditional method is more smooth than from the trained DNN controller, and it takes less time to finish the interception, by less than one percent.

4.3 Comparison with common RL methods

Apply the EBDDPG algorithm described in Section 3, the DDPG algorithm [34] and the actor-critic (AC) algorithm [35] to this environment. The optimizer is the Adam optimizer [40] and the learning rates for the actor and the critic networks are 10^{-4} and 10^{-3}, respectively. The episode versus the success rate calculated by counting every other 200 episodes of the policy actor network is shown in Fig. 12.

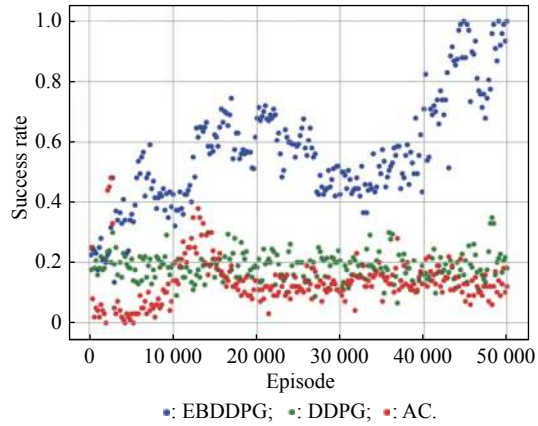

Fig. 12 Success rate vs. episode

As shown in Fig. 12, the success rate is approximately 20% at the beginning and increases to nearly 100% as the training proceeds by EBDDPG. However, without prior information, the success rate maintains around 20% as the training proceeds by DDPG or AC. It is clear that, by taking advantage of the prior information, the convergence problem in the common RL in the coplanar orbital interception can be successfully resolved.

Actually, as stated in Subection 3.1, since the orbital interception environment is much more complex than the traditional environments designed for comparing RL algorithms, for example, the cartpole environment, applying the common reinforcement learning algorithm, for example, the DDPG algorithm or the AC algorithm, to this environment directly will always encounter the convergence problem.

4.4 Monte Carlo simulation

To test the robustness of the model proposed above, the Monte Carlo simulation is performed. The data of this scenario mission is the same as Subection 4.2.2, while the initial condition is dispersed as shown in Table 2 according to the measurement accuracy.

Table 2 Dispersion used in Monte Carlo simulation

Deviation	r/km	v/(m/s)	θ/(°)	A/(°)
3σ	±1	±10	±0.5	±0.5

Fig. 13 to Fig. 16 show $N = 100$ histories of the state elements of the chaser spacecraft in Monte Carlo simulation. Moreover, Fig. 17 shows the dispersion of the final distance between the chaser spacecraft and the target spacecraft. The mean value μ and standard deviation σ of final distance d_f are introduced to quantify the Monte Carlo simulation result as follows:

$$\mu = \sum^{N}(d_f),$$

$$\sigma = \sqrt{\frac{\sum^{N}(d_f-\mu)^2}{N-1}}.$$

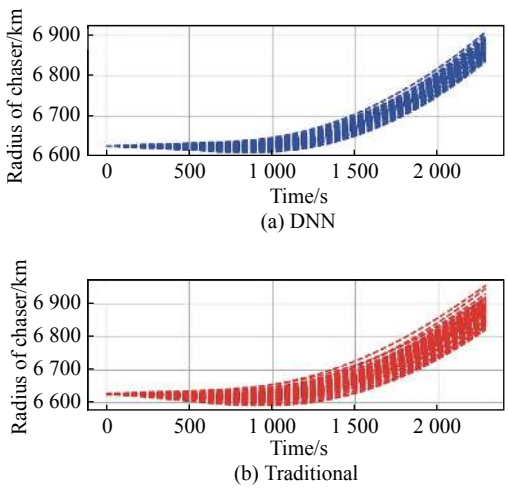

Fig. 13 Radius of chaser vs. time t

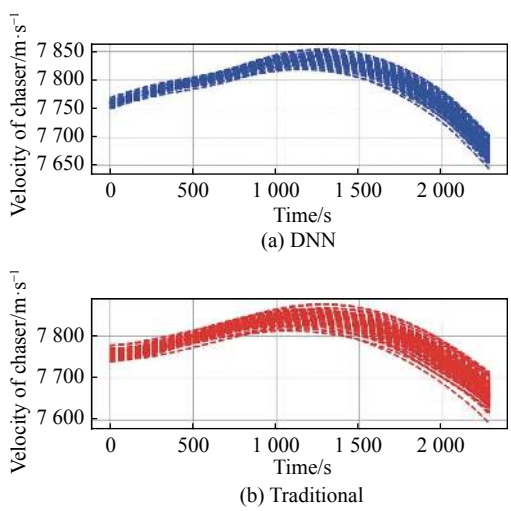

Fig. 14 Velocity of chaser vs. time t

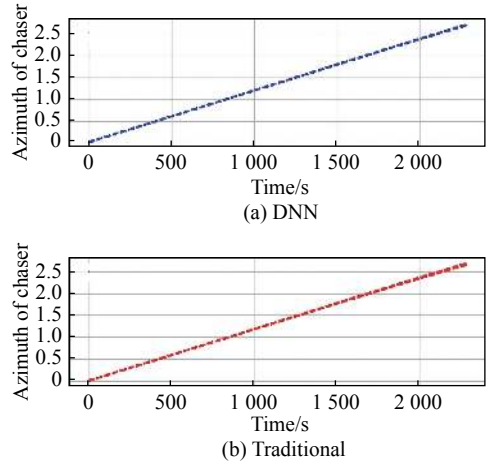

Fig. 15 Azimuth of chaser vs. time t

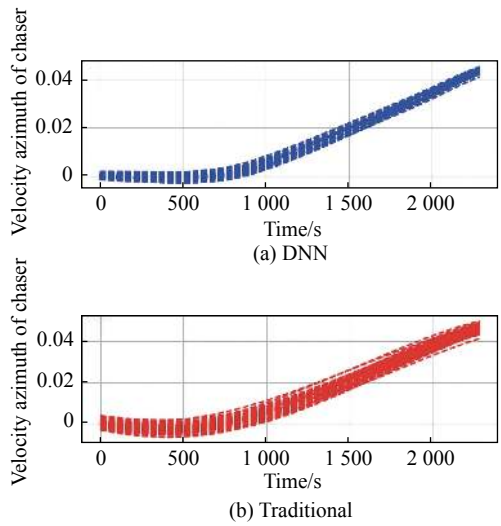

Fig. 16 Velocity azimuth of chaser vs. time t

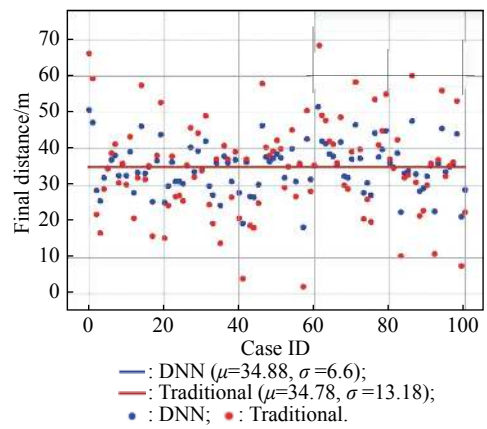

Fig. 17 Dispersion of the final distance

As shown in Fig. 13, Fig. 14, and Fig. 16, the accumulated error grows significantly as time progresses for the radius, velocity and velocity azimuth, while there is little accumulated error for the azimuth as shown in Fig. 15.

Moreover, as shown in Fig. 17, the mean value of the final distance is almost the same for the trained DNN controller and the traditional optimization method, while the standard deviation for the trained DNN controller is much smaller than the traditional optimization method. It indicates that the trained DNN is more stable than the traditional optimization method. In other words, when the noise is added, by using the trained DNN controller, the final distance between the chaser and the target will be much closer than that using the traditional optimization method, and this is better for the terminal guidance to guarantee the interception.

In conclusion, the trained DNN controller is much more efficient and stabler than the traditional optimization methods, though it can only guarantee a near optimized control. However, compared with the improvement of efficiency, roughly two orders of magnitude, the difference between the numerical results, less than one percent, is acceptable.

5. Conclusions

Concentrating on the coplanar orbital interception, this paper presents a well-designed RL algorithm. Some useful conclusions are drawn as follows:

(i) The application of the RL to the coplanar orbital interception is feasible;

(ii) By taking advantage of prior information, the convergence problem in the common RL algorithm can be successfully resolved;

(iii) The control sequence generated by the trained DNN controller is much more efficient and stabler than the traditional optimization methods. Though it is a near optimized result, the difference is acceptable.

However, there are still some issues worthy of research:

(i) The method to smooth the control sequence requires further research;

(ii) The technology of adjusting the DNN controller online is worthy of exploration;

(iii) The method to handle the convergence problem of three-dimensional orbital interception needs to be investigated.

These issues will be addressed in subsequent research.

References

[1] GARCIA E, CASBEER D W, PACHTER M. Design and analysis of state-feedback optimal strategies for the differential game of active defense. IEEE Trans. on Automatic Control, 2019, 64(2): 553–568.

[2] JIANG J X, ZENG X Y, GUZZETTI D, et al. Path planning for asteroid hopping rovers with pre-trained deep reinforcement learning architectures. Acta Astronautica, 2020, 171: 265–279.

[3] FENG Y T, WANG C Q, LI A J. Optimal trajectory tracking control based on reinforcement learning for the deployment process of space tether system. IFAC-PapersOnLine , 2020, 53(1): 679–684.

[4] WANG X, SHI P, WEN C X, et al. Design of parameter-self-tuning controller based on reinforcement learning for tracking noncooperative targets in space. IEEE Trans. on Aerospace and Electronic Systems, 2020, 56(6): 4192–4208.

[5] ZAVOLI A, FEDERICI L. Reinforcement learning for low-thrust trajectory design of interplanetary missions. arXiv: 2008.08501, 2020. https://arxiv.org/abs/2008.08501.

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

[7] LIU P F, LIU Y M, HUANG T Y, et al. Decentralized automotive radar spectrum allocation to avoid mutual interference using reinforcement learning. IEEE Trans. on Aerospace and Electronic Systems, 2021, 57(1): 190–205.

[8] SELVI E, BUEHRER R M, MARTONE A, et al. Reinforcement learning for adaptable bandwidth tracking radars. IEEE Trans. on Aerospace and Electronic Systems, 2020, 56(5): 3904–3921.

[9] XIE Z C, SUN T, KWAN T, et al. Motion control of a space manipulator using fuzzy sliding mode control with reinforcement learning. Acta Astronautica, 2020, 176: 156–172.

[10] XU K, WU F G, ZHAO J S. Model-based deep reinforcement learning with heuristic search for satellite attitude control. Industrial Robot, 2019, 46(3): 415−420.

[11] SHARMA S, CUTLER J W. Robust orbit determination and classification: a learning theoretic approach. Interplanetary Network Progress Report, 2015, 42-203(Nov.): 1−20. DOI: 10.1620/tjem.201.127.

[12] SANCHEZ C, IZZO D. Real-time optimal control via deep neural networks: study on landing problems. Journal of Guidance, Control, and Dynamics, 2018, 41(5): 1122–1135.

[13] SHEN H X, CASALINO L. Revisit of the three-dimensional orbital pursuit-evasion game. Journal of Guidance, Control, and Dynamics, 2018, 41(8): 1823–1831.

[14] GONG H R, GONG S P, LI J F. Pursuit-evasion game for satellites based on continuous thrust reachable domain. IEEE Trans. on Aerospace and Electronic Systems, 2020, 56(6): 4626–4637.

[15] LI Z Y, ZHU H, YANG Z, et al. A dimension-reduction solution of free-time differential games for spacecraft pursuit-evasion. Acta Astronautica, 2019, 163: 201–210.

[16] YE D, SHI M M, SUN Z W. Satellite proximate interception vector guidance based on differential games. Chinese Journal of Aeronautics, 2018, 31(6): 1352–1361.

[17] ZENG X, YANG L P, ZHU Y W, et al. Comparison of two optimal guidance methods for the long-distance orbital pursuit-evasion game. IEEE Trans. on Aerospace and Electronic Systems, 2021, 57(1): 521–539.

[18] YE D, SHI M M, SUN Z W. Satellite proximate pursuit-evasion game with different thrust configurations. Aerospace Science and Technology, 2020, 99: 105715.

[19] LI Z Y, ZHU H, LUO Y Z. An escape strategy in orbital pur-

suit-evasion games with incomplete information. Science China Technological Sciences, 2021, 64: 559–570.

[20] PRINCE E R, HESS J A, COBB R G, et al. Elliptical orbit proximity operations differential games. Journal of Guidance, Control, and Dynamics, 2019, 42(7): 1458–1472.

[21] SHE H, HETTEL W, LUBIN P. Directed energy interception of satellites. Advances in Space Research, 2019, 63(12): 3795–3815.

[22] MENG E T, BU X Y, WANG C H. A novel anti-interception waveform in LEO satellite system. Proc. of the IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference, 2020: 1183–1187.

[23] GUO Y J, KANG G H, JIN C D, et al. A cooperative interception strategy of small satellite swarm based on bifurcation potential field. Proc. of the IEEE CSAA Guidance, Navigation and Control Conference, 2018: 1–6.

[24] HAN H W, QIAO D, CHEN H B. Optimization of aeroassisted rendezvous and interception trajectories between non-coplanar elliptical orbits. Acta Astronautica, 2019, 163(Part B): 190–200.

[25] LI Y F, ZHANG Y S. A method of determining multiple-targets interception based on traversing points. Applied Mechanics & Materials, 2013, 437: 1081–1084.

[26] HAFER W, REED H L, TURNER J, et al. Sensitivity methods applied to orbital pursuit evasion. Journal of Guidance, Control, and Dynamics, 2015, 38(6): 1–9.

[27] CARR R W, COBB R G, PACHTER M, et al. Solution of a pursuit-evasion game using a near-optimal strategy. Journal of Guidance, Control, and Dynamics, 2018, 41(4): 841–850.

[28] SUTTON R, BARTO A. Reinforcement learning: an introduction. Massachusetts: MIT Press, 1998.

[29] CHU W M, WU S N, HE X, et al. Deep learning-based inertia tensor identification of the combined spacecraft. Proc. of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 2020, 234(7): 1356–1366.

[30] FURFARO R, SCORSOGLIO A, LINARES R, et al. Adaptive generalized ZEM-ZEV feedback guidance for planetary landing via a deep reinforcement learning approach. Acta Astronautica, 2020, 171: 156–171.

[31] GAUDET B, LINARES R, FURFARO R. Deep reinforcement learning for six degree-of-freedom planetary landing. Advances in Space Research, 2020, 65(7): 1723–1741.

[32] JIANG X Q, LI S, FURFARO R. Integrated guidance for Mars entry and powered descent using reinforcement learning and pseudospectral method. Acta Astronautica, 2019, 163: 114–129.

[33] DAS-STUART A, HOWELL K, FOLTA D. Rapid trajectory design in complex environments enabled by reinforcement learning and graph search strategies. Acta Astronautica, 2020, 171: 172–195.

[34] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning. arXiv: 1509.02971, 2019. https://arxiv.org/abs/1509.02971v2.

[35] MNIH V, BADIA A P, MIRZA M, et al. Asynchronous methods for deep reinforcement learning. Proc. of the 33rd International Conference on Machine Learning, 2016, 48: 1928–1937.

[36] SCHULMAN J, LEVINE S, MORITZ P, et al. Trust region policy optimization. arXiv: 1502.05477v5. https://arxiv.org/pdf/1502.05477.pdf.

[37] RUNARSSON T P, YAO X. Search biases in constrained evolutionary optimization. IEEE Trans. on Systems, Man and Cybernetics, Part C (Applications and Reviews), 2005, 35(2): 233–243.

[38] ANDERSON D. Algorithms for minimization without derivatives. IEEE Trans. on Automatic Control, 1974, 19(5): 632–633.

[39] AWHEDA M D, SCHWARTZ H M. A residual gradient fuzzy reinforcement learning algorithm for differential games. International Journal of Fuzzy Systems, 2017, 19(4): 1–19.

[40] KINGMA D, BA J. Adam: a method for stochastic optimization. Computer Science, 2014. https://arxiv.org/abs/1412.6980v8.

Biographies

ZENG Xin was born in 1992. He received his B.S. and M.S. degrees from the National University of Defense Technology (NUDT), Changsha, China, in 2014 and 2016, respectively. He is a Ph.D. student with the College of Aeronautics and Astronautics, NUDT. His research interests include aerospace dynamics, guidance and control, and application of artificial intelligence to the control of astronautic systems.
E-mail: xzavier0214@outlook.com

ZHU Yanwei was born in 1981. He received his B.S., M.S., and Ph.D. degrees from the National University of Defense Technology (NUDT), Changsha, China, in 2002, 2004, and 2009 respectively. He is an associate professor with the College of Aeronautics and Astronautics, NUDT. His research interests include aerospace dynamics, guidance and control, and astronautic mission planning and design.
E-mail: zywnudt@163.com

YANG Leping was born in 1964. He received his B.S. and M.S. degrees from the National University of Defense Technology (NUDT), Changsha, China, in 1984 and 1987, respectively. He is a professor with the College of Aeronautics and Astronautics, NUDT. His research interests include aerospace dynamics, guidance and control, and astronautic mission planning and design.
E-mail: ylpnudt@163.com

ZHANG Chengming was born in 1998. He received his B.S. degree from the National University of Defense Technology (NUDT), Changsha, China, in 2019. He is a student with the College of Aeronautics and Astronautics, NUDT. His research interests include aerospace dynamics, guidance and control, and application of artificial intelligence to the control of astronautic systems.
E-mail: zhchm_vincent@163.com