Journal of Systems Engineering and Electronics
Vol. 33, No. 2, April 2022, pp.438 – 446

A DNN based trajectory optimization method for intercepting non-cooperative maneuvering spacecraft

YANG Fuyunxiang, YANG Leping, ZHU Yanwei*, and ZENG Xin

College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, China

Abstract: Current successes in artificial intelligence domain have revitalized interest in neural networks and demonstrated their potential in solving spacecraft trajectory optimization problems. This paper presents a data-free deep neural network (DNN) based trajectory optimization method for intercepting non-cooperative maneuvering spacecraft, in a continuous low-thrust scenario. Firstly, the problem is formulated as a standard constrained optimization problem through differential game theory and minimax principle. Secondly, a new DNN is designed to integrate interception dynamic model into the network and involve it in the process of gradient descent, which makes the network endowed with the knowledge of physical constraints and reduces the learning burden of the network. Thus, a DNN based method is proposed, which completely eliminates the demand of training datasets and improves the generalization capacity. Finally, numerical results demonstrate the feasibility and efficiency of our proposed method.

Keywords: non-cooperative maneuvering spacecraft, neural network, differential game, trajectory optimization.

DOI: 10.23919/JSEE.2022.000044

1. Introduction

Recently, the trajectory optimization problem for intercepting non-cooperative maneuvering spacecraft has drawn increasing attention from scholars [1−3]. The problem can be considered as a two-player zero-sum pursuit-evasion game, which is essentially a two-sided optimization problem with completely conflicting goals [4]. Finding the solution of such a problem generally results in solving a high-dimensional two-point boundary value problem (TPBVP) [5].

The traditional methods for this problem usually utilize collocation, swarm optimization, evolutionary algorithm or their combination. Tuomas and Harri [6] divided the problem into two one-sided optimization problems using the direct collocation method. Due to the sensitivity to initial solution guess, it is difficult for this method to obtain global optimal solutions when the solutions are inappropriate initialized. Stupik et al. [7] proposed an approach through particle swarm optimization (PSO) to obtain open-loop optimal trajectories, but computation errors were not well handled in some specific scenarios. Li et al. [8] presented a dimension-reduction method, where differential evolution (DE) was utilized to provide good initial guess for Newton's method. Horie and Conway [9] proposed a semi-direct method, where optimal control laws of two players were obtained in different methods. One is obtained by analyzing necessary conditions and the other is solved by nonlinear programming. This method is computationally extensive and it needs appropriate initialization. Pontani and Conway [4] improved this semi-direct method by adding a genetic algorithm preprocessor which was utilized to obtain the initial guess, solving a more sophisticated three-dimensional case, but the method of genetic algorithm parameters selection is not mentioned. Sun et al. [10] developed a semi-direct control parameterization (SDCP) method and a hybrid method combined SDCP method proposed with multiple shooting method to solve scenarios in low earth orbit.

Due to the deep reform brought to practical applications by deep neural networks (DNNs), a new kind of methods has emerged. Since the neural network has good approximation capacity [11], Wu et al. [12] proposed a method directly using networks learning optimal control law from datasets generated by traditional methods to derive the optimal interception trajectory. George [13] presented a method in reinforcement learning manner, combining networks and evolutionary algorithm to solve optimal trajectories for interception and rendezvous scenarios with three different thrust models. However, this type of method usually has the following three disadvantages: Firstly, large datasets are required for training and situations beyond the training area are difficult to handle, es-

Manuscript received November 30, 2020.
*Corresponding author.
This work was supported by the National Defense Science and Technology Innovation (18-163-15-Lz-001-004-13).

pecially for supervised learning method. Secondly, data generated by orbital dynamics is subject to exact physical constraints and it is challenging to enforce these constraints on neural network models without sacrificing exact physics. Thirdly, learning burden of network is too heavy. When faced with trajectory optimization problem with complex physical constraints, networks not only need to search optimization, but also have to learn physical constraints.

Thanks to the latest achievement in network structure domain, the above difficulties can be effectively solved. It has been demonstrated that a large amount of information is captured by the structure of network instead of any learning ability [14] and this development is quickly applied in style transfer in fonts [15] and data upsampling in medical imaging [16]. This idea is also proven to be effective in complex dynamic system beyond natural images domain. Long et al. [17] designed a network structure called PDE-Net. It is used to learn partial differential equations (PDEs) and predict future state of system. Zhu et al. [18] emploied a convolutional encoder-decoder neural network approach as well as a conditional flow-based generative model for the solution of PDEs, surrogate model construction, and uncertainty quantification tasks. Hoyer et al. [19] proposed a DNN-based method which greatly improves the optimization results for the analysis of 116 structural optimization cases.

In this study, a new DNN is designed to help the generation of optimal trajectories for intercepting non-cooperative maneuvering spacecraft, where the initial states of two spacecraft and interception trajectories serve as inputs and outputs, respectively. Orbital dynamics as well as other physical constraints are integrated into the network, expressed by a secondary structure composed of multiple layers of neurons. Then the predicted terminal condition is calculated based on trajectories generated by DNN, serving as a new set of outputs. The difference between predicted terminal condition and real terminal constraints instead of training datasets is utilized to guide the training. The proposed DNN-based method successfully deepens network understanding of physical constraints, reduces the learning burden of network, improves the generalization capacity of method, and completely eliminates the demand for training data.

The rest of the paper is organized as follows: Section 2 describes the derivation of the dynamics of intercepting non-cooperative maneuvering target as well as the process of formulating a standard optimization problem. Section 3 presents our proposed data-free DNN based trajectory optimization method, followed by Section 4, where simulations are conducted to evaluate the effectiveness of the proposed method by comparing with previous methods. Finally, conclusions are summarized in Section 5.

2. Problem formulation

The relative states of two spacecraft are modeled using Hill-Clohessy-Wiltshire (HCW) equations and the problem is written as a TPBVP based on time-free differential game theory at first. Then, according to minimax principle, the relationships between co-states and control laws are derived to transform TPBVP into a standard constrained optimization problem.

2.1 Relative dynamics

Here, the interception scenario is set as two spacecraft orbiting near a circular reference orbit, as shown in Fig. 1. Target denotes the target spacecraft and pursuer denotes the interception spacecraft. Since two spacecraft both can maneuver, it is convenient to describe the relative translational motion in the Hill orbit frame whose origin is referred to a virtual spacecraft orbiting on reference orbit with no maneuvers. As illustrated in Fig. 1, the Cartesian coordinates of x, y, and z are aligned with the directions of the orbital radial, orbital velocity vector and normal vector with respect to the orbital plane, respectively.

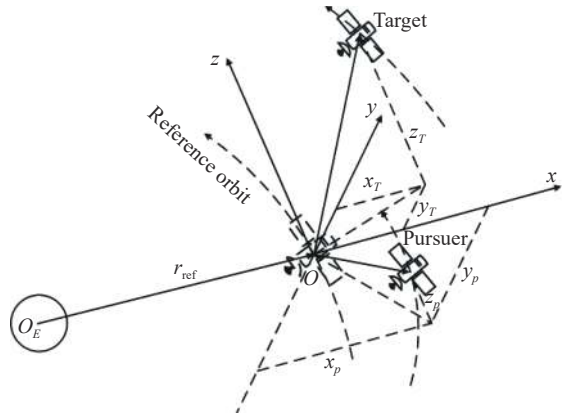

Fig. 1 Illustration of Hill frame

The linearized HCW equations of relative motion are used. The dynamics of spacecraft can be given as follows:

$$\dot{\boldsymbol{x}}_i = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{B}\boldsymbol{u}_i,\ i = T, P \tag{1}$$

where subscript i denotes the identity tag; T represents the target; P represents the pursuer; $\boldsymbol{u}_i = [u_{ix}, u_{iy}, u_{iz}]^{\mathrm{T}}$ is the acceleration vector; the spacecraft state is denoted by $\boldsymbol{x}_i = [x_i, y_i, z_i, \dot{x}_i, \dot{y}_i, \dot{z}_i]^{\mathrm{T}}$; $\omega = \sqrt{\mu/\gamma_{\text{ref}}^3}$ is average angular velocity of the reference orbit, μ is the geocentric gravitational constant and γ_{ref} is the radius of reference orbit, matrix $\boldsymbol{A}$ and $\boldsymbol{B}$ can be expressed as

$$\boldsymbol{A}=\begin{bmatrix}0&0&0&1&0&0\\0&0&0&0&1&0\\0&0&0&0&0&1\\3\omega^2&0&0&0&2\omega&0\\0&0&0&-2\omega&0&0\\0&0&-\omega^2&0&0&0\end{bmatrix}$$

$$\boldsymbol{B}=\begin{bmatrix}0&0&0\\0&0&0\\0&0&0\\1&0&0\\0&1&0\\0&0&1\end{bmatrix}$$

Define a low, continuous, and constant thrust in Fig. 2, where α and β denote the thrust pointing angles, and F_i denotes the constant thrust acceleration.

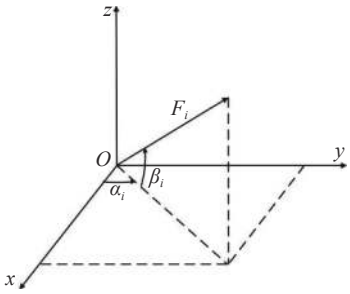

Fig. 2 Illustration of acceleration vector

The acceleration can be expressed as

$$\boldsymbol{u}_i=\begin{bmatrix}u_{ix}\\u_{iy}\\u_{iz}\end{bmatrix}=F_i\begin{bmatrix}\cos\beta_i\cos\alpha_i\\\cos\beta_i\sin\alpha_i\\\sin\beta_i\end{bmatrix},\ i=T,P \tag{2}$$

where $F_T<F_P$ needs to be satisfied [20]. The in-plane angel $\alpha_i\in[0,2\pi]$ and the out-of-plane angel $\beta_i\in[-\pi/2,\pi/2]$.

2.2 Two-player zero-sum time-free differential game

Generally, the dynamics of the two players in a zero-sum differential game [21,22] is described by

$$\dot{\boldsymbol{x}}=\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v},t),\ \boldsymbol{x}(t_0)=\boldsymbol{x}_0. \tag{3}$$

where $\boldsymbol{x}$ denotes state vector, $\boldsymbol{\mu}$ and $\boldsymbol{v}$ control sequences of the two players, respectively.

Terminal constraint function G is described by

$$G[\boldsymbol{x}(t_f),t_f]=\boldsymbol{0} \tag{4}$$

where t_f, indicating terminal time, is unknown.

Find $(\boldsymbol{u}^*(t),\boldsymbol{v}^*(t))$ for objective functional, which combines terminal performance index Φ and integral performance index L.

$$J=\Phi[\boldsymbol{x}(t_f),t_f]+\int_{t_0}^{t_f}L[\boldsymbol{x}(\tau),\boldsymbol{u},\boldsymbol{v},\tau]\mathrm{d}\tau \tag{5}$$

satisfying [23]

$$J(\boldsymbol{u}^*,\boldsymbol{v})\leqslant J(\boldsymbol{u}^*,\boldsymbol{v}^*)\leqslant J(\boldsymbol{u},\boldsymbol{v}^*). \tag{6}$$

It is equivalent to solving the following problem:

$$\min_{\boldsymbol{u}}\max_{\boldsymbol{v}}J$$
$$\text{s.t.}\begin{cases}\dot{\boldsymbol{x}}=f(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v},t),\boldsymbol{x}(t_0)=\boldsymbol{x}_0\\G[\boldsymbol{x}(t_f),t_f]=\boldsymbol{0}\end{cases}. \tag{7}$$

To solve the constrained functional extremum problem (7), augmented functional (8) is introduced through Lagrange multiplier method.

$$\begin{aligned}J_a=&\Phi[\boldsymbol{x}(t_f),t_f]+\boldsymbol{\kappa}^{\mathrm{T}}G[\boldsymbol{x}(t_f),t_f]+\\&\int_{t_0}^{t_f}\Big[L[\boldsymbol{x}(\tau),\boldsymbol{u},\boldsymbol{v},\tau]+\\&\boldsymbol{\lambda}^{\mathrm{T}}[\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v},t)-\dot{\boldsymbol{x}}]\Big]\mathrm{d}\tau\end{aligned} \tag{8}$$

where $\boldsymbol{\kappa}$ and $\boldsymbol{\lambda}$ are two sets of Lagrange multipliers.

The constrained problem is transformed into an unconstrained functional extremum problem:

$$\min_{\boldsymbol{u}}\max_{\boldsymbol{v}}J_a. \tag{9}$$

Hamiltonian function is defined as

$$\begin{aligned}H(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v},\boldsymbol{\lambda},t)=&L[\boldsymbol{x}(\tau),\boldsymbol{u},\boldsymbol{v},\tau]+\\&\boldsymbol{\lambda}^{\mathrm{T}}f(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v},t).\end{aligned} \tag{10}$$

According to minimax principle, the following equations are obtained, faced with the condition that t_f is unknown:

$$\begin{cases}\dot{\boldsymbol{x}}=\dfrac{\partial H}{\partial\boldsymbol{\lambda}}\\\dot{\boldsymbol{\lambda}}=-\dfrac{\partial H}{\partial\boldsymbol{x}}\end{cases}, \tag{11}$$

$$\begin{aligned}H(\boldsymbol{x}^*,\boldsymbol{u}^*,\boldsymbol{v}^*,\boldsymbol{\lambda},t)=&\min_{\boldsymbol{u}}\max_{\boldsymbol{v}}H(\boldsymbol{x}^*,\boldsymbol{u},\boldsymbol{v},\boldsymbol{\lambda},t)=\\&\max_{\boldsymbol{v}}\min_{\boldsymbol{u}}H(\boldsymbol{x}^*,\boldsymbol{u},\boldsymbol{v},\boldsymbol{\lambda},t),\end{aligned} \tag{12}$$

$$\begin{cases}\boldsymbol{\lambda}(t_f)=\dfrac{\partial\Phi}{\partial\boldsymbol{x}(t_f)}+\left(\dfrac{\partial G}{\partial\boldsymbol{x}(t_f)}\right)^{\mathrm{T}}\boldsymbol{\kappa}\\H(t_f)=-\dfrac{\partial\Phi}{\partial t_f}-\left(\dfrac{\partial G}{\partial t_f}\right)^{\mathrm{T}}\boldsymbol{\kappa}\end{cases}, \tag{13}$$

$$\begin{cases}\boldsymbol{x}(t_0)=\boldsymbol{x}_0\\G[\boldsymbol{x}(t_f),t_f]=\boldsymbol{0}\end{cases} \tag{14}$$

where (11) is also called the canonical equation.

2.3 Trajectory optimization model

In this work, it can be determined that $\boldsymbol{x}=[\boldsymbol{x}_T^{\mathrm{T}}, \boldsymbol{x}_P^{\mathrm{T}}]^{\mathrm{T}}$, $\boldsymbol{v}=\boldsymbol{u}_T$, and $\boldsymbol{u}=\boldsymbol{u}_P$ by combining dynamics and differential game theory. The following state equation is defined by combining dynamics and differential game theory:

$$f(\boldsymbol{x},\boldsymbol{u}_T,\boldsymbol{u}_P,t)=\bar{\boldsymbol{A}}\boldsymbol{x}+\boldsymbol{B}_T\boldsymbol{u}_T+\boldsymbol{B}_P\boldsymbol{u}_P \tag{15}$$

where

$$\boldsymbol{x}=[\boldsymbol{x}_T,\boldsymbol{x}_P]^{\mathrm{T}},$$

$$\bar{\boldsymbol{A}}=\begin{bmatrix}\boldsymbol{A} & \boldsymbol{0}_{6\times6}\\ \boldsymbol{0}_{6\times6} & \boldsymbol{A}\end{bmatrix},$$

$$\boldsymbol{B}_T=\begin{bmatrix}\boldsymbol{B}\\ \boldsymbol{0}_{6\times3}\end{bmatrix},$$

$$\boldsymbol{B}_P=\begin{bmatrix}\boldsymbol{0}_{6\times3}\\ \boldsymbol{B}\end{bmatrix}.$$

It is considered that when the interception mission ends, both spacecraft will be in the same position. Therefore, the terminal constraints are set as

$$G[\boldsymbol{x}(t_f),t_f]=\boldsymbol{C}\boldsymbol{x}(t_f)=\begin{bmatrix}x_T(t_f)-x_P(t_f)\\ y_T(t_f)-y_P(t_f)\\ z_T(t_f)-z_P(t_f)\end{bmatrix}=\boldsymbol{0}_{3\times1} \tag{16}$$

where

$$\boldsymbol{C}=\begin{bmatrix}\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} & -\boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3}\\ & & \boldsymbol{0}_{9\times12} & \end{bmatrix}.$$

Terminal time t_f,also known as interception time can be considered as primary indicator of evaluation task [8]. The target will delay the rendezvous time as long as possible, when the pursuer wants to complete the rendezvous as quickly as possible because of the characteristic of non-cooperative maneuvering target. Therefore, objective generalized functional is given as

$$\begin{cases}\Phi[\boldsymbol{x}(t_f),t_f]=0\\ L[\boldsymbol{x}(t),\boldsymbol{u}_T,\boldsymbol{u}_P,t]=1\\ J=0+\int_{t_0}^{t_f}\mathrm{d}\tau=t_f-t_0\end{cases}. \tag{17}$$

Substituting (15)−(17) into (10)−(14), a TPBVP is obtained:

$$H(\boldsymbol{x},\boldsymbol{u}_T,\boldsymbol{u}_P,\boldsymbol{\lambda},t)=1+\boldsymbol{\lambda}^{\mathrm{T}}(\bar{\boldsymbol{A}}\boldsymbol{x}+\boldsymbol{B}_T\boldsymbol{u}_T+\boldsymbol{B}_P\boldsymbol{u}_P) \tag{18}$$

$$\begin{cases}\dot{\boldsymbol{x}}=\bar{\boldsymbol{A}}\boldsymbol{x}+\boldsymbol{B}_T\boldsymbol{u}_T+\boldsymbol{B}_P\boldsymbol{u}_P\\ \dot{\boldsymbol{\lambda}}=-\bar{\boldsymbol{A}}^{\mathrm{T}}\boldsymbol{\lambda}\end{cases} \tag{19}$$

$$\begin{cases}\left.\dfrac{\partial H(\boldsymbol{x}^*,\boldsymbol{u}_P,\boldsymbol{u}_T{}^*,\boldsymbol{\lambda},t)}{\partial\boldsymbol{u}_P}\right|_{\boldsymbol{u}_P{}^*}=0\\ \left.\dfrac{\partial^2 H(\boldsymbol{x}^*,\boldsymbol{u}_P,\boldsymbol{u}_T{}^*,\boldsymbol{\lambda},t)}{\partial\boldsymbol{u}_P}\right|_{\boldsymbol{u}_P{}^*}\geqslant0\end{cases} \tag{20}$$

$$\begin{cases}\left.\dfrac{\partial H(\boldsymbol{x}^*,\boldsymbol{u}_P{}^*,\boldsymbol{u}_T,\boldsymbol{\lambda},t)}{\partial\boldsymbol{u}_T}\right|_{\boldsymbol{u}_T{}^*}=0\\ \left.\dfrac{\partial^2 H(\boldsymbol{x}^*,\boldsymbol{u}_P{}^*,\boldsymbol{u}_T,\boldsymbol{\lambda},t)}{\partial\boldsymbol{u}_T}\right|_{\boldsymbol{u}_T{}^*}\leqslant0\end{cases} \tag{21}$$

$$\boldsymbol{\lambda}(t_f)=\boldsymbol{C}^{\mathrm{T}}\boldsymbol{\kappa}=[\kappa_1,\kappa_2,\kappa_3,0,0,0,-\kappa_1,-\kappa_2,-\kappa_3,0,0,0]^{\mathrm{T}} \tag{22}$$

$$H(t_f)=0 \tag{23}$$

$$\boldsymbol{C}\boldsymbol{x}(t_f)=\boldsymbol{0}. \tag{24}$$

According to (2), (18), (20), and (21), the relationship between control vector and co-state is given

$$\begin{cases}\cos\alpha_P^*=\dfrac{-\lambda_{10}}{\cos\beta_P^*\sqrt{\lambda_{10}{}^2+\lambda_{11}{}^2+\lambda_{12}{}^2}}\\ \sin\alpha_P^*=\dfrac{-\lambda_{11}}{\cos\beta_P^*\sqrt{\lambda_{10}{}^2+\lambda_{11}{}^2+\lambda_{12}{}^2}}\\ \sin\beta_P^*=\dfrac{-\lambda_{12}}{\sqrt{\lambda_{10}{}^2+\lambda_{11}{}^2+\lambda_{12}{}^2}}\end{cases}, \tag{25}$$

$$\begin{cases}\cos\alpha_T^*=\dfrac{\lambda_4}{\cos\beta_T^*\sqrt{\lambda_4{}^2+\lambda_5{}^2+\lambda_6{}^2}}\\ \sin\alpha_T^*=\dfrac{\lambda_5}{\cos\beta_T^*\sqrt{\lambda_4{}^2+\lambda_5{}^2+\lambda_6{}^2}}\\ \sin\beta_T^*=\dfrac{\lambda_6}{\sqrt{\lambda_4{}^2+\lambda_5{}^2+\lambda_6{}^2}}\end{cases}. \tag{26}$$

As a convenience, (25) and (26) is written as

$$\begin{cases}\boldsymbol{u}_P=\boldsymbol{u}_P(\boldsymbol{\lambda})\\ \boldsymbol{u}_T=\boldsymbol{u}_T(\boldsymbol{\lambda})\end{cases}. \tag{27}$$

Substitute (27) into (18)−(24), then

$$\begin{cases} \dot{\boldsymbol{\lambda}} = -\bar{\boldsymbol{A}}^{\mathrm{T}}\boldsymbol{\lambda} \\ \boldsymbol{\lambda}(t_f) = [\kappa_1, \kappa_2, \kappa_3, 0, 0, 0, -\kappa_1, -\kappa_2, -\kappa_3, 0, 0, 0]^{\mathrm{T}} \\ \dot{\boldsymbol{x}} = \bar{\boldsymbol{A}}\boldsymbol{x} + \boldsymbol{B}_T\boldsymbol{u}_T(\boldsymbol{\lambda}) + \boldsymbol{B}_P\boldsymbol{u}_P(\boldsymbol{\lambda}) \\ \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \\ \boldsymbol{C}\boldsymbol{x}(t_f) = \boldsymbol{0} \\ H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\lambda}, t)|_{t_f} = H(\boldsymbol{\lambda}(t_f), t_f) = 0 \end{cases}. \tag{28}$$

Considering (28) only determined by four unknown parameters $t_f, \kappa_1, \kappa_2, \kappa_3$, a standard optimization problem with constraints is derived as

$$\min_{t_f, \kappa_1, \kappa_2, \kappa_3} \boldsymbol{g}(t_f, \kappa_1, \kappa_2, \kappa_3)$$
$$\text{s.t.} \begin{cases} \dot{\boldsymbol{\lambda}} = -\bar{\boldsymbol{A}}^{\mathrm{T}}\boldsymbol{\lambda} \\ \boldsymbol{\lambda}(t_f) = [\kappa_1, \kappa_2, \kappa_3, 0, 0, 0, -\kappa_1, -\kappa_2, -\kappa_3, 0, 0, 0]^{\mathrm{T}} \\ \dot{\boldsymbol{x}} = \bar{\boldsymbol{A}}\boldsymbol{x} + \boldsymbol{B}_T\boldsymbol{u}_T(\boldsymbol{\lambda}) + \boldsymbol{B}_P\boldsymbol{u}_P(\boldsymbol{\lambda}) \\ \boldsymbol{x}(t_0) = \boldsymbol{x}_0 \end{cases} \tag{29}$$

where

$$\boldsymbol{g}(t_f, \kappa_1, \kappa_2, \kappa_3) = \begin{bmatrix} |\boldsymbol{C}\boldsymbol{x}(t_f)| \\ |H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\lambda}, t)|_{t_f}| \end{bmatrix}. \tag{30}$$

3. Algorithm

In the above sections, the problem is transformed into an optimization problem with strong physical constraints. In order to make the neural network better handle physical constraints in the problem, a new DNN structure is designed to help generate optimal interception trajectories, where physics model is represented by secondary structure in network.

Fig. 3 gives an overview of the whole structure and data flow in forward propagation and gradient backward pass. The inputs are the initial states of two spacecraft $\boldsymbol{x}_0$. Both interception trajectories $\boldsymbol{x}(t)(t \in [t_0, t_f])$ and predictive value $\boldsymbol{h}$ serve as outputs, where $\boldsymbol{x}(t)(t \in [t_0, t_f])$ is the solution of the problem and $\boldsymbol{h}$ is used for network training.

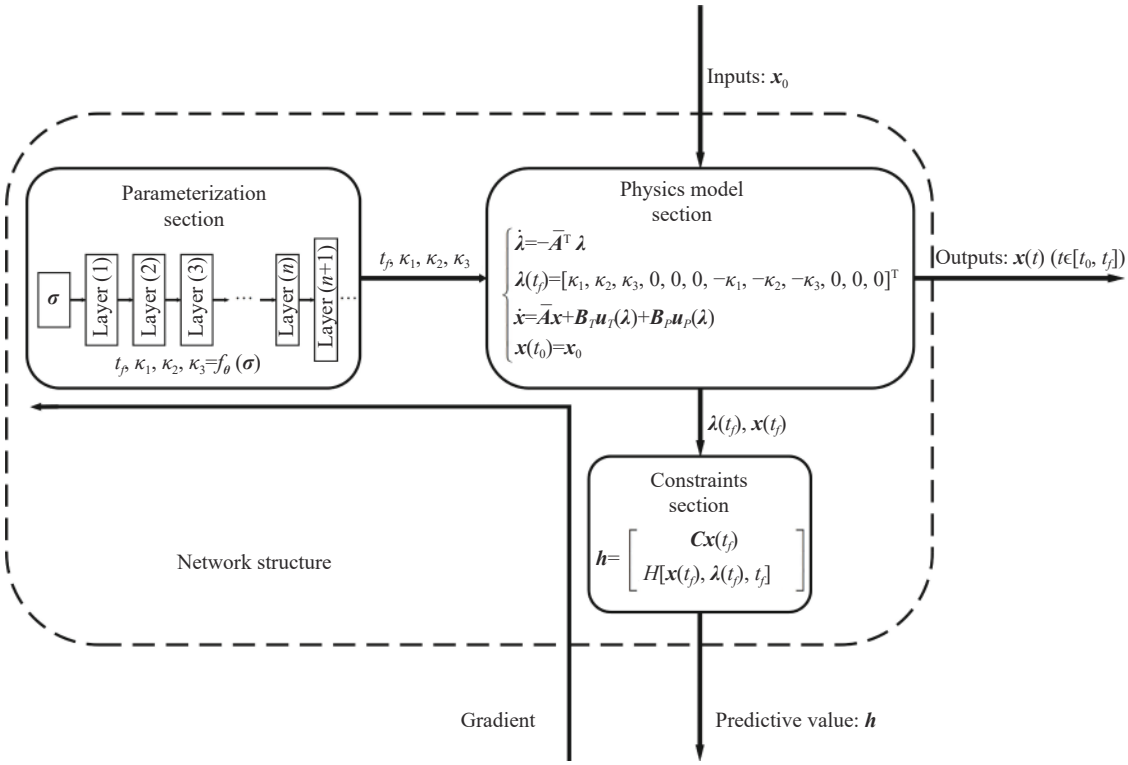

Fig. 3 Proposed network structure

There are three sections in this network, namely parameterization section, physical model section and constraints section. Parameterization section $f_{\boldsymbol{\theta}}(\boldsymbol{\sigma})$ is a secondary structure formed by several layers of neurons. $\boldsymbol{\theta}$ and $\boldsymbol{\sigma}$ represent all trainable parameters in this secondary structure. The role of this section is to generate $t_f, \kappa_1, \kappa_2, \kappa_3$ for physical model section, autonomously. Physical model section enforces dynamics on neural networks, playing the same role as canonical equation constraints. Given inputs, initial state $\boldsymbol{x}_0$, this section calcu-

lates state $\boldsymbol{x}(t)$ and co-state $\boldsymbol{\lambda}(t)$ with the assistance of generated $t_f, \kappa_1, \kappa_2, \kappa_3$. While outputting trajectory $\boldsymbol{x}(t)$ $(t \in [t_0, t_f])$, terminal value $\boldsymbol{\lambda}(t_f)$ and $\boldsymbol{x}(t_f)$ are transferred into constraints section. In constraints section, $\boldsymbol{\lambda}(t_f)$ and $\boldsymbol{x}(t_f)$ are substituted into (23) and (24) to compute predictive value $\boldsymbol{h}$, which is another output of our network, guiding the training process.

$$\boldsymbol{h} = \begin{bmatrix} \boldsymbol{C}\boldsymbol{x}(t_f) \\ H[\boldsymbol{x}(t_f), \boldsymbol{\lambda}(t_f), t_f] \end{bmatrix}. \tag{31}$$

Orbital dynamics and other physical constraints are integrated into the network, expressed by a secondary structure composed of multiple layers of neurons, so that this network is endowed with the knowledge of physical constraints. The task of the network is simplified. Since all trajectories generated by it perfectly accord with dynamics without training, network only needs to focus on searching the optimal trajectories, which also leads to the substantial reduction in training data.

After the new DNNs are determined, the optimization method can be proposed. The whole process of method is summarized as the following six steps and an overview of data flow is illustrated in Fig. 4.

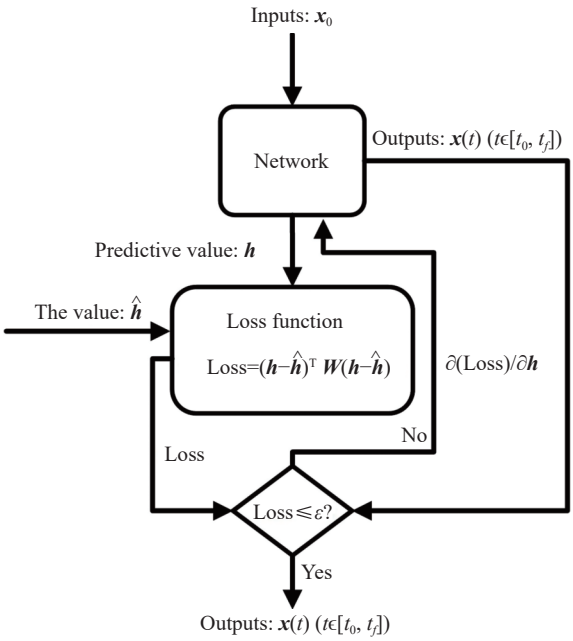

Fig. 4 Schema of proposed approach

Step 1 Initialization. Threshold ε is determined and true value $\hat{\boldsymbol{h}}$ is given, according to terminal constraints (23) and (24)

$$\hat{\boldsymbol{h}} = \boldsymbol{0}. \tag{32}$$

All trainable parameter in our network, $\boldsymbol{\theta}$ and $\boldsymbol{\sigma}$, are initialized.

Step 2 Forward propagation. Input $\boldsymbol{x}_0$ is transferred into our network. Predictive value $\boldsymbol{h}$ and output $\boldsymbol{x}(t)(t \in [t_0, t_f])$ are obtained.

Step 3 Loss computation. Difference between predicted value $\boldsymbol{h}$ and true value $\hat{\boldsymbol{h}}$ is defined as loss function

$$\text{Loss} = \frac{(\boldsymbol{h}-\hat{\boldsymbol{h}})^{\mathrm{T}}\boldsymbol{W}(\boldsymbol{h}-\hat{\boldsymbol{h}})}{\dim(\boldsymbol{h})} \tag{33}$$

where $\boldsymbol{W}$ is weight matrix.

Step 4 Judgment. Result Loss is compared with threshold ε. If Loss $> \varepsilon$, go to Step 5. If Loss $\leqslant \varepsilon$, go to Step 6.

Step 5 Gradient backward pass. Gradient $\partial(\text{Loss})/\partial\boldsymbol{h}$ is calculated and transferred into network for modifying parameters $\boldsymbol{\theta}$ and $\boldsymbol{\sigma}$. After parameters update, network regenerates predictive value $\boldsymbol{h}$ and outputs $\boldsymbol{x}(t)(t \in [t_0, t_f])$. Go to Step 3.

Step 6 Output. The optimal trajectory $\boldsymbol{x}^*(t)(t \in [t_0, t_f])$ is obtained and outputted.

In the proposed method, terminal constraints serve as labeled data for network training, so that all information required for training comes from interception trajectory optimization problem itself. By changing the training mechanism from using training datasets to using terminal constraints, this method completely eliminates the demand of training data, which helps improve generalization capacity of method. A stationary mapping is created between scenarios and optimal trajectories instead of building a time-varying mapping between states and control strategies, which reduces the training data demand and makes training process easier for network. Physical model is embedded in the network structure as part of the network so that the network is bound to the physical process, which also reduces the training difficulty. Since no training data is needed, it will not happen that the training is incomplete due to insufficient data.

4. Numerical simulation

To verify the DNN in the proposed method endowed with the knowledge of dynamics, a simulation scenario is given in the absence of training data to compare the results of our method with the results of a previous traditional method. Besides, ten cases are utilized for comparing the generalization capacity of our proposed method with a previous DNN based method. The threshold and weight matrix in the method are

$$\varepsilon = 10^{-5}, \tag{34}$$

$$\boldsymbol{W} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}. \tag{35}$$

Learning rate η is self-adaptive and its initial value is given:

$$\eta_0 = 10^{-3}. \tag{36}$$

4.1 Feasibility verification

Reference orbit is geostationary orbit. The initial states are listed in Table 1.

The thrust magnitudes of initial states can be given as

$$\begin{cases} F_T = 0.000\,4g \\ F_P = 0.000\,6g \end{cases} \tag{37}$$

where $g = 9.78$ m/s^2 denotes the magnitude of the gravitational acceleration at sea level.

Table 1 Scenario parameter setting

ID	x/km	y/km	z/km	$\dot{x}$/(m/s)	$\dot{y}$/(m/s)	$\dot{z}$/(m/s)
Target	−33.8	−50.4	6.7	2.0	4.9	0.0
Pursuer	0.0	0.0	0.0	0.0	0.0	0.0

This interception scenario is solved independently in our proposed method and method in [8]. Our method is based on network. After inputting the initial state, the network outputs the optimal trajectories directly after learning, while method in [8] is a traditional method, where the Newton's iteration method is used to find the accurate costate vector solution after obtaining an initial guess searched by differential evolution algorithm and the optimal trajectories are generated by using accurate solution. In our method, network generates the optimal trajectories after 58 epochs of learning and the pursuer intercepts the target successfully in 5 913.15 s, while the interception time is 5 912.24 s by the method in [8]. The comparison of the results is illustrated in Fig. 5−Fig. 9.

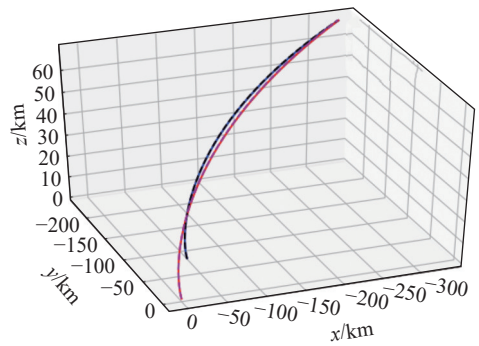

Fig. 5 Trajectories of two players in scenario

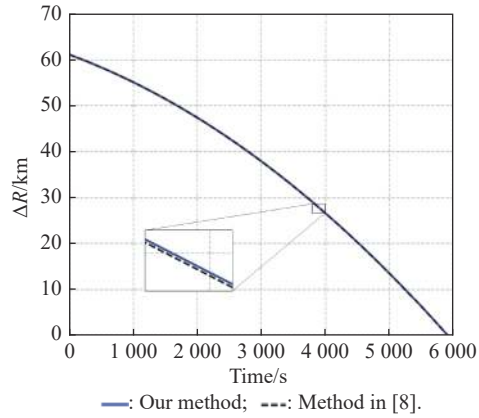

Fig. 6 Time histories of the relative distance between two spacecraft

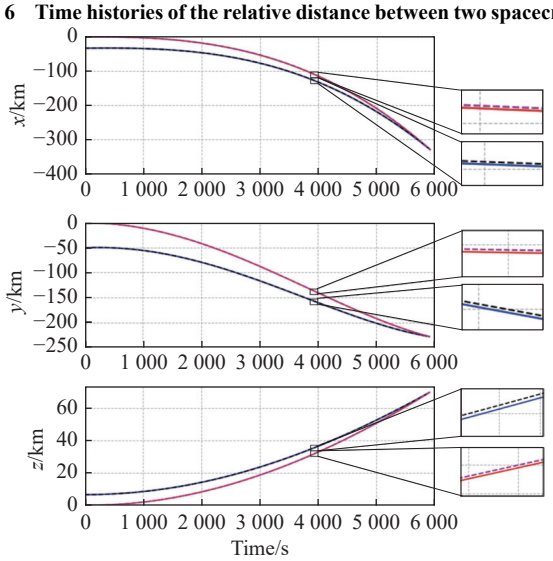

Fig. 7 Time histories of the position elements

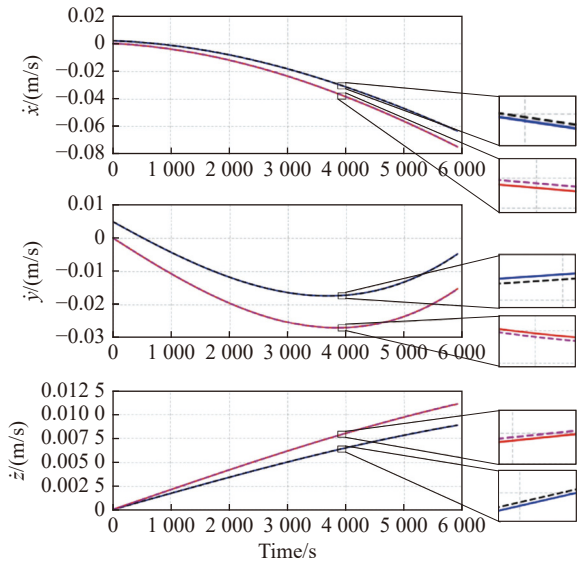

Fig. 8 Time histories of the velocity elements

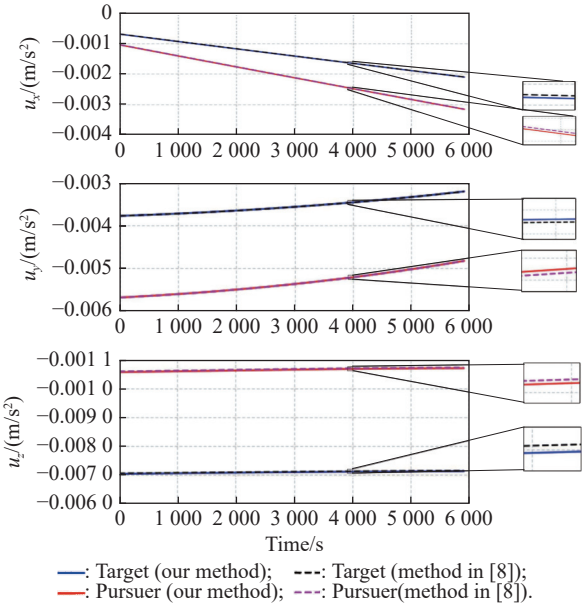

Fig. 9 Time histories of the acceleration

The result curves generated by two methods basically coincide, indicating that our method is effective. Integrating dynamics into network is a feasible method to enforce physical constraints on neural network models while maintaining accuracy. The idea of using terminal constraints as labeled data to guide network training works in the absence of training datasets.

4.2 Performance comparison

To show the efficiency and generalization capacity of the proposed method, it is further compared with the method proposed in [12]. The method in [12] is a supervised learning method, establishing a network utilizing states and control strategies as inputs and outputs, respectively. Data generated by traditional methods is used to train the network. This method attempts to establish the time sequence correspondence between states and control strategies, which is a time-vary relationship, while our method is an unsupervised learning method, generating the entire trajectory based on initial state without training data and establishing a stationary relationship between scenarios and optimal trajectories. Here, the method in [12] is denoted by Method 1, while our proposed method is denoted by Method 2, for simplicity. Reference orbit is still set as geostationary orbit and thrust magnitudes keep the same as the previous section. Then, 500 cases are selected to generate training dataset $\boldsymbol{D}$ for Method 1. The selection range is shown in Table 2. After the networks in Method 1 are trained, another ten cases are further selected for comparison. In the ten cases, Case 1−Case 7 are in $\boldsymbol{D}$, but Case 8−Case 10 are not in $\boldsymbol{D}$.

Table 2 Parameter selection range

ID	Relative distance/km	Relative velocity/(m/s)
Target	$\sqrt{x^2+y^2+z^2} \leqslant 100$	$\sqrt{v_x^2+v_y^2+v_z^2} \leqslant 10$
Pursuer	$\sqrt{x^2+y^2+z^2} \leqslant 100$	$\sqrt{v_x^2+v_y^2+v_z^2} \leqslant 10$

For different methods, the success rate is compared. Success number of times is denoted by n_s, and success rate is denoted by r_s. All the performance is summarized in Table 3.

Table 3 Performance of different methods

Method name	Dataset	n_s in Case 1−Case 7	n_s in Case 8−Case 10	r_s/%
Method 1	500	3	0	30
Method 2	–	7	3	100

The performance of Method 1 is unsatisfactory. The poor performance handling cases in training area indicates that the network neither fully understands dynamic characteristics nor learns how to optimize trajectories well. On the one hand, it can be explained that 500 training cases are obviously not enough for network training, on the other hand, it can be considered that learning burden for dual tasks is excessive. The failure in cases beyond training area demonstrates that Method 1 lacks generalization capacity. Method 2 successfully solves all ten cases with no training data, which indicates that integrating the physical constraints into the network and using terminal constraints as labeled data improve the generalization capacity of method.

5. Conclusions

Concentrating on trajectory optimization problem of intercepting non-cooperative maneuvering spacecraft, the paper presents a new designed DNN and a data-free method base on it. Some useful conclusions are drawn as follows:

(i) Integrating dynamics into neural network structure is an efficient method to reduce the learning burden of networks;

(ii) Results generated by our method successfully enforces orbital dynamics and other physical constraints on neural network models without sacrificing exact physics;

(iii) The proposed DNN based trajectory optimization method completely eliminates the demand of training data, which helps improve generalization capacity of method.

In addition, there are still some points worth further study:

(i) The performance of the proposed method will be evaluated, when the problem possesses a more complex

physical model;

(ii) It is worthy analyzing how to extend the one-to-one scenarios to the many-to-many scenarios;

(iii) How to modify the proposed method so that it can be utilized to solve scenarios that spacecraft process different thrust configurations.

References

[1] SHEN H C, LORENZO C. Revisit of the three-dimensional orbital pursuit-evasion game. Journal of Guidance, Control and Dynamics, 2018, 41(8): 1820–1828.

[2] ZHAI G, BI X Z, ZHAO H Y, et al. Non-cooperative maneuvering spacecraft tracking via a variable structure estimator. Aerospace Science and Technology, 2018, 79: 352–363.

[3] HU Q L, LIU Y Y, ZHANG Y M. Control of non-cooperative spacecraft in final phase proximity operations under input constraints. Control Engineering Practice, 2019, 87: 83–96.

[4] PONTANI M, CONWAY B A. Numerical solution of the three-dimensional orbital pursuit-evasion game. Journal of Guidance, Control and Dynamics, 2009, 32(2): 474–487.

[5] BASAR T, OLSDER G J. Dynamic noncooperative game theory. Philadelphia: Academic Press, 1999.

[6] TUOMAS R, HARRI E. Visual aircraft identification as a pursuit-evasion game. Journal of Guidance, Control and Dynamics, 2000, 23(4): 701–708.

[7] STUPIK J, PONTANI M, CONWAY B A. Optimal pursuit/evasion spacecraft trajectories in the Hill reference frame. Proc. of the AIAA/AAS Astrodynamics Specialist Conference, 2012. DOI: 10.2514/6.2012-4882.

[8] 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.

[9] HORIE K, CONWAY B A. Optimal fighter pursuit-evasion maneuvers found via two-sided optimization. Journal of Guidance, Control and Dynamics, 2006, 29(1): 105–112.

[10] SUN S T, ZHANG Q H, LOXTON R, et al. Numerical solution of a pursuit-evasion differential game involving two spacecraft in low earth orbit. Journal of Industrial and Management Optimization, 2015, 11(4): 1127–1147.

[11] SANCHEZ-SANCHEZ C, IZZO D, HENNES D. Learning the optimal state-feedback using deep networks. Proc. of the IEEE Symposium Series on Computational Intelligence Conference, 2016. DOI: 10.1109/SSCI.2016.7850105.

[12] WU Q C, LI B, LI J, et al. Solution of infinite time domain spacecraft pursuit strategy based on deep neural network. Aerospace Control, 2019, 37(6): 13–18,58. (in Chinese)

[13] GEORGE B C. Optimal and robust neural network controllers for proximal spacecraft maneuvers. Wright-Patterson Air Force Base, America: Air Force Institute of Technology, 2019.

[14] ULYANOV D, VEDALDI A, LEMPITSKY V. Deep image prior. International Journal of Computer Vision, 2020, 128(7): 1867–1888.

[15] AZADI S, FISHER M, KIM V, et al. Multi-content GAN for few-shot font style transfer. Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018: 7564–7573.

[16] DITTMER S, KLUTH T, MAASS P, et al. Regularization by architecture: a deep prior approach for inverse problems. Journal of Mathematical Imaging and Vision, 2019, 62: 456–470.

[17] LONG Z C, LU Y P, DONG B. PDE-Net 2. 0:learning PDEs from data with a numeric-symbolic hybrid deep network. Journal of Computational Physics, 2019, 399: 108–125.

[18] ZHU Y H, ZABARAS N, KOUTSOURELAKIS P S, et al. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. Journal of Computational Physics, 2019, 394: 56–81.

[19] HOYER S, SOHL-DICKSTEIN J, GREYDANUS S. Neural reparameterization improves structural optimization. https://arxiv.org/abs/1909.04240.

[20] CHOI H L, HYEOK R, TAHK M J, et al. A co-evolutionary method for pursuit-evasion games with non-zero lethal radii. Engineering Optimization, 2004, 36(1): 19–36.

[21] ISAACS R. Differential games. New York: Wiley, 1965.

[22] FRIEDMAN A. Differential games. Rhode Island: American Mathematical Society, 1974.

[23] YANG H W, LI J Y, BAOYIN H X. Low-cost transfer between asteroids with distant orbits using multiple gravity assists. Advances in Space Research, 2015, 56(5): 837–847.

Biographies

YANG Fuyunxiang was born in 1996. He received his B.S. degree from National University of Defense Technology (NUDT), Changsha, China, in 2018. He is a Ph.D. candidate with the College of Aeronautics and Astronautics, NUDT. His research interests include aerospace dynamics, guidance and control, application of artificial intelligence to the control of astronautic systems.

E-mail: yfyxgood@hotmail.com

YANG Leping was born in 1964. He received his B.S. and M.S. degrees from 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, astronautic mission planning and design.

E-mail: ylp_1964@163.com

ZHU Yanwei was born in 1981. He received his B.S., M.S., and Ph.D. degrees from 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, astronautic mission planning and design.

E-mail: zywnudt@163.com

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

E-mail: xzavier0214@outlook.com