aerospace

MDPI

Article

Reentry Flight Capability Assessment Based on Dynamics–Informed Neural Network and Piecewise Guidance

Kai Liu [1], Jili Zhang [2,*] and Xinlu Guo [1]

[1] School of Aeronautics and Astronautics, Dalian University of Technology, Dalian 116024, China
[2] Yangzhou Collaborative Innovation Research Institute, Shenyang Aircraft Design & Research Institute, Yangzhou 225000, China
* Correspondence: ahzjl0@126.com

Abstract: To improve the flexibility of the trajectory and the diversity of the drop point of the reentry vehicle, a flight capability assessment method based on a dynamics–informed neural network (DINN) is proposed. Firstly, the concept of a reachable domain is introduced to characterize the flight capability of the reentry vehicle and to estimate whether there are appropriate TAEM points in the area. Secondly, after the impact characteristic analysis, the reachable domains corresponding to different initial flight states are obtained through moderate dynamic simulations and reasonable mathematical expansion. The flight states and boundary point positions of the reachable domain are used as the training database of DINN, and the acquired DINN can realize the fast solution of reachable domains. Finally, the effectiveness of DINN in solving the reachable domain is verified using simulation. The simulation results show that DINN manifests the same accuracy as the existing solving methods and can meet the demand of determining whether the target point is located in the reachable domain. Additionally, the running time is shortened to one–800th of the existing methods, reaching the millisecond level, to support real–time assessment and decision–making. A predictor–corrector guidance algorithm with the piecewise objective function is also introduced. The simulation results illustrate that the proposed algorithm can stably guide the vehicle from the initial state points to the target points in the reachable domain.

Keywords: dynamics–informed neural network; piecewise predictor–corrector; reentry reachable domain; flight capability assessment

Citation: Liu, K.; Zhang, J.; Guo, X. Reentry Flight Capability Assessment Based on Dynamics–Informed Neural Network and Piecewise Guidance. *Aerospace* **2022**, *9*, 790. https://doi.org/10.3390/aerospace9120790

Academic Editor: Mikhail Ovchinnikov

Received: 15 October 2022
Accepted: 1 December 2022
Published: 3 December 2022

1. Introduction

Flight capability is a broad notion that can be defined by several different parameters, including the maximum deflection of control surfaces, the magnitude of the attitude angle, and the assessment of the predetermined target point reachability. It is intricate and abstract if these characteristics are utilized to define the flight capability of a reentry vehicle without any additional explanation. The maximum attitude angle and control surface deflection, in fact, can be specified as constraints during the trajectory design process. There is a "set" of reachable target points, regardless of the state that the reentry vehicles are in, that satisfies all constraints and can combine all flight capability measurements. It is also possible to refer to the set as the "reachable domain" because it consists of reachable target points that constitute an area on the "longitude–latitude" profile [1,2]. The concept of the reachable domain has already been proposed by Zhang et al. [3] under a single upper–bounded impulse for a given initial orbit. In contrast to the above abstract parameters, the reachable domain reflects their constraints while making the flight capability more visible.

Since it is clearly impossible to obtain this reachable domain by listing every target point, numerous researchers have integrated physical analysis to determine the edge points for working out the reachable domain [4–7]. The principal methods include the constant bank angle method, the profile translation design method, and the optimization method.

The constant bank angle method characterizes high stability and timeliness, thus becoming a common method in engineering. The profile translation design method translates the reentry process constraints to the H–V profile for the sake of forming a reentry corridor. The optimization method is the most accurate one but possesses low solution speed and poor stability. Ref. [6] considered the smoothness of the optimization–generated guidance commands and argued that it is poorly feasible to directly obtain the profiles through using the bank angle and the angle of attack commands optimized using the pseudospectral method, so they chose to use the second derivative as the optimized variable to produce smoother guidance commands. The reachable domain of spacecraft is also considered in Refs. [8–10]. Ref. [8] gives a reachable domain calculation model of near–rectilinear halo orbit (NR). The collection of all positions accessible for a spacecraft under a given initial orbit and fuel constraints with free time of flight is solved in Ref. [9]. Aiming to meet the requirement of collision–free asteroid landing with a given time of flight (TOF), the authors give a fast generation method of landing in the reachable domain based on section and expansion [10].

Approaches for reentry guidance have been the subject of numerous studies. A robust state feedback guidance law in real–time generated by using the indirect Legendre pseudospectral feedback method was proposed [11]. The authors in Ref. [12] came up with a guidance approach for tracking the nominal trajectory designed from the flight profile. A standard trajectory optimization method was proposed, coupled with an alternate convex–relaxed optimal control formulation [13]. There was also a classical method of tracking the nominal trajectory that put the PID controller to use [14]. This method utilized the error of the trajectory parameter as a feedback input to adjust the guidance command. Ref. [15] presented the flight corridor satisfying multiple conditions and designed the flight profile and bank reversion strategies. Moreover, in the early stages, the analytical predictor–corrector guidance method carried out some applications [16]. With the improvement of computing capacity, the numerical predictor–corrector guidance, predicting the terminal state by integrating dynamics equations, has become the focus [1,17–20]. In another approach, pigeon–inspired optimization (PIO) was utilized in Refs. [21,22] to reduce the error of the drop point in the guidance.

Artificial intelligence is considered a crucial component that has previously been employed to address challenging problems in aerospace engineering [23]. It is frequently integrated with flight trajectory generation and correction, reentry guidance, fault detection [24], and other fields. The most recent survey on machine learning techniques used for spacecraft dynamics, navigation, and control is presented in Refs. [25,26]. To compute the low–thrust trajectories, Hennes et al. [27] and Mereta et al. [28], as well as Yin et al. [29] and Xie et al. [30], utilized artificial neural networks (ANNs) and deep neural networks (DNNs), respectively. Ref. [31] demonstrated the use of neural networks for missing thrust events to correct flight trajectories autonomously. The issue of automatic path planning for a spaceship that resembled the manipulator was dealt with using a machine learning strategy in Ref. [32]. Attitude control of flight vehicles using neural networks was introduced in Refs. [33–35]. Additionally, Viavattene et al. [36] employed ANNs for real–time calculation of the cost and duration required to transfer a range of debris objects out of orbit. Ref. [37] provided a solution to the adaptive control problem of a type of noncanonical neural network nonlinear system with unknown input dead zones. The application of neural network theory on vehicle reentry guidance was discussed in Ref. [38], which proposed a reentry guidance method based on the generalized regression neural network (GRNN).

According to the aforementioned works, machine learning was typically used for flight trajectories and attitude control of flight vehicles and spacecraft. Compared with previous work that pays more attention to the reachable domain of spacecraft, this research focuses more on reentry vehicles. To give reentry vehicles preliminary online decision making capability, more sophisticated and more intelligent algorithms are provided. These algorithms take into account the use of machine learning in the reachable domain calculation of a reentry vehicle as well as the current development trend of more intelligent and

autonomous vehicles. In some circumstances, there may be multiple terminal targets for the reentry vehicle. The attainable points can be determined based on the reentry flight states at this time. Typically, the early phases of reentry are characterized by extremely high flight speeds, thus the evaluation outcome will be worthless if it takes a long time to obtain the calculated area. As a result, the time consumption should be taken into account to prevent missing the opportunity for decision making during development of the assessment process. To measure reentry capabilities more intelligently and quickly, a dynamic–informed neural network (DINN) is therefore presented based on a priori dynamic knowledge. DINN is an optimized method that originated from the comprehension of the BPNN databases. It can assess the reachability of the terminal target point, namely the terminal area energy management (TAEM) point, in real time based on the flight states. After confirming that the TAEM point is reachable, a predictor–corrector guidance algorithm is developed based on the piecewise objective function, which has a higher guiding accuracy than the conventional predictor–corrector technique.

2. Dynamic Modeling

2.1. Reentry Constraints

2.1.1. Path Constraints

Typical reentry trajectory constraints mainly include the heating rate constraint, the dynamic pressure constraint, and the overload constraint, which are given by Equations (1)–(3):

$$\dot{Q} = k_Q \rho^{0.5} (V_c V)^{3.15} \leq \dot{Q}_{\max} \tag{1}$$

$$q = \frac{1}{2} \rho (V_c V)^2 \leq q_{\max} \tag{2}$$

$$n = \sqrt{L^2 + D^2} \leq n_{\max} \tag{3}$$

where $\dot{Q}$ is the heating rate; q is dynamic pressure; n is overload; $\dot{Q}_{\max}$, $q_{\max}$, and $n_{\max}$ are the corresponding constraints; k_Q is a constant associated with the reentry vehicle; $V_c = \sqrt{g_0 R_e}$ is a dimensionless coefficient; R_e is the radius of the Earth; $g_0 = 9.8067$ m/s^2; L is lift; and D is drag.

2.1.2. Terminal Constraints

When the reentry vehicles arrive at the terminal point, they should satisfy the altitude, velocity, and range constraints, which are given as follows:

$$r_f = r_{TAEM} \tag{4}$$

$$V_f = V_{TAEM} \tag{5}$$

$$S \leq S_f \tag{6}$$

Altitude and velocity can also be uniformly represented by energy, as indicated in Equation (7).

$$e_{TAEM} = \frac{1}{r_{TAEM}} - \frac{V_{TAEM}^2}{2} \tag{7}$$

Therefore, the terminal constraints could also be described by energy. Equations (4)–(6) can be equivalent to Equation (8):

$$\begin{cases} e_f = e_{TAEM} \\ S \leq S_f \end{cases} \tag{8}$$

where e_f is the terminal energy; S is the residual range; e_{TAEM} and S_f are the corresponding terminal constraints.

The magnitude of the bank angle must satisfy the following formula:

$$|\sigma| \leq |\sigma_{\max}| \tag{9}$$

The $|\sigma_{\max}|$ is related to the constraints of the reentry process, and the specific solution method will be discussed in the following section.

2.1.3. Constraint Transformation and Reentry Corridor

Throughout the reentry process, the vehicle must adhere to all constraints. However, assessing whether all constraints can be satisfied is difficult due to the greater computational complexity. In fact, all path constraints can be restrained by the magnitude of the bank angle.

As shown in Figure 1, the upper bound of the corridor on the H–V profile is determined by quasi–equilibrium glide condition (QEGC), while the lower bound is given by the maximum values of dynamic pressure, overload, and heating rate constraints, as indicated in Equation (10).

$$\begin{aligned} H_{up} &= H_{QEGC} \\ H_{down} &= \max(H_q, H_n, H_q) \end{aligned} \tag{10}$$

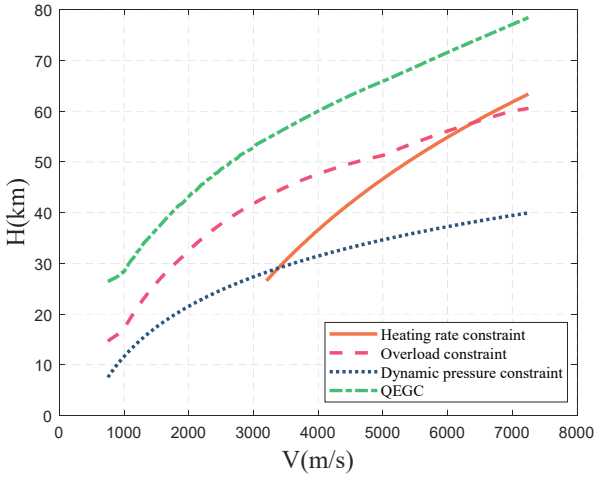


Figure 1. Reentry corridor under constraints.

In the glide phase, the vehicles should satisfy QEGC. The magnitude of bank angle σ might be calculated using altitude and velocity through Equation (11).

$$\frac{L\cos\sigma}{m} + \frac{V^2}{r} - \frac{\mu}{r^2} = 0 \tag{11}$$

On the boundary H_{up}, the matching σ can be discovered if r and V are known. The acquired H_{up} allows for the evaluation of the related geocentric distance r_{up} and the subsequent lower bound solution $|\sigma|_{\min}$ of the bank angle. The upper bound $|\sigma|_{\max}$ of the bank angle can also be solved by obtaining H_{down} and r_{down} in accordance with the relationship between the other constraints and the bank angle. The solution of the bounded bank angle value is shown as follows:

$$\begin{aligned} |\sigma|_{\max}(V) &= |\sigma|(r_{down}(V), V) \\ |\sigma|_{\min}(V) &= |\sigma|_{QEGC}(r_{up}(V), V) \end{aligned} \tag{12}$$

Therefore, the guidance command of σ needs to satisfy:

$$|\sigma(V)| = \begin{cases} |\sigma|_{\min}(V) & \sigma_i < |\sigma|_{\min}(V) \\ \sigma_i & |\sigma|_{\min}(V) < \sigma_i < |\sigma|_{\max}(V) \\ |\sigma|_{\max}(V) & \sigma_i > |\sigma|_{\max}(V) \end{cases} \tag{13}$$

where σ_i is the magnitude of the bank angle given by the guidance algorithm. The value of $|\sigma|_{\min}$ should not be too low because $|\sigma|$ controls the capacity of the vehicle to reorient during reentry.

The other guidance command of α is given by a predesigned α-V profile described in Equation (14).

$$\alpha = \begin{cases} \alpha_{\max} & V_1 \le V \le V_0 \\ \frac{\alpha_{\max}-\alpha_{\min}}{V_2-V_1}(V-V_1)+\alpha_{\max} & V_2 \le V < V_1 \\ \alpha_{\min} & V_t \le V < V_2 \end{cases} \tag{14}$$

2.2. Dynamic Equations

During the reentry process, the vehicle is powered off and only affected by gravity and aerodynamic forces. The dynamic model of the reentry vehicle is established in the trajectory coordinate system with the bank to turn (BTT) mode. With a spherical Earth and Earth rotation in mind, Equation (15) gives the three degrees of freedom point–mass dynamic Equation [39],

$$\begin{cases} \dot{V} = -\frac{D}{m} - g\sin\gamma + \omega_e^2 r(\cos^2\phi\sin\gamma - \cos\phi\sin\phi\cos\psi\cos\gamma) \\ \dot{\gamma} = \frac{1}{V}\left(\frac{L\cos\sigma}{m} - g\cos\gamma + \frac{V^2\cos\gamma}{r}\right) + 2\omega_e\cos\phi\sin\psi + \\ \qquad \frac{\omega_e^2 r\left(\cos^2\phi\cos\gamma - \cos\phi\sin\phi\cos\psi\sin\gamma\right)}{V} \\ \dot{\psi} = \frac{1}{V}\left(\frac{L\sin\sigma}{m\cos\gamma} + \frac{V^2\cos\gamma\tan\phi\sin\psi}{r}\right) - \frac{2\omega_e(\cos\phi\cos\psi\sin\gamma - \sin\phi\cos\gamma)}{\cos\gamma} \\ \qquad + \frac{\omega_e^2 r\cos\phi\sin\phi\sin\psi}{V\cos\gamma} \\ \dot{h} = V\sin\gamma \\ \dot{\lambda} = \frac{V\cos\gamma\sin\psi}{r\cos\phi} \\ \dot{\phi} = \frac{V\cos\gamma\cos\psi}{r} \end{cases} \tag{15}$$

where V is the flight speed; γ is the path angle; ψ is the course angle; h is the altitude; λ and ϕ denote the longitude and latitude, which correspond to the subpoint of the vehicle on the surface of the Earth, respectively; g is the gravitational acceleration, and g_0 is the gravitational acceleration at sea level; m is the mass of the vehicle; ω_e represents the angular velocity of the Earth rotation; σ is the bank angle; L and D are lift and drag.

3. Dynamic–Informed Neural Network (DINN) Reachable Domain Generator

Achieving an effective and real–time flight capability assessment during the reentry flight is a prerequisite for autonomous decision making. An efficient flight capability assessment can predict whether a feasible reentry trajectory exists between the reentry point and the TAEM point in advance. Furthermore, its promptness is also essential because, under the conditions of tremendous speed and rapidly changing position of the reentry vehicle, it will be valueless to make decisions if it cannot be completed in a short period. The lack of instantaneity in the present techniques restricts their use. The absence of immediacy can be admirably addressed by a neural network, which is called DINN in this paper, built on a priori dynamic knowledge. DINN is an optimized method with a small amount of data. The computation of the reachable domain can be resolved immediately with the use of DINN. It satisfies the requirements of validity and immediacy and performs close to existing methods as well.

Figure 2 illustrates the construction of DINN training and the calculation of the reachable domain in this paper.

This section introduces a DINN–based reachable domain generator to construct a fast computational mapping from flight states to the reachable domain.

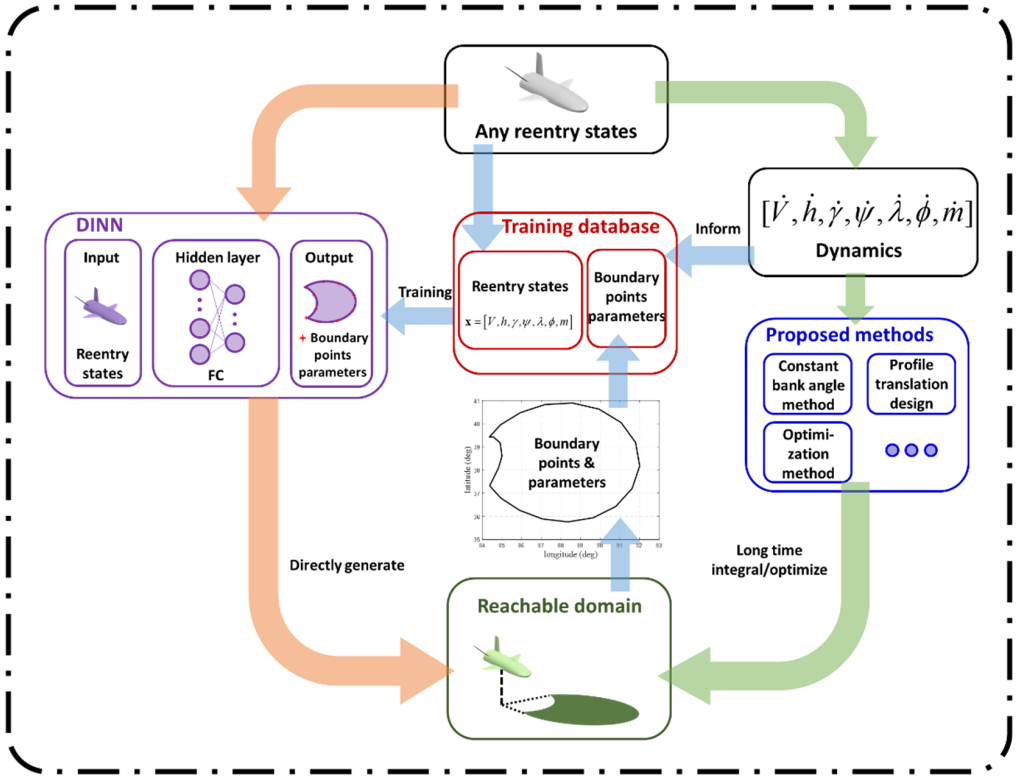


Figure 2. A framework for solving the reachable domain based on DINN.

3.1. Generation of DINN Training Data

To perform a real–time flight capability assessment procedure, the fundamental idea behind DINN is to employ a neural network to extract and reproduce the mapping relationship between flight states and reachable domains. The input and output of DINN, whose training data are derived from the dynamic simulation, are the flight state parameters and the reachable domain boundary points of the reentry vehicle, respectively. Several boundary points that link consecutively to form closed areas serve to define the reachable domain. To simplify the network structure and improve DINN efficiency of the online operation, the impact factors of the reachable domain should be first screened and analyzed through a priori dynamic knowledge.

The boundary points of the reachable domain are affected by seven flight states, which are velocity V, altitude h, longitude λ, latitude ϕ, path angle γ, course angle ψ, and mass m. The state vector composed of them is

$$\mathbf{x} = [V, h, \gamma, \psi, \lambda, \phi, m] \tag{16}$$

The reachable domain corresponding to this initial state is obtained via dynamic simulation and can be characterized in the form of boundary points that

$$\mathbf{b} = [\lambda_1, \phi_1, \lambda_2, \phi_2, \ldots, \lambda_n, \phi_n] \tag{17}$$

where n could describe the number of boundary points of the reachable domain.

DINN implements the rapid–solving **b** process based on **x**. **x** and **b**, respectively, constitute the input and output information of DINN, so that

$$\mathbf{y}=[\mathbf{x},\mathbf{b}] \tag{18}$$

can be used as one set of training data for DINN.

DINN training requires a large amount of training data. The vector **x** in training data is customized. The setting of that first ensures the convergence accuracy of DINN, and

next reduces the number of $\mathbf{x}$. Because it is necessary to calculate the corresponding $\mathbf{b}_i$ by dynamic simulation for each $\mathbf{x}_i$, and the consumption of computational time is proportional to the number of $\mathbf{x}$, $\mathbf{x}_i$ and $\mathbf{b}_i$ actually constitute the training data $\mathbf{y}_i$ which contain the dynamic information of the reentry vehicle.

3.1.1. Vector $\mathbf{x}$ and State–Matrix $\mathbf{X}$ Composition

The influence strength and linearity of these seven variates on the reachable domain are shown in Figure 3.

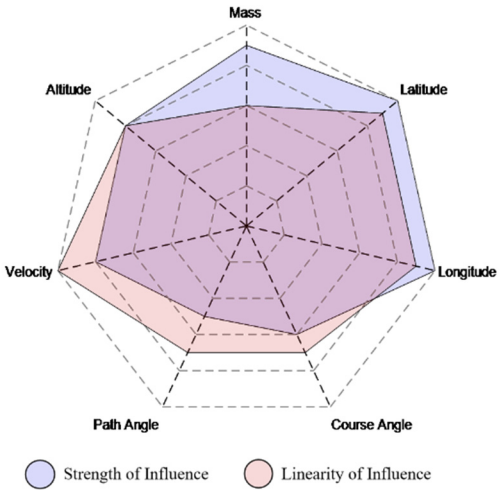


Figure 3. Influence strength and linearity of state variates on the reachable domain.

To build a training database for DINN based on dynamic simulation, a lot of time and computational resources are needed. The generated data cover most of the possible states in the reentry process. For example, the reentry vehicle is preset in an ideal flight state:

$$\mathbf{x}_0 = [V_0, h_0, \gamma_0, \psi_0, \lambda_0, \phi_0, m_0] \tag{19}$$

There may be large uncertainties between the actual and the ideal flight state. To obtain the reachable domain in any actual flight state, the training data need to cover all above states and consider the uncertainties shown in Table 1.

Table 1. Uncertainty of flight state.

States	Uncertain Interval
Velocity V_0	$\pm\Delta V_{0\max}$
Altitude h_0	$\pm\Delta h_{0\max}$
Path angle γ_0	$\pm\Delta \gamma_{0\max}$
Course angle ψ_0	$\pm\Delta \psi_{0\max}$
Longitude λ_0	$\pm\Delta \lambda_{0\max}$
Latitude ϕ_0	$\pm\Delta \phi_{0\max}$
Mass m_0	$\pm\Delta m_{0\max}$

The possible states of velocity v_p, altitude h_p, path angle λ_p, course angle ψ_p, longitude γ_p, latitude ϕ_p, and mass m_p are shown in Equation (20), which should satisfy Equation (21).

$$\begin{cases} v_p = [V_0 - k_1\Delta V_0, V_0 - (k_1-1)\Delta V_0, \ldots, V_0 - \Delta V_0, V_0, V_0 + \Delta V_0, \ldots, V_0 + (k_1-1)\Delta V_0, V_0 + k_1\Delta V_0] \\ h_p = [h_0 - k_2\Delta h_0, h_0 - (k_2-1)\Delta h_0, \ldots, h_0 - \Delta h_0, h_0, h_0 + \Delta h_0, \ldots, h_0 + (k_2-1)\Delta h_0, h_0 + k_2\Delta h_0] \\ \lambda_p = [\lambda_0 - k_5\Delta\lambda_0, \lambda_0 - (k_5-1)\Delta\lambda_0, \ldots, \lambda_0 - \Delta\lambda_0, \lambda_0, \lambda_0 + \Delta\lambda_0, \ldots, \lambda_0 + (k_5-1)\Delta\lambda_0, \lambda_0 + k_5\Delta\lambda_0] \\ \psi_p = [\psi_0 - k_4\Delta\psi_0, \psi_0 - (k_4-1)\Delta\psi_0, \ldots, \psi_0 - \Delta\psi_0, \psi_0, \psi_0 + \Delta\psi_0, \ldots, \psi_0 + (k_4-1)\Delta\psi_0, \psi_0 + k_4\Delta\psi_0] \\ \gamma_p = [\gamma_0 - k_3\Delta\gamma_0, \gamma_0 - (k_3-1)\Delta\gamma_0, \ldots, \gamma_0 - \Delta\gamma_0, \gamma_0, \gamma_0 + \Delta\gamma_0, \ldots, \gamma_0 + (k_3-1)\Delta\gamma_0, \gamma_0 + k_3\Delta\gamma_0] \\ \phi_p = [\phi_0 - k_6\Delta\phi_0, \phi_0 - (k_6-1)\Delta\phi_0, \ldots, \phi_0 - \Delta\phi_0, \phi_0, \phi_0 + \Delta\phi_0, \ldots, \phi_0 + (k_6-1)\Delta\phi_0, \phi_0 + k_6\Delta\phi_0] \\ m_p = [m_0 - k_7\Delta m_0, m_0 - (k_7-1)\Delta m_0, \ldots, m_0 - \Delta m_0, m_0, m_0 + \Delta m_0, \ldots, m_0 + (k_7-1)\Delta m_0, m_0 + k_7\Delta m_0] \end{cases} \tag{20}$$

$$k_1\Delta V_0 = \Delta V_{0\max}, k_2\Delta h_0 = \Delta h_{0\max}, k_3\Delta\gamma_0 = \Delta\gamma_{0\max}, k_4\Delta\psi_0 = \Delta\psi_{0\max}, k_5\Delta\lambda_0 = \Delta\lambda_{0\max}, k_6\Delta\phi_0 = \Delta\phi_{0\max}, k_7\Delta m_0 = \Delta m_{0\max} \tag{21}$$

The final state–matrix **X** consisting of all possible states is illustrated in Equation (22).

$$\mathbf{X} = \begin{bmatrix} v_p(1) & h_p(1) & \gamma_p(1) & \psi_p(1) & \lambda_p(1) & \phi_p(1) & m_p(1) \\ v_p(2) & h_p(1) & \gamma_p(1) & \psi_p(1) & \lambda_p(1) & \phi_p(1) & m_p(1) \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ v_p(2k_1+1) & h_p(2k_2+1) & \gamma_p(2k_3+1) & \psi_p(2k_4+1) & \lambda_p(2k_5+1) & \phi_p(2k_6+1) & m_p(2k_7+1) \end{bmatrix} \tag{22}$$

Apparently, k_1 to k_7 decide the size of the final state–matrix **X**, which has 7 columns (corresponding to the flight states), and the rows of that satisfy:

$$\mathrm{row}(\mathbf{X}) = \prod_{i=1}^{7}(2k_i + 1) \tag{23}$$

3.1.2. Vector **b** and Domain–Matrix **B** Composition

The vector **b** characterizes the reachable domain of the reentry vehicle based on the state of vector **x**. Numerous approaches, such as the optimization method and constant bank angle method, are in use to solve **b**. Since the process of solving the vector **b** is not the main subject of this paper, the constant bank angle method with sufficient performance is adopted to realize the solution of **b** based on **x**.

The constant bank angle method is simple and effective in solving the reachable domain. It starts from any state of the reentry process, uses a constant bank angle as a command, and generates a trajectory by integrating the dynamic formula until the terminal constraints are satisfied. By selecting different constant command values at suitable intervals between $[-|\sigma_{\max}|, |\sigma_{\max}|]$, the corresponding trajectories can be generated. The reachable domain of the reentry vehicle in any initial state, shown in Figure 4, can be formed by connecting all endpoints of the obtained trajectories in the longitude–latitude profile.

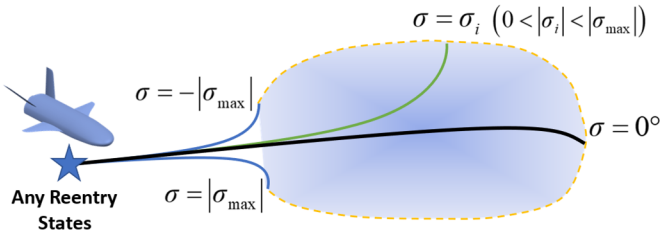


Figure 4. Reachable domain solved using constant bank angle method in the longitude–latitude profile.

For any state $\mathbf{x} = [V, h, \gamma, \psi, \lambda, \phi, m]$, its corresponding domain boundary $\mathbf{b} = [\lambda_1, \phi_1, \lambda_2, \phi_2, \ldots, \lambda_n, \phi_n]$ can be obtained through the dynamic simulation based on the constant bank angle method. Therefore, the domain–matrix **B** calculated using the state–matrix **X** can be expressed as Equation (24).

$$\mathbf{B} = \begin{bmatrix} \lambda_1(1) & \phi_1(1) & \lambda_2(1) & \phi_2(1) & \dots & \lambda_n(1) & \phi_n(1) \\ \lambda_1(2) & \phi_1(2) & \lambda_2(2) & \phi_2(2) & \dots & \lambda_n(2) & \phi_n(2) \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \lambda_1[\mathrm{row}(\mathbf{X})] & \phi_1[\mathrm{row}(\mathbf{X})] & \lambda_2[\mathrm{row}(\mathbf{X})] & \phi_2[\mathrm{row}(\mathbf{X})] & \dots & \lambda_n[\mathrm{row}(\mathbf{X})] & \phi_n[\mathrm{row}(\mathbf{X})] \end{bmatrix} \tag{24}$$

3.1.3. Impact Characteristics between Variates and Reachable Domain

The number of k_1 to k_7 determines the size of the training database and the computational performance of DINN generated by this database. Increasing the number of k_1 to k_7 is undesirable and will make the data generation time explode. When training a perfect DINN, it is necessary to guarantee its performance and to control the size of the training database. The focus is making the acquiring time manageable. As a result, the impact characteristics of each state in **x** on **b** can be analyzed to fulfill the above requests.

To avoid increasing the database size and improving the efficiency of data generation for a single object, the input and output of the training database can be directly expanded linearly because of some impact factors with high linearities, such as longitude λ and latitude ϕ.

For example, if all state variates are expanded without considering the linear characteristics as the back–propagation neural network (BPNN) is performed, the values of k_1 to k_7 are shown in Table 2.

Table 2. The values of k_1 to k_7 after expanding.

k_1	k_2	k_3	k_4	k_5	k_6	k_7
1	1	2	1	1	1	1

In the above condition, each ideal states $\mathbf{x}_0$ shown in Equation (19) can be expanded to a matrix $\mathbf{X}_1$ possessing 3645 rows through calculating with Equation (23). The expanded $\mathbf{X}_1$ is shown in Equation (25).

$$\mathbf{X}_1 = \begin{bmatrix} v_p(1) & h_p(1) & \gamma_p(1) & \psi_p(1) & \lambda_p(1) & \phi_p(1) & m_p(1) \\ v_p(2) & h_p(1) & \gamma_p(1) & \psi_p(1) & \lambda_p(1) & \phi_p(1) & m_p(1) \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ v_p(2k_1+1) & h_p(2k_2+1) & \gamma_p(2k_3+1) & \psi_p(2k_4+1) & \lambda_p(2k_5+1) & \phi_p(2k_6+1) & m_p(2k_7+1) \end{bmatrix} \tag{25}$$

That is, to generate the training data, the corresponding dynamic simulation calculations that equal the row number of $\mathbf{X}_1$, need to be performed.

To reduce the calculational burden of dynamic simulation without changing the size of the training data, the variates that have stronger linearity can be expanded by pure mathematical computation based on a priori dynamic knowledge. This approach is utilized in DINN. If all state variates are expanded with the linear characteristics considered, the values of k_1 to k_7 are shown in Table 3.

Table 3. The values of k_1 to k_7 after expanding with the consideration of linear characteristics.

k_1	k_2	k_3	k_4	k_5	k_6	k_7
1	1	2	1	0	0	1

In the above condition, each ideal state $\mathbf{x}_0$ shown in Equation (19) can be expanded to a matrix $\mathbf{X}_1$ with 405 rows through calculating with Equation (23). The calculations are only 1/9 of the BPNN. The expanded $\mathbf{X}_2$ is shown in Equation (26).

$$\mathbf{X}_2 = \begin{bmatrix} v_p(1) & h_p(1) & \gamma_p(1) & \psi_p(1) & \lambda_p(1) & \phi_p(1) & m_p(1) \\ v_p(2) & h_p(1) & \gamma_p(1) & \psi_p(1) & \lambda_p(1) & \phi_p(1) & m_p(1) \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ v_p(2k_1+1) & h_p(2k_2+1) & \gamma_p(2k_3+1) & \psi_p(2k_4+1) & \lambda_p(1) & \phi_p(1) & m_p(2k_7+1) \end{bmatrix} \tag{26}$$

To make the number of the state–matrices $\mathbf{X}_1$ and $\mathbf{X}_2$ rows consistent, $\mathbf{X}_2$ need to be further expanded using a pure mathematical computation. Define $\mathbf{X}_2^i$ as the i–th row of $\mathbf{X}_2$. Each row of $\mathbf{X}_2$ should be expanded to 9 rows to make $\mathbf{X}_1$ and $\mathbf{X}_2$ have the same size. Supposing the expanded $\mathbf{X}_2^i$ to be $\widetilde{\mathbf{X}}_2^i$, Equation (27) is then given:

$$\widetilde{\mathbf{X}}_2^i = \begin{bmatrix} \mathbf{X}_2^i + (0,0,0,0,-\Delta\lambda,-\Delta\phi,0) \\ \mathbf{X}_2^i + (0,0,0,0,-\Delta\lambda,0,0) \\ \mathbf{X}_2^i + (0,0,0,0,-\Delta\lambda,\Delta\phi,0) \\ \mathbf{X}_2^i + (0,0,0,0,0,-\Delta\phi,0) \\ \mathbf{X}_2^i \\ \mathbf{X}_2^i + (0,0,0,0,0,\Delta\phi,0) \\ \mathbf{X}_2^i + (0,0,0,0,\Delta\lambda,-\Delta\phi,0) \\ \mathbf{X}_2^i + (0,0,0,0,\Delta\lambda,-0,0) \\ \mathbf{X}_2^i + (0,0,0,0,\Delta\lambda,\Delta\phi,0) \end{bmatrix} \tag{27}$$

Correspondingly, the expanded domain–matrix $\widetilde{\mathbf{B}}_2^i$ is:

$$\widetilde{\mathbf{B}}_2^i = \begin{bmatrix} \mathbf{B}_2^i + (-\Delta\lambda,-\Delta\phi,\ldots,-\Delta\lambda,-\Delta\phi) \\ \mathbf{B}_2^i + (-\Delta\lambda,0,\ldots,-\Delta\lambda,0) \\ \mathbf{B}_2^i + (-\Delta\lambda,+\Delta\phi,\ldots,-\Delta\lambda,+\Delta\phi) \\ \mathbf{B}_2^i + (0,-\Delta\phi,\ldots,0,-\Delta\phi) \\ \mathbf{B}_2^i \\ \mathbf{B}_2^i + (0,+\Delta\phi,\ldots,0,+\Delta\phi) \\ \mathbf{B}_2^i + (\Delta\lambda,-\Delta\phi,\ldots,\Delta\lambda,-\Delta\phi) \\ \mathbf{B}_2^i + (\Delta\lambda,0,\ldots,\Delta\lambda,0) \\ \mathbf{B}_2^i + (\Delta\lambda,+\Delta\phi,\ldots,\Delta\lambda,+\Delta\phi) \end{bmatrix} \tag{28}$$

The expanded state–matrix $\widetilde{\mathbf{X}}_2$ and the domain–matrix $\widetilde{\mathbf{B}}_2$ are denoted as

$$\begin{cases} \widetilde{\mathbf{X}}_2 = \left[\mathbf{X}_2^1, \mathbf{X}_2^2, \ldots, \mathbf{X}_2^{\mathrm{row}(\mathbf{X}_2)}\right] \\ \widetilde{\mathbf{B}}_2 = \left[\mathbf{B}_2^1, \mathbf{B}_2^2, \ldots, \mathbf{B}_2^{\mathrm{row}(\mathbf{B}_2)}\right] \end{cases} \tag{29}$$

At present, the size of $\widetilde{\mathbf{X}}_2$ and $\widetilde{\mathbf{B}}_2$ in DINN is the same as that of $\mathbf{X}_1$ and $\mathbf{B}_1$ in BPNN, which means that the size of their database is the same for network training. Meanwhile, the computational time of the DINN training database is one–ninth that of BPNN.

3.2. The Construction Design of DINN

Based on the discussion in Section 3.1, the length of vector **x** and vector **b** match the input and output layer dimensions of DINN. The length of vector **x** is 7, which corresponds to velocity V, altitude h, longitude λ, latitude ϕ, path angle γ, course angle ψ, and mass m, forming a seven–dimensional input layer. N_2 is determined by the boundary points, namely the double of that, because each boundary point contains longitude λ_i and latitude ϕ_i. N_1 is determined based on N_2.

The designed structure of the DINN is shown in Table 4.

Table 4. Main parameters of DINN.

Label	Input Layer	Hidden Layer	Output Layer
Connection mode to next layer	Fully connected	Fully connected	–
Number of neurons	7	N_1	N_2
Activation function	–	Sigmoid	linear

4. Piecewise Predictor–Corrector Guidance for Reachable Target Points

To enhance the accuracy of the guidance, a predictor–corrector guidance algorithm using a piecewise objective function is presented in this section.

4.1. Longitudinal Guidance Logic

4.1.1. Range Prediction

By integrating Equation (15), the position and velocity of the reentry vehicle can be obtained at any time. The prediction portion of the predictor–corrector algorithm can anticipate the ultimate land position based on the initial bank angle. Using the bank angle $\widetilde{\sigma}_i$ as the guidance command, the predicted reentry range $S_p(\widetilde{\sigma}_i)$ defined in the longitude profile can be calculated as follows:

$$\Omega_p(\widetilde{\sigma}_i) = \arccos\left[\cos\phi_p cos\phi\cos(\lambda_p - \lambda) + \sin\phi_p \sin\phi\right] \tag{30}$$

$$S_p(\widetilde{\sigma}_i) = R_e \cdot \Omega_p(\widetilde{\sigma}_i) \tag{31}$$

where λ and ϕ are the latitude and longitude of the reentry vehicle's actual position during the prediction execution; λ_p and ϕ_p are the latitude and longitude of the drop point, which are obtained from the prediction part of the predictor–corrector algorithm; $\Omega_p(\widetilde{\sigma}_i)$ is the geocentric angle corresponding to $S_p(\widetilde{\sigma}_i)$.

4.1.2. Command Correction Based on the Piecewise Objective Function

The iteration in the command correction part works out the bank angle. The proposed correction methods usually find σ through iteration of a single function (i.e., range–to–go). However, when the vehicle approaches the terminal, the coupling effect of longitudinal and lateral guidance will be more significant. In addition, the high values of the bank angle and the tighter heading corridor will produce frequent heading switches, resulting in the oscillation of the guidance commands and a decrease in the accuracy of the guidance. To generate stable guidance commands and improve the accuracy of reentry guidance, the piecewise objective function method is raised, which is taken as:

$$f = \begin{cases} S_p - S_f & , V > V_s \\ S_e & , V \le V_s \end{cases} \tag{32}$$

As opposed to the conventional objective function, the distance between the predicted and target drop point is introduced, which essentially represents the drop point error.

In Equation (32), V_s is the velocity when the objective function switches; S_p is the predicted range from the current point to the predicted drop point, which has been given by Equations (30) and (31); S_f is the range from the current point to the target drop point, given by:

$$\Omega_f = \arccos\left[\cos\phi_f cos\phi\cos(\lambda_f - \lambda) + \sin\phi_f \sin\phi\right] \tag{33}$$

$$S_f = R_e \cdot \Omega_f \tag{34}$$

where λ_f and ϕ_f are the latitude and longitude of the target drop point and Ω_f is the geocentric angle corresponding to S_f.

S_e is the distance between the predicted drop point and the target drop point. The calculation functions of S_e for given $\widetilde{\sigma}_i$ are Equations (35) and (36).

$$\Omega_e(\widetilde{\sigma}_i) = \arccos\left[\cos\phi_p \cos\phi_f \cos(\lambda_p - \lambda_f) + \sin\phi_p \sin\phi_f\right] \tag{35}$$

$$S_e(\widetilde{\sigma}_i) = R_e \Omega_e(\widetilde{\sigma}_i) \tag{36}$$

where $\Omega_e(\widetilde{\sigma}_i)$ is the geocentric angle corresponding to $S_e(\widetilde{\sigma}_i)$.

After the range has been determined using the supplied bank angle in the range prediction part, there will be two instances. In the instance of $V > V_s$, when the Newton iteration approach is used to solve the bank angle command described in Equation (37):

$$\widetilde{\sigma}_{i+1} = \widetilde{\sigma}_i - a\frac{f_i(\widetilde{\sigma}_i)}{\dot{f}_i(\widetilde{\sigma}_i)} \tag{37}$$

Because S_e is a nonnegative function and might not have a zero solution in the case of $V \leq V_s$, the iterative calculation is ineffective and easily divergent. In this situation, the bank angle can be solved using the equidistance test method because the vehicle is close to the flight terminal and the running time of the prediction part is greatly reduced. This method takes the bank angle into various values at a fixed interval $\Delta\sigma$ and selects the bank angle that minimizes S_e as the guidance command at that time.

The key benefit of the piecewise objective function is that S_e is the direct representation of the drop point error, making it possible to use S_e as the objective function of iteration, thus significantly lowering the drop point error.

4.2. Lateral Guidance Logic

In the related research, a reasonable reversal logic definition of the bank angle, which is designed by defining the cross–range and cross–range corridor, is used to achieve lateral guidance. There are numerous ways to define the cross–range, and each one relates to a specific cross–range corridor design methodology. The cross–range corridor often resembles a funnel. When the reentry vehicle is approaching the terminal, the tighter funnel boundary will allow the bank angle to flip more frequently. Furthermore, if a wider cross–range corridor is specified, it is difficult to satisfy the precision of the drop point. By setting the update time of the guidance command, the piecewise objective function is used to limit the reversal of the bank angle and avoid the challenge of designing the terminal boundary.

When $V > V_s$, the cross–range represents the drop point error of flight to the terminal point along with the current state, which is defined as follows:

$$Z = \arcsin\left(\sin\Omega_f \sin\Delta\psi\right) \tag{38}$$

where $\Delta\psi$ is the heading error, namely the angle between the velocity and the displacement–to–go. This strategy has a greater impact on guiding since the cross–range may react quickly when the bank angle reserves. Additionally, this approach may guarantee that the trajectory will gradually tend to the target drop point because the heading inaccuracy is based on that point.

The edge of the cross–range is given by:

$$\begin{cases} Z_{\max} = k_1 \arcsin(\sin\Omega \sin\Delta\psi_{\max}) \\ Z_{\min} = -k_2 \arcsin(\sin\Omega \sin\Delta\psi_{\max}) \end{cases} \tag{39}$$

where k_1 and k_2 are adjustable constants and $\Delta\psi_{\max}$ is the max heading error. Therefore, the sign of the bank angle is determined by:

$$\mathrm{sgn}(\widetilde{\sigma}_i) = \begin{cases} 1 & , Z < Z_{\min} \\ -1 & , Z > Z_{\max} \end{cases} \tag{40}$$

Therefore, the guidance command becomes as follows:

$$\widetilde{\sigma} = |\widetilde{\sigma}_i| \mathrm{sgn}(\widetilde{\sigma}_i) \tag{41}$$

when $V \leq V_s$ and the longitudinal guidance uses the equidistance test method, which essentially can solve the magnitude and the sign of the bank angle simultaneously. Therefore, it is not necessary to build an additional lateral guidance strategy and design the cross–range corridor for reversal.

The framework of piecewise predictor–corrector guidance is shown in Figure 5.

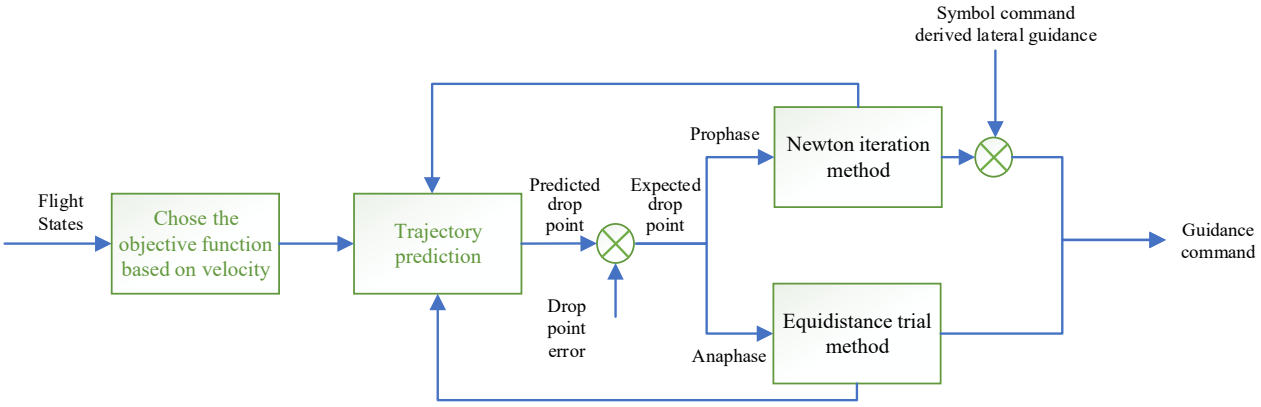


Figure 5. Framework of piecewise predictor–corrector guidance.

5. Simulation Validation

To evaluate the efficiency of DINN utilized to assess vehicle flight capability in any reentry state and compare DINN with BPNN, simulation tests are carried out in various situations, including high energy status and low energy status. These two statuses represent, respectively, the prophase and anaphase of reentry.

The flight corridor obeys the path constraints as: $\dot{Q}_{\max} = 3$ MW, $q_{\max} = 10$ kPa, $n_{\max} = 6$. The final constraints go with $h_f = 21.9$ km, $v_f = 763$ m/s.

Building upon the constraints above, the max magnitude of the bank angle is [–80°, 80°].

Sections 5.1 and 5.2 will give the result of the flight capability assessment in different energy states using various methods.

In the simulation, two TAEM points are selected as (90° N, 40° E) and (97° N, 44° E) for guidance after the flight capability assessment. By simulating the guidance algorithm from the initial state point to the TAEM point and observing the TAEM point errors, it is possible to confirm the effectiveness of the piecewise predictor–corrector method.

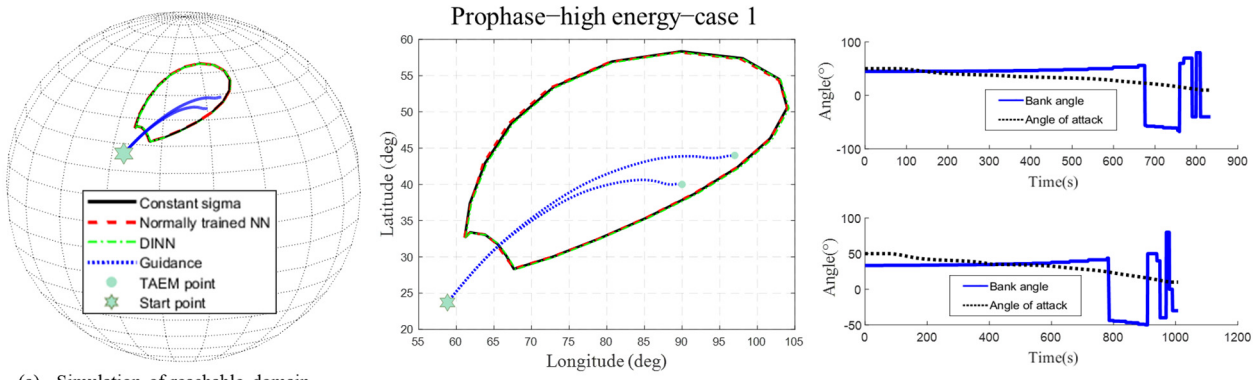
5.1. Example 1: High Energy Status (Prophase of Reentry)

The high energy status is first considered, namely the prophase of reentry. In this status, the altitude of the vehicle is 75–85 km and the velocity is 6500–7500 m/s. Other details are shown in Table 5.

Table 5. Four cases in high energy status.

Case	Altitude	Velocity	Path Angle	Course Angle	Longitude	Latitude	Mass
Case1	80,163	7153	−1.54	35.05	58.82	23.71	5705
Case2	77,328	7006	−1.28	36.27	59.90	25.09	5433
Case3	75,328	6896	−1.15	36.92	59.22	25.34	5833
Case4	74,988	6854	−0.95	37.32	60.60	25.93	5988

The reachable domains calculated using the constant bank angle method, BPNN and DINN are shown in Figures 6–9.

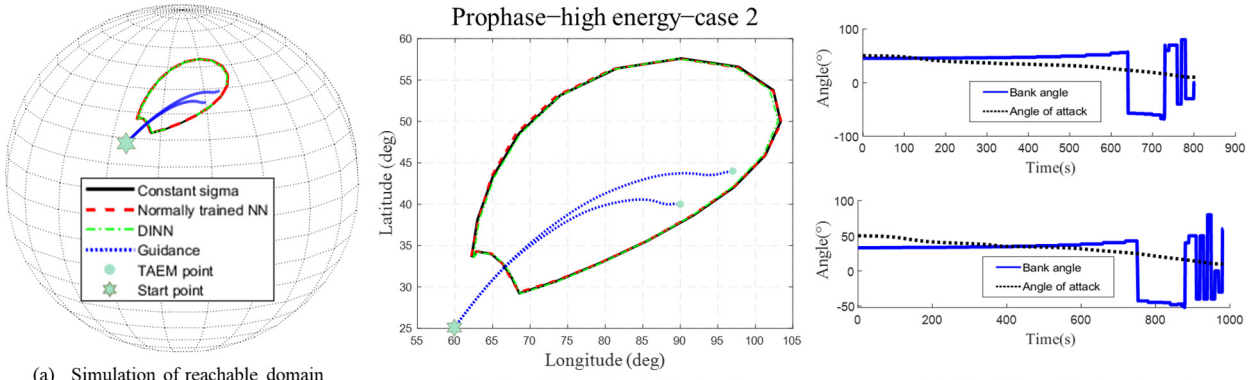


(a) Simulation of reachable domain on the Earth

(b) Simulation of reachable domain on the longitude−latitude profile

(c) Simulation of bank angle and angle of attack

Figure 6. Simulation of high energy Case 1.

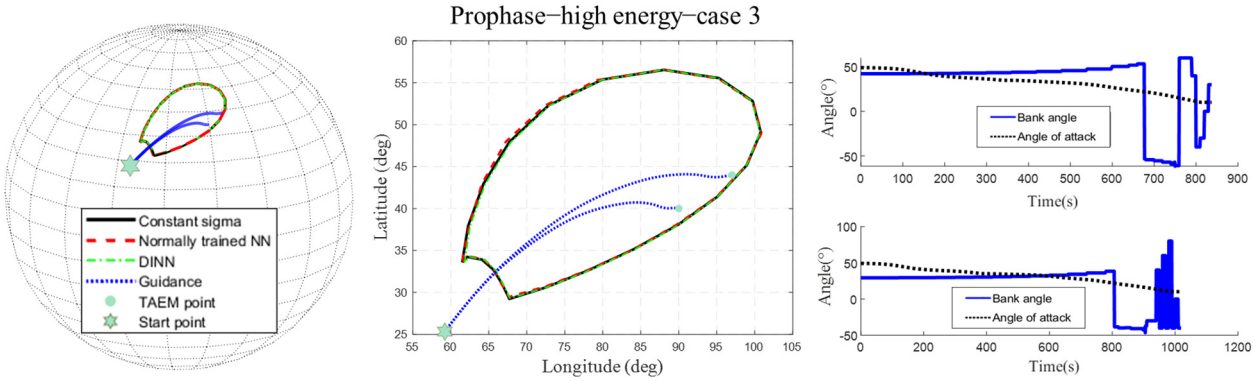


(a) Simulation of reachable domain on the Earth

(b) Simulation of reachable domain on the longitude−latitude profile

(c) Simulation of bank angle and angle of attack

Figure 7. Simulation of high energy Case 2.



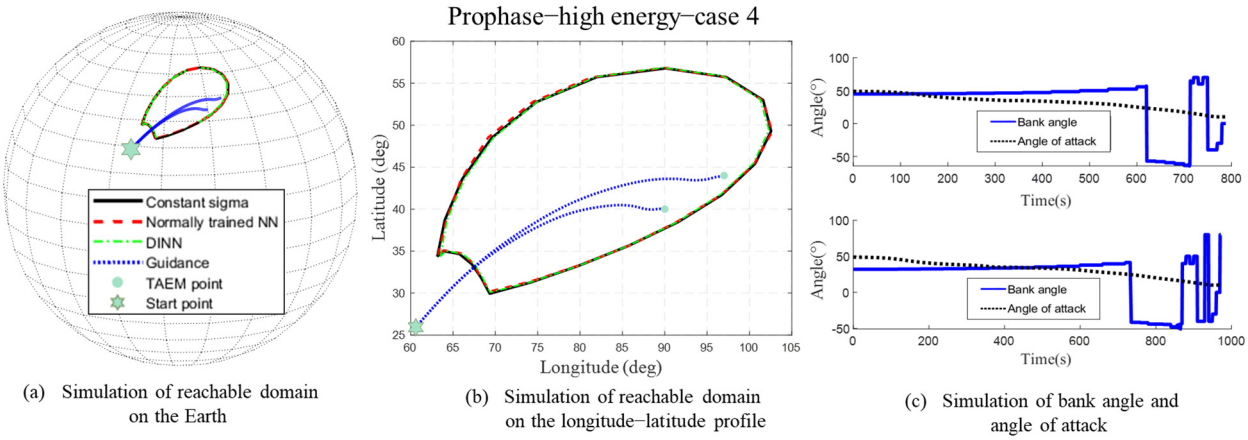
(a) Simulation of reachable domain on the Earth

(b) Simulation of reachable domain on the longitude−latitude profile

(c) Simulation of bank angle and angle of attack

Figure 8. Simulation of high energy Case 3.

As can be seen in the above figures, the reachable domains achieved using the constant bank angle method, BPNN, and DINN are nearly identical. The average computational time of the constant bank angle method is 15.84 s (15,842 s in 1000 times), while the DINN requires 0.019 s (19.17 s in 1000 times). Table 6 shows training time and average computational time of the three methods.



(a) Simulation of reachable domain on the Earth

(b) Simulation of reachable domain on the longitude−latitude profile

(c) Simulation of bank angle and angle of attack

Figure 9. Simulation of high energy Case 4.

Table 6. Comparison of each method.

Methods	Training Time (s)	Average Computational Time (s)
Constant bank angle method	–	15.84
BPNN	31,656	–
DINN	3517	0.019

Additionally, feasible trajectories from the initial state point to two TAEM points constructed by the guidance algorithm in four high energy cases. Table 7 displays the drop point error for high energy examples 1 to 4. Therefore, the piecewise predictor–corrector algorithm can accurately guide the reentry vehicle to the selected TAEM points.

Table 7. The drop point error of high energy cases 1–4.

TAEM Point	Case 1	Case 2	Case 3	Case 4
(90° N, 40° E)	586.5	240.9	564.8	212.9
(97° N, 44° E)	719.9	616.4	518.4	505.5

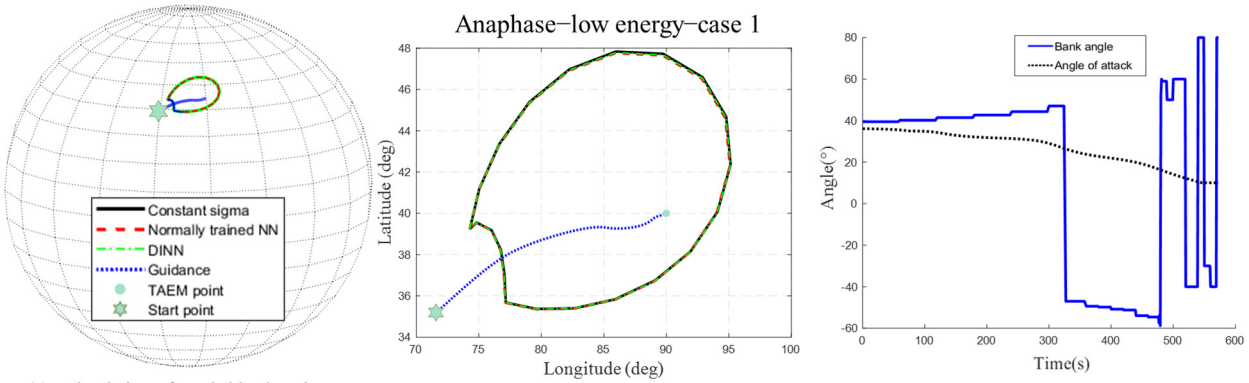
5.2. Example 2: Low Energy Status (Anaphase of Reentry)

The low energy status is then considered, namely the metaphase of reentry. In this status, the altitude of the vehicle is 60–70 km and the velocity is 4500–5500 m/s. Other details are shown in Table 8.

Table 8. Four cases in low energy status.

Case	Altitude	Velocity	Path Angle	Course Angle	Longitude	Latitude	Mass
Case 1	67,057	5151	−1.00	54.95	71.57	35.16	5705
Case 2	66,704	5073	−1.04	54.82	71.57	35.19	5433
Case 3	65,807	5004	−0.73	56.83	72.30	35.57	5833
Case 4	64,993	4850	−0.42	58.84	73.04	35.94	5988

The reachable domains calculated using the constant bank angle method, BPNN, and DINN are shown in Figures 10–13.

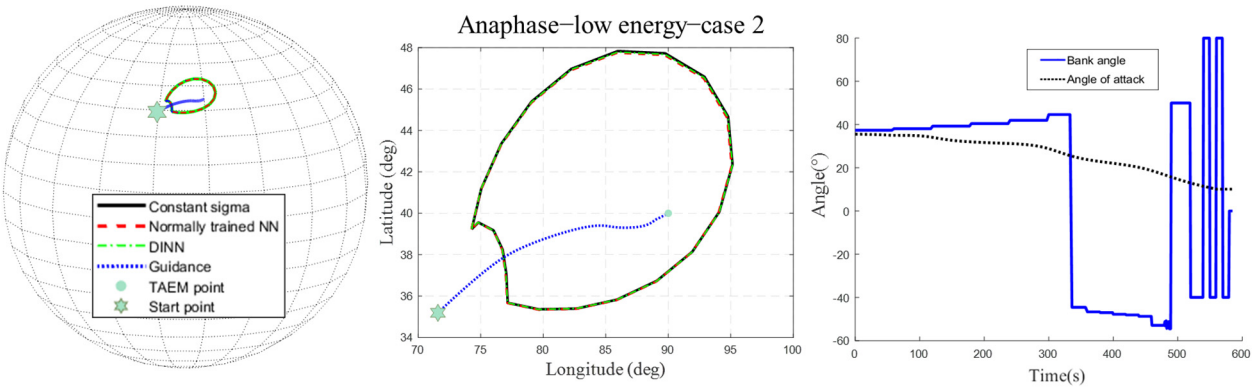


(a) Simulation of reachable domain on the Earth

(b) Simulation of reachable domain on the longitude−latitude profile

(c) Simulation of bank angle and angle of attack

Figure 10. Simulation of low energy Case 1.

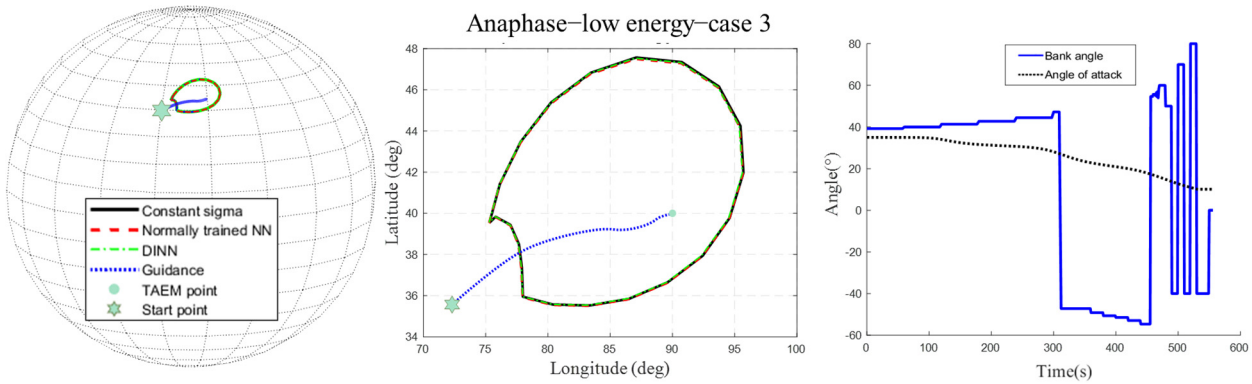


(a) Simulation of reachable domain on the Earth

(b) Simulation of reachable domain on the longitude−latitude profile

(c) Simulation of bank angle and angle of attack

Figure 11. Simulation of low energy Case 2.



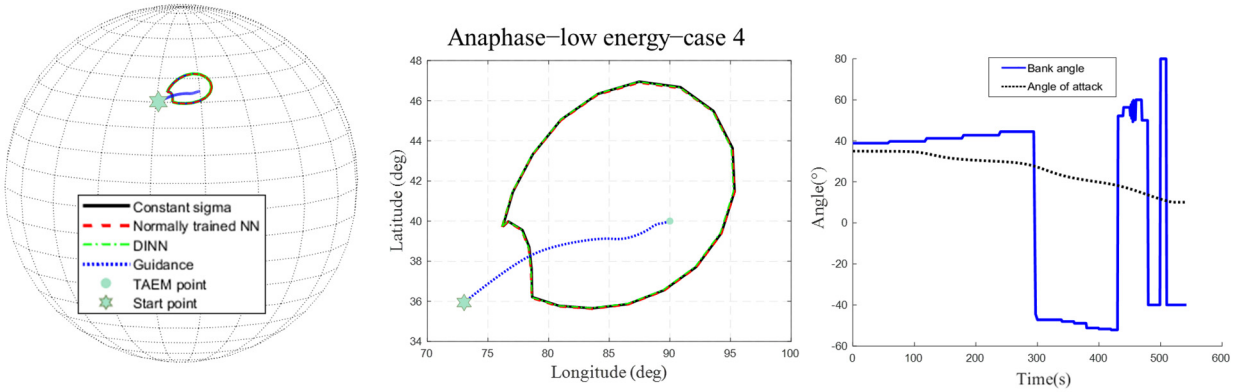
(a) Simulation of reachable domain on the Earth

(b) Simulation of reachable domain on the longitude−latitude profile

(c) Simulation of bank angle and angle of attack

Figure 12. Simulation of low energy Case 3.

As can be seen in the above figures, the reachable domains achieved by the constant bank angle method, BPNN, and DINN are nearly identical. The average computational time of the constant bank angle method is 10.75 s (10,750 s in 1000 times), while the DINN requires 0.018 s (18.28 s in 1000 times). Table 9 shows training time and average computational time of the three methods.



(a) Simulation of reachable domain on the Earth

(b) Simulation of reachable domain on the longitude−latitude profile

(c) Simulation of bank angle and angle of attack

Figure 13. Simulation of low energy Case 4.

Table 9. Comparison of each method.

Methods	Training Time (s)	Average Computational Time (s)
Constant bank angle method	–	10.75
BPNN	31,656	–
DINN	3517	0.018

Additionally, feasible trajectories from the initial state point to the TAEM point (90° N, 40° E) were constructed by the guidance algorithm in four low energy cases. Table 10 displays the drop point error for low energy examples 1 to 4. Therefore, the piecewise predictor–corrector algorithm can accurately guide the reentry vehicle to the selected TAEM point.

Table 10. The drop point error of low energy cases 1–4.

TAEM Point	Case 1	Case 2	Case 3	Case 4
(90° N, 40° E)	757.6	1890.3	869.5	464.4

Some primary benefits of DINN may be observed from the above simulation findings. When creating a database, DINN is one–ninth as efficient as BPNN. The constant bank angle method, as the most effective related strategy, still requires 15.84 s and 10.75 s for calculation during the reachable domain generation phase, while DINN needs only 0.019 s and 0.018 s. Since the reachable domains obtained using each approach are nearly identical, the accuracy of DINN is equal to that of the constant bank angle method and BPNN.

6. Conclusions

Reentry is one of the crucial steps in completing the space roundtrip. Reentry vehicles will become more intelligent owing to a reentry guidance algorithm with high precision as well as great autonomous online decision–making and target–changing capabilities. In this paper, the reachable domain is used to analyze the flight capabilities of the reentry vehicle. Taking into account the high immediacy of decision making, DINN is also introduced to achieve rapid calculation of the reachable domain, which is based on a priori dynamic knowledge. In the training process, by utilizing a priori dynamic knowledge on the evaluation of the impact characteristics between variates and the reachable domain, the performance of DINN is maintained while the dimension of the training database is reduced. The feasibility of the reentry trajectories can be assessed in advance by determining the relative position between the reachable domain and the TAEM point. The judgment results can be utilized in decision making and enhancing the autonomy of vehicles, which can be deployed in more

scenarios. Additionally, the piecewise objective function and the equidistant test strategy are proposed to perfect the predictor–corrector guidance algorithm used for the reentry vehicle. A piecewise objective function is introduced, which uses "range–to–go" as the iterative objective function at the prophase and "predicted error of drop points" as the iterative objective function at the metaphase. The proposed function solves the iteration divergence brought by range–to–go converging to zero. This method successfully enhances the accuracy and stability of the predictor–corrector guidance algorithm.

Author Contributions: Conceptualization, K.L. and J.Z.; methodology, K.L. and J.Z.; software, K.L., J.Z. and X.G.; validation, K.L., J.Z. and X.G.; formal analysis, K.L., J.Z. and X.G.; investigation, K.L., J.Z. and X.G.; resources, K.L., J.Z. and X.G.; data curation, K.L., J.Z. and X.G.; writing—original draft preparation, K.L., J.Z. and X.G.; writing—review and editing, K.L., J.Z. and X.G.; visualization, K.L., J.Z. and X.G.; supervision, K.L., J.Z. and X.G.; project administration, K.L., J.Z. and X.G.; funding acquisition, K.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under grant NSFC U2141229 and the Chinese Aeronautical Establishment, Aviation Science Foundation under grant ASF 2019ZC063001, and the JCJQ Funding under grant 2019–JCJQ–DA–001–131.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Benito, J.; Mease, K.D. Reachable and Controllable Sets for Planetary Entry and Landing. *J. Guid. Control. Dyn.* **2010**, *33*, 641–654. [CrossRef]
2. Hsu, F.-K.; Kuo, T.-S.; Chern, J.-S. Landing Domain Analysis of Shuttle Re–Entry Vehicles. *Int. J. Syst. Sci.* **1991**, *22*, 1145–1158. [CrossRef]
3. Zhang, H.; Zhang, G. Reachable Domain of Ground Track with a Single Impulse. *IEEE Trans. Aerosp. Electron. Syst.* **2020**, *57*, 1105–1122. [CrossRef]
4. Arslantaş, Y.E.; Oehlschlägel, T.; Sagliano, M. Safe Landing Area Determination for a Moon Lander by Reachability Analysis. *Acta Astronaut.* **2016**, *128*, 607–615. [CrossRef]
5. Cui, P.; Ge, D.; Gao, A. Optimal Landing Site Selection Based on Safety Index during Planetary Descent. *Acta Astronaut.* **2017**, *132*, 326–336. [CrossRef]
6. Gao, C.; Jiang, C.; Jing, W. Optimization of Projectile State and Trajectory of Reentry Body Considering Attainment of Carrying Aircraft. *J. Syst. Eng. Electron.* **2017**, *28*, 137–144. [CrossRef]
7. Liu, X.; Li, S.; Jiang, X.; Huang, X. Planetary Landing Site Detection and Selection Using Multilevel Optimization Strategy. *Acta Astronaut.* **2019**, *163*, 272–286. [CrossRef]
8. Lu, L.; Li, H.; Zhou, W.; Liu, J. Design and Analysis of a Direct Transfer Trajectory from a near Rectilinear Halo Orbit to a Low Lunar Orbit. *Adv. Space Res.* **2021**, *67*, 1143–1154. [CrossRef]
9. Wen, C.; Zhao, Y.; Shi, P. Precise Determination of Reachable Domain for Spacecraft with Single Impulse. *J. Guid. Control Dyn.* **2014**, *37*, 1767–1779. [CrossRef]
10. Yingjie, Z.; Hongwei, Y.; Jincheng, H. The Fast Generation of the Reachable Domain for Collision–Free Asteroid Landing. *Mathematics* **2022**, *10*, 3763. [CrossRef]
11. Tian, B.; Zong, Q. Optimal Guidance for Reentry Vehicles Based on Indirect Legendre Pseudospectral Method. *Acta Astronaut.* **2011**, *68*, 1176–1184. [CrossRef]
12. Li, D.-W.; Yang, B. Reentry Guidance for Reusable Launching Vehicle. *J. Solid Rocket. Technol.* **2010**, *33*, 119–124.
13. Chai, R.; Tsourdos, A.; Savvaris, A.; Chai, S.; Xia, Y. Trajectory Planning for Hypersonic Reentry Vehicle Satisfying Deterministic and Probabilistic Constraints. *Acta Astronaut.* **2020**, *177*, 30–38. [CrossRef]
14. Causey, W.; Sohoni, V. Reentry Guidance for Space Shuttle. In Proceedings of the Symposium on Automatic Control in Space, Genova, Italy, 4–8 June 1973.
15. Pan, L.; Peng, S.; Xie, Y.; Liu, Y.; Wang, J. 3D Guidance for Hypersonic Reentry Gliders Based on Analytical Prediction. *Acta Astronaut.* **2020**, *167*, 42–51. [CrossRef]
16. Lees, L.; Hartwig, F.W.; Cohen, C.B. The Use of Aerodynamic Lift During Entry into the Earth's Atmosphere. *ARS J.* **1959**, *29*, 633–641. [CrossRef]
17. Shen, Z.; Lu, P. Dynamic Lateral Entry Guidance Logic. *J. Guid. Control Dyn.* **2004**, *27*, 949–959. [CrossRef]
18. Lu, P. Predictor–Corrector Entry Guidance for Low–Lifting Vehicles. *J. Guid. Control Dyn.* **2008**, *31*, 1067–1075. [CrossRef]

19. Zhang, D.; Liu, L.; Wang, Y. On–Line Reentry Guidance Algorithm with Both Path and No–Fly Zone Constraints. *Acta Astronaut.* **2015**, *117*, 243–253. [CrossRef]
20. Wu, J.; Xiong, F. Predictor–Corrector Guidance Law Considering Multiple Terminal Constraints. In Proceedings of the 39th Chinese Control Conference, CCC 2020, Shenyang, China, 27–29 July 2020; IEEE Computer Society: Shenyang, China, 2020; pp. 3316–3320.
21. Sushnigdha, G.; Joshi, A. Trajectory Design of Re–Entry Vehicles Using Combined Pigeon Inspired Optimization and Orthogonal Collocation Method. *IFAC-PapersOnLine* **2018**, *51*, 656–662. [CrossRef]
22. Wu, Y.; Yao, J.; Qu, X. An Adaptive Reentry Guidance Method Considering the Influence of Blackout Zone. *Acta Astronaut.* **2018**, *142*, 253–264. [CrossRef]
23. Izzo, D.; Märtens, M.; Pan, B. A Survey on Artificial Intelligence Trends in Spacecraft Guidance Dynamics and Control. *Astrodynamics* **2019**, *3*, 287–299. [CrossRef]
24. Dong, Y.; Wen, J.; Zhang, Y.; Ai, J. Deep Neural Networks–Based Air Data Sensors Fault Detection for Aircraft. In Proceedings of the 2021 33rd Chinese Control and Decision Conference (CCDC), Kunming, China, 22–24 May 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 442–447.
25. Shirobokov, M.; Trofimov, S.; Ovchinnikov, M. Survey of Machine Learning Techniques in Spacecraft Control Design. *Acta Astronaut.* **2021**, *186*, 87–97. [CrossRef]
26. Silvestrini, S.; Lavagna, M. Deep Learning and Artificial Neural Networks for Spacecraft Dynamics, Navigation and Control. *Drones* **2022**, *6*, 270. [CrossRef]
27. Hennes, D.; Izzo, D.; Landau, D. Fast Approximators for Optimal Low–Thrust Hops between Main Belt Asteroids. In Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece, 6–9 December 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–7.
28. Mereta, A.; Izzo, D.; Wittig, A. Machine Learning of Optimal Low–Thrust Transfers between near–Earth Objects. In Proceedings of the International Conference on Hybrid Artificial Intelligence Systems, La Rioja, Spain, 21–23 June 2017; Springer: Berlin/Heidelberg, Germany, 2017; pp. 543–553.
29. Yin, S.; Li, J.; Cheng, L. Low–Thrust Spacecraft Trajectory Optimization via a DNN–Based Method. *Adv. Space Res.* **2020**, *66*, 1635–1646. [CrossRef]
30. Xie, R.; Dempster, A.G. An On–Line Deep Learning Framework for Low–Thrust Trajectory Optimisation. *Aerosp. Sci. Technol.* **2021**, *118*, 107002. [CrossRef]
31. Rubinsztejn, A.; Sood, R.; Laipert, F.E. Neural Network Optimal Control in Astrodynamics: Application to the Missed Thrust Problem. *Acta Astronaut.* **2020**, *176*, 192–203. [CrossRef]
32. Santos, R.R.; Rade, D.A.; da Fonseca, I.M. A Machine Learning Strategy for Optimal Path Planning of Space Robotic Manipulator in On–Orbit Servicing. *Acta Astronaut.* **2022**, *191*, 41–54. [CrossRef]
33. Galchenko, P.; Pernicka, H.J.; Balakrishnan, S.N. Pointing System Design for the COronal Diagnostic Experiment (CODEX) Using a Modified State Observer and a Neural Network Controller. In Proceedings of the 2020 AAS/AIAA Astrodynamics Specialist Conference, South Lake Tahoe, CA, USA, 9–13 August 2020; pp. 645–660.
34. Biggs, J.D.; Fournier, H. Neural–Network–Based Optimal Attitude Control Using Four Impulsive Thrusters. *J. Guid. Control Dyn.* **2020**, *43*, 299–309. [CrossRef]
35. Eldad, O.; Lightsey, E.G.; Claudel, C. Minimum–Time Attitude Control of Deformable Solar Sails with Model Uncertainty. *J. Spacecr Rocket.* **2017**, *54*, 863–870. [CrossRef]
36. Viavattene, G.; Devereux, E.; Snelling, D.; Payne, N.; Wokes, S.; Ceriotti, M. Design of Multiple Space Debris Removal Missions Using Machine Learning. *Acta Astronaut.* **2022**, *193*, 277–286. [CrossRef]
37. Lai, G.; Tao, G.; Zhang, Y.; Liu, Z. Adaptive Control of Noncanonical Neural–Network Nonlinear Systems with Unknown Input Dead–Zone Characteristics. *IEEE Trans. Neural. Netw. Learn Syst.* **2019**, *31*, 3346–3360. [CrossRef]
38. Zeng, Z.-F.; Tang, Y.-H.; Xu, M.; Chen, S.-L. Study of Reentry Guidance Based on Neural Network. *Flight Dyn.* **2011**, *29*, 64–67.
39. Hilton, W.F.; Vinh, N.X.; Busemann, A.; Culp, R.D. *Hypersonic and Planetary Entry Flight Mechanics*; University of Michigan Press: Ann Arbor, MI, USA, 1980; 357p. [CrossRef]