

Article#

Online Trajectory Optimization Based on Pseudospectra Convex Optimization for Morphing Gliding Reentry Vehicles

Tong Wei ¹, Jiale Huang ^{1,2}, Xingyu Zhu ^{1,2}, Fengqi Ni ¹, Xinyue Zhou ^{1,2}, Mengdie Liu ^{1,2} and Enmi Yong ^{1,*}

¹ Computational Aerodynamics Institute, China Aerodynamics Research and Development Center, Mianyang 621000, China

² School of Computer Science, Southwest University of Science and Technology, Mianyang 621000, China

* Correspondence: 13261267330@163.com

Abstract

Trajectory planning for morphing gliding reentry vehicles is a nonconvex optimization problem driven by nonlinearity, parameter uncertainty, and multiple constraints. No-fly zones (NFZs) are a critical constraint because their rapid movement and expansion hinder the real-time generation of optimal flight trajectories and wing morphing strategies. Therefore, this study proposes an innovative online trajectory optimization method based on sequential convex optimization integrated with a deep neural network (DNN). The proposed method first uses the Radau pseudospectral method to discretize continuous dynamics and convert the non-convex trajectory planning problem into a relaxed convex subproblem. The subproblem is reformulated as an augmented Lagrangian function through linearization and is iteratively solved using the interior-point method. Finally, the DNN learns the mapping between flight states and optimal control variables (angle of attack rate, bank angle rate, and wing sweep angle rate) to rapidly generate control variables. Different from the time-consuming offline optimization method, the proposed model only requires 0.4 ms to predict three groups of control variables, with the predicted control errors remaining below 2.25%. This method efficiently provides high-precision and stable reentry trajectories and morphing strategies for gliding reentry vehicles. Thus, the proposed method achieves synchronous flight path and wing deformation optimization and demonstrates strong robustness under time-varying mission conditions.

Keywords: trajectory planning; no-fly zones; sequential convex optimization; Radau pseudospectral; deep neural network

Academic Editors: Shufan Wu,
Qiang Shen and Yixin Huang

Received: 24 May 2026

Revised: 21 June 2026

Accepted: 29 June 2026

Published: 30 June 2026

Copyright: © 2026 by the authors.
Licensee MDPI, Basel, Switzerland.
This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Morphing gliding reentry vehicles have several advantages, including high maneuverability, flexible trajectories, and robust anti-interception capabilities [1,2]. These vehicles have been extensively researched by major global aerospace powers because of their wide velocity profiles and extended operational altitudes [3,4]. Among the related technologies, trajectory planning is crucial for analyzing the performance of aircraft, such as range, maneuverability, and ballistic characteristics. This allows morphing gliding reentry vehicles to achieve optimal flight performance by altering wing shapes according to different mission scenarios (e.g., take-off, cruise, reconnaissance, attack, and landing).

Trajectory planning typically requires solving nonlinear optimal control problems under various constraints, including boundary conditions, shape parameter constraints,

no-fly-zone (NFZ) constraints, and path constraints [5]. Previous research primarily categorizes proposed trajectory planning algorithms into indirect and direct methods [6,7]. The indirect method derived from Pontryagin's maximum principle transforms the trajectory optimization problem into a Hamiltonian boundary value problem, which is typically solved using gradient-based algorithms, and offers high solution accuracy and strong optimality. The direct method primarily adopts parameterization to convert the optimal control problem into a parameter optimization problem that is solved using nonlinear programming methods such as the sequential quadratic programming algorithm. This method has been widely investigated because of its intuitive formulation [8–10]; however, its unpredictable solution times and unstable convergence restrict its application to online scenarios [11].

Within the direct method, convex optimization frameworks meet the requirements for rapid computational solutions, which increase their utility in spacecraft trajectory optimization [12–15]. These frameworks excel in solving low-complexity problems and deliver results quickly using interior-point methods. However, most aerospace-related problems are constrained by nonlinear, nonconvex dynamics, and path constraints, which cannot be directly solved within the convex optimization framework. Therefore, a convexification method that effectively minimizes approximation errors is necessary. The mainstream convexification methods include lossless and sequential convexification [16].

To address the Mars landing trajectory planning problem, Ackimese et al. utilized a lossless convexification method. This approach efficiently replaces nonconvex constraints with relaxed convex constraints without compromising the solution accuracy [17,18]. For reentry trajectory optimization, the sequential convex programming (SCP) methodology proposed by Liu et al. differentiates energy-related dynamic equations and integrates linearization and relaxation operations within iterations [13,19–21]. The feasibility of this methodology was substantiated by the minimum time and minimum heat flow reentry trajectories. Furthermore, Wang and Zhang improved the conventional SCP algorithm with two enhanced strategies, line-search SCP and trust-region SCP, which significantly enhanced its convergence performance [22]. Zhou et al. proposed an improved SCP method using an adaptive mesh refinement [23], which balanced computational efficiency and solution accuracy by dynamically adjusting the node distribution according to the linearization error after each iteration.

However, the SCP methodology requires further improvements in terms of discretization, convergence, and penalty parameter selection. The classic trapezoidal discretization approach has been used to convert continuous optimization problems into a discrete form [22,24]. Nevertheless, it often introduces considerable discrepancies between the approximation and actual models during computation. To improve the solution accuracy, the density of the equidistant discretization nodes must be increased. This inevitably increases the number of optimization variables and significantly extends the solution time. In contrast, the pseudospectral discretization method achieves higher precision without increasing the number of nodes and has been widely applied to solve optimal control problems [14,25]. Nevertheless, most studies have investigated trajectory optimization for conventional aircraft with fixed configurations, while rarely addressing morphing strategies during flight. For a fixed-configuration aircraft, trajectory optimization aims to obtain the optimal trajectory described by time-varying variables, including the angle of attack, velocity, altitude, and flight-path angle.

Morphing gliding reentry vehicles require the generation of optimal trajectories and the determination of the optimal wing morphing strategy under dynamic mission conditions. The complexity and uncertainty associated with actual flight missions have increased considerably. For instance, no-fly zones in combat regions can be altered at any

time. In these scenarios, a vehicle must generate a feasible and safe trajectory, which circumvents restricted zones, ensures flight safety, and satisfies mission constraints by adjusting its wing configuration and other external parameters accordingly. Accordingly, establishing an efficient online trajectory optimization framework that satisfies the practical demands of complex morphing–gliding reentry missions is critical.

However, the application of convex optimization to online trajectory optimization is limited by the real-time requirements and convergence performance. Next-generation artificial intelligence has advanced significantly in numerous fields, and its applications in aerospace guidance, control, and dynamics have been increasingly explored [26–28], resulting in several promising achievements. Sanchez-Sanchez and Izzo investigated a deep neural network (DNN) for optimal state-feedback control of nonlinear systems and developed a real-time optimal control architecture [29] capable of handling numerous distinct initial conditions and maintaining optimal feedback responses. Li et al. exploited the strong approximation capabilities and high computational efficiencies of DNNs and used these networks to rapidly generate real-time control commands [30]. Even under deviations from nominal conditions, the framework generates high-precision control commands and exhibits favorable generalization performance. However, research on real-time optimal trajectory generation for morphing gliding reentry vehicles operating in time-varying NFZ environments is insufficient. To address this limitation, this study investigates trajectory optimization and morphing strategies for such vehicles. An online trajectory optimization framework is proposed, which integrates an improved sequential convexification method and a DNN to improve the trajectory planning performance for morphing vehicles.

The remainder of this paper is organized as follows: Section 2 presents an online trajectory optimization scheme that combines deep neural networks and Radau pseudospectral sequential convex optimization. Section 3 presents numerical simulations to verify the effectiveness of the proposed method. Finally, Section 4 concludes the paper.

2. Materials and Methods

2.1. Aerodynamics and Modeling of Morphing Gliding Reentry Vehicle

In this study, the morphing–gliding reentry vehicles designed in a prior study [31], which consists of two movable components (left and right wings) and a fixed fuselage, were investigated. The wing sweep angle varies smoothly between 20° and 90° . As illustrated in Figure 1, this morphing vehicle has three typical configurations, namely, the loiter, sweep, and dash configurations with sweep angles of 20° , 40° , and 60° , respectively. The wing parameters for the three configurations are listed in Table 1.

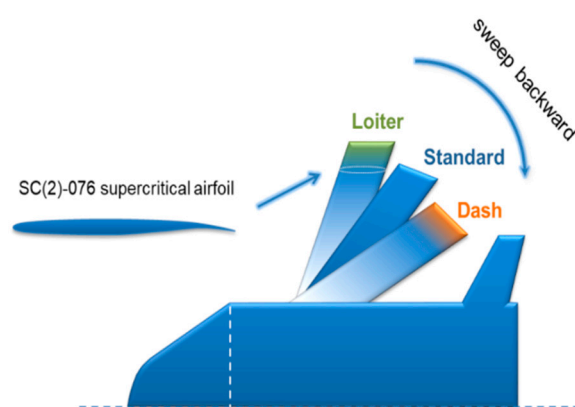


Figure 1. Geometric models of three typical configurations.

Table 1. Wing parameters of three configurations.

Configuration	Loiter	Standard	Dash
Sweep, $\Lambda(^{\circ})$	20	40	60
S_x (m)	−0.093	−0.198	−0.283
Pitch moment of inertia of each wing I_{ky} (kg m ²)	0.6	2.4	4.4
Span (m)	1.99	1.73	1.36

A detailed aerodynamic analysis of this morphing gliding reentry vehicle under three configurations, which analyzes flight conditions with Mach numbers of 2, 4, 5, and 7, and angles of attack ranging from -2° to 10° , is presented [32]. In addition, approximate formulas for the aerodynamic behavior are provided. Because the lift and drag coefficients can be approximated as linear and quadratic functions of the angle of attack, respectively, they were modeled as functions of multiple variables, including the angle of attack (α), Mach number (M), and sweep angle (Λ), as follows:

$$\begin{aligned} C_L &= C_L^0(M, \bar{\Lambda}) + C_L^{\alpha}(M, \bar{\Lambda})\alpha + \Delta C_L \\ C_D &= C_D^0(M, \bar{\Lambda}) + C_D^{\alpha}(M, \bar{\Lambda})\alpha + C_D^{\alpha^2}(M, \bar{\Lambda})\alpha^2 + \Delta C_D \end{aligned} \quad (1)$$

where the terms ΔC_L and ΔC_D quantify the aerodynamic uncertainty coefficients. The variable M denotes the Mach number, and $\bar{\Lambda}$ represents the normalized sweep angle, defined by the equation $\bar{\Lambda} = (\Lambda - 20)/70$, where $\bar{\Lambda} \in [0, 1]$. Based on the three typical configurations and four Mach numbers, the coefficients of $C_D^0(M, \bar{\Lambda})$, $C_D^{\alpha}(M, \bar{\Lambda})$, $C_D^{\alpha^2}(M, \bar{\Lambda})$, $C_L^0(M, \bar{\Lambda})$, and $C_L^{\alpha}(M, \bar{\Lambda})$ are described by the following functions:

$$\begin{aligned} C_D^0 &= -3.116 \times 10^{-2} - 6.272 \times 10^{-3} \bar{\Lambda} + 9.989 \times 10^{-2} M \\ &\quad + 4.593 \times 10^{-3} \bar{\Lambda}^2 - 3.722 \times 10^{-3} \bar{\Lambda} M - 4.037 \times 10^{-2} M^2 \\ &\quad - 8.766 \times 10^{-4} \bar{\Lambda}^2 M + 1.125 \times 10^{-3} \bar{\Lambda} M^2 + 6.118 \times 10^{-3} M^3 \\ &\quad + 8.732 \times 10^{-5} \bar{\Lambda}^2 M^2 - 8.575 \times 10^{-5} \bar{\Lambda} M^3 - 3.143 \times 10^{-4} M^4 \\ C_D^{\alpha} &= 2.037 \times 10^{-3} - 4.622 \times 10^{-3} \bar{\Lambda} + 4.129 \times 10^{-4} M \\ &\quad + 2.728 \times 10^{-3} \bar{\Lambda}^2 + 2.229 \times 10^{-3} \bar{\Lambda} M - 1.909 \times 10^{-4} M^2 \\ &\quad - 1.189 \times 10^{-3} \bar{\Lambda} M - 2.98 \times 10^{-4} \bar{\Lambda} M^2 + 2.783 \times 10^{-5} M^3 \\ &\quad + 1.106 \times 10^{-4} \bar{\Lambda}^2 M^2 + 1.083 \times 10^{-5} \bar{\Lambda} M^3 - 1.322 \times 10^{-6} M^4 \\ C_D^{\alpha^2} &= 9.879 \times 10^{-4} + 1.743 \times 10^{-4} \bar{\Lambda} - 1.408 \times 10^{-4} M \\ &\quad - 4.794 \times 10^{-4} \bar{\Lambda}^2 - 3.67 \times 10^{-5} \bar{\Lambda} M - 3.129 \times 10^{-5} M^2 \\ &\quad + 1.292 \times 10^{-4} \bar{\Lambda}^2 M + 3.196 \times 10^{-7} \bar{\Lambda} M^2 + 9.769 \times 10^{-6} M^3 \\ &\quad - 1.057 \times 10^{-5} \bar{\Lambda}^2 M^2 + 3.356 \times 10^{-7} \bar{\Lambda} M^3 - 6.292 \times 10^{-7} M^4 \\ C_L^0 &= 0.2186 - 0.228 \bar{\Lambda} - 0.1676 M + 0.1008 \bar{\Lambda}^2 \\ &\quad + 0.127 \bar{\Lambda} M + 5.618 \times 10^{-2} M^2 - 4.77 \times 10^{-2} \bar{\Lambda}^2 M \\ &\quad - 2.006 \times 10^{-2} \bar{\Lambda} M^2 - 7.836 \times 10^{-3} M^3 + 4.627 \times 10^{-3} \bar{\Lambda}^2 M^2 \\ &\quad + 9.454 \times 10^{-4} \bar{\Lambda} M^3 + 3.866 \times 10^{-4} M^4 \\ C_L^{\alpha} &= 5.183 \times 10^{-2} - 1.395 \times 10^{-3} \bar{\Lambda} - 1.626 \times 10^{-3} M \\ &\quad - 2.217 \times 10^{-2} \bar{\Lambda}^2 + 4.354 \times 10^{-3} \bar{\Lambda} M - 4.733 \times 10^{-2} M^2 \\ &\quad + 4.556 \times 10^{-3} \bar{\Lambda}^2 M - 9.406 \times 10^{-4} \bar{\Lambda} M^2 + 1.036 \times 10^{-3} M^3 \\ &\quad - 3.023 \times 10^{-4} \bar{\Lambda}^2 M^2 + 5.77 \times 10^{-5} \bar{\Lambda} M^3 - 6.171 \times 10^{-5} M^4 \end{aligned} \quad (2)$$

2.2. Dynamic Model and Constraints

2.2.1. Dynamic Model in the Distance Domain

To generate the trajectory of a vehicle, its dynamic model is typically non-dimensionalized in the time domain. The nondimensional dynamic equations are formulated as

$$\begin{cases}
 \dot{r} = V \sin \gamma \\
 \dot{\theta} = \frac{V \cos \gamma \sin \psi}{\cos \phi} \\
 \dot{\phi} = \frac{V \cos \gamma \cos \psi}{r} \\
 \dot{V} = -D - \frac{\sin \gamma}{r^2} + \omega_e^2 r \cos \phi (\sin \gamma \cos \phi - \cos \gamma \cos \psi \sin \phi) \\
 \dot{\gamma} = \frac{L \cos \sigma}{V} - \frac{\cos \gamma}{r^2 V} + \frac{V \cos \gamma}{r} + 2\omega_e \cos \phi \sin \psi \\
 \quad + \frac{\omega_e^2 r \cos \phi (\cos \phi \cos \gamma + \cos \psi \sin \phi \sin \gamma)}{V} \\
 \dot{\psi} = \frac{L \sin \sigma}{V \cos \gamma} + \frac{V \cos \gamma \sin \psi \tan \phi}{r} + 2\omega_e (\sin \phi - \cos \phi \tan \gamma \cos \psi) \\
 \quad + \frac{\omega_e^2 r \sin \phi \cos \phi \sin \psi}{V \cos \gamma}
 \end{cases} \quad (3)$$

The radial distance between the vehicle and Earth's center is denoted by r , with the radius of the Earth (R_0) as the nondimensional parameter ($R_0 = 6378\text{km}$). The spherical coordinates are defined by the longitude θ and latitude ϕ . V is the flight speed of the aircraft, and its nondimensional parameter is $V_c = \sqrt{g_0 R_0}$. γ , ψ , σ , and ω_e denote the flight path angle, heading angle, bank angle, and angular rotation rate of the Earth, respectively. The temporal domain is normalized using the time scale $t_c = \sqrt{R_0 / g_0}$. L and D , which represent the nondimensional lift and drag accelerations, are given by

$$\begin{cases}
 L = R_0 \rho V^2 A_{ref} C_L / 2m \\
 D = R_0 \rho V^2 A_{ref} C_D / 2m
 \end{cases} \quad (4)$$

Here, m , A_{ref} , and ρ represent the vehicle mass, aerodynamic reference area, and atmospheric density, respectively, while C_L and C_D represent the lift and drag coefficients, respectively. For the differential kinematic equations with time t as the independent variable, the total flight time must be assigned a fixed value in the reentry trajectory optimization. This limits the flexibility of optimization algorithms when searching for optimal trajectories. To circumvent this limitation, the longitudinal range angle S is adopted as an independent variable of the differential kinematic equations. Thus, the trajectory optimization problem with a fixed terminal time is transformed into one with a fixed terminal range angle. The initial and final boundary values for the longitudinal range angle are defined as follows:

$$s_f = \arccos(\sin \phi_0 \sin \phi_f + \cos \phi_f \cos \phi_0 \cos(\theta_f - \theta_0)) \quad (5)$$

where θ_0 and ϕ_0 denote the longitudinal and latitudinal coordinates, respectively, of the initial deployment point of the vehicle, while θ_f and ϕ_f represent the longitude and latitude of the target destination, respectively.

As shown in Figure 2, the problem of generating an effective trajectory can be regarded as the minimization of the projected range between the initial and target points after establishing the initial deployment and target points. S denotes the projected flight path angle. The differential transformation governing the relationship between the time and range domains is formulated as

$$ds / dt = -v \cos \theta / r \quad (6)$$

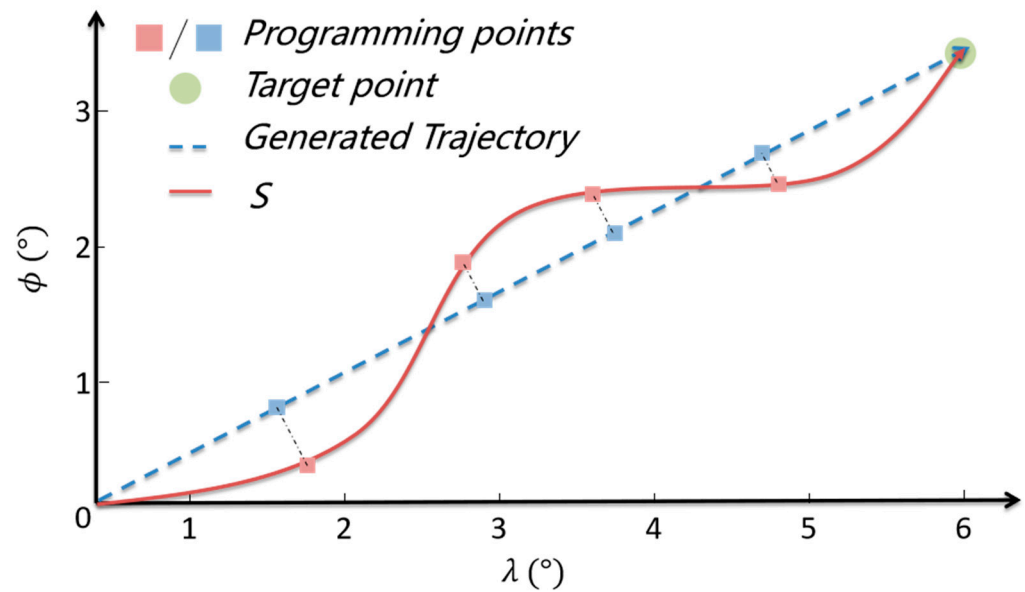


Figure 2. Generated trajectory in the flight range domain.

By integrating Equations (3) and (6), the dynamic model of the vehicle in the distance domain is expressed as

$$\left\{ \begin{array}{l} \frac{dr}{ds} = r \tan \gamma \\ \frac{d\theta}{ds} = \frac{\sin \psi}{\cos \phi} \\ \frac{d\phi}{ds} = \cos \psi \\ \frac{dV}{ds} = -\frac{rD}{V \cos \gamma} - \frac{\tan \gamma}{rV} + \frac{r^2 \omega_e^2 \cos \phi (\cos \phi \tan \gamma - \sin \phi \cos \psi)}{2} \\ \frac{d\gamma}{ds} = \frac{rL \cos \sigma}{V^2 \cos \gamma} - \frac{1}{r^2 V} + 1 + \frac{2r\omega_e \cos \phi \sin \psi}{V \cos \gamma} \\ \quad + \frac{r^2 \omega_e^2 \cos \phi (\cos \phi \tan \gamma + \cos \psi \sin \phi \tan \gamma)}{V^2} \\ \frac{d\psi}{ds} = \frac{rL \sin \sigma}{V^2 \cos^2 \gamma} + \sin \psi \tan \phi + \frac{2r\omega_e (\sin \phi - \cos \phi \tan \gamma \cos \psi)}{V \cos \gamma} \\ \quad + \frac{(r^2 \omega_e^2 \sin \phi \cos \phi \cos \psi)}{V^2 \cos^2 \gamma} \end{array} \right. \quad (7)$$

Here, ω_e denotes the angular rotation rate of the Earth. The flight trajectory is primarily affected by the angle of attack α , bank angle σ , and wing sweep angle Λ . To suppress high-frequency oscillations of Λ and avoid excessive loads on control surfaces or structural stress problems caused by rapid state variations, this study designates the time derivatives of these parameters, the rates of change in the angle of attack, bank angle, and the vehicle's wing leading-edge sweep angle, which are denoted as $\dot{\alpha}$, $\dot{\sigma}$, and $\dot{\Lambda}$, respectively, as control variables. On this basis, the following augmented differential kinematic equations are obtained:

$$\begin{cases}
\frac{dr}{ds} = r \tan \gamma \\
\frac{d\theta}{ds} = \frac{\sin \psi}{\cos \phi} \\
\frac{d\phi}{ds} = \cos \psi \\
\frac{dV}{ds} = -\frac{rD}{V \cos \gamma} - \frac{\tan \gamma}{rV} + \frac{r^2 \omega_e^2 \cos \phi (\cos \phi \tan \gamma - \sin \phi \cos \psi)}{2} \\
\frac{d\gamma}{ds} = \frac{rL \cos \sigma}{V^2 \cos \gamma} - \frac{1}{r^2 V} + 1 + \frac{2r\omega_e \cos \phi \sin \psi}{V \cos \gamma} \\
\quad + \frac{r^2 \omega_e^2 \cos \phi (\cos \phi \tan \gamma + \cos \psi \sin \phi \tan \gamma)}{V^2} \\
\frac{d\psi}{ds} = \frac{rL \sin \sigma}{V^2 \cos^2 \gamma} + \sin \psi \tan \phi + \frac{2r\omega_e (\sin \phi - \cos \phi \tan \gamma \cos \psi)}{V \cos \gamma} \\
\quad + \frac{(r^2 \omega_e^2 \sin \phi \cos \phi \cos \psi)}{V^2 \cos^2 \gamma} \\
\frac{d\alpha}{ds} = u_\alpha \\
\frac{d\sigma}{ds} = u_\sigma \\
\frac{d\Lambda}{ds} = u_\Lambda
\end{cases} \quad (8)$$

2.2.2. Constraint Settings

Throughout the operational reentry phase, the vehicle must adhere to a rigorous set of path constraints, NFZ restrictions, and terminal conditions. The terminal constraints are expressed as

$$\begin{cases}
r(s=0) = r_f \\
v(s=0) = v_f \\
\phi(s=0) = \phi_f \\
\theta(s=0) = \theta_f
\end{cases} \quad (9)$$

The parameters r_f , v_f , ϕ_f , and θ_f represent the target terminal altitude, terminal velocity, latitude, and longitude, respectively. To ensure flight safety and structural integrity, the trajectory-planning problem is subject to path constraints, including heat flux $\dot{Q}_{max}$, dynamic pressure q_{max} , and aerodynamic overload n_{max} limits. In addition, the physical feasibility of the actuators is ensured by constraining the rates of change in the angle of attack $\dot{\alpha}$, bank angle $\dot{\sigma}$, and wing leading-edge sweep angle $\dot{\Lambda}$. The inequality constraints are as follows.

$$\begin{cases}
\dot{\alpha}_{min} \leq \dot{\alpha} \leq \dot{\alpha}_{max} \\
\dot{\sigma}_{min} \leq \dot{\sigma} \leq \dot{\sigma}_{max} \\
\dot{\Lambda}_{min} \leq \dot{\Lambda} \leq \dot{\Lambda}_{max}
\end{cases} \quad (10)$$

The terms $\dot{\alpha}_{min}$ and $\dot{\alpha}_{max}$ denote the lower and upper bounds of the rate of change in the angle of attack, respectively. Similarly, $\dot{\sigma}_{min}$ and $\dot{\sigma}_{max}$ represent the constraints on the bank angle rate. $\dot{\Lambda}_{min}$ and $\dot{\Lambda}_{max}$ represent the minimum and maximum rates of change in the wing leading-edge sweep angle, respectively. In addition to the aforementioned actuator constraints, vehicles are subject to strict path constraints. The expressions for heat flux, dynamic pressure, and aerodynamic overload are as follows:

$$\begin{cases} \dot{Q} = k_q \rho^{0.5} (v \cdot v_c)^{3.15} \leq \dot{Q}_{max} \\ q = 0.5 \rho (v \cdot v_c)^2 \leq q_{max} \\ n = \sqrt{L^2 + D^2} r^2 \leq n_{max} \end{cases} \quad (11)$$

The scalars $\dot{Q}_{max}$, q_{max} , and n_{max} denote the upper bounds of heat flux density, dynamic pressure, and aerodynamic overload, respectively, with corresponding values of 1.5×10^5 W/m², 1.9×10^5 kPa, and 10 g. k_q is the aerodynamic heating coefficient. In this work, the NFZ is modeled as a circular area. To ensure mission safety, the trajectory is constrained to remain strictly exterior to this area, which is formulated by the following:

$$NFZ_i: (\theta - \theta_c)^2 + (\phi - \phi_c)^2 \geq d^2 \quad (12)$$

The variables ϕ_c and θ_c denote the latitude and longitude coordinates of the NFZ center. The parameter d represents the radius of the NFZ, while the index $i = 1, \dots, n$, is the total quantity of NFZs.

2.3. Convexification and Discretization of Dynamic Mode

The optimization problem in Section 2.2 is nonlinear, nonconvex, and difficult to solve directly; therefore, its convexification is indispensable. The pseudospectral method adopts a global polynomial interpolation. Compared with traditional equidistant trapezoidal discretization, the pseudospectral method achieves higher accuracy. Thus, the Radau pseudospectral discretization strategy is adopted in this study to obtain higher discretization accuracy.

2.3.1. Linearization of Dynamics Equation

To facilitate subsequent convexification, dimensionless time is introduced as an additional state variable and incorporated into the dynamic equations. The optimization is simplified by neglecting the influence of the angular velocity of the Earth. Under these assumptions, the augmented differential equations are rewritten in the following affine form:

$$\mathbf{F}(x) = \begin{bmatrix} r \tan \gamma \\ \sin \psi / \cos \phi \\ \cos \psi \\ -rD/V \cos \gamma - \tan \gamma / rV \\ rL \cos \sigma / V^2 \cos \gamma - 1/r^2 V + 1, \\ rL \sin \sigma / V^2 \cos^2 \gamma + \sin \psi \tan \phi \\ 0 \\ 0 \\ 0 \\ r/V \cos \gamma \end{bmatrix} \quad (13)$$

$$\mathbf{B}(x) = [0, 0, 0, 0, 0, 0, r/V \cos \gamma, r/V \cos \gamma, r/V \cos \gamma, 0]^T$$

The augmented differential state equations are then linearized at the reference trajectory $\mathbf{x}^k = (x^k, u^k)$ using the first-order Taylor expansion:

$$\dot{\mathbf{x}} = \mathbf{A}(\mathbf{x}^k) \mathbf{x} + \mathbf{B}(\mathbf{x}^k) u + \mathbf{F}(\mathbf{x}^k) + \mathbf{C}(\mathbf{x}^k) \quad (14)$$

where $\mathbf{x}^k = [r^k, \theta^k, \phi^k, V^k, \gamma^k, \psi^k, \alpha^k, \sigma^k, \Lambda^k, t^k]^T$

$$\mathbf{A}(\mathbf{x}^k) = \frac{\partial \mathbf{F}(\mathbf{x})}{\partial \mathbf{x}} \bigg|_{\mathbf{x}=\mathbf{x}^k} = \begin{bmatrix} a_{11} & 0 & 0 & 0 & a_{15} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_{23} & 0 & 0 & a_{26} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_{36} & 0 & 0 & 0 & 0 \\ a_{41} & 0 & 0 & a_{44} & a_{45} & 0 & a_{47} & 0 & a_{49} & 0 \\ a_{51} & 0 & 0 & a_{54} & a_{55} & 0 & a_{57} & a_{58} & a_{59} & 0 \\ a_{61} & 0 & a_{63} & 0 & a_{65} & a_{66} & a_{67} & a_{68} & a_{69} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ a_{101} & 0 & 0 & a_{104} & a_{105} & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (15)$$

$$\mathbf{C}(\mathbf{x}^*) = -(\mathbf{A}(\mathbf{x}^*)\mathbf{x}^* + \mathbf{B}(\mathbf{x}^*)\mathbf{u}^*) \quad (16)$$

In Equation (15), the expressions for the elements a_{ij} ($i, j = 1, 2, \dots, 10$) in matrix $\mathbf{A}(\mathbf{x})$ are as follows:

$$\begin{cases} a_{11} = \tan \gamma, a_{15} = \frac{r}{\cos^2 \gamma}, a_{23} = \frac{\sin \gamma \tan \phi}{\cos \phi}, a_{26} = \frac{\cos \gamma}{\cos \phi} \\ a_{36} = -\sin \gamma, a_{41} = -\frac{D+rD_r}{V \cos \gamma} + \frac{\tan \gamma}{r^2 V}, a_{44} = -\frac{rD_r}{V \cos \gamma} + \frac{rD}{V^2 \cos \gamma} + \frac{\tan \gamma}{rV^2} \\ a_{45} = \frac{rD \tan \gamma}{V \cos \gamma} - \frac{1}{rV \cos^2 \gamma}, a_{47} = \frac{-r}{V \cos \gamma} \frac{\partial D}{\partial \alpha}, a_{49} = \frac{-r}{V \cos \gamma} \frac{\partial D}{\partial \Lambda} \\ a_{51} = \frac{L \cos \sigma + rL_r \cos \sigma}{V^2 \cos \gamma} + \frac{1}{r^2 V}, a_{54} = \frac{2}{rV^3}, a_{55} = \frac{rL \cos \sigma}{V^2 \cos \gamma} \tan \gamma \\ a_{57} = \frac{r \cos \sigma}{V^2 \cos \gamma} \frac{\partial L}{\partial \alpha}, a_{58} = -\frac{rL \sin \sigma}{V^2 \cos \gamma}, a_{59} = \frac{r \cos \sigma}{V^2 \cos \gamma} \frac{\partial L}{\partial \Lambda} \\ a_{61} = \frac{L \sin \sigma + rL_r \sin \sigma}{V^2 \cos^2 \gamma}, a_{63} = \frac{\sin \gamma}{\cos^2 \phi}, a_{65} = \frac{2 \tan \gamma rL \sin \sigma}{V^2 \cos \gamma} \\ a_{66} = \cos \gamma \tan \phi, a_{67} = \frac{r \sin \sigma}{V^2 \cos^2 \gamma} \frac{\partial L}{\partial \alpha}, a_{68} = \frac{rL \sin \sigma}{V^2 \cos^2 \gamma}, a_{69} = \frac{r \sin \sigma}{V^2 \cos^2 \gamma} \frac{\partial L}{\partial \Lambda} \\ a_{101} = \frac{1}{V \cos \gamma}, a_{104} = \frac{r}{V^2 \cos \gamma}, a_{105} = \frac{r \tan \gamma}{V \cos \gamma} \end{cases} \quad (17)$$

$$D_r = \frac{-DR_0}{H}, D_v = \frac{R_0 \rho V A_{ref} C_D}{m}$$

$$L_r = \frac{-LR_0}{H}, L_v = \frac{R_0 \rho V A_{ref} C_L}{m}$$

After linearization, the differential dynamics equations are discretized using the Radau pseudospectral method, and the constraints are formulated as follows:

$$\sum_{i=0}^K \mathbf{D}_{ki} \mathbf{x} = \frac{S_f - S_0}{2} [\mathbf{A}(\mathbf{x}^k) \mathbf{x} + \mathbf{B}(\mathbf{x}^k) \mathbf{u} + \mathbf{F}(\mathbf{x}^k) + \mathbf{C}(\mathbf{x}^k)] \quad (18)$$

2.3.2. Linearization of Path Constraints

The path constraints on the heat flux density $\dot{Q}$, dynamic pressure q , and aerodynamic overload n are nonlinear. These constraints are integrated into the convex optimization framework by linearizing them into linear inequality constraints via a first-order Taylor expansion. The linearized constraints are expressed as

$$\begin{cases} f_{\dot{Q}}(r^k, V^k) + \frac{\partial f_{\dot{Q}}}{\partial r}(r - r^k) + \frac{\partial f_{\dot{Q}}}{\partial V}(V - V^k) \leq \dot{Q}_{\max} \\ f_q(r^k, V^k) + \frac{\partial f_q}{\partial r}(r - r^k) + \frac{\partial f_q}{\partial V}(V - V^k) \leq q_{\max} \\ f_n(r^k, V^k) + \frac{\partial f_n}{\partial r}(r - r^k) + \frac{\partial f_n}{\partial V}(V - V^k) \leq n_{\max} \end{cases} \quad (19)$$

where $f_{\dot{Q}}(r^k, V^k) \triangleq \dot{Q}(r^k, V^k)$, $f_q(r^k, V^k) \triangleq q(r^k, V^k)$, and $f_n(r^k, V^k) \triangleq n(r^k, V^k)$ denote the constraint functions of heat flux density, dynamic pressure, and aerodynamic overload,

respectively. To support the gradient-based optimization, the partial derivative expressions are given as follows:

$$\begin{cases} \frac{\partial f_{\phi}}{\partial r} = -\frac{R_0 k_{\phi} (V_c V)^{3.15} \sqrt{\rho}}{2H}, & \frac{\partial f_{\phi}}{\partial V} = 3.15 k_{\phi} V_c (V_c V)^{2.15} \sqrt{\rho} \\ \frac{\partial f_q}{\partial r} = -\frac{R_0 \rho (V_c V)^2}{2H}, & \frac{\partial f_q}{\partial V} = V_c V \rho \\ \frac{\partial f_n}{\partial r} = -\frac{R_0^2 A_{ref} \sqrt{C_L^2 + C_D^2} \rho V^2}{2mH}, & \frac{\partial f_n}{\partial V} = \frac{R_0 A_{ref} \sqrt{C_L^2 + C_D^2} \rho V}{m} \end{cases} \quad (20)$$

2.3.3. Linearization of NFZ Constraints

The NFZ geometric constraints are linearized using a first-order Taylor expansion, which is expressed as

$$2(\theta^k - \theta_c)\theta + 2(\phi^k - \phi_c)\phi \geq d^2 + \bar{d} \quad (21)$$

where θ^k and ϕ^k denote the longitude and latitude, respectively, at the k -th iteration of the sequential optimization:

$$\bar{d} = -(\theta^k - \theta_c)^2 - (\phi^k - \phi_c)^2 + 2(\theta^k - \theta_c)\theta^k + 2(\phi^k - \phi_c)\phi^k \quad (22)$$

2.3.4. Objective Function

To ensure accurate terminal guidance, the cost function is designed to minimize the terminal Euclidean error to the target landing point. The corresponding objective function is given by

$$J = |r(\tau_K) - r_f| + |(\theta(\tau_K) - \theta_f) + (\phi(\tau_K) - \phi_f)| \quad (23)$$

where $(r(\tau_K), \theta(\tau_K), \phi(\tau_K))$ denotes the terminal position obtained by the optimization algorithm and (r_f, θ_f, ϕ_f) represents the coordinates of the target landing point. A compensation term $\varpi \in \mathbb{R}^{8 \times (N)}$ is introduced to compensate for the linearization error introduced during the linearization process. Thus, the augmented differential kinematic constraints with the compensation term are expressed as:

$$\sum_{i=0}^K \mathbf{D}_i \mathbf{x} = \frac{S_f - S_0}{2} [\mathbf{A}(\mathbf{x}^k) \mathbf{x} + \mathbf{B}(\mathbf{x}^k) \mathbf{u} + \mathbf{F}(\mathbf{x}^k) + \mathbf{C}(\mathbf{x}^k) + \varpi] \quad (24)$$

In this framework, the compensation parameter $\varpi \in \mathbb{R}^{8 \times (N)}$ is an unconstrained variable. To regulate the magnitude of this virtual control input and ensure the convergence of the algorithm, specific penalty and regularization terms are integrated into the global objective function:

$$J = p_1 |(\theta(\tau_n) - \theta_f) + (\phi(\tau_n) - \phi_f)| + p_2 |r(\tau_n) - r_f| + w \|\varpi\| \quad (25)$$

where p_1 and p_2 denote the regularization coefficients, and w represents the penalty coefficient. Fixed trust-region constraints are imposed on the state variations. These constraints ensure the convergence stability of the sequential convex optimization algorithm and accelerate the convergence to the optimal solution of the original nonlinear problem. These are defined as follows:

$$\|\mathbf{x}_i - \mathbf{x}_i^k\|_2 \leq \delta_i, \quad i = 1, 2, \dots, 8 \quad (26)$$

where δ is the radius of the trust-region constraints.

2.4. Fast Trajectory Optimization Based on DNN

2.4.1. Optimization Scheme Based on DNN

Owing to their universal approximation capability, DNNs efficiently capture complex nonlinear mappings from the input state vector to the optimal control output. After sufficient offline training, the network serves as an embedded controller in the onboard guidance system to generate guidance commands in real time. Because both data sampling and network training are implemented offline, the presented DNN-based approach significantly reduces online computational latency while maintaining high control precision.

Figure 3 shows a schematic of the DNN-based optimal trajectory generation strategy tailored for a morphing gliding reentry vehicle. Within a predefined flight mission profile, the required optimal state-control dataset is generated using the direct method. Following a rigorous dataset preprocessing step, the data are fed into the DNN for offline training. The model learns complex nonlinear mapping in the dataset to predict the optimal control actions of the vehicle based on the flight state feedback. In practical operation, the trained DNN estimates the corresponding optimal control action u_i (i.e., angle of attack α_i , bank angle σ_i , and sweep angle Λ_i) by integrating the current input flight state x_i and the NFZ range (R, x, y) .

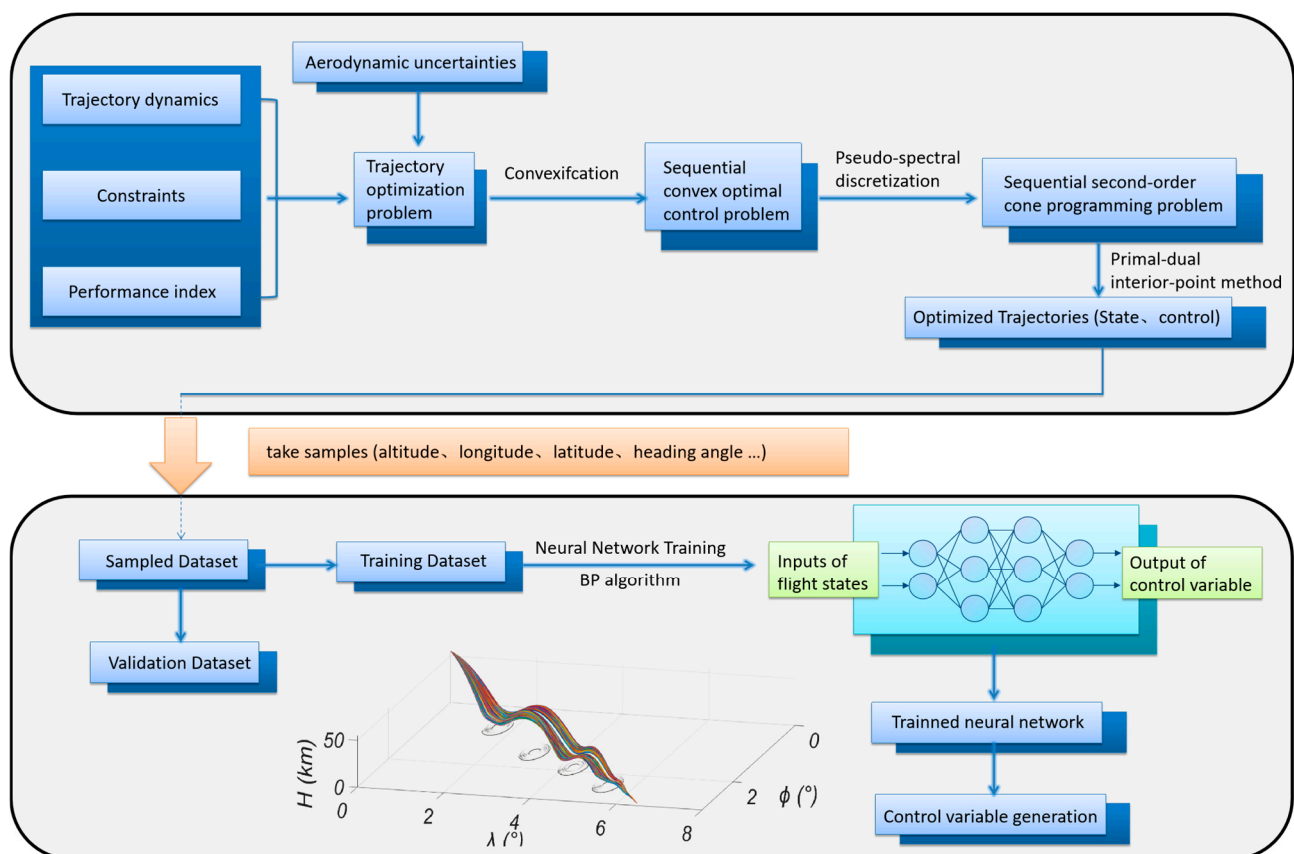


Figure 3. Optimal trajectory generation method for morphing gliding reentry vehicles based on DNN.

These DNN-based guidance controllers involve two operational paradigms. First, the trained controller can directly generate optimal guidance commands in real time (usually on the millisecond scale) by processing the current flight state feedback. The other system can rapidly generate a complete reference reentry trajectory (typically within seconds) based on specific initial entry conditions. A detailed schematic of the DNN architecture, which explicitly characterizes its inputs, outputs, and topological structure, is shown in Figure 4.

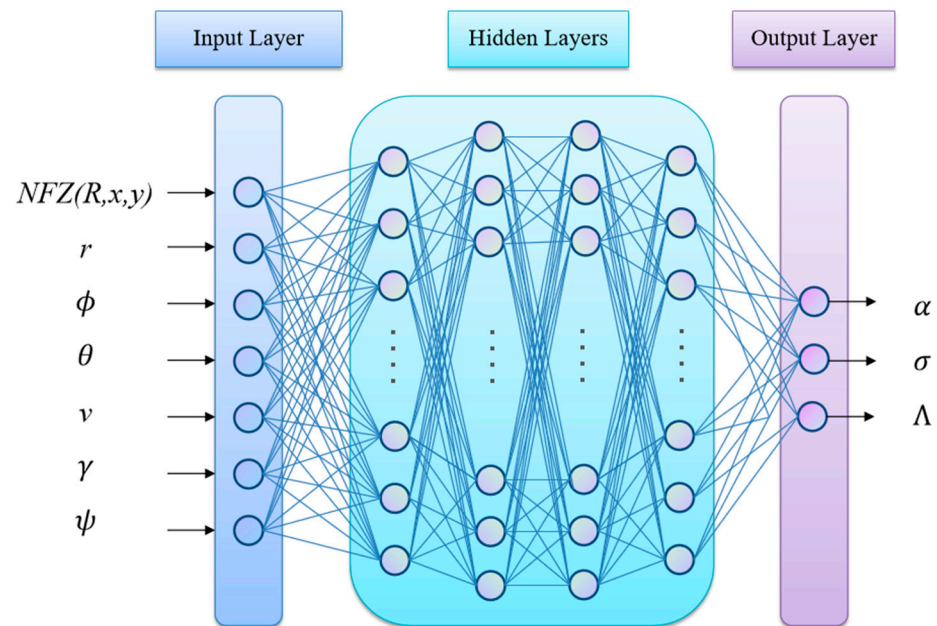


Figure 4. Schematic diagram of the DNN structure.

2.4.2. Design and Optimization of DNN

This subsection describes the design of the DNN model and methodology for hyperparameter optimization. A fully connected feedforward neural network comprising a dedicated input layer, series of hidden layers, and final output layer is employed. The network input consists of the state variables of the optimal trajectory, which is composed of $x = [r, \theta, \phi, v, \gamma, \psi, \alpha, \sigma, \Lambda]$. During the training process, the DNN learns the intrinsic nonlinear mapping between the state vector and the optimal control actions, $u = [\alpha, \sigma, \Lambda]$. The output o_{ij} corresponding to the (i) unit in the (j) layer of the network is formulated as

$$o_{ij} = g(w_{ij}o_{i-1} + b_{ij}) \quad (27)$$

where w_{ij} designates the weight vector and b_{ij} is the bias vector associated with this unit. Additionally, $O_{(j-1)}$ represents the aggregate output vector derived from the preceding layer and φ denotes the nonlinear activation function characterizing the unit.

To improve the performance of the optimization algorithm, accelerate the convergence speed, and reduce the computational cost, the altitude and velocity in the input vector are normalized as follows:

$$r_{pre} = \frac{r}{R_e}, v_{pre} = \frac{v}{v(t_0)} \quad (28)$$

where R_e denotes the radius of the Earth; $v(t_0)$ is the initial reentry flight velocity of the vehicle; and r_{pre} and v_{pre} represent the altitude and velocity values, respectively, after preprocessing.

Within the deep learning framework, the loss function L quantifies the divergence between the predicted outputs of the model and actual ground truth labels. This function guides and optimizes the model training, aiming to minimize the loss function to improve the accuracy of the predictions. Owing to its superior robustness against outliers, the mean absolute error (MAE) is selected as the specific loss function. Its definition is presented in Equation (29):

$$L = \frac{1}{N} \sum_{i=1}^N |Net(x_i) - y_i| \quad (29)$$

where N denotes the total number of training samples; $Net(x_i)$ represents the output predicted by the neural network with the input state vector x_i ; and y_i denotes the optimization data obtained using the direct method.

The first hyperparameter of a DNN is the network structure, which includes the number of hidden layers and neurons in each layer. An overly simple structure limits the ability of the network to learn nonlinear input–output mapping, resulting in underfitting. Conversely, an excessive number of layers or neurons increases the computational cost and risk of overfitting, where the model performs well on the training set but poorly on the unseen validation data. Accordingly, the final network adopted in this study comprises eight layers with 512 neurons per layer. The hidden-layer activation function is ReLU, and the learning-rate scheduler employs a patience-based strategy with a patience value of 10 and a minimum learning rate of 10^{-6} . The model weights are updated iteratively using mini-batch gradient descent. Thus, batch size and initial learning rate are another set of essential hyperparameters. The batch size defines the number of samples used in each training iteration and is chosen to balance computational efficiency and gradient stability. Large batch sizes may slow convergence and cause large fluctuations in the validation set, whereas extremely small sizes significantly increase the training time. Therefore, a batch size of 256 is used. The learning rate is the updating step for the network parameters. A low learning rate leads to slow convergence and a long training duration. In contrast, a high learning rate may cause unstable training or convergence to inferior solutions. Therefore, an initial learning rate of 10^{-4} is adopted.

All numerical experiments were conducted on a PC with an Intel Core i9-14900HX processor, NVIDIA GeForce RTX 4070 GPU, 32 GB DDR5 RAM, and 1 TB SSD, running the 64-bit Windows 11 operating system. The required software environments including MATLAB 2020 integrated with the CVX toolbox and Python with the PyTorch framework. Detailed hardware specifications, software versions, and key parameter settings are fully documented to ensure the reproducibility of all results presented.

3. Results

3.1. Generation Scheme of Optimal Trajectory Cluster

Variable reentry glide vehicles often encounter dynamic NFZs that may shift or expand over time. Therefore, the DNN used as a trajectory generator is designed to adapt to a wide range of NFZ conditions. The simulation environment includes mobile NFZs with variable ranges to validate performance. When the NFZ expands, the vehicle switches from Trajectory 1 (orange) to Trajectory 2 (green), as shown in Figure 5a. The spatial translation of the NFZ changes from Trajectory 2 (green) to Trajectory 3 (purple), as illustrated in Figure 5b.

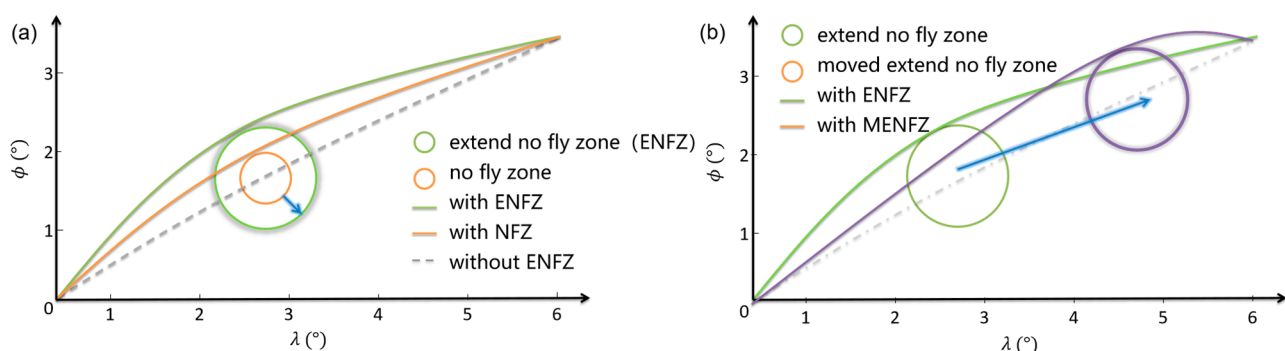


Figure 5. Generation idea of optimal trajectory cluster for variable gliding reentry vehicles based on dynamic NFZs. (a) radial expansion of the no-fly zone at a fixed location, forming ENFZ; (b) spatial relocation of the no-fly zone, forming MENFZ.

To ensure that the dataset covers the broadest possible range of spatial configurations, 2500 NFZs with different positions and sizes were set up in the experiment. The central positions and radius ranges of each NFZ are listed in Table 2.

Table 2. Range of initial state values for NFZ.

Center of No-Fly Zone	Range of the No-Fly Zone (km)
[1.6, 0.7]	[11.1, 33.3]
[3.0, 1.5]	[11.1, 33.3]
[4.2, 1.9]	[11.1, 33.3]
[5.4, 2.4]	[11.1, 33.3]

Figure 6 presents the optimal trajectory clusters for variable gliding reentry vehicles under different NFZ configurations. These clusters cover all scenarios where the NFZ shifts among the four NFZ centers and its range varies from 11.1 km to 33.3 km.

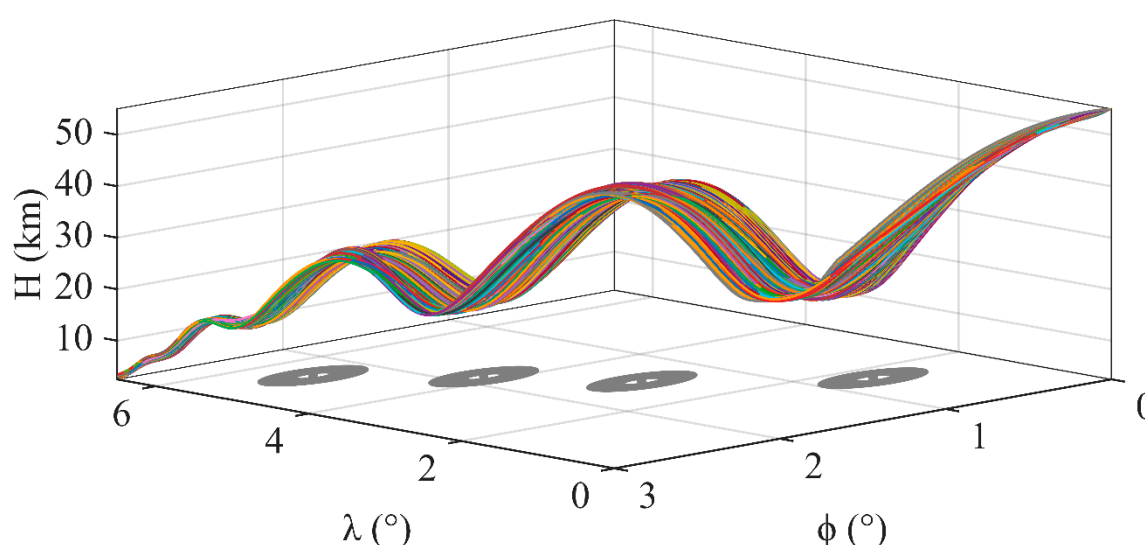


Figure 6. Optimal trajectory clusters of variable gliding reentry vehicles based on different NFZs.

3.2. Parameter Settings of Radau Pseudospectral Convex Optimization

In this study, the Radau pseudospectral convex optimization method was employed to generate an optimal trajectory library. The number of Gaussian collocation points per trajectory is a key influencing factor in numerical trajectory computation. Insufficient collocation points degrade the quality of trajectory optimization, whereas an excessive number substantially increases the time cost for both trajectory generation and subsequent network training. For the specific scenario investigated in this study, a 2 s interval between collocation points ensures sufficient trajectory accuracy. The reentry flight duration is about 700 s. After balancing the generation time and optimization quality, each trajectory contains approximately 250 discrete optimal state-action sets on average. This setup generates reliable trajectories within acceptable time limits. Although using more Gaussian collocation points is feasible, the time cost increases considerably. The currently selected discretization points guarantee trajectory reliability, and the 2500 samples are divided into 2000 training, 300 validation, and 200 test trajectories.

As shown in Table 3, the DNN model is evaluated on 200 independent test trajectories. Three error metrics are adopted to assess the prediction performance. RMSE reflects the overall prediction error and is sensitive to large local deviations. MAE denotes the average prediction bias. Mean $\pm$ std and 95% confidence intervals (CI) are calculated from all valid trajectories to quantify error dispersion and statistical reliability. The model

yields stable predictions for angle of attack, bank angle, and sweep angle. The corresponding RMSE values are 0.239, 0.107, and 0.462, with 95% CIs of [0.215, 0.264], [0.098, 0.117], and [0.385, 0.539], respectively. The relatively higher error of sweep angle is mainly caused by a small number of extreme trajectories. For most test cases, the prediction accuracy remains acceptable.

Table 3. Statistical confidence bounds of test-set metrics.

Control Variable	RMSE	95% CI	MAE
Angle of attack	0.239 ± 0.168	[0.215, 0.264]	0.099 ± 0.052
Bank angle	0.107 ± 0.064	[0.098, 0.117]	0.073 ± 0.034
Wing sweep angle	0.462 ± 0.530	[0.385, 0.539]	0.319 ± 0.387

In the simulation, the initial downrange, initial reentry altitude, and initial velocity of the reentry mission are 796.6 km, 55 km, and 2150 m/s, respectively. The bank angle rate is limited to 10°/s. Table 4 lists the bank angle bounds and path constraint limits.

Table 4. Range of initial state values for trajectories.

Parameter	Value Range
Initial altitude $r(t_0)$	55 km
Initial longitude $\theta(t_0)$	0°
Initial latitude $\phi(t_0)$	0°
Initial velocity $V(t_0)$	2150 m/s
Initial flight path angle $\gamma(t_0)$	66°
Initial Azimuth $\psi(t_0)$	−0.5°

For the two examples investigated in this section, the parameter settings of the trust region size and convergence criterion are given by Equations (30) and (31), respectively, as follows:

$$\delta = \left[\frac{10000}{R_0}, \frac{20\pi}{180}, \frac{20\pi}{180}, \frac{500}{V_0}, \frac{20\pi}{180}, \frac{20\pi}{180}, \frac{20\pi}{180} \right]^T \quad (30)$$

$$\varepsilon = \left[\frac{100}{R_0}, \frac{0.05\pi}{180}, \frac{0.05\pi}{180}, \frac{1}{V_0}, \frac{0.05\pi}{180}, \frac{0.05\pi}{180}, \frac{1\pi}{180} \right]^T \quad (31)$$

To verify the rationality and robustness of the parameter configuration in the proposed SCP algorithm, single-factor sensitivity analysis is performed on parameters ρ , λ_1 , and λ_2 . The default values are set as $\rho_0 = 10^{-2}$, $\lambda_1 = 10^2$, and $\lambda_2 = 10^1$. In each test, only one parameter is modified, while the rest remain at default settings. Several key indicators are adopted for quantitative evaluation, including solution success rate, average iteration number, average computation time, and maximum path constraint violation. Test results show that all parameter groups achieve a 100% success rate in solving convex subproblems. The parameter ρ primarily governs algorithm convergence. A small ρ value increases the reliance on virtual controls, while an excessively large value raises the difficulty of numerical computation. λ_1 mainly determines terminal position accuracy, and λ_2 has limited impacts within the test range. The default parameters adopted in this work effectively balance terminal accuracy, path constraint satisfaction, virtual control suppression, and computational efficiency.

For the sequential convex programming (SCP) method, ρ is the most sensitive parameter for convergence performance. A smaller ρ relaxes dynamic consistency constraints, whereas a larger ρ complicates numerical solving. The default ρ_0 lies in a reasonable range, which well balances virtual control suppression, terminal precision and

computational cost. λ_1 barely affects solution feasibility but significantly influences terminal position accuracy. A larger λ_1 can reduce terminal errors, accompanied by a slight increase in computational burden. Within the test scope, λ_2 exerts negligible influences on SCP solution stability, which verifies the good parameter robustness of the proposed algorithm, as shown in Table 5.

Table 5. SCP optimization performance with varying parameter magnitudes.

Parameter Perturbation	Success Rate of Solution	Average Iterations	Average Time	Terminal Error Mean Range
$Q = 0.1Q_0 \sim 10Q_0$	100%	2.5–4.3	29.9–59.9 s	$3.99 \times 10^{-7} \sim 4.21 \times 10^{-3}$
$\lambda_1 = 0.1\lambda_{10} \sim 10\lambda_{10}$	100%	3.3–4.0	38.7–52.9 s	$1.40 \times 10^{-3} \sim 4.30 \times 10^{-3}$
$\lambda_2 = 0.1\lambda_{20} \sim 10\lambda_{20}$	100%	3.6–3.9	47.5–52.5 s	$2.91 \times 10^{-3} \sim 3.97 \times 10^{-3}$

To further explore the convergence characteristics and anti-disturbance capability of the SCP algorithm, this paper investigates its robustness against perturbed initial reference trajectories. Four trajectory levels with different perturbation magnitudes are established, including nominal, small, moderate, and large perturbation trajectories. All perturbations are imposed only on the reference trajectory for SCP linearization, while the practical boundary conditions and path constraints are fixed during all tests.

The test results shown in Table 6 verify that the SCP algorithm converges stably when the reference trajectory is close to the feasible domain. All test cases converge successfully under nominal trajectories, achieving a 100% core state convergence rate. For small perturbation scenarios, the solution success rate decreases to 65%. Most convergent cases reach the maximum iteration number, but their terminal errors and path constraints still meet the requirements. Only 5% of test cases converge under moderate perturbations, with a sharp rise in virtual control norm. No effective solutions can be obtained when large perturbations are applied. These results confirm the local convergence feature of the SCP algorithm. A reference trajectory close to the feasible domain guarantees stable optimization performance. In contrast, trajectories far from the feasible domain cause severe convexification errors. These errors reduce solution accuracy and success rate, raise iteration costs and virtual control amplitude. In extreme perturbation scenarios, the convex subproblems fail to generate feasible solutions.

Table 6. Quantitative performance under different initial reference trajectory perturbations.

Initial Reference	Solver Success	Mean SCP Iterations	Mean Terminal Error
Nominal	100%	1.2	6.5×10^{-5}
Small perturbation	65%	5.0	5.5×10^{-5}
Medium perturbation	5%	5.0	6.2×10^{-5}
Large perturbation	0%	—	—

3.3. Generation Results of Training and Testing Data

Figure 7a shows 2500 optimal trajectories under random NFZs. The constraints of the heat flux, dynamic pressure, and overload are shown in Figure 8, with the maximum heat flux appearing at the start of gliding. To satisfy the heat-flux constraint, the vehicle adjusts the wing sweep angle to generate lift for trajectory pull-up, thereby reducing energy loss and severe aerothermal environments from high-angle-of-attack flights. Figure 7c illustrates the variation in the bank angle. For a safe flight and large-scale lateral maneuvering, the bank angle is adjusted several times during the mission. Figure 9 shows the variation in the wing sweep angle. The deformation results show that the sweep angle coordinates with the attitude angle during large-scale lateral maneuvers. The angle of attack remains

within a narrow range throughout the flight, which benefits gliding range extension and thermal protection design.

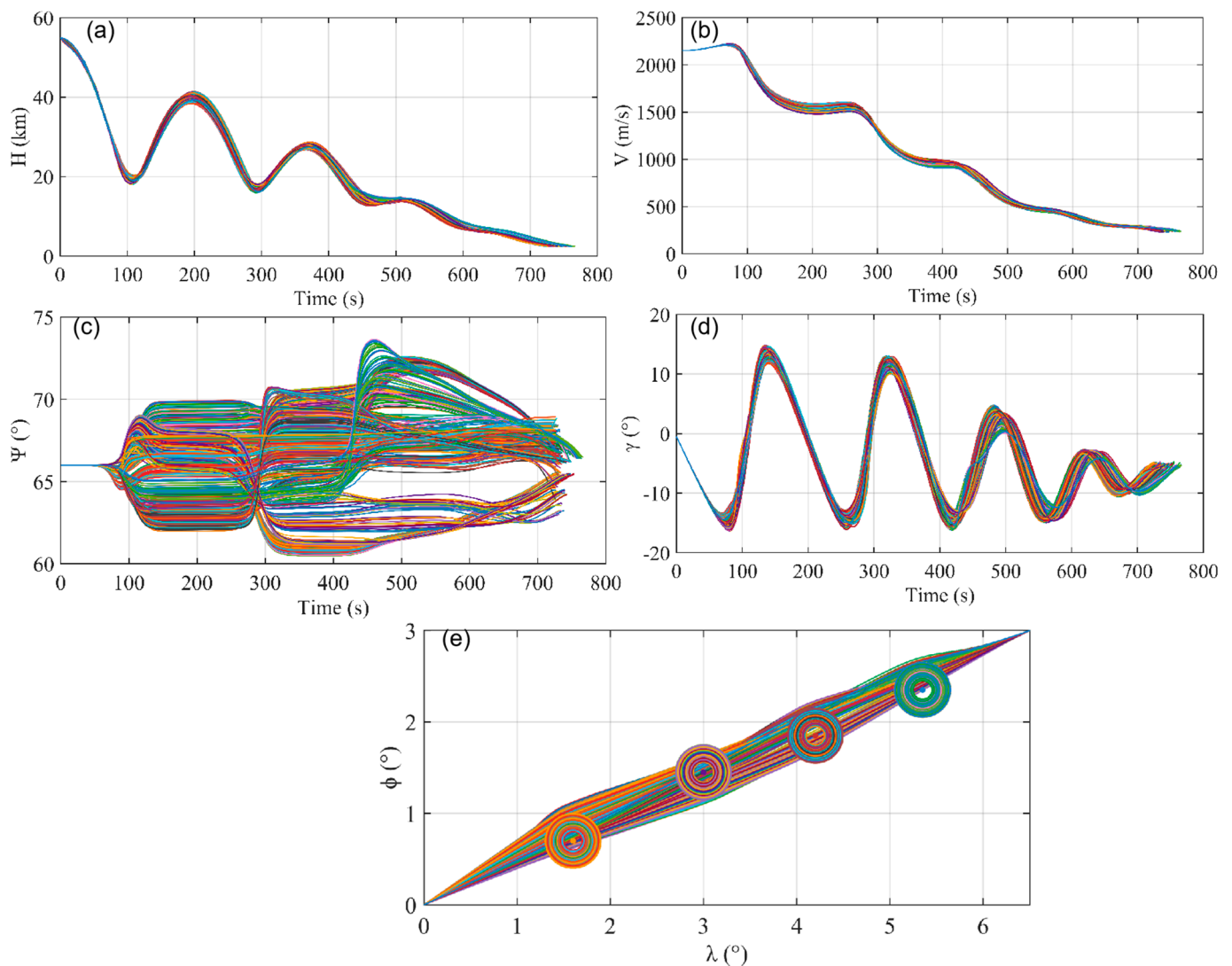


Figure 7. Optimal trajectory with random NFZs. (a) Altitude–time curve, (b) Velocity–time curve; (c) Flight path–time curve; (d) Azimuth–time curve; (e) Longitude–latitude curve.

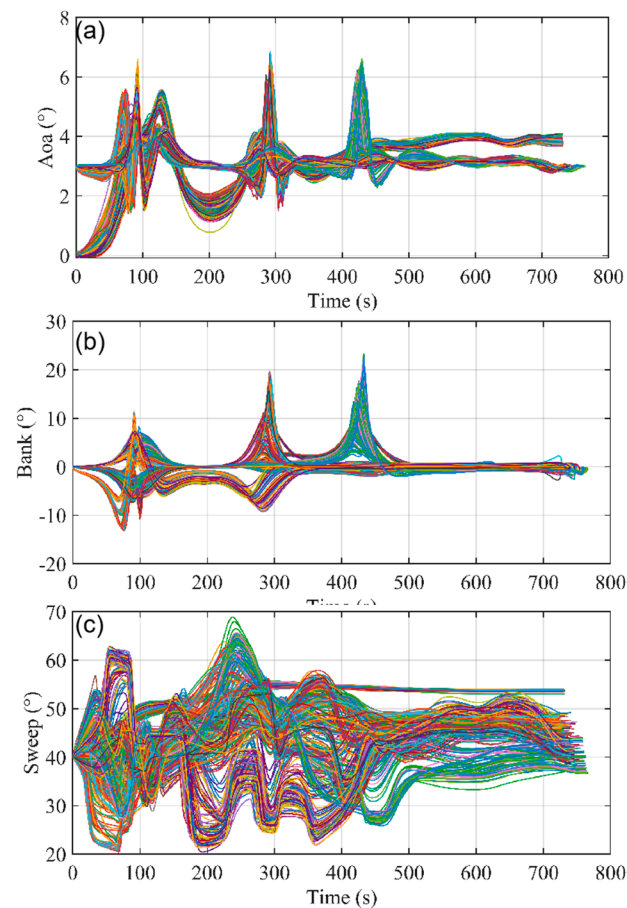


Figure 8. Optimal trajectory with random NFZs. (a) Angle of attack–time curve; (b) Bank angle–time curve; (c) Sweep angle–time curve.

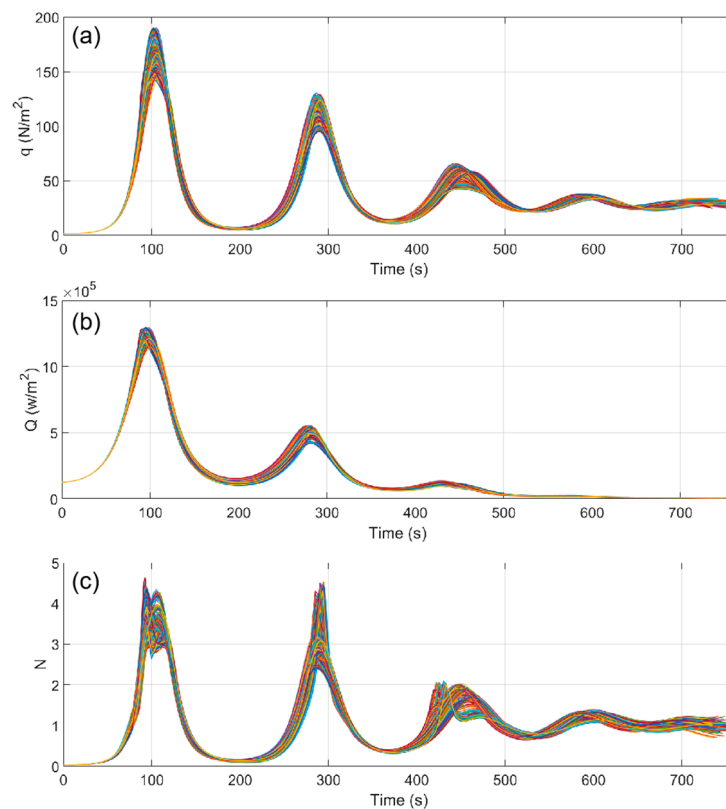


Figure 9. Optimal trajectory with random NFZs. (a) Dynamic pressure–time curve; (b) Heat flux density–time curve; (c) Overload–time curve.

3.4. Neural Network Prediction Results

To further verify the performance of the proposed DNN, the trained model uses the current flight state vector to generate predictions and provides control inputs (angle of attack α , bank angle σ , and sweep angle Λ) at the corresponding time. Figure 10 compares a trajectory randomly selected from the network validation dataset with that predicted by the neural network. The results demonstrate that the proposed neural network accurately predicts trajectories based on the current flight state.

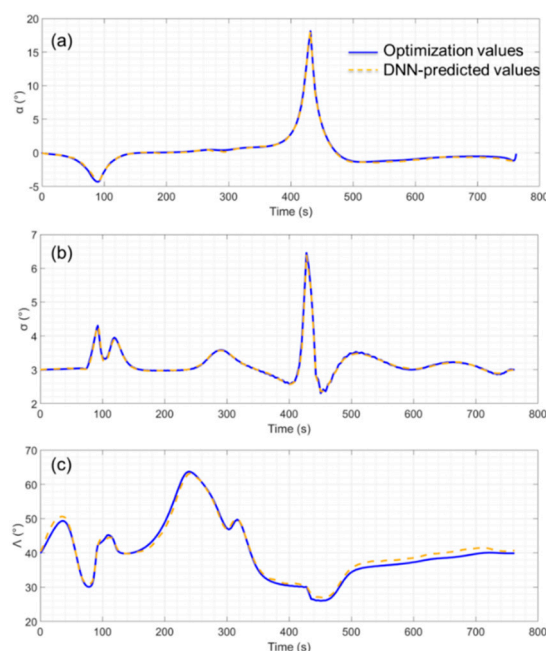


Figure 10. The comparison between a trajectory selected from the network validation dataset and the trajectory predicted by the neural network. (a) Time history of bank angle; (b) Time history of angle of attack; (c) Time history of sweep angle.

Figure 11 shows the absolute error distribution of the optimal actions predicted by the neural network. Analysis reveals that the angle of attack α has an average relative error of 2.25% and an absolute error of 0.912°; the bank angle σ has a relative error of 0.61% and an absolute error of 0.02°; the wing sweep angle Λ has a relative error of 2.25% and an absolute error of 0.912°. The designed DNN model demonstrates high accuracy in predicting optimal actions, thereby facilitating precise real-time guidance (only 0.4 ms is required to predict three groups of control variables).

To verify the effectiveness of the proposed algorithm, 200 trajectory cases are tested via Monte Carlo experiments, as presented in Figure 12. Diverse initial flight states are employed to produce validation trajectories, whose key parameters are summarized in Table 7. The average terminal position error, altitude error, and velocity error are measured as 12.85 km, 1.02 km, and 70.87 m/s correspondingly. It is shown that the DNN method delivers satisfactory terminal tracking accuracy and satisfies the requirements imposed by strong path constraints.

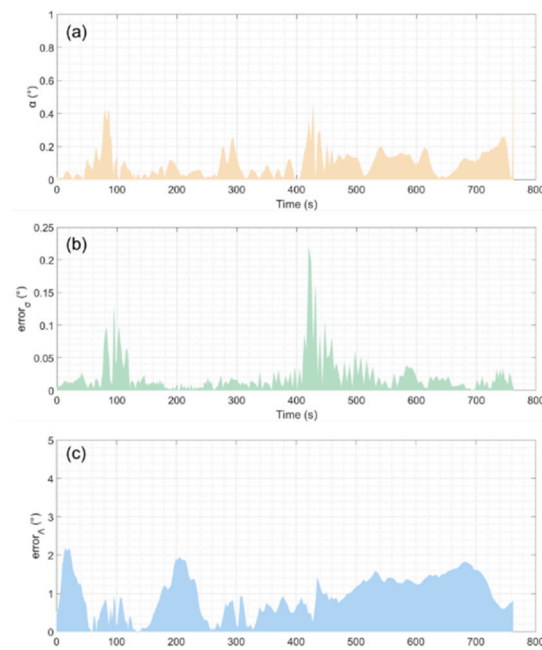


Figure 11. The absolute errors of the optimal actions predicted by the neural network. (a) absolute error of bank angle; (b) absolute error of angle of attack; (c) absolute error of sweep angle.

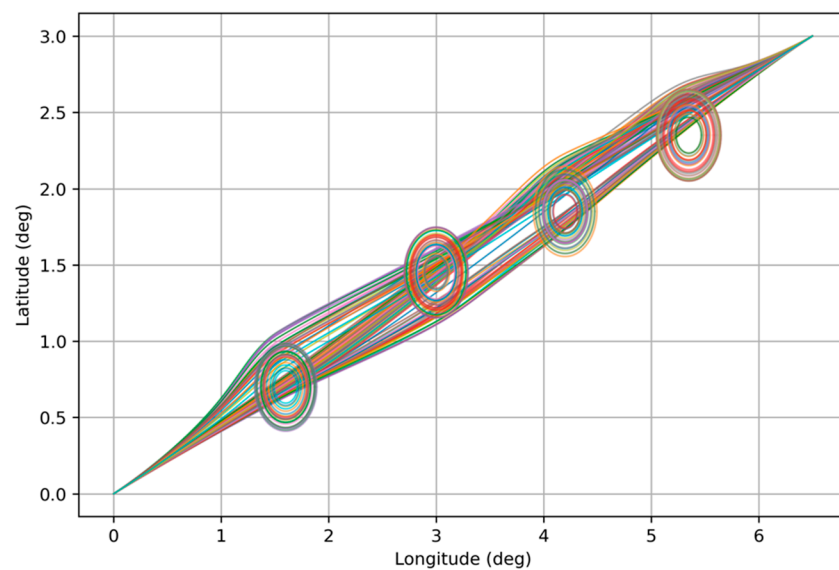


Figure 12. Statistical Results of Trajectories from Monte Carlo Experiments.

Table 7. Range of state values for Monte Carlo experiments.

Parameter	Value Range
Initial altitude r	55 ± 0.5 km
No-fly zone radius range	The no-fly zone radius is randomly chosen between 0.10° and 0.30° .
Initial velocity V	2150 ± 20 m/s
Initial flight path angle $\gamma(t_0)$	$66 \pm 0.5^\circ$
Initial Azimuth $\psi(t_0)$	$-0.5 \pm 1^\circ$

4. Discussion

This study proposes an online trajectory co-optimization framework for morphing gliding reentry vehicles, which integrates deep neural networks (DNNs) with sequential

Radau pseudospectral convex optimization methods. The established framework transforms the strongly non-convex trajectory planning problem with complex flight constraints into a series of tractable convex subproblems. Specifically, DNNs are employed to construct the nonlinear mapping between flight state parameters and optimal control variables, forming a hybrid optimization paradigm that combines high-precision numerical optimization and rapid intelligent inference. This strategy enables the real-time generation of optimal flight trajectories and synchronous wing morphing strategies throughout the full flight envelope, and exhibits excellent adaptability to complex flight missions involving dynamic no-fly zone constraints and time-varying range limitations.

Despite the promising performance of the developed framework in nominal flight scenarios, several inherent limitations restrict its practical engineering application. First, the training dataset of the DNN model is limited to conventional flight envelopes and common disturbance conditions, lacking coverage of extreme working scenarios. Abrupt variations in no-fly zone boundaries and intense external disturbances will degrade the model inference accuracy, and may further lead to the failure of trajectory optimization. Perturbation simulation experiments verify the local convergence characteristic of the adopted SCP algorithm. The algorithm can achieve complete convergence when the initial reference trajectory is close to the feasible solution domain. Nevertheless, small trajectory perturbations reduce the solution success rate to 65%, even though the convergent solutions still meet the basic accuracy and constraint requirements. Severe trajectory perturbations will completely invalidate the optimization results and fail to generate feasible flight trajectories. Second, the SCP algorithm relies on linearization and convex relaxation operations to approximate the original non-convex flight constraints. Such numerical approximation operations will inevitably introduce minor systematic errors, especially under extreme boundary constraint conditions. For flight missions with extremely narrow safety margins, these approximation errors make it difficult for the framework to guarantee the strict global optimality of the optimized trajectory and morphing strategies.

Future research will focus on addressing the aforementioned limitations to further improve the generalization performance and engineering practicability of the proposed framework. First, extended datasets containing diverse extreme flight scenarios and drastic constraint variation cases will be supplemented to expand the adaptive boundary of the DNN model, so as to enhance the robustness of intelligent inference against severe disturbances and abrupt mission changes. Second, adaptive convex relaxation strategies and dynamic error correction algorithms will be developed to compensate the linearization errors of SCP, thereby improving the optimization accuracy under boundary-limited flight conditions. Third, high-fidelity flight perturbation factors including aerodynamic uncertainty, sensor measurement noise and actuator execution deviation will be incorporated to construct a simulation platform that is highly consistent with real flight environments. Furthermore, online parameter updating and incremental learning mechanisms will be explored to realize adaptive adjustment of the framework for unknown and complex mission scenarios. This work provides a solid theoretical and technical foundation for the engineering implementation of online intelligent co-optimization of trajectories and morphing strategies for morphing reentry vehicles.

Author Contributions: Formal analysis, J.H.; investigation, F.N.; resources, X.Z. (Xinyue Zhou); data curation, M.L.; writing—original draft preparation, X.Z. (Xingyu Zhu); writing—review and editing, T.W.; supervision, E.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available on request from the corresponding author due to privacy or ethical restrictions.

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

References

1. Phoenix, A.; Rogers, R.E.; Maxwell, J.R.; Goodwin, G.B. Mach five to ten morphing waverider: Control point study. *J. Aircr.* **2019**, *56*, 493–504.
2. Bowcutt, K. Hypersonic Waverider Variable Leading Edge Flaps. U.S. Patent US6634594B1, 21 October 2003.
3. Maxwell, J.R. Hypersonic waverider stream surface actuation for variable design point operation. In Proceedings of the 52nd AIAA/SAE/ASEE Joint Propulsion Conference, Salt Lake City, UT, USA, 25–27 July 2016; p. 4706.
4. Phoenix, A.; Maxwell, J.R.; Goodwin, G.B. Morphing high-temperature surfaces for shape control hypersonic waverider vehicles. In Proceedings of the Modeling, Simulation, Control & Adaptive Systems Design and Implementation, Structural Health Monitoring, Snowbird, UT, USA, 18–20 September 2017; pp. 1–5.
5. Xie, Y.; Liu, L.; Tang, G.; Zheng, W. A reentry trajectory planning approach satisfying waypoint and no-fly zone constraints. In Proceedings of the 5th IEEE International Conference on Recent Advances in Space Technologies (RAST), Istanbul, Turkey, 9–11 June 2011; pp. 241–246.
6. Ottesen, D.; Russell, R.P. Direct-to-indirect mapping for optimal low-thrust trajectories. *Astrodynamics* **2024**, *8*, 27–46.
7. Xue, S.; Ping, L. Constrained Predictor-Corrector Entry Guidance. *J. Guid. Control Dyn.* **2010**, *33*, 1273–1281.
8. Zhang, J.; Liu, K.; Fan, Y.; She, Z. A Piecewise Predictor-corrector Reentry Guidance Algorithm with No-fly Zone Avoidance. *J. Astronaut.* **2021**, *42*, 122–131.
9. Li, M.; Zhou, C.; Shao, L.; Lei, H.; Luo, C. An Improved Predictor-Corrector Guidance Algorithm for Reentry Glide Vehicle Based on Intelligent Flight Range Prediction and Adaptive Crossrange Corridor. *Int. J. Aerosp. Eng.* **2022**, *2022*, 731586.
10. Yong, E.M.; Qian, W.Q.; He, K.F. An adaptive predictor-corrector reentry guidance based on self-definition way-points. *Aerosp. Sci. Technol.* **2014**, *39*, 221–225.
11. Zhu, J.; Liu, L.; Tang, G.; Bao, W. Highly constrained optimal gliding guidance. *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.* **2015**, *229*, 2321–2335.
12. Liu, X.; Lu, P. Survey of convex optimization for aerospace applications. *Astrodynamics* **2017**, *1*, 23–40.
13. Liu, X.; Shen, Z.; Lu, P. Entry trajectory optimization by second-order cone programming. *J. Guid. Control Dyn.* **2016**, *39*, 227–241.
14. Wang, Z.; Grant, M.J. Constrained trajectory optimization for planetary entry via sequential convex programming. *J. Guid. Control Dyn.* **2017**, *40*, 2603–2615.
15. Wang, Z.; Grant, M.J. Autonomous entry guidance for hypersonic vehicles by convex optimization. *J. Spacecr. Rocket.* **2018**, *55*, 993–1006.
16. Sandberg, A.; Sands, T. Autonomous trajectory generation algorithms for spacecraft slew maneuvers. *Aerospace* **2022**, *9*, 135.
17. Rozza, G.; Sands, T. Autonomous trajectory generation comparison for de-orbiting with multiple collision avoidance. *Sensors* **2022**, *22*, 7066.
18. Wang, J.; Cui, N.; Wei, C. Rapid trajectory optimization for hypersonic entry using convex optimization and pseudospectral method. *Aircr. Eng. Aerosp. Technol.* **2019**, *91*, 669–679.
19. Liu, X.; Li, S.; Xin, M. Mars Entry Trajectory Planning with Range Discretization and Successive Convexification. *J. Guid. Control Dyn.* **2022**, *45*, 755–763. <https://doi.org/10.2514/1.G006237>.
20. Li, S.; Liu, X.; Jiang, X.-Q.; Peng, Y.-M. *Trajectory Optimization and Guidance Methods for Mars Entry*; Springer: Singapore, 2024.
21. Liu, X.; Li, S. Survey of Trajectory Optimization Methods for Mars Entry and Powered Descent. *J. Guid. Control Dyn.* <https://doi.org/10.2514/1.G009183>.
22. Wang, Z.; Zhang, H. Improved sequential convex programming algorithms for entry trajectory optimization. *J. Spacecr. Rocket.* **2020**, *57*, 1373–1386.
23. Zhou, X.; He, R.; Zhang, H.; Tang, G.; Bao, W. Sequential convex programming method using adaptive mesh refinement for entry trajectory planning problem. *J. Astronaut. Technol.* **2021**, *109*, 106374.
24. Zhou, X.; Zhang, H.B.; Xie, L.; Tang, G.J.; Bao, W.M. An improved solution method via the pole-transformation process for the maximum-endurance problem. *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.* **2020**, *234*, 1941–1956.
25. Liu, X.; Shen, Z.; Lu, P. Exact convex relaxation for optimal flight of aerodynamically controlled missiles. *IEEE Trans. Aerosp. Electron. Syst.* **2019**, *55*, 1881–1892.
26. Izzo, D.; Martens, M.; Pan, B. A Survey on Artificial Intelligence Trends in Spacecraft Guidance Dynamics and Control. *Astrodynamics* **2019**, *3*, 287–299.

27. Sanchez-Sanchez, C.; Izzo, D. Learning the Optimal State-Feedback Using Deep Networks. In Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece, 6–9 December 2016.
28. Izzo, D.; Sprague, C.; Tabor, D. Machine Learning and Evolutionary Techniques in Interplanetary Trajectory Design. *arXiv* **2018**, arXiv:1802.00180.
29. Sanchez-Sanchez, C.; Izzo, D. Real-Time Optimal Control via Deep Neural Networks: Study on Landing Problems. *J. Guid. Control Dyn.* **2018**, *41*, 1122–1135.
30. Li, H.; Chen, H.; Tan, C.; Jiang, Z.; Xu, X. Fast Trajectory Generation with a Deep Neural Network for Hypersonic Entry Flight. *Aerospace* **2023**, *10*, 931.
31. Dai, P.; Yan, B.; Huang, W.; Zhen, Y.; Wang, M.; Liu, S. Design and aerodynamic performance analysis of a variable-sweep-wing morphing waverider. *Aerosp. Sci. Technol.* **2020**, *98*, 105703.
32. Dai, P.; Yan, B.; Liu, R.; Liu, S.; Wang, M. Integrated Morphing Strategy and Trajectory Optimization of a Morphing Waverider and Its Online Implementation Based on the Neural Network. *IEEE Access* **2021**, *99*, 59383–59393.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.