

NEURAL NETWORK BASED SLIDING MODE CONTROL FOR HYPERSONIC LONGITUDINAL CONTROL

Haejun Kim[1] & Jongho Shin[1]

[1]School of Mechanical Engineering, Chungbuk National University, Cheongju, Korea

Abstract

This paper applies and evaluates a neural-network-based tuning method that adjusts the design parameters of Sliding Mode Control (SMC) online to improve simultaneous velocity V and altitude h tracking in hypersonic longitudinal dynamics. The proposed method preserves the SMC structure. It makes a neural network tuner adjust the sliding surface parameters of the velocity and altitude channels over time. The neural network tuner receives the command, current state, and sliding surface observed during control as inputs. It outputs λ and is updated online during simulation. The performance is evaluated by comparing a conventional SMC with fixed λ and the proposed neural-network-aided NN-SMC under the same flight condition. The comparison uses the time responses of the velocity and altitude tracking errors and their RMSE/MAE values. It also presents the control input RMS and control energy to analyze both tracking performance and control characteristics.

Keywords: Hypersonic vehicle, sliding mode control, neural network tuner, online parameter adaptation, tracking

1. Introduction

In hypersonic longitudinal guidance, it is important to track velocity V and altitude h at the same time and to maintain robust control performance under strong nonlinearity and model uncertainty. A hypersonic vehicle is a nonlinear system in which aerodynamics, propulsion, gravity, and attitude dynamics are strongly coupled in the high-speed flight regime. Its control performance can also change significantly because of uncertainties in mass, moment of inertia, atmospheric density, and aerodynamic coefficients [1, 2]. Sliding Mode Control (SMC) has been widely used as a robust control method in this environment. However, the control response can vary significantly depending on the selection of control parameters. In particular, when various flight conditions and dynamic model uncertainties exist, fixed control parameters may not provide consistent performance over the entire operating range [3, 4, 5].

Various studies have been conducted to overcome these limitations of SMC and to apply it to hypersonic vehicle control. Qu et al. proposed dynamic-matching adaptive SMC for hypersonic vehicles under time-varying dynamic uncertainty and control saturation [6]. In that study, the bound of the dynamic uncertainty and the degree of control saturation were estimated online. The robust coefficient and response rate of SMC were then adjusted accordingly. This improved robustness, response speed, chattering reduction, and input saturation suppression. The study shows that it is important to adjust controller parameters according to the operating condition in hypersonic vehicle SMC. However, this approach includes uncertainty bound estimation, saturation degree calculation, and coefficient adaptation laws. Therefore, additional design parameter selection and tuning may be required in practical applications.

Xia et al. proposed a neural-network-based integral sliding mode optimal flight control method for a near-space hypersonic vehicle [7]. The method combines a Radial Basis Function (RBF) neural

network, integral SMC, an auxiliary system, and Adaptive Dynamic Programming (ADP). It achieves robust optimal tracking under unknown modeling errors, external disturbances, and input saturation. This study shows that a neural network can effectively compensate for modeling errors and disturbances in hypersonic vehicle control. However, the controller can become relatively complex because it uses an ADP-based optimal controller, a critic network, and an auxiliary system together.

Meanwhile, Fatemi and Akbarimajd proposed an MLP-based adaptive SMC method for a quadrotor UAV. They showed that an MLP can use real-time feedback to adjust SMC parameters and improve tracking accuracy and chattering suppression [8]. This study shows that a neural network can be used to tune the control parameters of an existing SMC structure. However, a quadrotor UAV operates in a relatively low-speed regime. Its disturbances are mainly caused by wind, payload changes, and sensor noise. In contrast, a hypersonic vehicle is affected at the same time by strong aerodynamic nonlinearity due to high-speed flight, coupling between propulsion and vehicle dynamics, and uncertainties in atmospheric density and aerodynamic coefficients. Therefore, applying this approach to the velocity and altitude tracking problem of a hypersonic vehicle requires separate design and validation that reflect the dynamic characteristics of the target system.

In this study, an NN-SMC is proposed by adding a neural network tuner that adjusts the sliding surface parameters of the velocity and altitude channels online while preserving the input calculation structure of the existing SMC. The proposed method does not require an offline dataset or a pre-trained target λ label. Instead, it updates the neural network weights online by using the tracking error and sliding variable calculated during closed-loop control. Therefore, the method can be added to the existing SMC structure in a relatively simple way. It can also adjust the convergence characteristics of the sliding surfaces according to the operating condition under uncertainty.

The rest of this paper is organized as follows. Section 2 defines the hypersonic longitudinal dynamic model and presents the input/output linearization and SMC design process for velocity and altitude outputs. Section 3 presents the structure and learning method of the MLP-based neural network tuner that adjusts the sliding surface parameters λ_V and λ_h online while preserving the existing SMC structure. Section 4 compares the tracking performance, control input usage, and online λ variation of Nominal SMC and NN-SMC using a plant model with parameter uncertainty. Finally, Section 5 summarizes the main results and discusses future work.

2. Longitudinal Dynamics and Sliding Mode Control

This section explains the hypersonic longitudinal dynamic model and the SMC design process based on this model. First, the state vector, control input, equations of motion, and aerodynamic model are defined. Then, the input/output linearization process is organized using velocity and altitude as the outputs. Finally, the SMC is designed using the linearized dynamics.

2.1 Vehicle Model

This study uses a hypersonic longitudinal dynamic model based on a vehicle developed by NASA Langley Research Center [9, 10]. The state vector and control input vector are defined as follows:

$$x = [V,\ \gamma,\ h,\ \alpha,\ q,\ \dot{\beta},\ \beta]^T, \qquad u = [\beta_c,\ \delta_e]^T. \tag{1}$$

Here, V is the flight velocity (ft/s), γ is the flight-path angle (rad), h is the altitude (ft), α is the angle of attack (rad), q is the pitch rate (rad/s), and β is the throttle setting. The control input β_c is the throttle command, and δ_e is the elevator deflection angle (rad).

The longitudinal equations of motion using these state variables are written as follows:

$$\dot{V} = \frac{T\cos\alpha - D}{m} - \frac{\mu\sin\gamma}{r^2}, \tag{2}$$

$$\dot{\gamma} = \frac{L + T\sin\alpha}{mV} - \frac{(\mu - V^2 r)\cos\gamma}{Vr^2}, \tag{3}$$

$$\dot{h} = V\sin\gamma, \tag{4}$$

$$\dot{\alpha} = q - \dot{\gamma}, \tag{5}$$

$$\dot{q} = \frac{M_{yy}}{I_{yy}}, \tag{6}$$

$$\ddot{\beta} = -2\zeta\omega_n\dot{\beta} - \omega_n^2\beta + \omega_n^2\beta_c. \tag{7}$$

Here, m is the vehicle mass, I_{yy} is the pitch moment of inertia, μ is the gravitational constant, $r = h + R_E$ is the distance from the center of the Earth, and R_E is the Earth radius. Equation (2) represents the velocity change caused by thrust, drag, and gravity. Equation (3) represents the flight-path angle change caused by lift, the normal component of thrust, and curvature effects. Equation (7) represents the second-order actuator dynamics through which the throttle command β_c is reflected in the actual throttle setting β. The parameters ζ and ω_n are the damping ratio and natural frequency, respectively. The lift L, drag D, thrust T, and pitch moment M_{yy} in the equations of motion are calculated from the dynamic pressure and aerodynamic coefficients:

$$L = \frac{1}{2}\rho V^2 S C_L, \qquad D = \frac{1}{2}\rho V^2 S C_D, \tag{8}$$

$$T = \frac{1}{2}\rho V^2 S C_T, \qquad M_{yy} = \frac{1}{2}\rho V^2 S\bar{c}\left[C_M(\alpha) + C_M(q) + C_M(\delta_e)\right]. \tag{9}$$

Here, ρ is the atmospheric density (slug/ft^3), S is the reference area (ft^2), and $\bar{c}$ is the mean aerodynamic chord (ft). The aerodynamic coefficients are defined as follows:

$$C_L = 0.6203\alpha, \tag{10}$$

$$C_D = 0.6450\alpha^2 + 0.0043378\alpha + 0.003772, \tag{11}$$

$$C_T = \begin{cases} 0.02576\beta, & \beta < 1, \\ 0.0224 + 0.00336\beta, & \beta \geq 1, \end{cases} \tag{12}$$

$$C_M(\alpha) = -0.035\alpha^2 + 0.036617\alpha + 5.3261 \times 10^{-6}, \tag{13}$$

$$C_M(q) = \frac{\bar{c}}{2V} q\left(-6.796\alpha^2 + 0.3015\alpha - 0.2289\right), \tag{14}$$

$$C_M(\delta_e) = c_e(\delta_e - \alpha). \tag{15}$$

Thus, C_L and C_D are aerodynamic coefficients determined by the angle of attack α, and C_T is defined piecewise according to the throttle setting β. The pitch moment consists of the static moment due to the angle of attack, the damping moment due to the pitch rate, and the control moment due to the elevator deflection. The parameter $c_e = 0.0292$ is the elevator effectiveness coefficient. The nominal cruise condition used in this study is $M = 15$, $V_0 = 15{,}060$ ft/s, and $h_0 = 110{,}000$ ft. This condition is used as the trim condition for controller design and simulation.

2.2 Input/Output Linearization

Input/output linearization is applied to track velocity V and altitude h at the same time. A general MIMO nonlinear system can be written as follows:

$$\dot{x} = f(x) + \sum_{k=1}^{m} g_k(x)u_k, \qquad y_i = h_i(x), \quad i = 1, \ldots, m. \tag{16}$$

In Eq. (16), $f(x)$ represents the state dynamics without input, $g_k(x)u_k$ represents the effect of each control input u_k on the system state, and $y_i = h_i(x)$ is the output to be linearized. In input/output

linearization, the selected output y_i is repeatedly differentiated to find the order at which the control input u_k appears explicitly. The resulting output dynamics are then arranged in a form that is linear with respect to the input. The control outputs in this study are velocity V and altitude h. In the original model, the control inputs β_c and δ_e appear explicitly when V is differentiated three times and h is differentiated four times. Therefore, the relative degrees of the two outputs are $r_V = 3$ and $r_h = 4$, respectively, and the total relative degree is $r = r_V + r_h = 7$. This is equal to the system order $n = 7$. Thus, the longitudinal nonlinear model can be fully input/output linearized for the selected outputs V and h [11].

To show this in detail, velocity V is first differentiated repeatedly:

$$\dot{V} = f_1(x), \tag{17}$$

$$\ddot{V} = \frac{\omega_1 \dot{x}_a}{m}, \tag{18}$$

$$\dddot{V} = \frac{\omega_1 \ddot{x}_a + \dot{x}_a^T \Omega_2 \dot{x}_a}{m}. \tag{19}$$

Here, $f_1(x)$ denotes the right-hand side of Eq. (2), and $x_a = [V, \gamma, \alpha, \beta, h]^T$ is the state subvector used in the differentiation. Also, $\omega_1 = \partial f_1 / \partial x_a$ and $\Omega_2 = \partial \omega_1 / \partial x_a$.

Altitude h is differentiated repeatedly as follows:

$$\dot{h} = V \sin\gamma, \tag{20}$$

$$\ddot{h} = \dot{V} \sin\gamma + V\dot{\gamma}\cos\gamma, \tag{21}$$

$$\dddot{h} = \ddot{V} \sin\gamma + 2\dot{V}\dot{\gamma}\cos\gamma - V\dot{\gamma}^2 \sin\gamma + V\ddot{\gamma}\cos\gamma, \tag{22}$$

$$h^{(4)} = \dddot{V} \sin\gamma + 3\ddot{V}\dot{\gamma}\cos\gamma - 3\dot{V}\dot{\gamma}^2 \sin\gamma + 3\dot{V}\ddot{\gamma}\cos\gamma - 3V\dot{\gamma}\ddot{\gamma}\sin\gamma - V\dot{\gamma}^3 \cos\gamma + V\dddot{\gamma}\cos\gamma. \tag{23}$$

Here, $\dot{\gamma}$ is calculated from the right-hand side of Eq. (3). If it is denoted by $f_2(x)$, the following relations are obtained:

$$\dot{\gamma} = f_2(x), \qquad \ddot{\gamma} = \pi_1 \dot{x}_a, \qquad \dddot{\gamma} = \pi_1 \ddot{x}_a + \dot{x}_a^T \Pi_2 \dot{x}_a. \tag{24}$$

Here, $\pi_1 = \partial f_2 / \partial x_a$ and $\Pi_2 = \partial \pi_1 / \partial x_a$.

Equations (19) and (23) include $\ddot{\alpha}$ and $\ddot{\beta}$. These two terms can be separated into input-related and input-free parts:

$$\ddot{\alpha} = \ddot{\alpha}_0 + \frac{c_e \rho V^2 S \bar{c}}{2 I_{yy}} \delta_e, \tag{25}$$

$$\ddot{\beta} = \ddot{\beta}_0 + \omega_n^2 \beta_c. \tag{26}$$

The input-free terms are given as follows:

$$\ddot{\alpha}_0 = \frac{1}{2I_{yy}} \rho V^2 S \bar{c} \left[C_M(\alpha) + C_M(q) - c_e \alpha \right] - \ddot{\gamma}, \tag{27}$$

$$\ddot{\beta}_0 = -2\zeta\omega_n \dot{\beta} - \omega_n^2 \beta. \tag{28}$$

Thus, δ_e affects the high-order derivatives of velocity and altitude through $\ddot{\alpha}$, and β_c affects the high-order derivatives related to thrust variation through $\ddot{\beta}$.

Using the input-free part, define $\ddot{x}_{a0} = [\dot{V}, \ddot{\gamma}, \ddot{\alpha}_0, \ddot{\beta}_0, \ddot{h}]^T$. Then, the high-order output dynamics of velocity and altitude are written as follows:

$$\dddot{V} = f_V(x) + b_{11}\beta_c + b_{12}\delta_e, \tag{29}$$

$$h^{(4)} = f_h(x) + b_{21}\beta_c + b_{22}\delta_e. \tag{30}$$

Here, $f_V(x)$ and $f_h(x)$ are nonlinear terms that do not include the control inputs. They are defined as follows:

$$f_V(x) = \frac{\omega_1 \ddot{x}_{a0} + \dot{x}_a^T \Omega_2 \dot{x}_a}{m}, \tag{31}$$

$$f_h(x) = 3\ddot{V}\dot{\gamma}\cos\gamma - 3\dot{V}\dot{\gamma}^2\sin\gamma + 3\dot{V}\ddot{\gamma}\cos\gamma - 3V\dot{\gamma}\ddot{\gamma}\sin\gamma - V\dot{\gamma}^3\cos\gamma + f_V(x)\sin\gamma + V\cos\gamma\left(\pi_1\ddot{x}_{a0} + \dot{x}_a^T\Pi_2\dot{x}_a\right). \tag{32}$$

Meanwhile, the elements of the input gain matrix $B(x) = [b_{ij}] \in \mathbb{R}^{2\times 2}$ are given as follows:

$$b_{11} = \frac{\rho V^2 S c_\beta \omega_n^2}{2m}\cos\alpha, \tag{33}$$

$$b_{12} = -\frac{c_e \rho V^2 S\bar{c}}{2mI_{yy}}\left(T\sin\alpha + \frac{\partial D}{\partial\alpha}\right), \tag{34}$$

$$b_{21} = \frac{\rho V^2 S c_\beta \omega_n^2}{2m}\sin(\alpha+\gamma), \tag{35}$$

$$b_{22} = \frac{c_e \rho V^2 S\bar{c}}{2mI_{yy}}\left[T\cos(\alpha+\gamma) + \frac{\partial L}{\partial\alpha}\cos\gamma - \frac{\partial D}{\partial\alpha}\sin\gamma\right]. \tag{36}$$

where

$$c_\beta = \begin{cases} 0.02576, & \beta < 1, \\ 0.00336, & \beta \geq 1, \end{cases} \tag{37}$$

$$\frac{\partial D}{\partial\alpha} = \frac{1}{2}\rho V^2 S(1.290\alpha + 0.0043378), \quad \frac{\partial L}{\partial\alpha} = \frac{1}{2}\rho V^2 S(0.6203). \tag{38}$$

Therefore, Eqs. (29) and (30) can be written in vector form as follows:

$$\begin{bmatrix} \dddot{V} \\ h^{(4)} \end{bmatrix} = \begin{bmatrix} f_V(x) \\ f_h(x) \end{bmatrix} + B(x)\begin{bmatrix} \beta_c \\ \delta_e \end{bmatrix}. \tag{39}$$

If $B(x)$ is nonsingular, the two inputs β_c and δ_e can be used to assign $\dddot{V}$ and $h^{(4)}$ to desired forms. This result is used in the next section to arrange the sliding variable derivatives $\dot{s}_V$ and $\dot{s}_h$ in affine form with respect to the control inputs and to derive the final SMC input law.

2.3 Sliding Mode Controller Design

This section designs an SMC for velocity and altitude tracking using the input/output linearization result obtained in the previous section [12]. The control objective is to determine the control inputs β_c and δ_e so that the velocity V and altitude h track the reference commands $V_d(t)$ and $h_d(t)$, respectively. The tracking errors are defined as follows:

$$e_V = V - V_d, \qquad e_h = h - h_d. \tag{40}$$

The sliding surfaces of the velocity and altitude channels are defined in the same form as in the reference paper. Because the relative degree of velocity V is 3 and that of altitude h is 4, the sliding surfaces are defined as follows:

$$s_V = \left(\frac{d}{dt} + \lambda_V\right)^3 \int_0^t e_V(\tau)d\tau, \tag{41}$$

$$s_h = \left(\frac{d}{dt} + \lambda_h\right)^4 \int_0^t e_h(\tau)d\tau. \tag{42}$$

Here, $\lambda_V > 0$ and $\lambda_h > 0$ are design parameters that determine the bandwidths of the velocity and altitude error dynamics, respectively. When $s_V = 0$ and $s_h = 0$, the errors e_V and e_h follow linear error dynamics. The integral terms in the sliding surfaces help reduce the steady-state error.
Expanding Eqs. (41) and (42) gives

$$s_V = \ddot{e}_V + 3\lambda_V\dot{e}_V + 3\lambda_V^2 e_V + \lambda_V^3\int_0^t e_V(\tau)d\tau, \tag{43}$$

$$s_h = \dddot{e}_h + 4\lambda_h\ddot{e}_h + 6\lambda_h^2\dot{e}_h + 4\lambda_h^3 e_h + \lambda_h^4\int_0^t e_h(\tau)d\tau. \tag{44}$$

These expressions are used when the sliding surfaces are differentiated once more to derive the actual control input. Differentiating Eqs. (43) and (44) with respect to time gives

$$\dot{s}_V = \dddot{e}_V + 3\lambda_V \ddot{e}_V + 3\lambda_V^2 \dot{e}_V + \lambda_V^3 e_V, \tag{45}$$

$$\dot{s}_h = e_h^{(4)} + 4\lambda_h \dddot{e}_h + 6\lambda_h^2 \ddot{e}_h + 4\lambda_h^3 \dot{e}_h + \lambda_h^4 e_h. \tag{46}$$

Substituting $e_V = V - V_d$ and $e_h = h - h_d$, and using the input/output linearization results $\dddot{V} = f_V(x) + b_{11}\beta_c + b_{12}\delta_e$ and $h^{(4)} = f_h(x) + b_{21}\beta_c + b_{22}\delta_e$, yields

$$\dot{s}_V = v_V(x,t) + b_{11}\beta_c + b_{12}\delta_e, \tag{47}$$

$$\dot{s}_h = v_h(x,t) + b_{21}\beta_c + b_{22}\delta_e. \tag{48}$$

Here, $v_V(x,t)$ and $v_h(x,t)$ include the reference command terms and the error dynamics terms except for the control inputs. They are defined as follows:

$$v_V(x,t) = -\dddot{V}_d + f_V(x) + 3\lambda_V \ddot{e}_V + 3\lambda_V^2 \dot{e}_V + \lambda_V^3 e_V, \tag{49}$$

$$v_h(x,t) = -h_d^{(4)} + f_h(x) + 4\lambda_h \dddot{e}_h + 6\lambda_h^2 \ddot{e}_h + 4\lambda_h^3 \dot{e}_h + \lambda_h^4 e_h. \tag{50}$$

Equations (47) and (48) can be written in vector form as

$$\begin{bmatrix} \dot{s}_V \\ \dot{s}_h \end{bmatrix} = \begin{bmatrix} v_V(x,t) \\ v_h(x,t) \end{bmatrix} + B(x) \begin{bmatrix} \beta_c \\ \delta_e \end{bmatrix}. \tag{51}$$

Here, $B(x)$ is the input gain matrix defined in the previous section. Equation (51) shows that the control input acts linearly on the time derivative of the sliding variables. Thus, if $B(x)$ is nonsingular, the input that produces the desired $\dot{s}_V$ and $\dot{s}_h$ can be calculated.

In SMC, the control input is selected so that the system trajectory reaches the sliding surface in finite time and stays on that surface after reaching it. For this purpose, the following sliding conditions are imposed:

$$\frac{1}{2}\frac{d}{dt}s_V^2 \le -k_V|s_V|, \tag{52}$$

$$\frac{1}{2}\frac{d}{dt}s_h^2 \le -k_h|s_h|. \tag{53}$$

Here, $k_V > 0$ and $k_h > 0$ are switching gains that determine the reaching speed to the sliding surfaces. Equations (52) and (53) make s_i^2 decrease when $s_i \neq 0$. Therefore, the state trajectory moves toward the sliding surface.

A representative reaching law that satisfies these conditions is given by

$$\begin{bmatrix} \dot{s}_V \\ \dot{s}_h \end{bmatrix} = \begin{bmatrix} -k_V \operatorname{sgn}(s_V) \\ -k_h \operatorname{sgn}(s_h) \end{bmatrix}. \tag{54}$$

Equating Eq. (51) and Eq. (54) gives the sign-based SMC input:

$$\begin{bmatrix} \beta_c \\ \delta_e \end{bmatrix} = B^{-1}(x) \begin{bmatrix} -v_V(x,t) - k_V \operatorname{sgn}(s_V) \\ -v_h(x,t) - k_h \operatorname{sgn}(s_h) \end{bmatrix}. \tag{55}$$

Equation (55) can be interpreted as two terms. The first term, $-B^{-1}[v_V \; v_h]^T$, is the equivalent control term. It cancels the sliding variable dynamics so that $\dot{s}_V = \dot{s}_h = 0$ in the nominal model. The second term, $-B^{-1}[k_V \operatorname{sgn}(s_V) \; k_h \operatorname{sgn}(s_h)]^T$, is the switching term. It is added to ensure reachability to the sliding surface even under model uncertainty and disturbance.

However, the sign function can cause chattering because it generates discontinuous input switching near the sliding surface. In practice, a saturation function with a boundary layer is used instead of the sign function to reduce this effect:

$$\operatorname{sat}(\sigma) = \begin{cases} 1, & \sigma > 1, \\ \sigma, & |\sigma| \le 1, \\ -1, & \sigma < -1. \end{cases} \tag{56}$$

Therefore, the final SMC input used in this study is

$$\begin{bmatrix}\beta_c\\ \delta_e\end{bmatrix} = B^{-1}(x)\begin{bmatrix}-v_V(x,t) - k_V\,\mathrm{sat}(s_V/\phi_V)\\ -v_h(x,t) - k_h\,\mathrm{sat}(s_h/\phi_h)\end{bmatrix}. \tag{57}$$

Here, $\phi_V > 0$ and $\phi_h > 0$ are the boundary layer widths of the velocity and altitude sliding surfaces, respectively. A smaller ϕ_i can increase tracking accuracy, but it can also increase input variation. A larger ϕ_i can reduce chattering, but it can increase the steady-state error. Therefore, saturation-based SMC has a trade-off between tracking accuracy and control input variation.
Both Nominal SMC and NN-SMC in this study use the same input calculation structure in Eq. (57). The difference is that Nominal SMC uses fixed values of λ_V and λ_h, whereas NN-SMC adjusts λ_V and λ_h online using the neural network tuner.

3. Neural Network Based Online Tuner

This section explains how the sliding surface parameters λ_V and λ_h are adjusted online by a neural network while preserving the SMC structure designed in the previous section. The proposed neural network tuner uses independent MLPs for the velocity and altitude channels. It receives the reference command, current state, integral of the error, and sliding variable as inputs. It also calculates the λ correction of each channel in real time using the output and online weight update law.

3.1 Network Architecture

In this study, the SMC sliding surface design parameters λ_V and λ_h are not fixed. They are updated at every control step by the neural-network-based online tuner. In Nominal SMC, λ_V and λ_h remain as preselected constants. In NN-SMC, the two parameters are corrected in real time using the current tracking condition and sliding variable information. For this purpose, independent Multi-Layer Perceptrons (MLPs) are constructed for the velocity and altitude channels.
The input vector of the velocity channel consists of the reference velocity command, actual velocity response, integral of the velocity error, and velocity sliding variable:

$$z_V = [V_d,\ \dot{V}_d,\ \ddot{V}_d,\ \dddot{V}_d,\ V,\ \dot{V},\ \ddot{V},\ V_I,\ s_V]^T. \tag{58}$$

Here, $V_I = \int_0^t e_V(\tau)d\tau$ is the integral of the velocity error, and s_V is the sliding variable of the velocity channel defined in Eq. (41). The input vector of the altitude channel consists of the reference altitude command, actual altitude response, integral of the altitude error, and altitude sliding variable:

$$z_h = [h_d,\ \dot{h}_d,\ \ddot{h}_d,\ \dddot{h}_d,\ h_d^{(4)},\ h,\ \dot{h},\ \ddot{h},\ \dddot{h},\ h_I,\ s_h]^T. \tag{59}$$

Here, $h_I = \int_0^t e_h(\tau)d\tau$ is the integral of the altitude error, and s_h is the sliding variable of the altitude channel defined in Eq. (42). Because each input component has a different physical unit and scale, the input is normalized by using a predefined scale factor before it is applied to the neural network. The scale factor is set empirically based on the nominal value of each variable and the expected variation range during simulation.
Each channel MLP has one hidden layer. The number of hidden nodes is set to $n_h = 8$ for both channels. The input dimension is 9 for the velocity channel and 11 for the altitude channel. Therefore, the size of the input-layer weight matrix is different for each channel. However, both channels use $\tanh$ as the hidden-layer activation function. In the output layer, a Leaky ReLU activation function is used to output one scalar correction value.
The forward propagation process for one channel is written as follows:

$$\bar{z}_i = \mathrm{scale}(z_i), \qquad i \in \{V,h\}, \tag{60}$$

$$\xi_i = \tanh(W_{h,i}\bar{z}_i + b_{h,i}), \tag{61}$$

$$o_i = W_{o,i}\xi_i + b_{o,i}, \tag{62}$$

$$\sigma_L(o_i) = \begin{cases} o_i, & o_i \geq 0,\\ a_{L,i}o_i, & o_i < 0. \end{cases} \tag{63}$$

Here, $\bar{z}_i$ is the input vector normalized by the scale factor, and $\xi_i \in \mathbb{R}^{n_h}$ is the hidden-layer output vector. That is, ξ_i is the feature vector generated after the normalized input passes through the hidden-layer weight $W_{h,i}$, bias $b_{h,i}$, and $\tanh$ nonlinear function. The variable $o_i \in \mathbb{R}$ is the linear output of the output layer, and $\sigma_L(o_i)$ is the scalar correction value after the Leaky ReLU. In $W_{h,i}$, the first subscript h denotes the hidden layer, while the channel index $i = h$ denotes the altitude channel.
Finally, λ_i of each channel is calculated by accumulating the neural network output correction:

$$\lambda_i(k+1) = \lambda_i(k) + \sigma_L(o_i(k)), \qquad i \in \{V, h\}. \tag{64}$$

Here, $i = V$ denotes the velocity channel, and $i = h$ denotes the altitude channel. The initial values are set to $\lambda_V(0) = 0.70$ and $\lambda_h(0) = 0.80$. The neural network output $\sigma_L(o_i(k))$ is used as the incremental correction of λ_i that reflects the current tracking condition and sliding variable information. Thus, if $\sigma_L(o_i) > 0$, the next λ_i increases. If $\sigma_L(o_i) < 0$, λ_i decreases. In the simulation, $a_{L,V} = 0.03$ is used for the velocity channel, and $a_{L,h} = 0.02$ is used for the altitude channel.
As a result, the proposed neural network does not directly generate the control inputs β_c and δ_e. It only adjusts the sliding surface parameters λ_V and λ_h used in the existing SMC input calculation. Therefore, NN-SMC can be regarded as an existing SMC law with an added neural-network-based parameter tuning module.

3.2 Online Learning Signal and Backpropagation

The proposed neural network tuner does not use a predefined target λ_i label. Instead, it updates the neural network weights online by using the tracking error and sliding variable calculated during closed-loop control. The online learning signals of the velocity and altitude channels are defined as follows:

$$g_V(k) = e_V(k)s_V(k), \qquad g_h(k) = e_h(k)s_h(k). \tag{65}$$

Here, $g_V(k)$ is the learning signal of the velocity-channel MLP, and $g_h(k)$ is the learning signal of the altitude-channel MLP. This signal becomes large when both the error and the sliding variable are large. Therefore, a larger neural network update occurs when the current control response is farther from the sliding surface. In contrast, the update becomes smaller when either the error or the sliding variable becomes small.
The online learning signals $g_V(k)$ and $g_h(k)$ can be interpreted as pseudo-gradients for reducing the magnitude of the sliding variables. The learning signal determines the adjustment direction of the neural network output correction according to the relative sign of the tracking error and the sliding variable. When the two values act in the same direction, the weights are updated in a direction that reduces the correction value. When they act in opposite directions, the weights are updated in a direction that increases the correction value. Therefore, the proposed update law does not directly minimize the tracking error itself. Instead, it is an approximate online learning method that adjusts λ_i in a direction that reduces $s_i(k)$, which represents the deviation from the sliding surface.
If $g_i(k)$ becomes too large during online learning, the neural network weights can change abruptly. Therefore, $g_i(k)$ is clipped in the implementation:

$$\bar{g}_i(k) = \mathrm{clip}\left(g_i(k), -g_{\max}, g_{\max}\right), \qquad i \in \{V, h\}. \tag{66}$$

Here, $g_{\max}$ is the constant that limits the maximum magnitude of the learning signal. In this study, $g_{\max} = 1.0$ is used. This reduces abrupt parameter changes during online learning.
Because the output layer uses a Leaky ReLU activation function, the derivative of the activation function with respect to the linear output $o_i(k)$ is given as follows:

$$\sigma_L'(o_i) = \begin{cases} 1, & o_i \geq 0, \\ a_{L,i}, & o_i < 0. \end{cases} \tag{67}$$

This means that the learning signal is passed directly when $o_i \geq 0$, while it is reduced by $a_{L,i}$ when $o_i < 0$. Therefore, unlike ReLU, the gradient does not completely disappear in the negative output region, and a correction that decreases λ_i is also possible.

Accordingly, the backpropagation signal delivered to the output layer of the ith channel is defined as follows:

$$\delta_{o,i}(k) = \bar{g}_i(k)\sigma_L'(o_i(k)), \qquad i \in \{V,h\}. \tag{68}$$

Here, $\bar{g}_i(k)$ is the online learning signal obtained from the closed-loop response, and $\sigma_L'(o_i(k))$ is the local slope of the output activation function. Thus, $\delta_{o,i}(k)$ is not an explicit supervised-learning error. It is interpreted as a pseudo-gradient signal used to update the output-layer weights.
The update directions of the output-layer weight and bias are

$$\Delta W_{o,i}(k) = \delta_{o,i}(k)\xi_i^T(k), \qquad \Delta b_{o,i}(k) = \delta_{o,i}(k). \tag{69}$$

Thus, the output-layer weight update direction is calculated as the product of the hidden-layer output vector $\xi_i(k)$ and the output-layer backpropagation signal $\delta_{o,i}(k)$.
The backpropagation signal delivered to the hidden layer is calculated using the derivative of the $\tanh$ activation function:

$$\delta_{h,i}(k) = \left(W_{o,i}^T\delta_{o,i}(k)\right) \odot \left(1-\xi_i^2(k)\right). \tag{70}$$

Here, $\odot$ denotes the element-wise product. Since the derivative of $\tanh$ is $1-\xi_i^2$, the update magnitude of each hidden node is determined by the signal delivered from the output layer and the activation level of the corresponding hidden node.
The update directions of the hidden-layer weight and bias are

$$\Delta W_{h,i}(k) = \delta_{h,i}(k)\bar{z}_i^T(k), \qquad \Delta b_{h,i}(k) = \delta_{h,i}(k). \tag{71}$$

Here, $\bar{z}_i(k)$ is the input vector normalized by the scale factor. Finally, the MLP parameters of each channel are updated as follows:

$$W_{h,i}(k+1) = W_{h,i}(k) - \eta_i\Delta W_{h,i}(k), \qquad b_{h,i}(k+1) = b_{h,i}(k) - \eta_i\Delta b_{h,i}(k), \tag{72}$$

$$W_{o,i}(k+1) = W_{o,i}(k) - \eta_i\Delta W_{o,i}(k), \qquad b_{o,i}(k+1) = b_{o,i}(k) - \eta_i\Delta b_{o,i}(k). \tag{73}$$

Here, η_i is the learning rate of the ith channel. In this study, $\eta_i = 0.001$ is used for both channels.
In summary, the proposed online learning method does not use a supervised-learning structure that tracks an explicit target λ_i. Instead, it adjusts the neural network output correction by using $e_i(k)s_i(k)$, which is calculated from the current closed-loop response. Therefore, no offline training data are required. The update direction of λ_i can be determined at every control step using only the error and sliding variable calculated by SMC.

4. Validation

This section compares the tracking performance, control input characteristics, and neural-network-adjusted sliding surface parameters of Nominal SMC and NN-SMC. The two controllers are evaluated under the same reference command and simulation condition. NN-SMC preserves the Nominal SMC input calculation structure and adjusts only λ_V and λ_h online.

4.1 Simulation Condition

This study compares the tracking performance of Nominal SMC and NN-SMC using a hypersonic longitudinal plant model with uncertainty. The controller is designed based on the nominal model. The simulation plant includes parameter uncertainties in the mass m, atmospheric density ρ, reference area S, pitch moment of inertia I_{yy}, mean aerodynamic chord $\bar{c}$, and elevator effectiveness coefficient c_e.
Each uncertain parameter is applied by multiplying the nominal value by a relative variation:

$$p = p_0(1+\Delta_p). \tag{74}$$

Here, p_0 is the nominal parameter, and Δ_p is the relative uncertainty. In this simulation, random uncertainties within $\pm 30\%$ are applied to m, ρ, and S, and random uncertainties within $\pm 20\%$ are applied to I_{yy}, $\bar{c}$, and c_e:

$$\Delta_m, \Delta_\rho, \Delta_S \in [-0.30, 0.30], \qquad \Delta_{I_{yy}}, \Delta_{\bar{c}}, \Delta_{c_e} \in [-0.20, 0.20]. \tag{75}$$

For an objective performance evaluation, the two controllers are tested under exactly the same simulation environment. Specifically, the nominal cruise state of the vehicle is used as the initial condition, and the same control command that requires a certain increase in velocity and altitude is applied. The control command consists of the velocity command $V_d(t)$ and the altitude command $h_d(t)$. From the nominal cruise condition, the velocity is commanded to increase by 100 ft/s and the altitude is commanded to increase by 2000 ft. In the result analysis, these values are converted to SI units and shown as reference changes of approximately 30.48 m/s and 609.6 m, respectively.
The controller performance is evaluated quantitatively from two viewpoints: tracking accuracy and control effort. The tracking accuracy is evaluated using RMSE and MAE, which reflect the error magnitude during both the transient and steady-state responses. The efficiency and variation of the control input are compared using the RMS and control energy of the throttle and elevator inputs.
The performance metrics are defined as follows. If the tracking error is $e_k = y_k - y_{d,k}$, then

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{k=1}^{N} e_k^2}, \qquad \mathrm{MAE} = \frac{1}{N}\sum_{k=1}^{N} |e_k|. \tag{76}$$

For the control input u_k, the input RMS and control energy are calculated as follows:

$$u_{\mathrm{RMS}} = \sqrt{\frac{1}{N}\sum_{k=1}^{N} u_k^2}, \qquad E_u = \sum_{k=1}^{N} u_k^2 \Delta t. \tag{77}$$

Here, N is the total number of samples, y_k and $y_{d,k}$ are the actual output and reference command, respectively, and Δt is the sampling time.

4.2 Results Analysis

Figure 1 first shows the velocity and altitude tracking results of Nominal SMC and NN-SMC. In the velocity response, both controllers track the reference command stably and show no large difference. In the altitude response, the difference is clearer. Nominal SMC still has a remaining altitude tracking error in the later part of the simulation. In contrast, NN-SMC gradually reduces the altitude error and converges closer to the reference altitude.
This trend is also observed in the tracking error comparison in Fig. 2. For the velocity error, NN-SMC shows a larger error temporarily in the early transient phase. However, the error decreases quickly afterward and remains small near the steady state. For the altitude error, NN-SMC shows a larger undershoot in the early phase, but the error decreases continuously and becomes close to zero at the end. In contrast, Nominal SMC retains an altitude error of more than about -10 m, which limits its steady-state tracking performance.
Table 1 quantifies the tracking performance observed in Figs. 1 and 2 using RMSE and MAE.

Table 1 – Tracking performance comparison between Nominal SMC and NN-SMC.

Output	Metric	Nominal SMC	NN-SMC
Velocity	RMSE (m/s)	0.0517	0.0548
Velocity	MAE (m/s)	0.0356	0.0194
Altitude	RMSE (m)	11.8903	8.8803
Altitude	MAE (m)	11.7470	6.3405

The velocity RMSE increases slightly from 0.0517 m/s for Nominal SMC to 0.0548 m/s for NN-SMC. This is because the large velocity error in the early transient phase is reflected in the RMSE. However, the steady-state error improves significantly, and the velocity MAE decreases from 0.0356 m/s to 0.0194 m/s. The improvement is clearer in the altitude channel. The altitude RMSE decreases from 11.8903 m to 8.8803 m, and the altitude MAE decreases from 11.7470 m to 6.3405 m. Therefore, NN-SMC shows better performance than Nominal SMC, especially in altitude tracking accuracy.
Next, Fig. 3 shows the time responses of the throttle setting β, elevator deflection δ_e, thrust T, and angle of attack α.

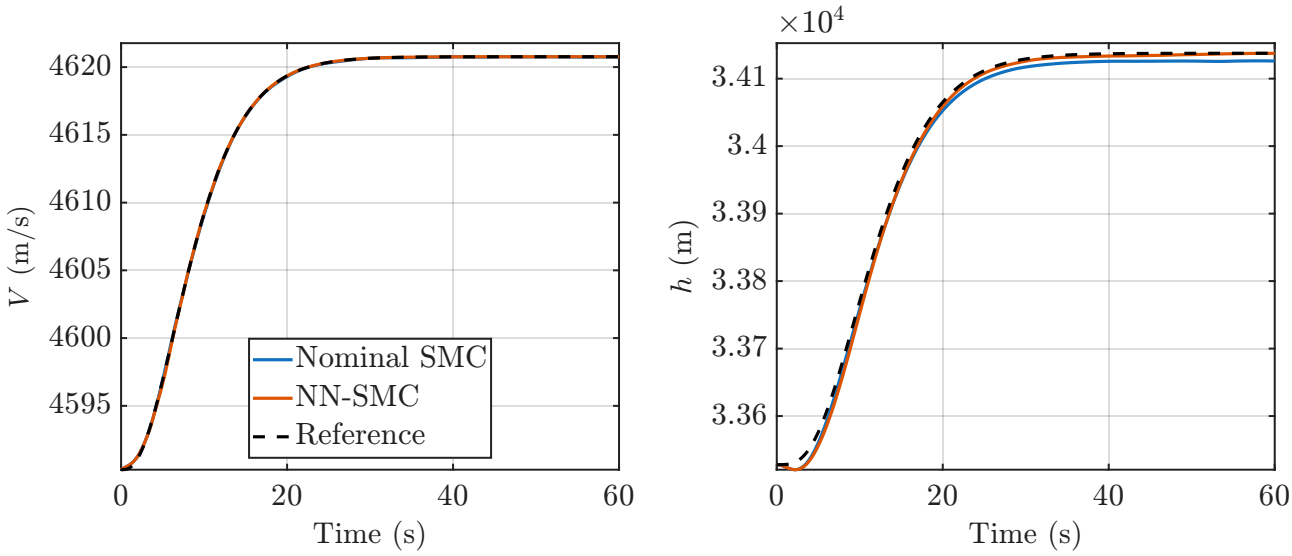


Figure 1 – Velocity and altitude tracking responses of Nominal SMC and NN-SMC.

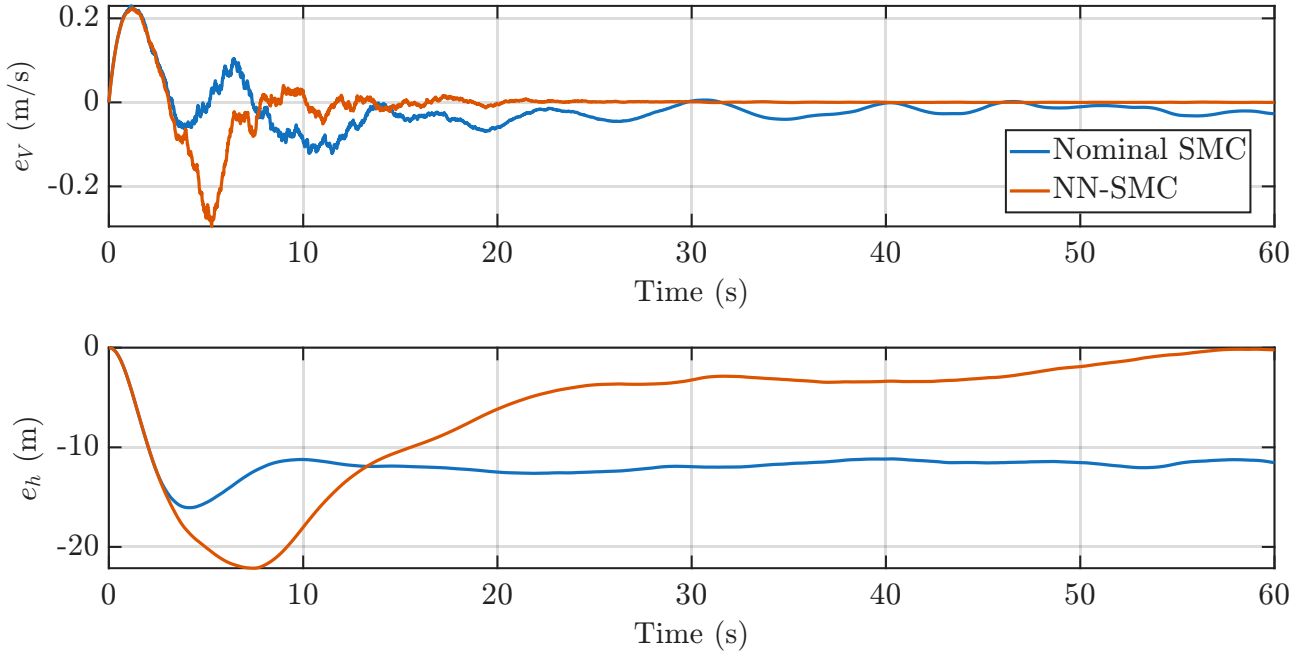


Figure 2 – Velocity and altitude tracking error comparison.

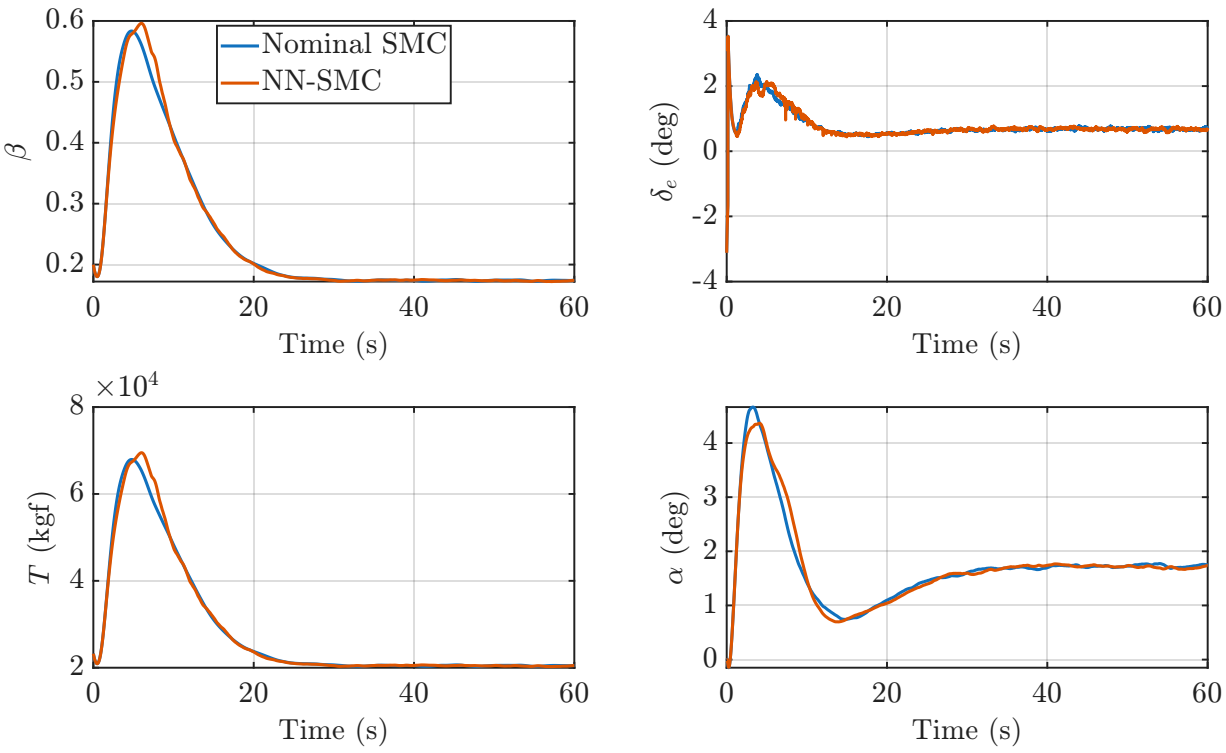


Figure 3 – Control input and state response comparison.

The two controllers show similar overall control input and state responses. That is, NN-SMC does not greatly increase the control input magnitude while improving the tracking performance. The throttle setting and thrust responses increase in the initial phase and then decrease toward similar steady-state values. The elevator deflection and angle of attack also vary within similar ranges for the two controllers.
Table 2 compares the RMS and control energy of the control inputs.

Table 2 – Control input RMS and energy comparison between Nominal SMC and NN-SMC.

Input	Metric	Nominal SMC	NN-SMC
β	RMS	1.2563	1.2606
δ_e	RMS (deg)	0.8833	0.8886
T	RMS (kgf)	31317.5	31564.3
β	Energy	94.712	95.356
δ_e	Energy	46.823	47.380
T	Energy	5.886×10^{10}	5.979×10^{10}

The RMS and control energy of β_c, δ_e, and T increase slightly for NN-SMC, but the difference between the two controllers is limited. For example, the RMS of β_c changes from 1.2563 to 1.2606, and the RMS of δ_e changes only from 0.8833 deg to 0.8886 deg. The control energy also does not show a large difference. Therefore, Fig. 3 and Table 2 show that NN-SMC achieves better tracking performance with a control effort similar to that of Nominal SMC.
Finally, Fig. 4 shows the time variation of the sliding surface parameters λ_V and λ_h that are adjusted online in NN-SMC.
In Nominal SMC, $\lambda_V = 0.70$ and $\lambda_h = 0.80$ are fixed. In NN-SMC, these two parameters change according to the tracking condition. The parameter λ_V increases to about 1.6 after the initial phase, which strengthens the convergence characteristics of the velocity channel. In contrast, λ_h decreases to about $0.75 \sim 0.78$ after the initial transient phase. This result suggests that the altitude channel reduces the long-term error without requiring a large increase in control input.
In summary, Fig. 1, Fig. 2, and Table 1 show that NN-SMC effectively reduces the average tracking

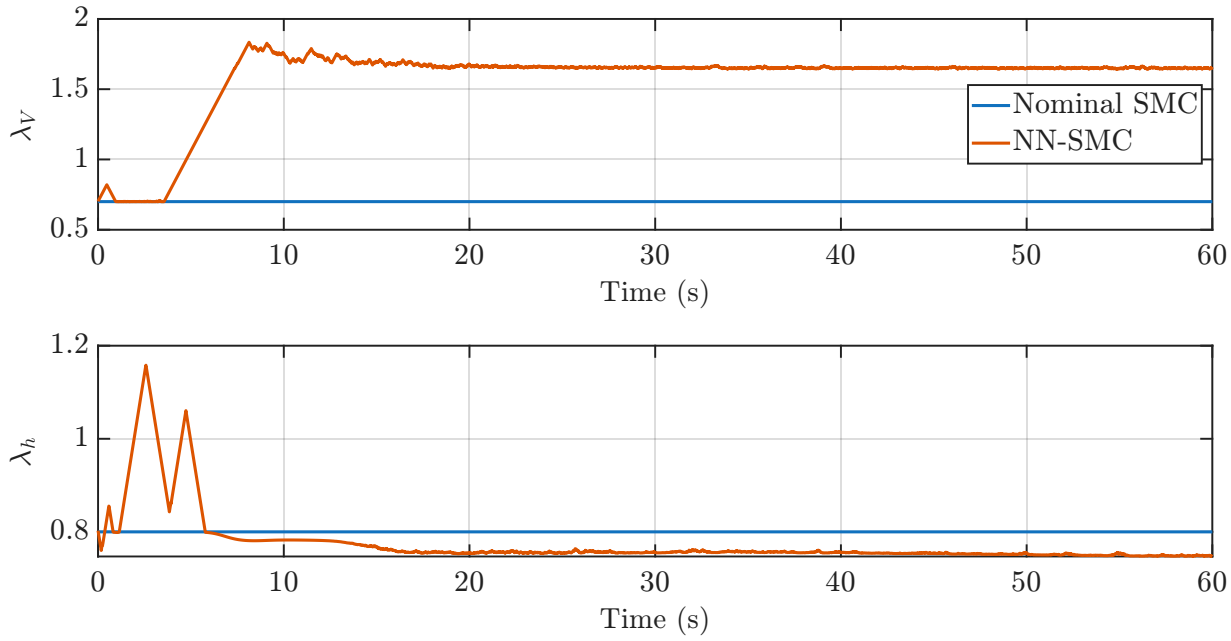


Figure 4 – Online variation of sliding surface parameters λ_V and λ_h.

error and the steady-state altitude error compared with Nominal SMC. Also, Fig. 3 and Table 2 show that this performance improvement is achieved without a large increase in control input. Finally, Fig. 4 confirms that the performance improvement is related to the adaptive change of λ_V and λ_h, which are adjusted online by the neural network.

5. Conclusion

This paper proposes an NN-SMC structure that combines a neural-network-based online tuner with an existing SMC to improve the longitudinal velocity and altitude tracking performance of a hypersonic vehicle. The proposed method preserves the basic input calculation structure of SMC. Its main feature is that the neural network adjusts the sliding surface parameters λ_V and λ_h in real time. This approach aims to reduce the limitation of Nominal SMC with fixed parameters and to adaptively adjust the convergence characteristics of the controller according to the flight state and tracking error.

The performance of the proposed method is validated using a hypersonic longitudinal plant model with parameter uncertainty. In the simulation, random uncertainties are applied to the mass, atmospheric density, reference area, pitch moment of inertia, mean aerodynamic chord, and elevator effectiveness coefficient. Nominal SMC and NN-SMC are compared under the condition of tracking given velocity and altitude reference commands. The performance evaluation is divided into tracking accuracy, control input usage, and parameter adjustment by the neural network tuner. The tracking accuracy is compared using RMSE and MAE calculated from the velocity and altitude responses and tracking errors. The control input usage is evaluated using the RMS and control energy of the throttle, elevator input, and thrust. The online variations of λ_V and λ_h in NN-SMC are also analyzed to confirm how the neural network tuner adjusts the sliding surface characteristics of each channel.

The simulation results show that NN-SMC provides a clear improvement over Nominal SMC, especially in altitude tracking performance. In the velocity channel, the RMSE increases slightly because of the initial transient error, but the MAE decreases, which indicates a reduction in the steady-state error. In the altitude channel, both RMSE and MAE decrease, and the steady-state altitude error remaining in Nominal SMC is effectively reduced by NN-SMC. In addition, the control input RMS and control energy remain at similar levels for the two controllers. This confirms that the proposed NN-SMC can achieve better tracking performance without a large increase in control effort.

Future work will improve the stability analysis of the proposed online tuner and test its generalization performance under various uncertainty and disturbance conditions. In addition, the learning signal will be extended to include control input variation or chattering suppression terms so that both tracking performance and control input stability can be improved.

Acknowledgement

This research was supported by the Regional Innovation System & Education(RISE) program through the (Chungbuk Regional Innovation System & Education Center), funded by the Ministry of Education(MOE) and the (Chungcheongbuk-do), Republic of Korea.(2026-RISE-11-014-01)

Contact Author Email Address

The contact author email addresses are as follows:
Haejun Kim: skh1111@chungbuk.ac.kr
Jongho Shin: jshin@chungbuk.ac.kr

Copyright Statement

The authors confirm that they, and/or their company or organization, hold copyright on all of the original material included in this paper. The authors also confirm that they have obtained permission, from the copyright holder of any third party material included in this paper, to publish it as part of their paper. The authors confirm that they give permission, or have obtained permission from the copyright holder of this paper, for the publication and distribution of this paper as part of the ICAS proceedings or as individual off-prints from the proceedings.

References

[1] Fang Wang, Linzhi Wen, Chao Zhou, and Changchun Hua. Adaptive preassigned-time controller design for a hypersonic vehicle with improved performance and uncertainties. *ISA transactions*, 132:309–328, 2023.

[2] Guan Wang, Hao An, Yiming Wang, Hongwei Xia, and Guangcheng Ma. Intelligent control of air-breathing hypersonic vehicles subject to path and angle-of-attack constraints. *Acta Astronautica*, 198:606–616, 2022.

[3] Wenxi Xu, Liguo Tan, Yan Zhen, and Mingzhe Hou. Intelligent parameter tuning sliding mode control for hypersonic vehicle based on the soft actor-critic. *IFAC-PapersOnLine*, 59(20):1338–1343, 2025.

[4] Rongyi Guo, Yibo Ding, and Xiaokui Yue. Active adaptive continuous nonsingular terminal sliding mode controller for hypersonic vehicle. *Aerospace Science and Technology*, 137:108279, 2023.

[5] Jinchao Shao, Wei-Wei Che, and Ke Shao. Nonlinear prescribed performance sliding mode control of hypersonic vehicles. *International Journal of Robust and Nonlinear Control*, 34(14):9928–9948, 2024.

[6] Chaoran Qu, Lin Cheng, Shengping Gong, and Xu Huang. Dynamic-matching adaptive sliding mode control for hypersonic vehicles. *Aerospace Science and Technology*, 149:109159, 2024.

[7] Rongsheng Xia, Mou Chen, Qingxian Wu, and Yuhui Wang. Neural network based integral sliding mode optimal flight control of near space hypersonic vehicle. *Neurocomputing*, 379:41–52, 2020.

[8] Mir Mikael Fatemi and Adel Akbarimajd. Adaptive sliding mode control for quadrotor uavs under disturbances using multi-layer perceptron. *IEEE Access*, 2025.

[9] Christopher I Marrison and Robert F Stengel. Design of robust control systems for a hypersonic aircraft. *Journal of Guidance, Control, and Dynamics*, 21(1):58–63, 1998.

[10] Qian Wang and Robert F Stengel. Robust nonlinear control of a hypersonic aircraft. *Journal of guidance, control, and dynamics*, 23(4):577–585, 2000.

[11] Jianli Wei and Huan Chen. Designing backstepping control system for hypersonic vehicle based on feedback linearization. *International Journal of Aerospace Engineering*, 2015(1):916328, 2015.

[12] Haojian Xu, Maj D Mirmirani, and Petros A Ioannou. Adaptive sliding mode control design for a hypersonic flight vehicle. *Journal of guidance, control, and dynamics*, 27(5):829–838, 2004.