Neurocomputing 128 (2014) 232–241

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Discrete-time hypersonic flight control based on extreme learning machine

Bin Xu [a,b,*], Yongping Pan [b], Danwei Wang [b], Fuchun Sun [c]

[a] *School of Automation, Northwestern Polytechnical University, Shaanxi, Xi'an 710072, China*
[b] *School of Electrical and Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798, Singapore*
[c] *Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China*

ARTICLE INFO

Article history:
Received 28 August 2012
Received in revised form
16 December 2012
Accepted 11 February 2013
Available online 28 October 2013

Keywords:
Hypersonic aircraft
Extreme learning machine
Neural networks
Single-hidden layer feedforward network

ABSTRACT

This paper describes the neural controller design for the longitudinal dynamics of a generic hypersonic flight vehicle (HFV). The dynamics are transformed into the strict-feedback form. Considering the uncertainty, the neural controller is constructed based on the single-hidden layer feedforward network (SLFN). The hidden node parameters are modified using extreme learning machine (ELM) by assigning random values. Instead of using online sequential learning algorithm (OSLA), the output weight is updated based on the Lyapunov synthesis approach to guarantee the stability of closed-loop system. By estimating the bound of output weight vector, a novel back-stepping design is presented where less online parameters are required to be tuned. The simulation study is presented to show the effectiveness of the proposed control approach.

1. Introduction

Hypersonic flight vehicles may offer a reliable and more cost efficient way to access space by reducing flight time. Also quick response and global attack became possible. Hypersonic flight control is challenging since the longitudinal model of the dynamics is known to be unstable, non-minimum phase with respect to the regulated output, and affected by significant model uncertainty. The main difficulty of the control law design for the hypersonic aircraft is due to the high complexity of the motion equations and there is little knowledge of the aerodynamic parameters of the vehicle.

Recently adaptive control and robust control are popularly studied on hypersonic flight controller design [1]. Back-stepping design [2] is an explicit tool for systematic nonlinear design. The HFV dynamics are written in the linearly parameterized form [3] and then robust adaptive dynamic inversion with back-stepping arguments is conducted. Dynamic surface control with control inputs saturation design is studied in [4].

Intelligent control is one important aspect for hypersonic flight control since it is with the capability of uncertainty approximation [5–7,22]. Since modern aircrafts are equipped with digital computers, the controller should be designed in discrete-time form [8]. Controller on the basis of continuous system is usually implemented by a digital computer with a certain sampling interval [9]. There are two methods for designing the digital controller. One method, called emulation, designs a controller with the continuous-time system, and then discretizes the controller. The other is to design the controllers directly based on the discrete system. In contrast to the emulation method, the discrete controller is designed in a discrete domain so that the performance of the controller may not depend on the sampling rate and the upper bounds of the neural network (NN) weight update rates guaranteeing the convergence can be estimated analytically while emulation method is otherwise [10].

Focused on discrete time design, the adaptive NN back-stepping HFV control [11] is studied to deal with the system uncertainty. The Kriging based adaptive controller is designed in [12] where the uncertainty is described as the realization of the Gaussian random functions [13]. The simulation shows the effectiveness of the controller design [11,14]. In the above schemes, the structure of NN is determined according to some prior information regarding the system to be approximated. Then the stable adaptive laws can be generated in a linear fashion. However in practice, systems are time-varying and the prior information is difficult to obtain. In this case, the exact values for the NN are hard to determine.

In this paper, a new stable neural control scheme is presented. The SLFN with RBF nodes is used as the function approximator to estimate the unknown nonlinearity. Different from the existing methods, the parameters of SLFN are adjusted based on the ELM. ELM has attracted widespread concern in recent years [15–17]

[*] Corresponding author at: School of Automation, Northwestern Polytechnical University, Shaanxi, Xi'an 710072, China. Tel.: +86 29 88431389.
E-mail address: smileface.binxu@gmail.com (B. Xu).

0925-2312/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.neucom.2013.02.049

since it overcomes some challenges faced by other techniques [18] such as (1) slow learning speed, (2) trivial human intervene, and/or (3) poor computational scalability. ELM works for generalized SLFN. The essence of ELM is that the hidden layer of SLFN needs not to be tuned. Compared with those traditional computational intelligence techniques, ELM provides better generalization performance at a much faster learning speed and with least human intervene. In [19], an approach for performing regression on large data sets in reasonable time is proposed using an ensemble of ELMs and the experiments show that competitive performance is obtained on the regression tasks. In [20], the ELM is utilized to train the controller by randomly assigning the parameters of hidden nodes. The output weights are synthesized using a Lyapunov function for guaranteeing the stability of the closed-loop system. Also it is indicated that original ELM cannot follow its reference trajectory well since the original ELM lacks stability proof of the whole control system and thus the convergence of the tracking error cannot be satisfied.

In this paper, considering the use of digital computer, the backstepping controller is designed with ELM by randomly assigning the parameters of hidden nodes. The updating law is designed with Lyapunov synthesis approach in discrete-time. Following the functional decomposition [11], we design the controller separately for the subsystems. Furthermore, the "minimal learning parameter" technique based on bound estimation of weight vector [14,21] is incorporated to reduce the computation burden. In this paper, only the cruise trajectories are considered for the control problem in this paper and we does not consider the ascent or the reentry of the vehicle.

This paper is organized as follows. Section 2 describes the longitudinal dynamics of a generic hypersonic flight vehicle. The strict-feedback form is formulated and the discrete analysis model is obtained in Section 3. SLFN based on ELM is illustrated in Section 4. Section 5 presents the adaptive controller design based on ELM. The weight bound estimation based controller is designed in Section 6. The simulation result is included in Section 7. Section 8 presents several comments and final remarks.

2. Hypersonic air vehicle model

The model of the longitudinal dynamics of a generic hypersonic aircraft in [1] is considered. This model is composed of five state variables $\mathbf{X}=[V,h,\alpha,\gamma,q]^T$ and two control inputs $\mathbf{U}_c=[\delta_e,\Phi]^T$ where V is the velocity, γ is the flight path angle, h is the altitude, α is the attack angle, q is the pitch rate, δ_e is the elevator deflection and Φ is the throttle setting.

The dynamics of hypersonic aircraft are described by the following nonlinear equations:

$$\dot{V}=\frac{T\cos\alpha-D}{m}-\frac{\mu\sin\gamma}{r^2} \tag{1}$$

$$\dot{h}=V\sin\gamma \tag{2}$$

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

$$\dot{\alpha}=q-\dot{\gamma} \tag{4}$$

$$\dot{q}=\frac{M_{yy}}{I_{yy}} \tag{5}$$

where T, D, L and M_{yy} represent thrust, drag, lift-force and pitching moment respectively, m, I_{yy} and μ represent the mass of aircraft, moment of inertia about pitch axis and gravity constant. r is the radial distance from center of the earth.

Refer to Appendix A for more information about the model.

3. System transformation

3.1. Strict-feedback formulation

Referred to [11,14], the formulation of the subsystems is presented in (6) and (8). The related definition of the system is listed in Appendix B.

The velocity subsystem (1) can be rewritten as follows:

$$\begin{aligned}&\dot{V}=f_v+g_vu_v\\&u_v=\Phi\\&y_v=V\end{aligned} \tag{6}$$

The tracking error of the altitude is defined as $\tilde{h}=h-h_d$ and the flight path command is chosen as

$$\gamma_d=\arcsin\left[\frac{-k_h(h-h_d)-k_I\int(h-h_d)\,dt+\dot{h}_d}{V}\right] \tag{7}$$

if $k_h>0$ and $k_I>0$ are chosen and the flight path angle is controlled to follow γ_d, the altitude tracking error is regulated to zero exponentially [5].

Assumption 1. The thrust term $T\sin\alpha$ in (3) is neglected because it is generally much smaller than L.

Define $\mathbf{X}_A=[x_1,x_2,x_3]^T$, $x_1=\gamma$, $x_2=\theta_p$, $x_3=q$ where $\theta_p=\alpha+\gamma$. Then the strict-feedback form equations of the attitude subsystem (3)–(5) are written as

$$\begin{aligned}&\dot{x}_1=f_1(x_1)+g_1(x_1)x_2\\&\dot{x}_2=f_2(x_1,x_2)+g_2(x_1,x_2)x_3\\&\dot{x}_3=f_3(x_1,x_2,x_3)+g_3(x_1,x_2,x_3)u_A\\&u_A=\delta_e\\&y=x_1\end{aligned} \tag{8}$$

Assumption 2. f_i and g_i are unknown smooth functions. There exist known constants $\overline{g_i}$ and $\underline{g}_i$ such that $\overline{g}_i\ge g_i\ge\underline{g}_i>0$, $i=1,\ 3,\ v$.

Remark 1. From Appendix B, the f_i is really complicated and it is considered to be totally unknown. $g(k)$ is time varying however according to the parameter perturbation, the bound of gain is easy to derive with the simple linear expression.

The goal pursued in this study is to design a dynamic controller δ_e and Φ to steer system altitude and velocity from a given set of initial values to desired trim conditions with the tracking reference h_d and V_d. With the command transformation (7), the control objective of system (8) is to design an adaptive controller, which makes $\gamma\to\gamma_d$, further $h\to h_d$ and all the signals involved are bounded.

3.2. Discrete-time model

By Euler expansion with sample time T_s, systems (6) and (8) can be approximated as

$$V(k+1)=V(k)+T_s[f_v(k)+g_v(k)u_v(k)] \tag{9}$$

$$\begin{aligned}&x_1(k+1)=x_1(k)+T_s[f_1(k)+g_1(k)x_2(k)]\\&x_2(k+1)=x_2(k)+T_s[f_2(k)+g_2(k)x_3(k)]\\&x_3(k+1)=x_3(k)+T_s[f_3(k)+g_3(k)u_A(k)]\end{aligned} \tag{10}$$

4. SLFN based on ELM

For N arbitrary distinct samples $(\mathbf{x}_i,\mathbf{t}_i)$, where $\mathbf{x}_i=[x_{i1},x_{i2},\ldots,x_{in}]^T\in R^n$ and $\mathbf{t}_i=[t_{i1},t_{i2},\ldots,t_{im}]^T\in R^m$, standard SLFN with

$\tilde{N}$ hidden neurons can be expressed as follows:

$$\sum_{i=1}^{\tilde{N}} \beta_i G(\mathbf{x}_j;\mathbf{w}_i,b_i)=\mathbf{o}_j,\quad j=1,\dots,N \tag{11}$$

where $\mathbf{w}_i=[w_{i1},w_{i2},\dots,w_{in}]^T$ and b_i are the learning parameters of hidden nodes, $\beta_i=[\beta_{i1},\beta_{i2},\dots,\beta_{im}]^T$ is the weight vector connecting the ith hidden neuron and the output neurons and $G(\mathbf{x}_j;\mathbf{w}_i,b_i)$ is the output of the ith hidden node with respect to input $\mathbf{x}_j$.

Let $\mathbf{x}=[\mathbf{x}_1,\mathbf{x}_2,\dots,\mathbf{x}_N]$, $\mathbf{w}=[\mathbf{w}_1,\mathbf{w}_2,\dots,\mathbf{w}_{\tilde{N}}]$ and $\mathbf{b}=[b_1,b_2,\dots,b_{\tilde{N}}]$. The standard SLFN with $\tilde{N}$ hidden neurons each with function $g(x)$ can approximate these N samples with zero error means that $\sum_{j=1}^{N}\|\mathbf{o}_j-\mathbf{t}_j\|=0$, *i.e.*, there exist β_i, $\mathbf{w}_i$ and b_i such that

$$\mathbf{H}(\mathbf{x};\mathbf{w},\mathbf{b})\beta=\mathbf{T} \tag{12}$$

in which

$$\mathbf{H}(\mathbf{x}_1,\dots,\mathbf{x}_N,\mathbf{w}_1,\dots,\mathbf{w}_{\tilde{N}},b_1,\dots,b_{\tilde{N}}) = \begin{bmatrix} G(\mathbf{x}_1;\mathbf{w}_1,b_1) & \cdots & G(\mathbf{x}_1;\mathbf{w}_{\tilde{N}},b_{\tilde{N}}) \\ \vdots & \cdots & \vdots \\ G(\mathbf{x}_N;\mathbf{w}_1,b_1) & \cdots & G(\mathbf{x}_N;\mathbf{w}_{\tilde{N}},b_{\tilde{N}}) \end{bmatrix}_{N\times\tilde{N}},$$

$$\beta=\begin{bmatrix}\beta_1^T\\ \vdots \\ \beta_{\tilde{N}}^T\end{bmatrix}_{\tilde{N}\times m}$$

and

$$\mathbf{T}=\begin{bmatrix}\mathbf{t}_1^T\\ \vdots \\ \mathbf{t}_N^T\end{bmatrix}_{N\times m}$$

In the case of $\tilde{N}\ll N$ and $\mathbf{H}$ being a nonsquare matrix, one may be interested to find $\hat{\mathbf{w}}_i,\hat{b}_i,\hat{\beta}_i$ $(i=1,\dots,\tilde{N})$ such that

$$\begin{aligned}&\|\mathbf{H}(\hat{\mathbf{w}}_1,\dots,\hat{\mathbf{w}}_{\tilde{N}},b_1,\dots,b_{\tilde{N}})\hat{\beta}-\mathbf{T}\|\\ &\quad=\min_{\mathbf{w}_i,b_i,\beta}\|\mathbf{H}(\mathbf{w}_1,\dots,\mathbf{w}_{\tilde{N}},b_1,\dots,b_{\tilde{N}})\beta-\mathbf{T}\|\end{aligned} \tag{13}$$

The popular back-propagation learning algorithm has several issues such as local minimal, overfitting, slow convergence rate. ELM is an efficient approach to solve the above issues. The essentials of ELM are as follows: (1) randomly generate hidden node without the knowledge of the training data; (2) learn without iterative tuning since it only adjusts the output weights; (3) minimize both MSE and adjusting parameters. For fixed input weights $\mathbf{w}_i$ and the hidden layer biases b_i, to train an SLFN is simply equivalent to finding a least-squares solution β of the linear system $\mathbf{H}\beta=\mathbf{T}$:

$$\begin{aligned}&\|\mathbf{H}(\mathbf{w}_1,\dots,\mathbf{w}_{\tilde{N}},b_1,\dots,b_{\tilde{N}})\beta-\mathbf{T}\|\\ &\quad=\min_{\beta}\|\mathbf{H}(\mathbf{w}_1,\dots,\mathbf{w}_{\tilde{N}},b_1,\dots,b_{\tilde{N}})\beta-\mathbf{T}\|\end{aligned} \tag{14}$$

The unique smallest norm least-squares solution of the above linear system is

$$\hat{\beta}=\mathbf{H}^{\dagger}\mathbf{T} \tag{15}$$

For RBF hidden nodes with Gaussian function $g(\cdot)$, $G(\mathbf{x}_j;\mathbf{w}_i,b_i)$ is given by

$$G(\mathbf{x}_j;\mathbf{w}_i,b_i)=g(b_i\|\mathbf{x}_j-\mathbf{w}_i\|) \tag{16}$$

where $\mathbf{w}_i$ and b_i are the center and impact factor of the ith RBF node, respectively. The RBF network is a special case of SLFN with RBF nodes in its hidden layer. Each RBF node has its own centroid and impact factor and its output is given by a radially symmetric function of the distance between the input and the center.

5. Adaptive control with ELM

When the SLFN is utilized to approximate the unknown nonlinear functions in the designed control scheme, according to the property of ELM, the hidden node parameters need not be tuned during training and may simply be assigned with random values.

The error definition is presented:

$$z_1(k)=x_1(k)-x_{1d}(k) \tag{17}$$

$$z_2(k)=x_2(k)-x_{2d}(k) \tag{18}$$

$$z_3(k)=x_3(k)-x_{3d}(k) \tag{19}$$

where $x_{1d}(k)$ is derived from (7), $x_{2d}(k)$, $x_{3d}(k)$ are the virtual control inputs to be designed.

For simplicity, we define $G_i(k)=T_sg_i(k)$, $\overline{G}_i=T_s\overline{g}_i$, $G_{iN}(k)=T_sg_{iN}(k)$, $\varepsilon_i(k)=\varepsilon_i(\mathbf{X}_i(k))$, $\theta_i(k)=\theta_i(\mathbf{X}_i(k))$, $i=1,2,3,v$.

The related controller design is illustrated in the following 3 steps:

Step 1: From the definition of $z_1(k)$ in (17), we have

$$z_1(k+1)=x_1(k)+T_s[f_1(k)+g_1(k)x_2(k)]-x_{1d}(k+1) \tag{20}$$

where $x_{1d}(k+1)$ is acquired from (7).

Define $\mathbf{X}_1(k)=[V(k),x_1(k),h_d(k),h_d(k+1)]^T$. The uncertainty is defined as

$$\begin{aligned}U_1(k)&=\frac{-x_1(k)-T_sf_1(k)+x_{1d}(k+1)}{G_1(k)}+\frac{z_1(k)}{\overline{G}_1}\\ &=\mathbf{H}_1(\mathbf{X}_1(k),\mathbf{w}_1,\mathbf{b}_1)\beta_1^*+\varepsilon_1(\mathbf{X}_1(k))\end{aligned} \tag{21}$$

where β_1^* is the optimal parameters for SLFN to approximate $U_1(k)$ while $\varepsilon_1(\mathbf{X}_1(k))$ is the SLFN reconstruction error.

For simplicity, we remark $\mathbf{H}_i(\mathbf{X}_i(k),\mathbf{w}_i,\mathbf{b}_i)\beta_i^*$ and $\varepsilon_i(\mathbf{X}_i(k))$ as $\mathbf{H}_i(k)$ and $\varepsilon_i(k)$ respectively.

Take $x_2(k)$ in (20) as the virtual control input and design its desired value as

$$x_{2d}(k)=\frac{(c_1-1)z_1(k)}{\overline{G}_1}+\mathbf{H}_1(k)\hat{\beta}_1(k) \tag{22}$$

with $0<c_1<1$, $\hat{\beta}_1(k)$ is the estimation of β_1^*.

Combining (18), (20) and (22), the following equation can be obtained:

$$\begin{aligned}z_1(k+1)&=x_1(k)+T_s[f_1(k)+g_1(k)x_2(k)]-x_{1d}(k+1)\\ &=G_1(k)z_2(k)+G_1(k)\left[x_{2d}(k)-\frac{-x_1(k)-T_sf_1(k)+x_{1d}(k+1)}{G_1(k)}\right]\\ &=G_1(k)z_2(k)+G_1(k)\left[\begin{array}{l}\dfrac{(c_1-1)z_1(k)}{\overline{G}_1}+\mathbf{H}_1(k)\hat{\beta}_1(k)\\ -\dfrac{-x_1(k)-T_sf_1(k)+x_{1d}(k+1)}{G_1(k)}\end{array}\right]\\ &=G_1(k)z_2(k)+\frac{G_1(k)}{\overline{G}_1}c_1z_1(k)+G_1(k)(\mathbf{H}_1(k)\tilde{\beta}_1(k)-\varepsilon_1(k))\end{aligned} \tag{23}$$

where $\tilde{\beta}_1(k)=\hat{\beta}_1(k)-\beta_1^*$.

The robust adaption law is designed as

$$\hat{\beta}_1^T(k+1)=\hat{\beta}_1^T(k)-\lambda_1\mathbf{H}_1(k)z_1(k+1)-\delta_1\hat{\beta}_1^T(k) \tag{24}$$

where $\lambda_1>0$ and $0<\delta_1<1$.

Step 2: From (18), we know

$$z_2(k+1)=x_2(k)+T_s[f_2(k)+g_2(k)x_3(k)]-x_{2d}(k+1) \tag{25}$$

Define $\mathbf{X}_2(k)=[V(k),x_1(k),x_2(k),h_d(k),h_d(k+1),h_d(k+2)]^T$.

The uncertainty $U_2(k)$ is defined and can be approximated by NN as

$$\begin{aligned}U_2(k)&=\frac{-x_2(k)-T_sf_2(k)+x_{2d}(k+1)}{G_2(k)}-\frac{z_2(k)}{\overline{G}_2}\\ &=H_2(\mathbf{X}_2(k),\mathbf{w}_2,\mathbf{b}_2)\beta_2^*+\varepsilon_2(\mathbf{X}_2(k))\end{aligned} \tag{26}$$

where β_2^* is the optimal parameters for SLFN to approximate $U_2(k)$ while $\varepsilon_2(k)$ is the SLFN reconstruction error.

Take $x_3(k)$ in (25) as the virtual control input and design its desired value as

$$x_{3d}(k)=\frac{(c_2-1)z_2(k)}{\overline{G}_2}+\mathbf{H}_2(k)\hat{\beta}_2(k) \tag{27}$$

with $0<c_2<1$, $\hat{\beta}_2(k)$ is the estimation of β_2^*.

The error dynamics are derived as

$$z_2(k+1)=G_2(k)z_2(k)+\frac{G_2(k)}{\overline{G}_2}c_2z_2(k)+G_2(k)(\mathbf{H}_2(k)\tilde{\beta}_2(k)-\varepsilon_2(k)) \tag{28}$$

where $\tilde{\beta}_2(k)=\hat{\beta}_2(k)-\beta_2^*$.

The adaption law is

$$\hat{\beta}_2^T(k+1)=\hat{\beta}_2^T(k)-\lambda_2\mathbf{H}_2(k)z_2(k+1)-\delta_2\hat{\beta}_2^T(k) \tag{29}$$

where $\lambda_2>0$ and $0<\delta_2<1$.

Step 3: From (19), we know

$$z_3(k+1)=x_3(k)+T_s[f_3(k)+g_3(k)u_A(k)]-x_{3d}(k+1) \tag{30}$$

Define $\mathbf{X}_3(k)=[V(k),\mathbf{X}_A^T(k),h_d(k),h_d(k+1),h_d(k+2),h_d(k+3)]^T$:

$$\begin{aligned}U_3(k)&=\frac{-x_3(k)-T_sf_3(k)+x_{3d}(k+1)}{G_3(k)}+\frac{z_3(k)}{\overline{G}_3}\\&=\mathbf{H}_3(\mathbf{X}_3(k),\mathbf{w}_3,\mathbf{b}_3)\beta_3^*+\varepsilon_3(\mathbf{X}_3(k))\end{aligned} \tag{31}$$

where β_3^* is the optimal parameters for SLFN to approximate $U_3(k)$ while $\varepsilon_3(k)$ is the SLFN reconstruction error.

The elevator deflection is designed as

$$u_A(k)=\frac{(c_3-1)z_3(k)}{\overline{G}_3(k)}+\mathbf{H}_3(k)\hat{\beta}_3(k) \tag{32}$$

with $0<c_3<1$, $\hat{\beta}_3(k)$ is the estimation of β_3^*.

The error dynamics are derived as

$$z_3(k+1)=\frac{G_3(k)}{\overline{G}_3}c_3z_3(k)+G_3(k)(\mathbf{H}_3(k)\tilde{\beta}_3(k)-\varepsilon_3(k)) \tag{33}$$

where $\tilde{\beta}_3(k)=\hat{\beta}_3(k)-\beta_3^*$.

The adaption law is

$$\hat{\beta}_3^T(k+1)=\hat{\beta}_3^T(k)-\lambda_3\mathbf{H}_3(k)z_3(k+1)-\delta_3\hat{\beta}_3^T(k) \tag{34}$$

where $\lambda_3>0$ and $0<\delta_3<1$.

Theorem 5.1. *Considering system (10) with the controller (32) and the update law (24), (29), (34), all the signals involved are semiglobal uniformly ultimately bounded.*

Follow the procedure in [11], the proof could be done. It is very similar and thus omitted here.

For the velocity subsystem, we define $\mathbf{X}_v(k)=[V(k),\mathbf{X}_A^T(k),V_d(k+1)]^T$ and $z_v(k)=V(k)-V_d(k)$

$$z_v(k+1)=V(k)+T_s[f_v(k)+g_v(k)u_v(k)]-V_d(k+1) \tag{35}$$

The throttle setting is designed as

$$u_v(k)=\frac{-V(k)+c_vz_v(k)+V_d(k+1)}{\overline{G}_v}+\mathbf{H}_v(k)\hat{\beta}_v(k) \tag{36}$$

with $0<c_v<1$, $\hat{\beta}_v(k)$ is the estimation of β_v^*.

The robust updating law is

$$\hat{\beta}_v^T(k+1)=\hat{\beta}_v^T(k)-\lambda_v\mathbf{H}_v(k)z_v(k+1)-\delta_v\hat{\beta}_v^T(k) \tag{37}$$

where $\lambda_v>0$ and $0<\delta_v<1$.

Theorem 5.2. *Considering system (9) with the controller (36) and the update law (37), the velocity is semiglobal uniformly ultimately bounded.*

Similar to the analysis of altitude subsystem, the proof is omitted here.

It is noted that the tuning rule of output weight β in the OS-ELM is conducted from the least-square error method where the training data arrives one-by-one. But for the closed-loop system, the algorithm is designed to ensure the Lyapunov stability of the system. Though the modification of the output weight is different, the hidden node parameters of the SLFN are randomly determined like the original ELM algorithm [20].

6. Weight vector bound estimation based controller design

As indicated in [14], the neural design could be further simplified by estimating the upper bound of $\|\beta_i\|$. With this idea, we have the following definition:

Assumption 3. For each $i=1,2,\ldots,n$ on the compact set Ω_i, β_i^* satisfies

$$\|\beta_i^*\|\le\phi_i\ \mathrm{sgn}(\phi_i) \tag{38}$$

where $\phi_i\ \mathrm{sgn}(\phi_i)=1$ if $\phi_i\ge0$ and -1 otherwise.

Remark 2. In [21], the algorithm is with restriction since the estimation can be positive only. But actually the NN approximation could be negative. Using the $\mathrm{sgn}(\cdot)$ function and redefining the assumption, it is more reasonable that the ϕ_i can be either positive or negative.

In the following, we design the controller with less online adaptive parameters. The procedure for the back-stepping design and the uncertainty definition for each subsystem are the same as in Section 5. To clearly demonstrate the idea, only the first step is specifically illustrated.

Step 1: From the definition of $z_1(k)$ in (17), we have

$$z_1(k+1)=x_1(k)+T_s[f_1(k)+g_1(k)x_2(k)]-x_{1d}(k+1) \tag{39}$$

where $x_{1d}(k+1)$ is acquired from (7).

Define $\mathbf{X}_1(k)=[V(k),x_1(k),h_d(k),h_d(k+1)]^T$. The uncertainty is defined as

$$\begin{aligned}U_1(k)&=\frac{-x_1(k)-T_sf_1(k)+x_{1d}(k+1)}{G_1(k)}+\frac{z_1(k)}{\overline{G}_1}\\&=\mathbf{H}_1(\mathbf{X}_1(k),\mathbf{w}_1,\mathbf{b}_1)\beta_1^*+\varepsilon_1(\mathbf{X}_1(k))\end{aligned} \tag{40}$$

where β_1^* is the optimal parameters for SLFN to approximate $U_1(k)$ while $\varepsilon_1(\mathbf{X}_1(k))$ is the SLFN reconstruction error.

Take $x_2(k)$ in (39) as the virtual control input and design its desired value as

$$x_{2d}(k)=\frac{(c_{11}-1)z_1(k)}{\overline{G}_1}+\frac{\phi_{s1}^2(k)}{\phi_{s1}(k)+\tau_1\ \mathrm{sgn}(\phi_{s1}(k))} \tag{41}$$

with $0<c_{11}<1$, $\tau_1>0$, $\phi_{s1}(k)=\hat{\phi}_1(k)\|\mathbf{H}_1(k)\|$ where $\hat{\phi}_1$ is the estimation of ϕ_1.

For the adaptive item, the transformation in [14] is modified as

$$\begin{aligned}\frac{\phi_{s1}^2(k)}{\phi_{s1}(k)+\tau_1\ \mathrm{sgn}(\phi_{s1}(k))}&=\frac{\phi_{s1}^2(k)}{\phi_{s1}(k)+\tau_1\ \mathrm{sgn}(\phi_{s1}(k))}-\phi_{s1}(k)\\&\quad+\phi_{s1}(k)-\phi_1\|\mathbf{H}_1(k)\|+\phi_1\|\mathbf{H}_1(k)\|\\&=\tilde{\phi}_1(k)\|\mathbf{H}_1(k)\|+\phi_1\|\mathbf{H}_1(k)\|-\frac{\tau_1\phi_{s1}(k)}{\phi_{s1}(k)+\tau_1}\end{aligned}$$

where $\tilde{\phi}_1=\hat{\phi}_1-\phi_1$.

Remark 3. With the new adaptive design and the estimation error, the transformation is closely related to the absolute value of the norm of SLFN weights. The confusion in [14,21] is eliminated.

The robust adaption law is designed as

$$\hat{\phi}_1(k+1)=\hat{\phi}_1(k)-\lambda_{11}\|\mathbf{H}_1(k)\|z_1(k+1)-\delta_{11}\hat{\phi}_1(k) \tag{42}$$

where $\lambda_{11}>0$ and $0<\delta_{11}<1$.

Combining (18), (39) and (41), the following equation can be obtained:

$$\begin{aligned}
z_1(k+1) &= x_1(k)+T_s[f_1(k)+g_1(k)x_2(k)]-x_{1d}(k+1)\\
&= G_1(k)z_2(k)+G_1(k)\left[x_{2d}(k)-\frac{-x_1(k)-T_sf_1(k)+x_{1d}(k+1)}{G_1(k)}\right]\\
&= G_1(k)z_2(k)+G_1(k)\left[\begin{array}{l}\dfrac{(c_{11}-1)z_1(k)}{\overline{G}_1}+\dfrac{\phi_{s1}^2(k)}{\phi_{s1}(k)+\tau_1\ \mathrm{sgn}(\phi_{s1}(k))}\\ -\dfrac{-x_1(k)-T_sf_1(k)+x_{1d}(k+1)}{G_1(k)}\end{array}\right]\\
&= G_1(k)z_2(k)+\frac{G_1(k)}{\overline{G}_1}c_{11}z_1(k)\\
&\quad +G_1(k)\Bigg(\tilde{\phi}_1(k)\|\mathbf{H}_1(k)\|+\phi_1\|\mathbf{H}_1(k)\|\\
&\quad -\frac{\tau_1\phi_{s1}(k)}{\phi_{s1}(k)+\tau_1}-\mathbf{H}_1(k)\beta_1^*(k)-\varepsilon_1(k)\Bigg)
\end{aligned}$$

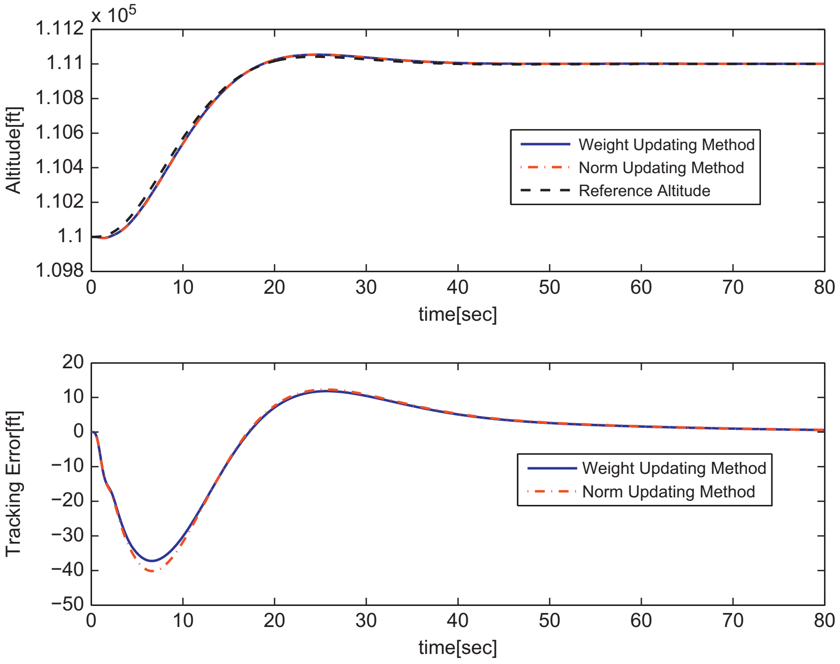

Fig. 1. Altitude tracking.

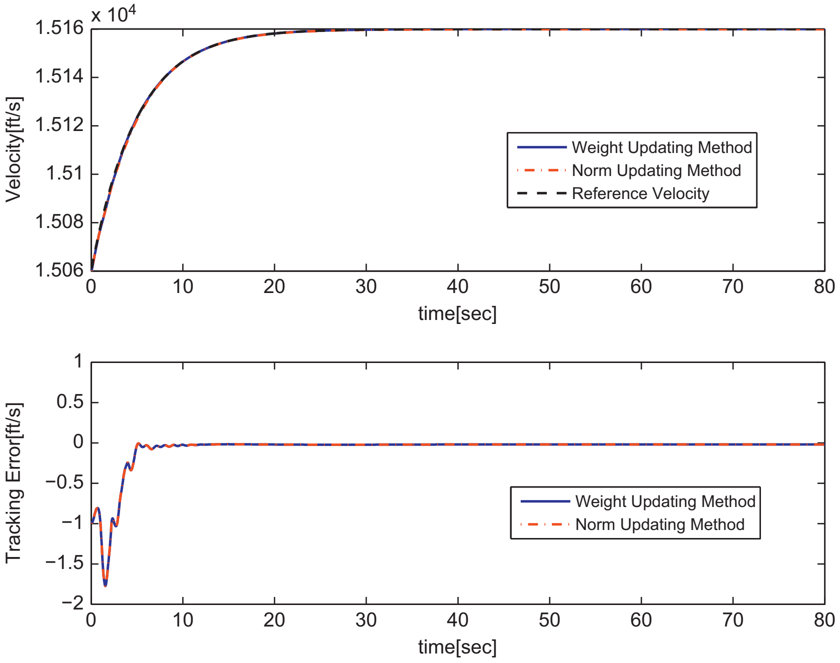

Fig. 2. Altitude tracking.

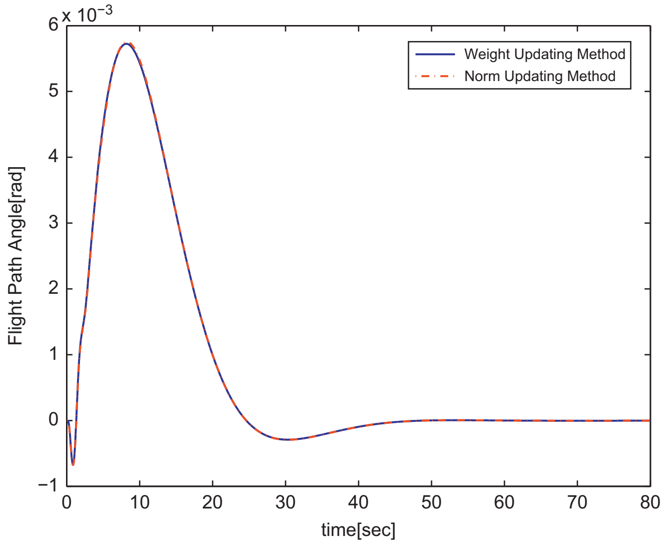

Fig. 3. Flight path angle.

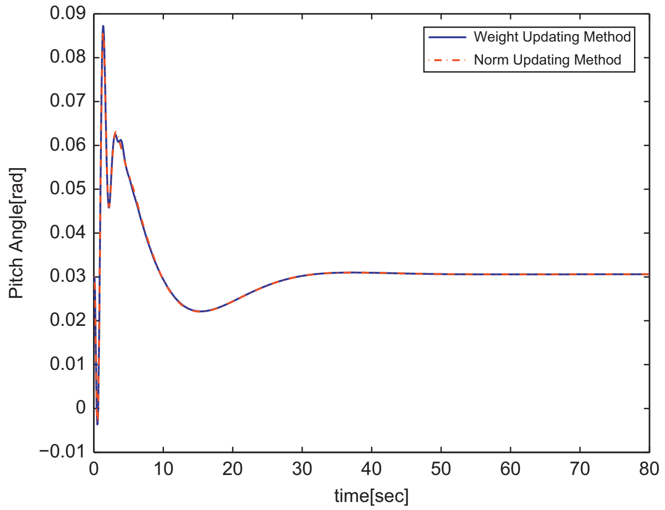

Fig. 4. Pitch angle.

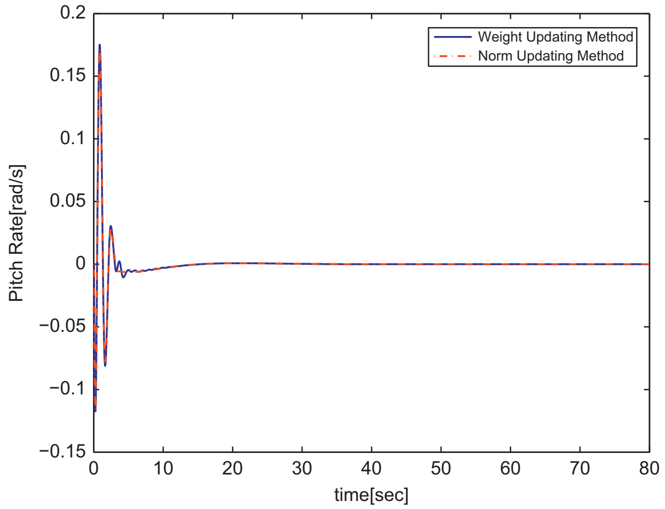

Fig. 5. Pitch rate.

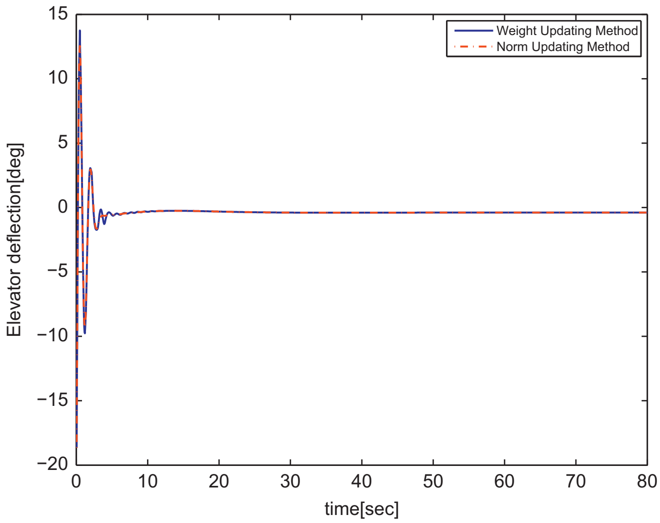

Fig. 6. Elevator deflection.

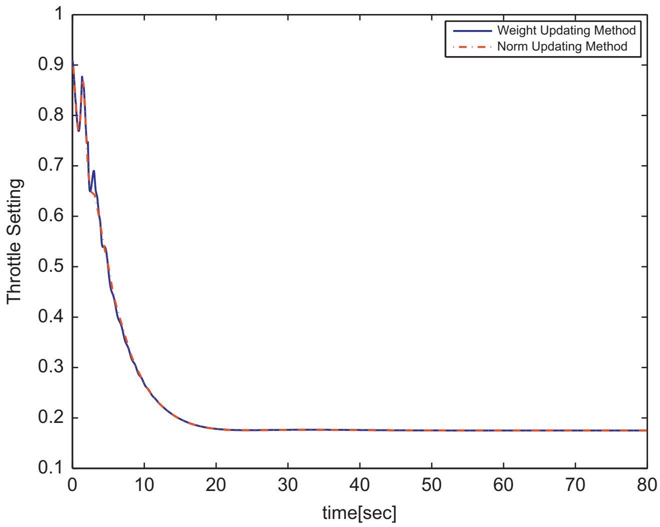

Fig. 7. Throttle setting.

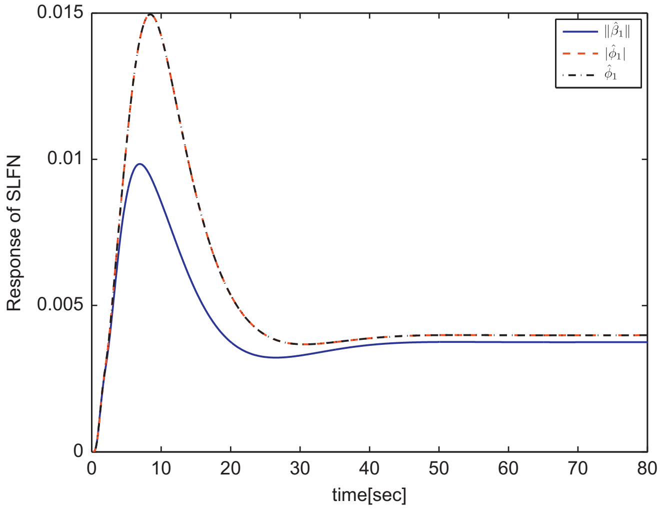

Fig. 8. SLFN response of x_{2d}.

Step 2: The controller is presented as

$$x_{3d}(k)=\frac{(c_{22}-1)z_2(k)}{\overline{G}_2}+\frac{\phi_{s2}^2(k)}{\phi_{s2}(k)+\tau_2\ \mathrm{sgn}(\phi_{s2}(k))} \tag{43}$$

with $0<c_{22}<1$, $\tau_2>0$, $\phi_{s2}(k)=\hat{\phi}_2(k)\|\mathbf{H}_2(k)\|$.

The adaption law is

$$\hat{\phi}_2(k+1)=\hat{\phi}_2(k)-\lambda_{22}\|\mathbf{H}_2(k)\|z_2(k+1)-\delta_{22}\hat{\phi}_2(k) \tag{44}$$

where $\lambda_{22}>0$ and $0<\delta_{22}<1$.

Step 3: The controller is presented as

$$u_A(k)=\frac{(c_{33}-1)z_3(k)}{\overline{G}_3}+\frac{\phi_{s3}^2(k)}{\phi_{s3}(k)+\tau_3\ \mathrm{sgn}(\phi_{s3}(k))} \tag{45}$$

with $0<c_{33}<1$, $\tau_3>0$, $\phi_{s3}(k)=\hat{\phi}_3(k)\|\mathbf{H}_3(k)\|$.

The adaption law is

$$\hat{\phi}_3(k+1)=\hat{\phi}_3(k)-\lambda_{33}\|\mathbf{H}_3(k)\|z_3(k+1)-\delta_{33}\hat{\phi}_3(k) \tag{46}$$

where $\lambda_{33}>0$ and $0<\delta_{33}<1$.

Theorem 6.1. *Considering system (10) with the controller (45) and the update law (42), (44), (46), all the signals involved are semiglobal uniformly ultimately bounded.*

The proof is easy to derive with the new error dynamics following the analysis in [14] and thus omitted here.

For the velocity subsystem, the controller is presented as

$$u_v(k)=\frac{-V(k)+c_{vv}z_v(k)+V_d(k+1)}{\overline{G}_v}+\frac{\phi_{sv}^2(k)}{\phi_{sv}(k)+\tau_v\ \mathrm{sgn}(\phi_{sv}(k))} \tag{47}$$

with $0<c_{vv}<1$, $\tau_v>0$, $\phi_{sv}(k)=\hat{\phi}_v(k)\|\mathbf{H}_v(k)\|$.

The adaption law is

$$\hat{\phi}_v(k+1)=\hat{\phi}_v(k)-\lambda_{vv}\|\mathbf{H}_v(k)\|z_v(k+1)-\delta_{vv}\hat{\phi}_v(k) \tag{48}$$

where $\lambda_{vv}>0$ and $0<\delta_{vv}<1$. □

Theorem 6.2. *Considering system (9) with the controller (47) and the update law (48), the velocity is semiglobal uniformly ultimately bounded.*

Similar to the analysis of altitude subsystem, the proof is omitted here.

Remark 4. The hidden node parameters of the SLFN are randomly determined like the original ELM algorithm. Furthermore, based on the design in Section 5 the new design with less online adaptive parameters is proposed to extend the notation of adaptive bound related estimation and overcome the singularity problem [21].

7. Simulations

The flight of the vehicle is at the condition $M=15$, $V=15{,}060$ ft/s, $h=110{,}000$ ft, $\alpha=0.03$ rad, $\gamma=0$, $q=0$. In this simulation, step commands of 1000 ft and 100 ft/s are selected for altitude and velocity separately. Reference commands are generated by the filter:

$$\frac{h_d}{h_c}=\frac{\omega_{n1}\omega_{n2}^2}{(s+\omega_{n1})(s^2+2\varepsilon_c\omega_{n2}s+\omega_{n2}^2)}$$

$$\frac{V_d}{V_c}=\frac{\omega_{v1}}{(s+\omega_{v1})}$$

where $\omega_{n1}=1$, $\omega_{n2}=0.2$, $\varepsilon_c=0.7$, $\omega_{v1}=0.2$. The perturbation is set to be 3% for the parameter set (m,μ,I_{yy},S).

To demonstrate the tracking performance, the control methods in Sections 5 and 6 are denoted as weight updating method and norm updating method respectively. The parameters for the ELM controller are selected as $k_h=2$, $k_I=0.1$, $T_s=0.05\,s$, $c_1=0.9$, $c_2=0.7$, $c_3=0.7$, $c_v=0.75$. The weight updating parameters

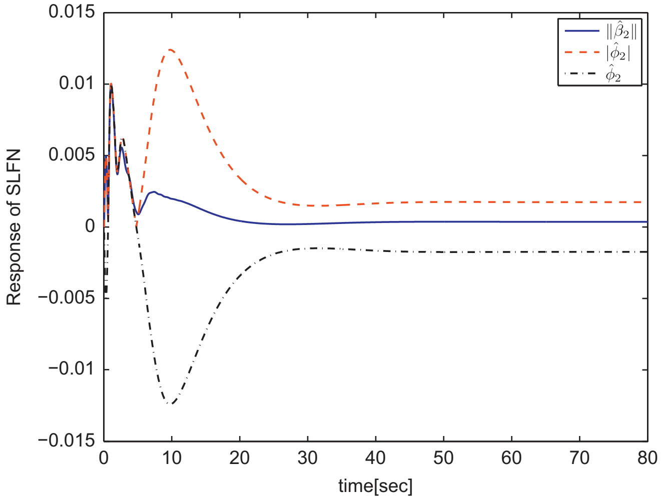

Fig. 9. SLFN response of x_{3d}.

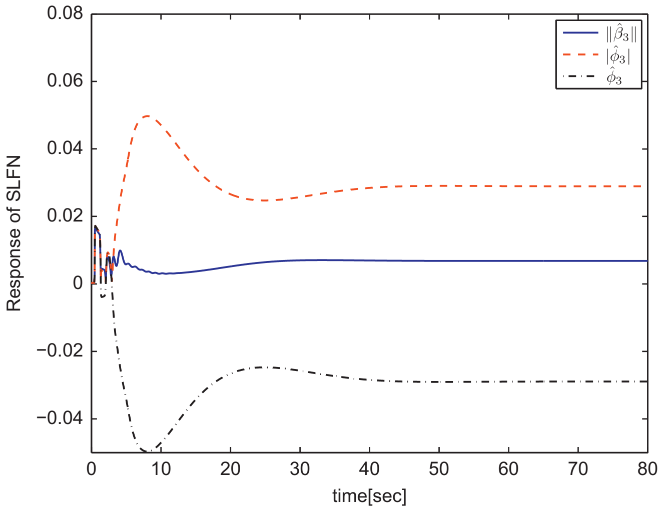

Fig. 10. SLFN response of u_A.

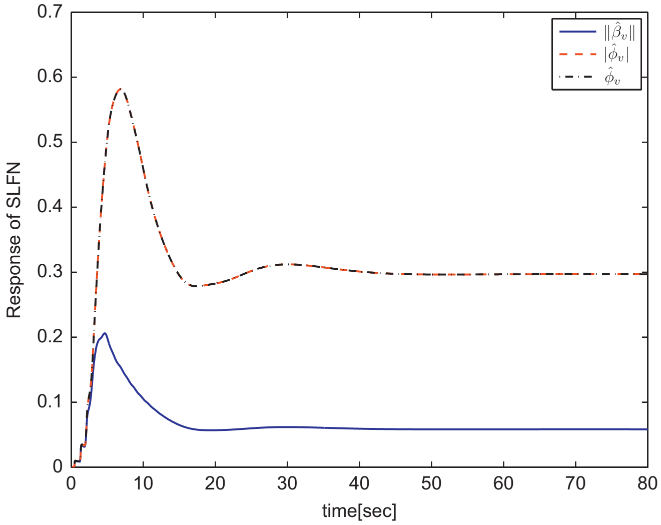

Fig. 11. SLFN response of u_v.

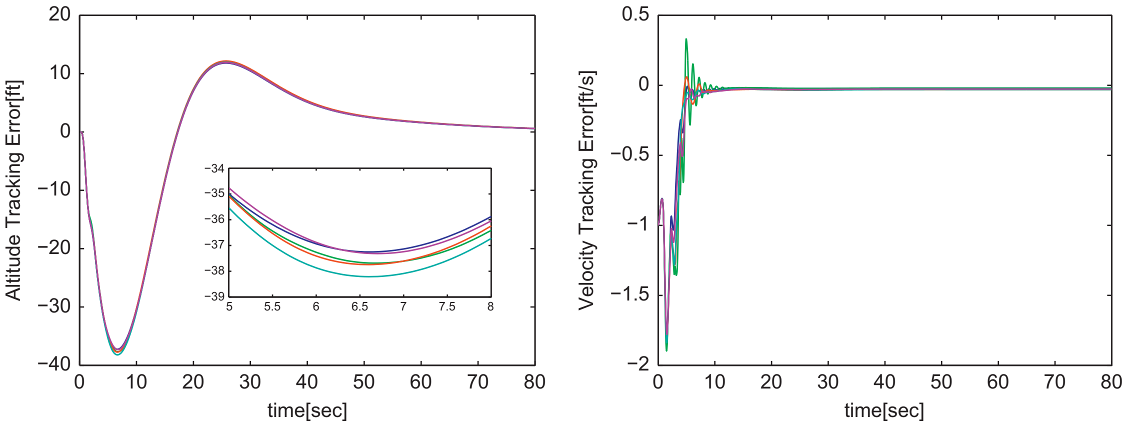

Fig. 12. System tracking of weight updating method.

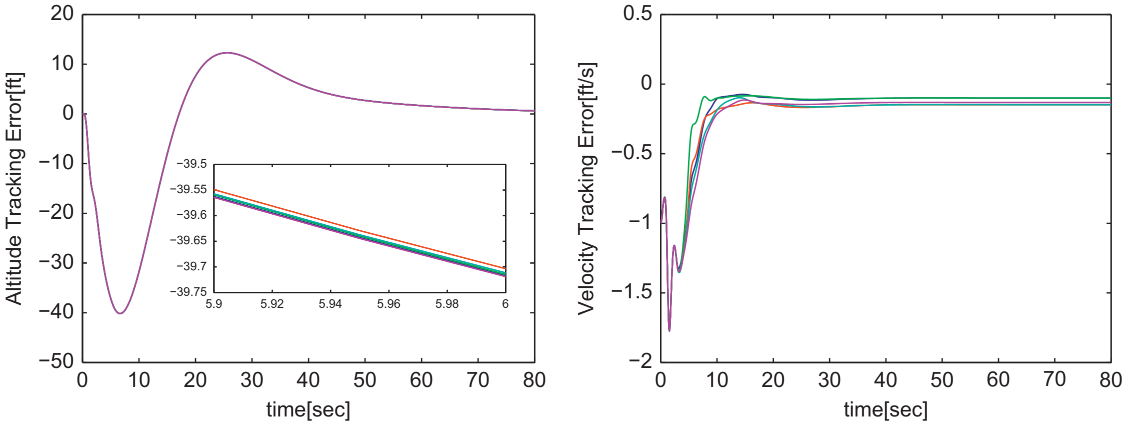

Fig. 13. System tracking of norm updating method.

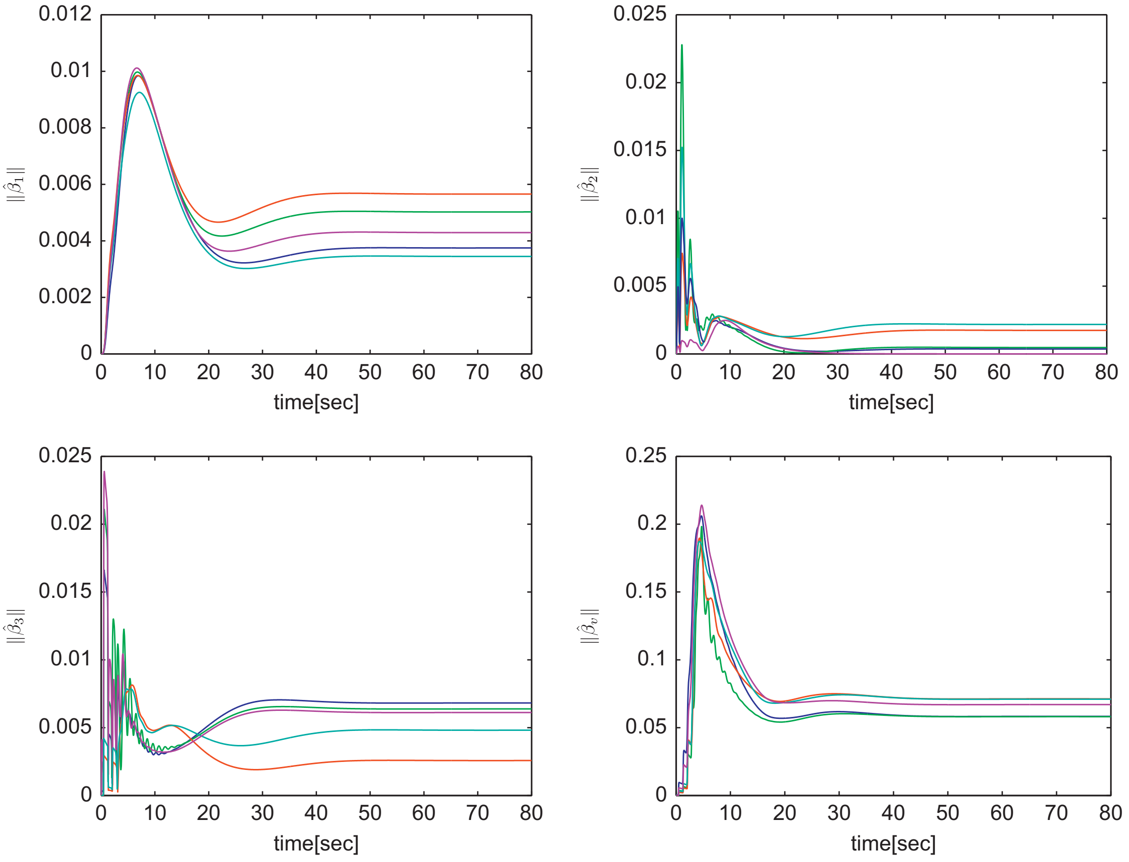

Fig. 14. SLFN response of $\|\hat{\beta}_i\|$.

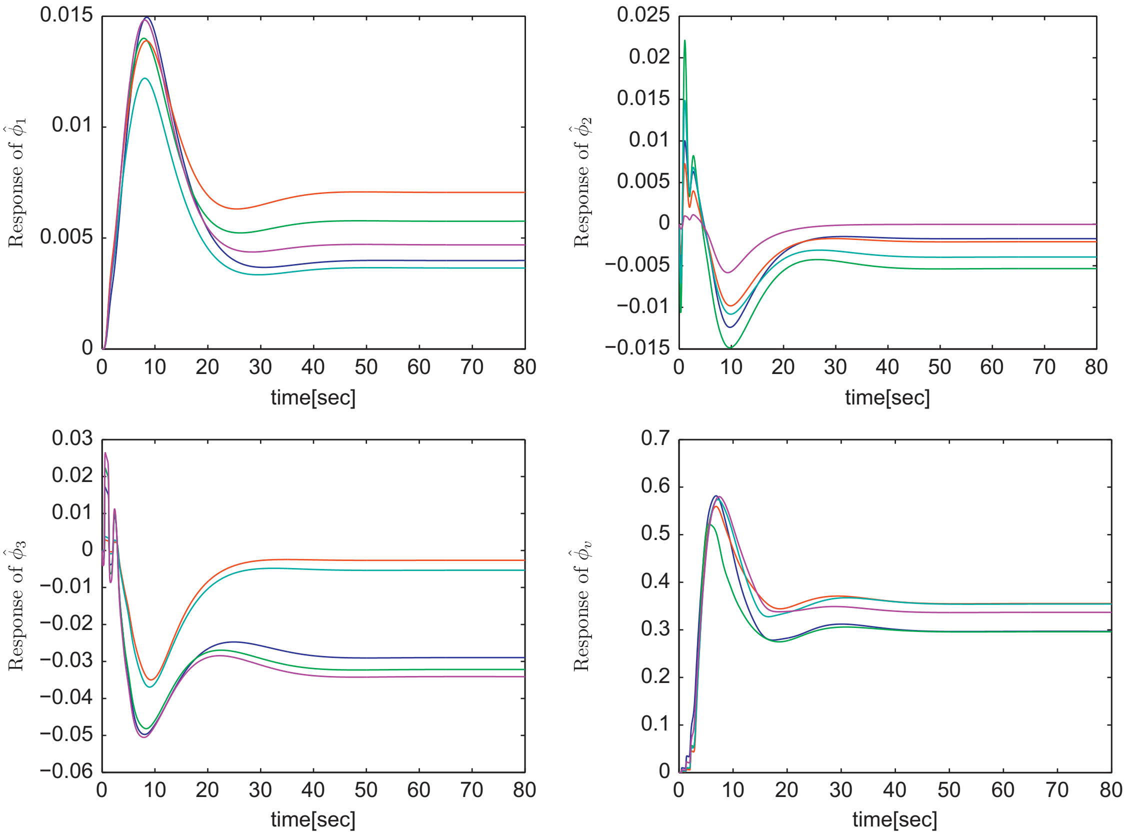

Fig. 15. SLFN response of $\|\hat{\phi}_i\|$.

are designed as $\lambda_i = 0.1$ and $\delta_i = 0.01$, $j = 1, 2, 3, v$. The parameters of the norm updating method are the same as weight updating method. Initially, the random hidden parameters are assigned. In this case, the number of hidden nodes is 50 for each subsystem.

The simulation results are illustrated in Figs. 1–11. Fig. 1 depicts the response performance that the altitude controller tracks the step change with magnitude 1000 ft while the velocity steps from 15,060 ft/s to 15,160 ft/s in Fig. 2. It can be seen that with less tuning parameters the norm updating method achieves almost the same good tracking performance as the weight updating method and the error converges to the small neighborhood of zero in the presence of uncertainty. The system states response of flight path angle, pitch angle and pith rate is demonstrated in Figs. 3–5. We can see that the system states respond in a quite similar way and this could be further confirmed from the controller response in Figs. 6 and 7. Figs. 8–11 show that the $\hat{\beta}_i$ and $\hat{\phi}_i$, $i = 1, 2, 3, v$ are bounded during the control process. Also we see the SLFN responses with a similar trend for x_{2d} and u_v while it is quite different for x_{3d} and u_A. This is due to the fact that $\hat{\phi}_1$ and $\hat{\phi}_v$ keep positive while the $\hat{\phi}_2$ and $\hat{\phi}_3$ change from positive to negative during the control application. It is interesting to find that for each virtual control, the absolute value of $\hat{\phi}_i$ is different $\|\beta_i\|$ but the controller input is quite similar while there is a little difference due to the mechanism of the adaptive design from SLFN. It extends the previous result in [14,21] to a more general case with bound estimation in Section 6.

To further show the performance evaluation, for each method, 5 groups are presented with the random hidden parameters. For both methods, with different hidden parameters, the system responds in a quite similar way and the tracking error converges to the small neighborhood of zero (Figs. 12 and 13). From the SLFN response, we see that the weight norm in Fig. 14 and the bound estimation in Fig. 15 are different since the hidden parameters are randomly assigned. However, the final value converges to some constant. It indicates that the updating algorithm and the controller are robust.

8. Conclusions

In this paper, discrete controller via back-stepping design is applied on hypersonic flight. In the design, the SLFN is used for uncertainty approximation. The hidden node parameters of SLFN are determined based on the ELM algorithm by assigning randomly the values. The robust updating law is provided to guarantee the closed-loop stability. Furthermore, the bound estimation based controller is presented to reduce the number of online adaptive parameters. Simulation results are presented to show the effectiveness of the proposed algorithm.

Acknowledgments

This work was supported by DSO National Laboratories of Singapore through a Strategic Project Grant (Project no. DSOCL10004), National Science Foundation of China (Grant nos. 61304098, 61134004, 61074185, 61005085), Fundamental Research Funds for the Central Universities (2012QNA4024) and NWPU Basic Research Funding (Grant no. JC20120236). The authors would like to thank the anonymous reviewers for constructive comments that helped to improve the quality and presentation of this paper.

Appendix A. Hypersonic aircraft model description

The related definition of the hypersonic aircraft model is as follows: $r=h+R_E$, $\overline{q}=\frac{1}{2}\rho V^2$, $L=\overline{q}SC_L$, $D=\overline{q}SC_D$, $T=\overline{q}SC_T$, $M_{yy}=\overline{q}S\overline{c}[C_M(\alpha)+C_M(\delta e)+C_M(q)]$, $C_L=0.6203\alpha$, $C_D=0.6450\alpha^2+0.0043378\alpha+0.003772$, $C_M(\alpha)=-0.035\alpha^2+0.036617\alpha+5.3261\times 10^{-6}$, $C_M(q)=(q\overline{c}/2V)\times(-6.796\alpha^2+0.3015\alpha-0.2289)$.

The control inputs related definition is as follows:

$$C_T=\begin{cases} 0.02576\Phi & \text{if } \Phi<1 \\ 0.0224+0.00336\Phi & \text{otherwise} \end{cases}$$

$$C_M(\delta e)=0.0292(\delta e-\alpha)$$

where ρ denotes the air density, S is the reference area, $\overline{c}$ is the reference length and R_E is the radius of the Earth. $C_x, x=L,D,T,M$ are the force and moment coefficients.

Appendix B. Definition for nonlinear functions f_i and g_i, $i=1, 2, 3, v$

$f_1=-(\mu-V^2r)\cos\ \gamma/(Vr^2)-0.6203\overline{q}S\gamma/(mV)$, $g_1=0.6203\overline{q}S/(mV)$, $f_2=0$, $g_2=1$, $f_3=\overline{q}S\overline{c}[C_M(\alpha)+C_M(q)-0.0292\alpha]/I_{yy}$, $g_3=0.0292\overline{q}S\overline{c}/I_{yy}$.

$f_v=-(D/m+\mu\ \sin\ \gamma/r^2)+\overline{q}S\times 0.0224\ \cos\ \alpha/m$, $g_v=\overline{q}S\times 0.00336\ \cos\ \alpha/m$ if $\Phi>1$. Otherwise $f_v=-(D/m+\mu\ \sin\ \gamma/r^2)$, $g_v=\overline{q}S\times 0.02576\ \cos\ \alpha/m$.

References

[1] Q. Wang, R. Stengel, Robust nonlinear control of a hypersonic aircraft, J. Guid. Control Dyn. 23 (4) (2000) 577–585.

[2] P. Kokotovic, The joy of feedback: nonlinear and adaptive: 1991 bode prize lecture, IEEE Control Syst. Mag. 12 (1991) 7–17.

[3] L. Fiorentini, A. Serrani, M. Bolender, D. Doman, Nonlinear robust adaptive control of flexible air-breathing hypersonic vehicles, J. Guid. Control Dyn. 32 (2) (2009) 401–416.

[4] B. Xu, X. Huang, D. Wang, F. Sun, Dynamic surface control of constrained hypersonic flight models with parameter estimation and actuator compensation, Asian J. Control 15 (6) (2013), http://dx.doi.org/10.1002/asjc.679.

[5] B. Xu, D. Gao, S. Wang, Adaptive neural control based on HGO for hypersonic flight vehicles, Sci. China Inf. Sci. 54 (3) (2011) 511–520.

[6] Y. Pan, M. Er, D. Huang, Q. Wang, Adaptive fuzzy control with guaranteed convergence of optimal approximation error, IEEE Trans. Fuzzy Syst. 19 (5) (2011) 807–818.

[7] Y. Pan, Y. Zhou, T. Sun, M. Er, Composite adaptive fuzzy H^∞ tracking control of uncertain nonlinear systems, Neurocomputing 99 (1) (2013) 15–24.

[8] K. Kanai, N. Hori, P. Nikiforuk, A discrete-time multivariable model-following method applied to decoupled flight control, J. Guid. Control Dyn. 9 (4) (1986) 403–407.

[9] R. Stengel, J. Broussard, P. Berry, Digital controllers for VTOL aircraft, IEEE Trans. Aerosp. Electron. Syst. (1) (1978) 54–63.

[10] D. Shin, Y. Kim, Nonlinear discrete-time reconfigurable flight control law using neural networks, IEEE Trans. Control Syst. Technol. 14 (3) (2006) 408–422.

[11] B. Xu, F. Sun, C. Yang, D. Gao, J. Ren, Adaptive discrete-time controller design with neural network for hypersonic flight vehicle via back-stepping, Int. J. Control 84 (9) (2011) 1543–1552.

[12] B. Xu, F. Sun, H. Liu, J. Ren, Adaptive Kriging controller design for hypersonic flight vehicle via back-stepping, IET Control Theory Appl. 6 (4) (2012) 487–497.

[13] B. Xu, Z. Shi, Universal kriging control of hypersonic aircraft model using predictor model without back-stepping, IET Control Theory Appl. 7 (4) (2013) 573–583.

[14] B. Xu, D. Wang, F. Sun, Z. Shi, Direct neural discrete control of hypersonic flight vehicle, Nonlinear Dyn. 70 (1) (2012) 269–278.

[15] G. Huang, L. Chen, Convex incremental extreme learning machine, Neurocomputing 70 (16–18) (2007) 3056–3062.

[16] G. Feng, G. Huang, Q. Lin, R. Gay, Error minimized extreme learning machine with growth of hidden nodes and incremental learning, IEEE Trans. Neural Netw. 20 (8) (2009) 1352–1357.

[17] G. Huang, X. Ding, H. Zhou, Optimization method based extreme learning machine for classification, Neurocomputing 74 (1) (2010) 155–163.

[18] G. Huang, D. Wang, Y. Lan, Extreme learning machines: a survey, Int. J. Mach. Learn. Cybern. 2 (2011) 107–122.

[19] M. van Heeswijk, Y. Miche, E. Oja, A. Lendasse, GPU-accelerated and parallelized ELM ensembles for large-scale regression, Neurocomputing 74 (16) (2011) 2430–2437.

[20] H. Rong, S. Suresh, G. Zhao, Stable indirect adaptive neural controller for a class of nonlinear system, Neurocomputing 74 (16) (2011) 2582–2590.

[21] Y. Liu, G. Wen, S. Tong, Direct adaptive NN control for a class of discrete-time nonlinear strict-feedback systems, Neurocomputing 73 (13–15) (2010) 2498–2505.

[22] Y.P. Pan, M.J. Er, Enhanced adaptive fuzzy control with optimal approximation error convergence, IEEE Trans. Fuzzy Syst. 21 (6) (2013) 1–10.

Bin Xu received the B.S. degree in measuring and control instrument from Northwestern Polytechnical University in 2006 and the Ph.D. degree in computer science from Tsinghua University, Beijing, China in 2012. During 2012–2013, he was a research fellow with Nanyang Technological University. He is currently a lecturer in School of Automation, Northwestern Polytechnical University. His research interests include intelligent control and adaptive control with application on flight dynamics, multi-robot formation and transportation system.

Yongping Pan The Ph.D. degree in control theory and control engineering from the South China University of Technology (SCUT), Guangzhou, in 2011. From 2007 to 2008, he was a Control Software Engineer in Santak Electronic (Shenzhen) Co., Ltd., Eaton Co., and an R&D Engineer in Light Engineering (China) Co., Ltd. He is currently a Research Fellow of the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests include adaptive approximation-based control, fuzzy logic and neural networks, modeling and prediction, and embedded control system design.

Danwei Wang received his Ph.D. and MSE degrees from the University of Michigan, Ann Arbor in 1989 and 1984, respectively. He received his B.E degree from the South China University of Technology, China, in 1982. Since 1989, he has been with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. Currently, he is a professor in the Division of Control and Instrumentation. He has served as general chairman, technical chairman and various positions in international conferences, such as International Conference on Control, Automation, Robotics and Vision (ICARCVs) and IROS conferences. He is an Associate Editor for the International Journal of Humanoid Robotics and served as an Associate Editor of Conference Editorial Board, IEEE Control Systems Society from 1998 to 2005. He was a recipient of Alexander von Humboldt fellowship, Germany. His research interests include robotics, control theory and applications. He has published widely in technical areas of iterative learning control, repetitive control, robust control and adaptive control systems, manipulator/mobile robot dynamics, path planning, and control, as well as model-based fault diagnosis and satellite formation flying (Personal home page: http://www.ntu.edu.sg/home/edwwang).

Fuchun Sun received the B.S. and M.S. degrees from the Naval Aeronautical Engineering Academy, Yantai, China, in 1986 and 1989, respectively, and the Ph.D. degree from Tsinghua University, Beijing, China, in 1998. He was with the Department of Automatic Control, Naval Aeronautical Engineering Academy. From 1998 to 2000, he was a Postdoctoral Fellow with the Department of Automation, Tsinghua University. He is currently a Professor with the Department of Computer Science and Technology, Tsinghua University. His research interests include intelligent control, neural networks, fuzzy systems, variable structure control, nonlinear systems, and robotics. Dr. Sun is the recipient of the excellent Doctoral Dissertation Prize of China in 2000 and the Choon-Gang Academic Award by Korea in 2003, and was recognized as a Distinguished Young Scholar in 2006 by the Natural Science Foundation of China.