Research Article

Extreme Learning Machine Assisted Adaptive Control of a Quadrotor Helicopter

Yu Zhang,¹ Zheng Fang,² and Hongbo Li³

¹School of Aeronautics and Astronautics, Zhejiang University, Zhejiang, Hangzhou 310027, China

²State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Liaoning, Shenyang 110189, China

³Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

Correspondence should be addressed to Zheng Fang; fangzheng@mail.neu.edu.cn

Received 21 August 2014; Revised 10 December 2014; Accepted 14 December 2014

Academic Editor: Yi Jin

Copyright © 2015 Yu Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Control of quadrotor helicopters is difficult because the problem is naturally nonlinear. The problem becomes more challenging for common model based controllers when unpredictable uncertainties and disturbances in physical control system are taken into account. This paper proposes a novel intelligent controller design based on a fast online learning method called extreme learning machine (ELM). Our neural controller does not require precise system modeling or prior knowledge of disturbances and well approximates the dynamics of the quadrotor at a fast speed. The proposed method also incorporates a sliding mode controller for further elimination of external disturbances. Simulation results demonstrate that the proposed controller can reliably stabilize a quadrotor helicopter in both agitated attitude and position control tasks.

1. Introduction

Unmanned aerial vehicles (UAVs) have received considerable attention in recent years due to their wide military and civilian applications. Their typical applications include collecting data, monitoring, surveillance, investigation, and inspection [1], which can be used in scenarios such as environmental monitoring, resource exploration, agriculture surveying, traffic control, weather forecasting, aerial photography, disasters search, and rescue. Particularly, unmanned rotorcrafts play an important role in these applications because of their flexibility such as hovering and vertical take-off and landing (VTOL). However, conventional rotorcraft with a main rotor and a tail rotor has extremely complex dynamics. Its maneuverability is greatly limited since it is very difficult to design a controller with high performance. Meanwhile, a special kind of rotorcrafts called quadrotor helicopter which has a compact form is becoming more and more popular than conventional rotorcrafts as they are mechanically and dynamically simpler and easier to control. In spite of this, the quadrotor is still a dynamically unstable system and its controller design is also challenging because of the inherent system characteristics

such as nonlinearities, cross couplings due to the gyroscopic effects, and underactuation [2]. Besides, all the applications mentioned above require a vehicle with stable and accurate performance of motion control. Therefore, how to design a high quality controller for quadrotor is an important and meaningful problem.

Many different control theories and methods are employed to design the attitude stabilizer or motion controller for quadrotors [3]. In the early stage, Bouabdallah et al. first apply two different control techniques, PID and linear quadratic (LQ) techniques, to a microquadrotor called OS4 [2]. Subsequently, other research works using PID [4, 5] or LQ [6] methods are also reported. These two kinds of controllers are easy to design and implement. However, they cannot handle the unmolded dynamics and external disturbances. Because of the nonlinearity, feedback linearization technology is adopted to design the controller for quadrotors [7, 8]. However, precise model of quadrotors which is required for linearization is difficult to obtain. Backstepping design method [9–11] is another choice to deal with the nonlinear models of quadrotors [12, 13], but the controller is usually vulnerable to parameters uncertainty. To deal with the dynamic

uncertainty and external disturbances, sliding mode control [14] and H infinite control [15, 16] algorithms are used to improve the robustness of the system. However, sliding mode controller is likely to have chattering phenomena in both sides of sliding mode surface due to the delay of sensors or actuators, and H infinite control method, which requires approximate linearization near the equilibrium point of the system, is not suitable for aggressive control.

The performance of model based controllers above degrades significantly in case of model uncertainties and unknown disturbances. One potential way to solve this problem is intelligent control methods such as fuzzy logic control [17, 18], neural networks control [19, 20], and learning based control [21]. One main challenge of utilizing these techniques is the convergence performance of controllers. Slow convergence may cause failure in real time control system.

In this paper, a novel computational intelligence technique called extreme learning machine (ELM) [22] is introduced to control the quadrotors by compensating the dynamic uncertainties and the external disturbances. ELM theories have been successfully improved recently by Cao et al. [23] and widely used in several control systems [24]. Essentially, it is a learning policy for generalized single hidden layer feedforward networks (SLFNs) whose input weight and hidden layer do not need to be tuned. Compared with backpropagation (BP) method and support vector machines (SVMs), ELM provides better generalization performance at a much faster learning speed and with least human intervention [25]. Thus, ELM can be used to estimate and compensate the uncertainties and disturbances of the systems simultaneously in real time.

There are three main contributions of this paper. First, we employ Lyapunov second method to minimize the cost function of ELM and satisfy the quadrotor control system stability simultaneously under the framework of ELM. So it is a development of ELM theory. Second, traditional neural network based controller is facing two problems. One is that too many parameters need to be initialized and tuned. The other is slow convergence. Since ELM can converge very fast with its input weight and hidden nodes parameters fixed, the two problems above are significantly alleviated when ELM is employed to the quadrotor control. Third, the quadrotor control system is a complicated dynamic system. The stability of the ELM based control system is proved.

This paper is organized as follows: In Section 2, the mathematical model of ELM is presented. Kinematics and dynamics models of quadrotors are described in Section 3. In Section 4, the details of designing an ELM-assisted quadrotor controller are presented. The stability of the proposed controller is also proved in this section. Simulation results are given in Section 5 to demonstrate the performance of the proposed controller. Finally, the paper is concluded in Section 6.

2. Preliminary on Extreme Learning Machine

In this section, the basic idea of ELM is briefly reviewed to provide a background for designing controller for the quadrotor. ELM is a special SLFN whose learning speed

can be much faster than conventional feedforward network learning algorithm such as BP algorithm while obtaining better generalization performance [26]. The essence of ELM is that the input weights and the parameters of the hidden layer do not need to adjust during the learning procedure. We take a SLFN with L hidden nodes as an example. The output of the SLFN can be modeled as

$$f_L(\mathbf{x}) = \sum_{i=1}^L \beta_i G(\mathbf{x}, \mathbf{c}_i, a_i), \quad \mathbf{x} \in \mathbf{R}^n, \quad \mathbf{c}_i \in \mathbf{R}^n, \quad (1)$$

where β_i is the output weight connecting the i th hidden node to the output node, $G(\mathbf{x}, \mathbf{c}_i, a_i)$ is the activation function of the i th hidden node, and $\mathbf{c}_i$ and a_i are the parameters of the activation function which are randomly generated and then fixed afterwards. Furthermore, there are two kinds of hidden nodes. Usually, additive hidden nodes use Sigmoid or threshold activation function as follows:

$$G(\mathbf{x}, \mathbf{c}_i, a_i) = \frac{1 - e^{-(\mathbf{c}_i \mathbf{x} + a_i)}}{1 + e^{-(\mathbf{c}_i \mathbf{x} + a_i)}}, \quad (2)$$

where $\mathbf{c}_i$ is the input weight vector for the i th hidden node and a_i is the bias of the i th hidden node. For RBF hidden nodes, Gaussian or triangular activation function is used for activation which can be given by

$$G(\mathbf{x}, \mathbf{c}_i, a_i) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_i\|^2}{2a_i^2}\right), \quad (3)$$

where $\mathbf{c}_i$ and a_i are the center and impact factor of the i th RBF node, respectively.

Then, N sample pairs $(\mathbf{x}_k, \mathbf{y}_k) \in \mathbf{R}^n \times \mathbf{R}^m$ ($k = 1, \dots, N$) are used to train the SLFN. If this network can approximate N samples with zero error, there must exist β_i^* , $\mathbf{c}_i$, and a_i such that

$$\sum_{i=1}^L \beta_i^* G(\mathbf{x}_k, \mathbf{c}_i, a_i) = \mathbf{y}_k, \quad k = 1, \dots, N. \quad (4)$$

The previous equation can be rewritten compactly as

$$\mathbf{H}\boldsymbol{\beta}^* = \mathbf{Y}, \quad (5)$$

where

$$\mathbf{H} = \begin{bmatrix} G(\mathbf{x}_1, \mathbf{c}_1, a_1) & \cdots & G(\mathbf{x}_1, \mathbf{c}_L, a_L) \\ \vdots & \ddots & \vdots \\ G(\mathbf{x}_N, \mathbf{c}_1, a_1) & \cdots & G(\mathbf{x}_N, \mathbf{c}_L, a_L) \end{bmatrix}_{N \times L}, \quad (6)$$

$$\boldsymbol{\beta}^* = [\beta_1^{*T} \cdots \beta_L^{*T}]^T,$$

$$\mathbf{Y} = [\mathbf{Y}_1^T \cdots \mathbf{Y}_L^T]^T.$$

ELM aims to minimize not only the training error but also the norm of output weights, which would yield a better generalization performance [25]. In other words, ELM try to minimize the training error as well as the norm of the output weights. So the objective function can be expressed as

$$\min \|\mathbf{H}\boldsymbol{\beta} - \mathbf{Y}\|, \|\boldsymbol{\beta}\|. \quad (7)$$

FIGURE 1: Quadrotor schematic. The white arrow width is proportional to rotor rotational speed, and the black arrows show the moving direction of the quadrotor.

Finally, the minimal norm least-square method instead of the standard optimization method was adopted in the original implementation of ELM [25], and the closed form solution is obtained:

$$\beta^* = \mathbf{H}^\dagger \mathbf{Y}, \quad (8)$$

where $\mathbf{H}^\dagger$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$.

3. Quadrotor Helicopters Model

The quadrotor helicopter has four rotors in cross configuration. As we can see from Figure 1, the two pairs of rotors (1, 3) and (2, 4) always turn in opposite directions. By changing the rotor speed, we can move the vehicles in different directions in 3D space. Initially, suppose all the rotors have the same speed as shown in Figure 1(c); increasing the four rotors speeds together generates upward movement. Then, increasing or decreasing 2 and 4 rotors speed inversely

will change the attitude of the quadrotor. It generates roll rotation as well as lateral motion. Changing 1 and 3 rotors speed in the same way produces the pitch rotation as well as the longitudinal movements (see Figure 1(d)). Finally, if the counter-torque resulting from rotor (1, 3) is different from that of rotor (2, 4), the yaw rotations of the quadrotor are generated as shown in Figures 1(a) and 1(b).

3.1. Kinematic Model. To facilitate the model description, quadrotor reference frames are defined first. We consider earth fixed frame E -XYZ as an inertia frame while frame B -xyz is a body fixed frame as shown in Figure 2.

To transform an attitude from the body frame (x , y , and z) to the inertia frame (X , Y , and Z), the coordinate transformation matrix [1]

$$\mathbf{R} = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}, \quad (9)$$

FIGURE 2: Quadrotor reference frames.

where c and s denote cosine and sine functions. Then, the relationship between the euler rates $\dot{\eta} = [\dot{\phi} \ \dot{\theta} \ \dot{\psi}]^T$ and angular body rates $\omega = [p \ q \ r]^T$ can be represented by

$$\omega = \mathbf{R}_r \dot{\eta}, \quad (10)$$

where

$$\mathbf{R}_r = \begin{bmatrix} 1 & 0 & -\sin(\theta) \\ 0 & \cos(\phi) & \sin(\phi)\cos(\theta) \\ 0 & -\sin(\phi) & \cos(\phi)\cos(\theta) \end{bmatrix}. \quad (11)$$

Particularly, when the quadrotor works around the hover position, the following condition can be approximately satisfied:

$$\mathbf{R}_r \approx \mathbf{I}_{3 \times 3}, \quad (12)$$

where $\mathbf{I}$ is the identity matrix.

3.2. Dynamic Model. Since the quadrotor helicopter can fly in 3D space, both rotational and translational motions should be considered in system modeling. The rotational equations of motion can be derived in the body frame using the Newton-Euler method while the translational equations of motion are derived in the inertia navigation frame using Newton's second law [1]. Many works about quadrotor modeling have been reported, so here we directly give the full state space model [27] of a quadrotor for controller design.

The state vector is defined as $\mathbf{X} = [\phi \ \dot{\phi} \ \theta \ \dot{\theta} \ \psi \ \dot{\psi} \ z \ \dot{z} \ x \ \dot{x} \ y \ \dot{y}]^T$, where ϕ , θ , and ψ are the attitude angles which represent roll, pitch, and yaw. (x, y, z) is the quadrotor's position in the inertia frame. If we define the speeds of the four rotors as Ω_1 , Ω_2 , Ω_3 , and Ω_4 , the control input vector can be further defined as $\mathbf{U} = [U_1 \ U_2 \ U_3 \ U_4]^T$, which is mapped by

$$\begin{aligned} U_1 &= k_F(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2), \\ U_2 &= k_F(-\Omega_2^2 + \Omega_4^2), \end{aligned}$$

$$\begin{aligned} U_3 &= k_F(\Omega_1^2 - \Omega_3^2), \\ U_4 &= k_M(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2), \end{aligned} \quad (13)$$

where k_F and k_M are the aerodynamic force and moment constants, respectively. In this case, U_1 is the total thrust generated from the four rotors. U_2 , U_3 , and U_4 are the equivalent of the torques around x -, y -, and z -axis in body frame.

Then, the state space model can be given by the following compact form:

$$\dot{\mathbf{X}} = \mathbf{f}(\mathbf{X}, \mathbf{U}), \quad (14)$$

where

$$\mathbf{f}(\mathbf{X}, \mathbf{U}) = \begin{pmatrix} \dot{\phi} \\ \dot{\theta}\psi a_1 + \dot{\theta}a_2\Omega_r + b_1U_2 \\ \dot{\theta} \\ \dot{\phi}\psi a_3 - \dot{\phi}a_4\Omega_r + b_2U_3 \\ \dot{\psi} \\ \dot{\theta}\phi a_5 + b_3U_4 \\ \dot{z} \\ g - \frac{(\cos \phi \cos \theta)U_1}{m} \\ \dot{x} \\ \frac{(\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi)U_1}{m} \\ \dot{y} \\ \frac{(\cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi)U_1}{m} \end{pmatrix}, \quad (15)$$

$$\begin{aligned} a_1 &= \frac{(I_{yy} - I_{zz})}{I_{xx}}, \\ a_2 &= \frac{J_r}{I_{xx}}, \\ a_3 &= \frac{(I_{zz} - I_{xx})}{I_{yy}}, \\ a_4 &= \frac{J_r}{I_{yy}}, \\ a_5 &= \frac{(I_{xx} - I_{yy})}{I_{zz}}, \\ b_1 &= \frac{l}{I_{xx}}, \\ b_2 &= \frac{l}{I_{yy}}, \\ b_3 &= \frac{l}{I_{zz}}, \end{aligned} \quad (16)$$

and I_{xx} , I_{yy} , and I_{zz} are the moment of inertia around x -, y -, and z -axis, respectively. J_r is the propeller inertia coefficient. l is the arm length of the quadrotor.

FIGURE 3: The structure of the quadrotor control system.

4. Controller Design Using ELM

By investigating the relationship between the state variables in (15), the attitude of the quadrotor does not depend on the translational motion while the translation of the quadrotor depends on the attitude angles. Thus, the whole system can be decoupled into two subsystems: inner loop attitude subsystem and outer loop position subsystem, respectively. The structure of the whole control system is described in Figure 3, where $(x_d, \dot{x}_d, y_d, \dot{y}_d, z_d, \dot{z}_d)$ is the coordinate of the desired trajectory and $(\phi_d, \theta_d, \psi_d, \dot{\phi}_d, \dot{\theta}_d, \dot{\psi}_d)$ is the desired attitude angles and their rates generated by the position controller. U_1 is produced by the position controller related to the altitude of the vehicle while the U_2, U_3 , and U_4 are calculated by the attitude controller.

4.1. Position Control Loop. The outer loop is a position control system which is much slower compared to the inner loop attitude control system. In addition, the desired roll, pitch, and yaw angles are normally small that makes the position subsystem an approximately linear system near the equilibrium points. A simple PD controller is sufficient to control the quadrotor's position.

For horizontal movement, according to the desired trajectory, the expected accelerations in X and Y direction can be calculated by the PD controller:

$$\begin{aligned}\ddot{x}_d &= k_{px}(x_d - x) + k_{dx}(\dot{x}_d - \dot{x}), \\ \ddot{y}_d &= k_{py}(y_d - y) + k_{dy}(\dot{y}_d - \dot{y}),\end{aligned}\quad (17)$$

where (k_{px}, k_{dx}) and (k_{py}, k_{dy}) are the proportional and differential control gains in X and Y directions. Control gains are acquired using pole placement design method to make the position system stable. When the yaw angle of the quadrotor keeps fixed, we can substitute $\ddot{x}_d$ and $\ddot{y}_d$ into (15) and obtain

$$\begin{aligned}\frac{(\cos \phi_d \sin \theta_d \cos \psi + \sin \phi_d \sin \psi)U_1}{m} &= \ddot{x}_d, \\ \frac{(\cos \phi_d \sin \theta_d \sin \psi - \sin \phi_d \cos \psi)U_1}{m} &= \ddot{y}_d.\end{aligned}\quad (18)$$

Using the small angle assumption around the equilibrium position, (18) can be simplified as

$$\begin{aligned}\frac{(\theta_d \cos \psi + \phi_d \sin \psi)U_1}{m} &= \ddot{x}_d, \\ \frac{(\theta_d \sin \psi - \phi_d \cos \psi)U_1}{m} &= \ddot{y}_d.\end{aligned}\quad (19)$$

Then, the reference roll and pitch angles can be solved by

$$\begin{bmatrix} \phi_d \\ \theta_d \end{bmatrix} = \frac{m}{U_1} \begin{bmatrix} \sin \psi & \cos \psi \\ -\cos \psi & \sin \psi \end{bmatrix}^{-1} \begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \end{bmatrix}.\quad (20)$$

For vertical movements, the altitude control input can be calculated by

$$\begin{aligned}U_1 &= \frac{1}{\cos(\phi) \cos(\theta)} \\ &\cdot ((1 + k_1 k_2)(z_d - z) + (k_1 + k_2)(\dot{z}_d - \dot{z}) + mg)\end{aligned}\quad (21)$$

using backstepping design method, where k_1 and k_2 are positive real numbers and mg is the term to balance the gravity. Please note that all the inputs have their own saturation functions.

4.2. Attitude Control Loop. The quadrotor's attitude system is sensitive to the disturbances. It is also a nonlinear system with unmodeled dynamic uncertainties compared to the real quadrotor system. Hence an ELM based SLFN with very fast learning speed is employed to adaptively learn and estimate the nonlinear model including the uncertainties of the quadrotor's attitude system in real time. Meanwhile, the proposed neural controller incorporates a sliding mode controller to deal with the external disturbances. The stability of the attitude control system is proved using the Lyapunov theory.

Since the roll, pitch, and yaw subsystems have the same form of expression, without loss of generality, we take roll subsystem as an example to introduce the design procedure. Although three attitude subsystems are coupled with each other, the coupled items can be considered the unknown nonlinear functions of dynamic system or internal disturbances and thus be adaptively estimated by the SLFN with ELM.

FIGURE 4: Roll and pitch angle regulation.

4.2.1. Model Based Control Law. According to (15), the roll subsystem is expressed in the following form:

$$\begin{aligned}\dot{x}_1 &= x_2, \\ \dot{x}_2 &= f(\mathbf{x}) + g(\mathbf{x})u + d, \\ y &= x_1,\end{aligned}\quad (22)$$

where $f(\mathbf{x})$ is assumed to be unknown but bounded and $g(\mathbf{x})$ is the input gain. Particularly, in roll subsystem $g(\mathbf{x}) = b_1 = l/I_{xx}$ which is known. d is the unknown disturbances. It is bounded and satisfies $|d| \leq d_{\max}$, where $d_{\max}$ is the upper bound of the disturbances. Then, the output tracking error $\mathbf{E} = [e, \dot{e}]^T$, where $e = y_d - x_1$ and $\dot{e} = \dot{y}_d - x_2$.

Suppose all the functions and parameters in (22) are precisely known; the perfect control law u^* can be obtained using feedback linearization method as follows:

$$u^* = \frac{I_{xx}}{l}(\ddot{y}_d - f(\mathbf{x}) - d + \mathbf{K}^T \mathbf{E}), \quad (23)$$

where $\mathbf{K} = [K_1, K_2]^T$ is a real number vector. Substitute (23) into (22); the following error equation is obtained:

$$\ddot{e} + K_2 \dot{e} + K_1 e = 0. \quad (24)$$

K_1 and K_2 can be determined when all the roots of the polynomial $s^2 + K_2 s + K_1 = 0$ are in the open left half plane, which indicates that the tracking error will converge to zero.

4.2.2. Neural Controller Structure. Since $f(\mathbf{x})$ is coupled with other variables and cannot be accurately modeled, d

is also unknown disturbances; control law of (23) cannot be directly implemented. Here, we present an ELM-assisted neural controller to solve the problem.

The inner loop controller is mainly composed of two parts. One is the neural controller u_n and the other is the sliding mode controller u_s . So the overall control law

$$u = u_n + u_s. \quad (25)$$

The neural controller is used to approximate the unknown function $f(\mathbf{x})$ while the sliding mode controller is employed to eliminate the external disturbances d .

For the neural controller, according to (23), the optimal neural control law is expected as

$$u_n^* = \frac{I_{xx}}{l}(\ddot{y}_d - f(\mathbf{x}) + \mathbf{K}^T \mathbf{E}). \quad (26)$$

Here, a SLFN whose parameters are determined based on the ELM is employed to approximate the above desired neural control law. Then, the actual neural control law u_n is given by

$$u_n = \mathbf{H}(\mathbf{r}, \mathbf{c}, \mathbf{a})\boldsymbol{\beta}, \quad (27)$$

where $\mathbf{r} = [\ddot{y}_d; \mathbf{x}; \mathbf{E}]^T$ is the input vector. As we mentioned in Section 2, $\mathbf{c}$ and $\mathbf{a}$ are hidden node parameters which are generated randomly and then fixed. Training a SLFN with ELM is the equivalent of finding a least-square solution of the output weights $\boldsymbol{\beta}^*$, such that

$$\hat{u}_n^* = \mathbf{H}(\mathbf{r}, \mathbf{c}, \mathbf{a})\boldsymbol{\beta}^* + \varepsilon(\mathbf{r}), \quad (28)$$

where $\varepsilon(\mathbf{r})$ is the approximation error. This error arises if the number of hidden nodes is much less than the number of

training samples [24], but it is bounded with the constant ε_N ; that is, $|\varepsilon(\mathbf{r})| \leq \varepsilon_N$.

For the sliding mode controller, standard sliding mode control law has the form

$$u_s = G_s(\mathbf{x}) \operatorname{sgn}(f(\mathbf{E})), \quad (29)$$

where $G_s(\mathbf{x})$ is the sliding mode gain and $f(\mathbf{E})$ is the sliding mode surface function.

Since we have the forms of the neural controller and the sliding mode controller, the next step is to derive the adaptive law $\dot{\boldsymbol{\beta}}$, so that $\boldsymbol{\beta}$ converges to $\boldsymbol{\beta}^*$. The parameters of the sliding mode controller are determined. The details are given in the next subsection.

4.3. Adaptive Control Law and Stability Analysis. The stable adaptive law for $\boldsymbol{\beta}$ and the parameters of the sliding mode controller are derived using Lyapunov second method.

Based on (22), (25), and (26), the tracking error equation can be obtained:

$$\dot{\mathbf{E}} = \mathbf{A}\mathbf{E} + \mathbf{B} \left[u_n^* - u_n - u_s - \frac{dI_{xx}}{l} \right], \quad (30)$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -K_1 & -K_2 \end{bmatrix}, \quad (31)$$

$$\mathbf{B} = \begin{bmatrix} 0 \\ \frac{l}{I_{xx}} \end{bmatrix}.$$

Substituting (27), (28) into (30) yields

$$\dot{\mathbf{E}} = \mathbf{A}\mathbf{E} + \mathbf{B} \left[\mathbf{H}(\mathbf{r}, \mathbf{c}, \mathbf{a})\tilde{\boldsymbol{\beta}} + \varepsilon(\mathbf{r}) - u_s - \frac{dI_{xx}}{l} \right], \quad (32)$$

where $\tilde{\boldsymbol{\beta}} = \boldsymbol{\beta}^* - \boldsymbol{\beta}$. Then, the following Lyapunov function is considered to derive the stable tuning law for $\boldsymbol{\beta}$:

$$V = \frac{1}{2} \mathbf{E}^T \mathbf{P} \mathbf{E} + \frac{1}{2\eta} \tilde{\boldsymbol{\beta}}^T \tilde{\boldsymbol{\beta}}, \quad (33)$$

where $\mathbf{P}$ is a symmetric and positive definite matrix and η is a positive constant which is referred to as the learning rate of the SLFN.

According to (32), the derivative of the Lyapunov function is acquired as

$$\begin{aligned} \dot{V} &= \frac{1}{2} [\dot{\mathbf{E}}^T \mathbf{P} \mathbf{E} + \mathbf{E}^T \mathbf{P} \dot{\mathbf{E}}] + \frac{1}{\eta} \tilde{\boldsymbol{\beta}}^T \dot{\tilde{\boldsymbol{\beta}}} \\ &= -\frac{1}{2} \mathbf{E}^T \mathbf{Q} \mathbf{E} + \left(\mathbf{E}^T \mathbf{P} \mathbf{B} \mathbf{H} + \frac{1}{\eta} \tilde{\boldsymbol{\beta}}^T \right) \tilde{\boldsymbol{\beta}} \\ &\quad - \mathbf{E}^T \mathbf{P} \mathbf{B} \left(\frac{dI_{xx}}{l} - \varepsilon(\mathbf{r}) \right) - \mathbf{E}^T \mathbf{P} \mathbf{B} u_s, \end{aligned} \quad (34)$$

where

$$\mathbf{Q} = -(\mathbf{A}^T \mathbf{P} + \mathbf{P} \mathbf{A}). \quad (35)$$

Suppose $\mathbf{Q}$ is selected as a symmetric definite matrix, the first item of the Lyapunov function will converge to zero. Then, $\mathbf{P}$ can be calculated by solving (35). According to the second item of the Lyapunov function, $\tilde{\boldsymbol{\beta}}$ can be eliminated if we let

$$\dot{\tilde{\boldsymbol{\beta}}}^T = -\eta \mathbf{E}^T \mathbf{P} \mathbf{B} \mathbf{H}. \quad (36)$$

Consider $\dot{\tilde{\boldsymbol{\beta}}} = -\dot{\boldsymbol{\beta}}$ based on the definition of the $\tilde{\boldsymbol{\beta}}$. Thus, the tuning rule for the output weight is acquired:

$$\dot{\boldsymbol{\beta}}^T = \eta \mathbf{E}^T \mathbf{P} \mathbf{B} \mathbf{H}. \quad (37)$$

After that, (34) becomes

$$\dot{V} = -\frac{1}{2} \mathbf{E}^T \mathbf{Q} \mathbf{E} - \mathbf{E}^T \mathbf{P} \mathbf{B} \left(\frac{dI_{xx}}{l} - \varepsilon(\mathbf{r}) \right) - \mathbf{E}^T \mathbf{P} \mathbf{B} u_s. \quad (38)$$

To make (38) less than or equal to zero, the sliding mode controller u_s can be determined as

$$u_s = \left(\frac{d_{\max} I_{xx}}{l} + \varepsilon_N \right) \operatorname{sgn}(\mathbf{E}^T \mathbf{P} \mathbf{B}). \quad (39)$$

Therefore, the following relationship can be derived from (39):

$$\begin{aligned} \dot{V} &\leq -\frac{1}{2} \mathbf{E}^T \mathbf{Q} \mathbf{E} + |\mathbf{E}^T \mathbf{P} \mathbf{B}| \left(\frac{|d| I_{xx}}{l} + |\varepsilon(\mathbf{r})| \right) \\ &\quad - \mathbf{E}^T \mathbf{P} \mathbf{B} \left(\left(\frac{d_{\max} I_{xx}}{l} + \varepsilon_N \right) \operatorname{sgn}(\mathbf{E}^T \mathbf{P} \mathbf{B}) \right) \\ &= -\frac{1}{2} \mathbf{E}^T \mathbf{Q} \mathbf{E} + |\mathbf{E}^T \mathbf{P} \mathbf{B}| \left(\frac{|d| I_{xx}}{l} + |\varepsilon(\mathbf{r})| \right) \\ &\quad - |\mathbf{E}^T \mathbf{P} \mathbf{B}| \left(\frac{d_{\max} I_{xx}}{l} + \varepsilon_N \right) \\ &= -\frac{1}{2} \mathbf{E}^T \mathbf{Q} \mathbf{E} + |\mathbf{E}^T \mathbf{P} \mathbf{B}| (|d| - d_{\max}) \frac{I_{xx}}{l} \\ &\quad + |\mathbf{E}^T \mathbf{P} \mathbf{B}| (|\varepsilon(\mathbf{r})| - \varepsilon_N) \\ &\leq 0. \end{aligned} \quad (40)$$

Equation (40) indicates that $\dot{V}$ is negative semidefinite and the control system is guaranteed to be stable. Please note that traditional ELM trains their output weights using the least-square error method, but in this paper, we derived the tuning rate of the output weights using Lyapunov second method. It is an online sequential learning process that makes the controller applicable in real time. The input weight and the parameters of the hidden nodes are randomly assigned as the normal ELM algorithm.

FIGURE 5: Control inputs U_2 and U_3 .

Finally, the overall control laws for quadrotor's attitude system are given by

$$\begin{aligned}
 U_2 &= \mathbf{H}_\phi(\mathbf{r}_\phi, \mathbf{c}_\phi, \mathbf{a}_\phi) \boldsymbol{\beta}_\phi \\
 &\quad + \left(\frac{d_{\max} I_{xx}}{l} + \varepsilon_N \right) \text{sgn}(\mathbf{E}_\phi^T \mathbf{P}_\phi \mathbf{B}_\phi), \\
 U_3 &= \mathbf{H}_\theta(\mathbf{r}_\theta, \mathbf{c}_\theta, \mathbf{a}_\theta) \boldsymbol{\beta}_\theta \\
 &\quad + \left(\frac{d_{\max} I_{yy}}{l} + \varepsilon_N \right) \text{sgn}(\mathbf{E}_\theta^T \mathbf{P}_\theta \mathbf{B}_\theta), \\
 U_4 &= \mathbf{H}_\psi(\mathbf{r}_\psi, \mathbf{c}_\psi, \mathbf{a}_\psi) \boldsymbol{\beta}_\psi \\
 &\quad + (d_{\max} I_{zz} + \varepsilon_N) \text{sgn}(\mathbf{E}_\psi^T \mathbf{P}_\psi \mathbf{B}_\psi),
 \end{aligned} \tag{41}$$

where the subscripts ϕ , θ , and ψ represent the roll, pitch, and yaw subsystem, respectively. To avoid chattering problem, the $\text{sgn}(\cdot)$ function can be replaced by the saturation function $\text{sat}(\cdot)$.

5. Simulations

In this section, attitude and position control simulations are both implemented on a nonlinear quadrotor model to evaluate the performance of the proposed controller. We also compare our method to some other controllers to demonstrate the effectiveness of ELM. The parameters of the quadrotor for simulation are measured from a real platform as listed in Table 1.

FIGURE 6: Learning curve of $\|\boldsymbol{\beta}\|$ in attitude regulation control.

TABLE 1: Parameters of the quadrotor.

Name	Parameter	Value	Unit
Mass	m	2.2	kg
Inertia around x axis	I_{xx}	$1.676e-2$	$\text{kg}\cdot\text{m}^2$
Inertia around y axis	I_{yy}	$1.676e-2$	$\text{kg}\cdot\text{m}^2$
Inertia around z axis	I_{zz}	$2.314e-2$	$\text{kg}\cdot\text{m}^2$
Rotor inertia	J_r	0.1	$\text{kg}\cdot\text{m}^2$
Arm length	l	0.18	m

5.1. Simulation Setup. Two cases are tested for attitude control simulations. The first case is to regulate the quadrotor helicopter from an initial attitude ($\phi_i = \pi/8$, $\theta_i = \pi/8$, $\psi_i = 0$) to a target attitude ($\phi_t = -\pi/16$, $\theta_t = -\pi/16$, $\psi_t = 0$).

FIGURE 7: Roll and pitch angle tracking.

The attitude response speed and accuracy can be evaluated in this case. The second case is an attitude tracking mission. The quadrotor's attitude is set to follow a trajectory ($\phi_d = (\pi/8) \sin(2\pi \times 0.25t)$, $\theta_d = (\pi/8) \sin(2\pi \times 0.25t)$) while the yaw angle keeps fixed.

To further demonstrate the controller performance, position tracking simulations are also implemented, where the quadrotor is expected to follow a trajectory such as straight line and circle.

In all the cases above, the model uncertainty d in (22) is given by $0.5 * \sin(\pi t)$, and $d_{\max} = 0.5$. The controller parameters are chosen as follows: $K_1 = 8$, $K_2 = 128$, $\mathbf{Q} = \text{diag}(100, 100)$, $\eta = 0.05$, and $\varepsilon_N = 14$. The number of hidden nodes $L = 6$. RBF nodes are selected as the hidden nodes whose parameter $\mathbf{c}$ is generated in the interval $[-5\pi/16, 5\pi/16]$ and $a = 0.4$.

To show the feasibility and advantage of the proposed controller, we compare our method (SMC with ELM) to a proportion-differentiation controller (PD), a standard sliding mode controller (SMC), and a SMC with back propagation based neural network (SMC with NN). The parameters of the PD controller are chosen as follows: $k_{p\phi} = 5$, $k_{d\phi} = 1.2$, $k_{p\theta} = 5$, $k_{d\theta} = 1.2$, $k_{p\psi} = 5$, and $k_{d\psi} = 1$. Sliding mode control gain $G = 14.5$. Sliding mode surface parameters $c_1 = 8$ and $c_2 = 1$. The initial parameters of the SMC with NN are the same as the SMC with ELM.

5.2. Results. The results of roll (ϕ) and pitch (θ) angles regulation using different controllers are illustrated in Figure 4. As we can see, the proposed controller makes the attitude angles converge faster to the reference values than the other

controllers. PD controller has attitude overshoot problem which may result in vehicle's vibration. This problem can be avoided by decreasing PD control gains, but the response speed is sacrificed. The result from SMC with NN has large errors because of the slow convergence problem. The control inputs related to roll and pitch are shown in Figure 5. All of them are bounded and applicable. Inputs in our method are large at the beginning but decrease quickly. This is the reason why the response speed and accuracy are obtained simultaneously. In Figure 6, it indicates that the output weights β trained by ELM can converge faster than BP based neural networks. Thus, the contribution of ELM in quadrotor controller design is further confirmed.

Attitude tracking results are shown in Figure 7. ELM-assisted controller makes the attitude follow the reference trajectory pretty well while PD and SMC have a delay in tracking process. The result from SMC with NN also cannot follow the reference input well. It shows that the proposed controller has the best tracking performance among the four controllers. Tracking errors of this case are shown in Figure 8. We can see that our proposed method has the smallest error. Figure 9 shows the better convergency of output weights β using ELM than traditional neural networks.

The results of straight line tracking are shown in Figure 10 which demonstrate better tracking performance of proposed controller. As we can see from Figure 11, SMC with ELM has the smallest error compared to the others. In addition, the PD and SMC with NN cannot damp the errors quickly and have some oscillations. The learning curve of ELM also converges fast as shown in Figure 12.

Position tracking on a circle trajectory is shown in Figure 13. It seems that all methods can give a stable tracking

FIGURE 8: Attitude tracking errors.

FIGURE 9: Learning curve of $\|\beta\|$ in attitude tracking control.

FIGURE 10: Position tracking on a straight line.

performance. However, as we can see, the SMC with ELM has the best performance in terms of stability and tracking accuracy. The training process of β in ELM is also better than BP based neural networks as given in Figure 14. ELM can converge quickly and remains unchanged afterwards in this test.

6. Conclusions

In this paper, an ELM-assisted adaptive controller combined with a sliding mode controller is designed for control of a quadrotor helicopter. A single hidden layer feedforward network whose input weights and hidden node parameters are generated randomly and fixed afterwards is used to approximate the unknown dynamic model and internal uncertainties. Different from the standard ELM algorithm, the output weights of this neural network are updated based on the Lyapunov second method to guarantee the stability of the attitude control system. A sliding mode controller is employed to compensate the approximation error of the SLFN and eliminate the external disturbances. Plenty of simulations on quadrotor's attitude and position control are implemented to validate the effectiveness of the proposed control scheme. The simulation results show that the proposed controller has better performance on response speed and control accuracy than simple PD controller. Furthermore, the comparison with the standard SMC and SMC with NN indicates that the extreme learning machine has potential ability to handle the unmodeled uncertainty problem in the control domain because of its fast convergence capability and good approximation ability.

FIGURE 11: Tracking errors in X and Y direction.

FIGURE 12: Learning curve of $\|\beta\|$ in position tracking control (straight line). FIGURE 14: Learning curve of $\|\beta\|$ in position tracking control (circle).

FIGURE 13: Position tracking on a circle trajectory.

For future work, the idea of composite design and reinforcement learning [28] could be borrowed to develop new ELM algorithm to achieve better results.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61005085) and Fundamental Research Funds for the Central Universities (2012QNA4024, N120408002).

References

- [1] A. Nagaty, S. Saeedi, C. Thibault, M. Seto, and H. Li, "Control and navigation framework for quadrotor helicopters," *Journal of Intelligent and Robotic Systems*, vol. 70, no. 1–4, pp. 1–12, 2013.
- [2] S. Bouabdallah, A. Noth, and R. Siegwart, "PID vs LQ control techniques applied to an indoor micro Quadrotor," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '04)*, pp. 2451–2456, IEEE, October 2004.
- [3] Y. Li and S. Song, "A survey of control algorithms for quadrotor unmanned helicopter," in *Proceedings of the IEEE 5th International Conference on Advanced Computational Intelligence (ICACI '12)*, pp. 365–369, October 2012.
- [4] S. Gonzalez-Vazquez and J. Moreno-Valenzuela, "A new non-linear PI/PID controller for quadrotor posture regulation," in *Proceedings of the Electronics, Robotics and Automotive Mechanics Conference (CERMA '10)*, pp. 642–647, Morelos, Mexico, September–October 2010.
- [5] A. L. Salih, M. Moghavvemi, H. A. F. Mohamed, and K. S. Gaeid, "Flight PID controller design for a UAV quadrotor," *Scientific Research and Essays*, vol. 5, no. 23, pp. 3660–3667, 2010.
- [6] L. Minh and C. Ha, "Modeling and control of quadrotor MAV using vision-based measurement," in *Proceedings of the International Forum on Strategic Technology (IFOST '10)*, pp. 70–75, IEEE, October 2010.
- [7] N. G. Shakev, A. V. Topalov, O. Kaynak, and K. B. Shiev, "Comparative results on stabilization of the quad-rotor rotorcraft using bounded feedback controllers," *Journal of Intelligent and Robotic Systems*, vol. 65, no. 1–4, pp. 389–408, 2012.
- [8] H. Voos, "Nonlinear control of a quadrotor micro-uav using feedback-linearization," in *Proceedings of the IEEE International Conference on Mechatronics (ICM '09)*, pp. 1–6, IEEE, April 2009.
- [9] B. Xu, F. Sun, H. Liu, and J. Ren, "Adaptive kriging controller design for hypersonic flight vehicle via back-stepping," *IET Control Theory & Applications*, vol. 6, no. 4, pp. 487–497, 2012.
- [10] B. Xu, X. Huang, D. Wang, and F. Sun, "Dynamic surface control of constrained hypersonic flight models with parameter estimation and actuator compensation," *Asian Journal of Control*, vol. 16, no. 1, pp. 162–174, 2014.
- [11] B. Xu, F. Sun, C. Yang, D. Gao, and J. Ren, "Adaptive discrete-time controller design with neural network for hypersonic flight vehicle via back-stepping," *International Journal of Control*, vol. 84, no. 9, pp. 1543–1552, 2011.
- [12] A. Das, F. Lewis, and K. Subbarao, "Backstepping approach for controlling a quadrotor using lagrange form dynamics," *Journal of Intelligent and Robotic Systems*, vol. 56, no. 1–2, pp. 127–151, 2009.
- [13] A. Honglei, L. Jie, W. Jian, W. Jianwen, and M. Hongxu, "Backstepping-based inverse optimal attitude control of quadrotor," *International Journal of Advanced Robotic Systems*, vol. 10, article no. 223, 2013.
- [14] A. R. Patel, M. A. Patel, and D. R. Vyas, "Modeling and analysis of quadrotor using sliding mode control," in *Proceedings of the 44th Southeastern Symposium on System Theory (SSST '12)*, pp. 111–114, IEEE, March 2012.
- [15] J. Gadewadikar, F. L. Lewis, K. Subbarao, K. Peng, and B. M. Chen, "H-infinity static output-feedback control for rotorcraft," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 54, no. 4, pp. 629–646, 2009.
- [16] G. V. Raffo, M. G. Ortega, and F. R. Rubio, "An integral predictive/nonlinear H_∞ control structure for a quadrotor helicopter," *Automatica*, vol. 46, no. 1, pp. 29–39, 2010.
- [17] C. Coza, C. Nicol, C. J. B. Macnab, and A. Ramirez-Serrano, "Adaptive fuzzy control for a quadrotor helicopter robust to wind buffeting," *Journal of Intelligent and Fuzzy Systems*, vol. 22, no. 5–6, pp. 267–283, 2011.
- [18] D. Gautam and C. Ha, "Control of a quadrotor using a smart self-tuning fuzzy PID controller," *International Journal of Advanced Robotic Systems*, vol. 10, article 380, 2013.
- [19] T. Dierks and S. Jagannathan, "Output feedback control of a quadrotor UAV using neural networks," *IEEE Transactions on Neural Networks*, vol. 21, no. 1, pp. 50–66, 2010.
- [20] B. Xu, Z. Shi, C. Yang, and S. Wang, "Neural control of hypersonic flight vehicle model via time-scale decomposition with throttle setting constraint," *Nonlinear Dynamics*, vol. 73, no. 3, pp. 1849–1861, 2013.
- [21] P. Bouffard, A. Aswani, and C. Tomlin, "Learning-based model predictive control on a quadrotor: onboard implementation and experimental results," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 279–284, IEEE, May 2012.
- [22] G.-B. Huang, D. H. Wang, and Y. Lan, "Extreme learning machines: a survey," *International Journal of Machine Learning and Cybernetics*, vol. 2, no. 2, pp. 107–122, 2011.
- [23] J. Cao, Z. Lin, G.-B. Huang, and N. Liu, "Voting based extreme learning machine," *Information Sciences*, vol. 185, pp. 66–77, 2012.
- [24] H.-J. Rong and G.-S. Zhao, "Direct adaptive neural control of nonlinear systems with extreme learning machine," *Neural Computing and Applications*, vol. 22, no. 3–4, pp. 577–586, 2013.
- [25] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Real-time learning capability of neural networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 863–878, 2006.
- [26] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1–3, pp. 489–501, 2006.
- [27] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '07)*, pp. 153–158, IEEE, October 2007.
- [28] B. Xu, C. Yang, and Z. Shi, "Reinforcement learning output feedback NN control using deterministic learning technique," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 3, pp. 635–641, 2014.