Adaptive Trajectory Tracking Algorithm of a Quadrotor with Sliding Mode Control and Multilayer Neural Network

Kang Niu, Di Yang, Xi Chen, Rong Wang, and Jianqiao Yu

1Department of Astronautics, Beijing Institute of Technology, Beijing 100081, China
2Shanghai Electromechanical Engineering Research Institute, Shanghai Academy of Spaceflight Technology, Shanghai 201109, China

Correspondence should be addressed to Kang Niu; astro_nuaank@163.com

Received 20 January 2022; Revised 19 April 2022; Accepted 27 April 2022; Published 22 August 2022

1.Introduction

Nowadays, the quadrotor UAV is widely applied both in military and civilian areas [1–3]. At the same time, all these applications drive a continuous progress in path planning, system stability analysis, and trajectory tracking areas [4]. All these areas consisted of tracking the desire trajectory, designing a high control accuracy algorithm, and improving the environment adaptability of a quadrotor and others. However, it is well known that the quadrotor UAV is a strong nonlinear, underactuated, and highly coupled system [5]. So, how to design a proper control algorithm to tackle the desire trajectory and improve the accuracy and stability of the system has become challenging research.

To research these problems, researchers have proposed many methods all over the world. FOPD (Robust Fractional-Order PD) Controller is researched for a Wheeled Mobile Robot (WMR) in reference [6], which firstly presents an improved flat phase property as a robust controller tuning specification to improve the controlled system robustness while this reference just considers how to guarantee the stability of the system and the tracking error is very large. To stabilize the translational and rotational motion of a quadrotor system and enforce it to track a given trajectory with minimum energy and error, a NLPID (nonlinear proportional integral derivative) controller is proposed in reference [7], where their parameters are tuned using a genetic algorithm (GA) to minimize a multiobjective output performance index. However, this reference just analyzes the certain conditions on the gains of the NLPID controllers and the tracking performance is very poor. To improve the system's transient responses, the genetic self-tuning PID algorithm is studied for the aerosonde UAV model in reference [8]. Though the GA is introduced to the PID algorithm to improve the intelligence degree, it does not analyze the pitching, rolling, and yawing characteristic of the UAV and the simulation is too simple. In reference [9], the strategies for Model Predictive Control (MPC) design and implementation for UAV are discussed. The main idea of this literature is the system identification techniques which
are used to derive an estimate of the system model while the tracking performance and the environmental adaptability is so poor. To avoid the frequent periodic execution of the control tasks and reduce the computing cost, an event-trigger-based fractional-order sliding mode control strategy is investigated in reference [10]. All simulations results demonstrated that the proposed controller based on the event-triggered technique can track the attitude command well. In reference [11], a new controller including an estimate of the external disturbance is proposed to deal with the presence of constant wind disturbance. When considering the external disturbance, this method just introduces the constant wind disturbance. A new adaptive controller is designed by using adaptive output feedback linearization and H∞ optimal control techniques in reference [12]. Simulation results show that the tracking errors are very large. An integrator backstepping approach is proposed in [13]. Several numerical simulation results verify the effectiveness in yaw direction. However, in this paper, it just uses this method in position control of the quadrotor. To deal with a quadrotor in the existence of matched disturbances, a barrier function adaptive nonsingular terminal sliding mode controller is designed in reference [14]. Several numerical simulations show the tracking performance using this method, while in this paper we will track more difficult curve and improve the external disturbance force. When considering input-delay, model uncertainty, and wind disturbance, an adaptive super-twisting terminal sliding mode control approach is designed in reference [15]. Comparing with the traditional sliding mode control method, the adaptive compensation term and the integral term are introduced to construct the sliding surfaces. Several simulations are provided to demonstrate the validation and success of planned control technique. As in [16], the Backstepping, Feedback Linearization control-oriented algorithms, NLGL, and Carrot-Chasing geometric algorithm are implemented and a quadrotor simulation platform is compared. Considering the disturbances and input constraints, the paper proposes a novel finite-time backstepping approach to tackle the trajectory tracking problem of quadrotors in Section 2. In Section 3, the paper gives the Lyapunov stability analysis. Several simulations of a quadcopter tracking a moving target are conducted in different environments to verify the performance of the system.

Above all these classical trajectory tracking methods, many researchers start introducing neural network into the UAV control system, especially in recent years. In reference [20], an adaptive switch controller (ASC) is proposed for the nonlinear multi-input multi-output system (MIMO). In this paper, the ASC is an online switch between the neural network adaptive PID (APID) controller and the neural network indirect adaptive controller (IAC). Through this method is workable, it does not consider any disturbance or parameter perturbation and just track the 2D curve. A robust neural adaptive BS control is designed to deal with the limitations proposed in [21]. Using this method, a neural approximator is introduced to estimate and compensate for the perturbations. In [22], the BP neural network is just used to train the flight data before constructing the related dynamic model and BP neural network plays a very weak role during the flight course. To deal with the UAV landing on a ground marker, an autonomous quadrotor landing algorithm using deep reinforcement learning is proposed in [23]. From all simulations, it is clear that this method just uses deep reinforcement learning approach to train the landing data. An adaptive dynamic controller to track a trajectory in 3D space is proposed in [24]. This controller structure consists of a kinematic controller that generates reference commands to a dynamic compensator in charge of changing the reference commands according to the system dynamics. Using this method, the parameters of the dynamic compensator are directly updated during navigation, configuring a directly updated self-tuning regulator with input error, aiming at reducing the tracking errors, thus improving the system performance in task accomplishment. However, the tracking performance is poor. In [25], the performance and accuracy of the inner control loop providing attitude control are investigated when using intelligent flight control systems trained with state-of-the-art RL algorithms, deep deterministic gradient policy (DDGP), trust region policy optimization (TRPO) and proximal policy optimization (PPO). However, this method just develops an open-source high-fidelity simulation environment to train a flight controller attitude control of a quadrotor through RL off-line training.

In a word, it is clear that when using neural network in the UAV control system, all those previous researchers often use signal layer neural network or use it to train the flight data off-line. However, the prominent disadvantage of using signal layer neural network and off-line calculation is that the environment adaptability is very poor. So, to improve the trajectory tracking accuracy and the environmental adaptability, the multilayer neural network is introduced into the control system and the major difference between previous methods is that the multiplayer neural network is online calculation. So, after introducing the conception of neural network, it is added to the quadrotor dynamic model to compensate the model error and external disturbance. Moreover, the neural network is added to the attitude and trajectory tracking control loop, and then taking three layers neural network as an example, the paper designs the adaptive neural network control law. At last, to illustrate the stability, the paper gives the Lyapunov stability analysis.

The rest of the paper is organized as follows: the trajectory tracking problem is defined in Section 2. In Section 3,
2. Trajectory Tracking Problem

In this paper, it addresses the trajectory tracking control problem of a quadrotor. This problem is defined as making a quadrotor track the pre-established trajectory in 3D space.

2.1. Trajectory Tracking Definition. The desired trajectory be described by a curve \( \tau_d(\gamma) = [x_d(\gamma), y_d(\gamma), z_d(\gamma)]^T \) where \( \gamma \in (0, \gamma_f) \). \( \gamma_f \) represents the total virtual arc length in the space. Then, the trajectory tracking problem can be regarded as designing an appropriate controller that ensures convergence of the position \( P(t) \) to \( \tau_d(\gamma) \). The desire trajectory \( \tau_d(\gamma) \) and position \( P(t) \) are shown in Figure 1.

2.2. Coordinate Definition. To describe the relative motion of the quadrotor, the coordinate frames are defined as follows:

\[
\begin{align*}
L_{eb} &= 
\begin{bmatrix}
  c(\theta)c(\psi) & c(\theta)s(\psi) & -s(\theta) \\
  s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) & s(\phi)s(\theta)s(\psi) + c(\phi)c(\psi) & s(\phi)c(\theta) \\
  c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) & c(\phi)c(\theta)
\end{bmatrix},
\end{align*}
\]

Using the same approach, the transformation matrix \( L_{be} \) that transforms from \( \zeta_b \) to \( \zeta_e \) is as follows:

\[
\begin{align*}
L_{be} &= 
\begin{bmatrix}
  c(\phi)c(\theta) & s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) & c(\phi)s(\theta)s(\psi) + c(\phi)c(\psi) \\
  c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) & c(\phi)c(\theta)
\end{bmatrix}.
\end{align*}
\]

In (1) and (2), \( s(*) \) and \( c(*) \) represents \( \sin(*) \) and \( \cos(*) \) function, respectively. Then, from (1) and (2), it is clear that

\[
L_{be} = L_{eb}^{-1}.
\]

3. Quadrotor Dynamic Model

Before presenting the dynamic model, the paper considers that the quadrotor is a grid body and be just affected by gravity, thrust, and aerodynamic drag during the flight. Then, using Newton’s equations of motion for translational and rotational dynamics, the dynamic model of the quadrotor will be obtained as follows:

\[
\begin{align*}
\dot{\mathbf{p}} &= \mathbf{V}, \\
\dot{\mathbf{V}} &= -g + L_{eb}F_{thrust} - 1/mK_t\mathbf{P}, \\
\dot{\mathbf{\Theta}} &= Rw, \\
J\dot{\mathbf{w}} &= -w \times (J\mathbf{w}) + \mathbf{M},
\end{align*}
\]

where \( \mathbf{P} = [x \ y \ z]^T \) and \( \mathbf{V} = [\dot{x} \ \dot{y} \ \dot{z}]^T \) refer to the quadrotor’s position and velocity vector, respectively. \( m \) is the mass, and \( g \) is the gravitational acceleration constant. \( e = [0 \ 0 \ 1]^T \) is a unitary vector. \( K_t \) denotes the aerodynamic drag coefficient, which is a diagonal matrix. \( \mathbf{w} = [\mathbf{p} \ \mathbf{q} \ \mathbf{r}]^T \) is body angular velocity and its time derivative \( \dot{\mathbf{w}} \). \( \mathbf{\Theta} = [\mathbf{\phi} \ \mathbf{\theta} \ \mathbf{\psi}]^T \) represents rate of change of Euler angles, which denote roll, pitch, and yaw angles, respectively. The rotation matrix \( R \) from \( \zeta_b \) and
\( \omega = \dot{\Theta} \). \hspace{1cm} (9)

Combining all these equations, the dynamic model of a quadrotor can be written in the following form:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3 \\
\dot{x}_4 \\
\dot{x}_5 \\
\dot{x}_6 \\
\dot{x}_7 \\
\dot{x}_8 \\
\dot{x}_9 \\
\dot{x}_{10} \\
\dot{x}_{11} \\
\dot{x}_{12}
\end{pmatrix} =
\begin{pmatrix}
x_2 \\
(I_y - I_z)x_4x_6/I_x + U_d d/I_x \\
x_3 \\
(I_z - I_x)x_6x_3/I_y + U_d d/I_y \\
x_4 \\
(I_x - I_y)x_3x_7/I_y + U_d d/I_z \\
x_5 \\
x_6 \\
x_7 \\
x_8 \\
x_9 \\
x_{10} \\
x_{11} \\
x_{12}
\end{pmatrix},
\]

(10)

where \( u_x, u_y, \) and \( u_z \) are

\[
\begin{pmatrix}
u_x \\
u_y \\
u_z
\end{pmatrix} =
\begin{pmatrix}
c(x_1) s(x_3) c(x_5) + s(x_1) s(x_5) \\
c(x_1) s(x_3) c(x_5) + s(x_1) s(x_5) \\
c(x_1) c(x_3)
\end{pmatrix},
\]

(11)

### 4. Sigma-Pi Neural Network and Adaptive Control Law Design

#### 4.1. Sigma-Pi Neural Network

Like other neural networks, the Sigma-Pi neural network also consists of input layer, hidden layer, and output layer. The hidden layer consisted of the summation neurons and quadrature neurons. The advantages of this structure are that it not only preserves the highly nonlinear mapping ability but also increases the flexibility of the network. So, according to the specific problem, one can construct the appreciate networks to improve the learning efficiency. The typical structure of Sigma-Pi neural network is shown in Figure 2.
As shown in Figure 2, the command and the state feedback consist the input space of neural network. The hidden layer is composed of alternating summation neurons and quadrature neurons. $X_i$ is an $N$-dimensional vector, and $x_i$ is the $i_{th}$ element of $X$. $W_{ij}$ is an adjustable weight from $X_i$ layer to $j_{th}$ quadrature neurons in hidden layer, and hidden layer also has $W_{jk}$ which is an adjustable weight from $j_{th}$ quadrature neurons to summation neurons. Choosing different hidden layer, there is an involved weight matrix between each layer. $h_{ij}$ is the $j_{th}$ quadrature neurons for $i_{th}$ output. $Y_N$ is the output of the neural network. Then, the equation of Sigma-Pi neural network is as follows:

$$
\begin{align*}
    h_{km} &= \prod_{i=1}^{m} \left( \sum_{j=i}^{k} \left( \prod_{t=1}^{j} x_{i}W_{ij}\beta + \theta_{ij} \right) \xi_{jk} \right) + \theta_{km}, \\
    Y_N &= \sigma \sum_{i=1}^{n} h_{km} \xi_{mn} (N = 1 \cdots N).
\end{align*}
$$

In (12), $m$ is the net layers of neural network excepting output layer. When $m = 1$, the hidden layer just has a signal net layer with quadrature neurons. $\sigma (x)$ denotes the non-linear activation function and is selected as the logistic function, $\sigma (x) = 1/(1 + e^{-x})$ for all the results reported in this paper. $\beta$ represents the basic function vector. $\theta_{ij}$ denotes the adjustable threshold of the $j_{th}$ quadrature neurons of the $i_{th}$ output. In this paper, it will talk about adding the neural network to attitude and position control system.

### 4.2. Sigma-Pi Neural Network Control Structure

To design the corresponding Sigma-Pi neural network control law, this paper will combine the sliding mode control method. The control structure is shown in Figure 3.

As shown in Figure 3, it includes trajectory references module, command filter module, controller, adaptive control module, dynamics module, and sliding mode control module. The adaptive control structure is shown in Figure 4.

As shown in Figures 4 and 5, $U_{NN}$ represents the control input calculated by neural networks. The number of hidden layers and neurons is determined by the specific problem. In this paper, we will discuss the one or three hidden layers Sigma-Pi neural network.
4.3. Adaptive Control Law Design. In this section, this paper will design the adaptive control law. Combining (4) and considering the modelling error, equation (4) can be written as follows:

\[
\begin{align*}
X_1 &= X_2 \\
\dot{X}_2 &= f(X_1, X_2, U, M) + U_{NN} - \tilde{e},
\end{align*}
\]  

where \(X_1 = [\phi \theta \psi x y z]^T\) presents the Euler angles and position of the quadrotor; \(U_{NN}\) represents the neural network control input; \(\tilde{e} = [e_\phi e_\theta e_\psi e_x e_y e_z]^T\) denotes the modelling error; \(U\) represents the input generated by rotors; and \(M\) denotes the moment generated by fours rotors.

In this paper, the neural network is used to compensate the tracking error. If we choose different hidden layers, the control input \(U_{NN}\) will be different. If we chose a signal hidden layer, the neural network will become a neural network with the input layer and a signal hidden layer with quadrature neurons. Then, the neural network control input \(U_{NN}\) can be written as follows:

\[
\begin{align*}
Y_N &= \prod_{i=1}^{j} x_i W_{ij}^T \beta + \theta_{ij} \\
U_{NN} &= Y_N \\
N &= 1 \cdots n
\end{align*}
\]  

where \(\beta\) represents basis function vector which is defined as follows:

\[
\beta = \text{kron[kron[C1, C2, C3],}
\]

where \(\text{kron}\) represents the Kronecker product.

To design the adaptive control law, the paper defines that \(U_{NN}^{*}\) is the optimal neural network control input. Then, we will get the following equation:

\[
\begin{align*}
Y_N^* &= \prod_{i=1}^{j} x_i W_{ij}^{*T} \beta + \theta_{ij}^* \\
U_{NN}^* &= Y_N^* (N = 1 \cdots n) \\
U_{NN} - U_{NN}^* &= Y_N - Y_N^* = \prod_{i=1}^{j} x_i W_{ij}^{*T} \beta + \theta_{ij}^*,
\end{align*}
\]

where \(\tilde{e}\) represents the optimal weight coefficient matrix; then, the error between the optimal control input and the modelling error can be expressed as follows:

\[
U_{NN} - \tilde{e} = \prod_{i=1}^{j} x_i W_{ij}^{*T} \beta + U_{NN} - \tilde{e}.
\]  

So, choosing different hidden layers, we will get different 
\(\text{Sigma-Pi neural network control input } U_{NN}^{*}.\)
4.3.1. Attitude Control Law Design. The attitude control law will be designed in this section, and then the paper will give the appropriate Lyapunov-type stability analysis.

**Theorem 1.** The adaptive attitude control law with three layers neural network for yaw, roll, and pitch is defined as follows.

Then, the paper defines the Lyapunov function related to and \( \tilde{W}_{ij} \).

Then, the first-order differential equation \( \dot{V}(s, \tilde{W}_{ij}^T) \) can be defined as the following equation:

\[
\dot{V}(s, \tilde{W}_{ij}^T) = s \dot{s} + \lambda (\tilde{W}_{ij}^T \tilde{W}_{ij})^* - \eta \text{sgn}(s) + \lambda \tilde{W}_{ij}^T \dot{x}^*.
\]

According to (26), we will define the adaptive control law as follows:

\[
\dot{x}^* = -\kappa_2 \sum_{i=1}^{j} x_i \beta
\]

Substituting into (26) and (27), the first-order differential \( \dot{V}(s, \tilde{W}_{ij}^T) \) function is as follows:

\[
\dot{V}(s, \tilde{W}_{ij}^T) = -\eta |s| + \kappa_2 \sum_{i=1}^{j} x_i \beta + \kappa_2 \tilde{\theta}^*_{ij} + \lambda \tilde{W}_{ij}^T \dot{x}^*.
\]

According to (26), when \( \eta \geq \kappa_2 \tilde{\theta}^*_{ij} \), the Lyapunov function \( V(s, \tilde{W}_{ij}^T) \leq 0 \). The system is stable.

Using the same method, we can get the adaptive Sigma-Pi neural network control law of the pitch and the yaw channel.
\[ \begin{align*}
  \kappa_1 &= \frac{(I_y - I_z)}{I_x}, \\
  \kappa_2 &= \frac{d}{I_x}, \\
  \kappa_3 &= \frac{(I_z - I_x)}{I_y}, \\
  \kappa_4 &= \frac{d}{I_y}, \\
  \kappa_5 &= \frac{(I_x - I_y)}{I_z}, \\
  \kappa_6 &= \frac{d}{I_z}.
\end{align*} \] (30)

4.3.2. Position Control Law Design

**Theorem 2.** The adaptive position control law with three-layer neural network in the \( x, y, \) and \( z \) direction is defined as follows:

\[ \begin{align*}
  \dot{W}_{ijx} &= -\frac{u_x}{\lambda_x m} \prod_{i=1}^{j} x_i \beta, \\
  \dot{W}_{ijy} &= -\frac{u_y}{\lambda_y m} \prod_{i=1}^{j} x_i \beta, \\
  \dot{W}_{ijz} &= -\frac{u_z}{\lambda_z m} \prod_{i=1}^{j} x_i \beta.
\end{align*} \] (31)

To reduce the trajectory tracking error, the paper also adds the Sigma-Pi neural network to quadrotor position control loop. Using the same method, we can get the Sigma-Pi neural network compensation control law in the \( x, y, \) and \( z \) direction. Firstly, we will define the tracking error in \( x \) direction as follows:

\[ e_x = x_x - x_{x_d}. \] (32)

Then, the first-order differential equation \( \dot{e}_x \) can be obtained as the following equations:

\[ \dot{e}_x = \dot{x}_x - \dot{x}_{x_d}. \] (33)

According to (32) and (33), we can construct the sliding surface as follows:

\[ s = \epsilon_x e_4 + \dot{e}_4. \] (34)

Then, the first-order differential function \( \dot{s} \) can be defined as the following equation:

\[ \dot{s} = c_4 \epsilon_x + \dot{e}_4 = c_4 \dot{x}_x + \dot{x}_8 - \dot{x}_{x_d} \]

\[ = c_4 (x_8 - \dot{x}_x - \dot{x}_{x_d} - \epsilon_x). \] (35)

According to (11), \( \dot{x}_8 \) is defined as follows:

\[ \dot{x}_8 = \frac{u_x (U_1 + U_{NNx})}{m} - \kappa_7 x_8, \] (36)

where \( \kappa_7 = \frac{k_7}{m} \). Then, we can get the function \( \dot{s} \) as follows:

\[ \dot{s} = c_4 \epsilon_x + \dot{e}_4 = c_4 \dot{x}_x + \dot{x}_8 - \dot{x}_{x_d} \]

\[ = c_4 (x_8 - \dot{x}_x - \dot{x}_{x_d} - \epsilon_x). \] (37)

Then,

\[ \dot{s} = \frac{u_x (U_1 - U_{NNx})}{m} - \eta \text{sgn}(x) \]

\[ = \frac{u_x \left( \prod_{i=1}^{j} x_i \dot{W}_{ijx} + \theta_{ijx}^* \right) }{m} - \eta \text{sgn}(x), \] (38)

where \( \eta \geq \theta_{ijx}^* \).

Then, we can define the Lyapunov function which is related to \( s \) and \( \tilde{W}_{ijx}^* \)

\[ V(s, \tilde{W}_{ijx}^*) = \frac{1}{2} \dot{s}^2 + \frac{1}{2} \lambda \tilde{W}_{ijx}^* \tilde{W}_{ijx}^*. \] (39)

Then, the first-order differential equation \( \dot{V}(s, \tilde{W}_{ijx}^*) \) can be defined as the following equation:

\[ \dot{V}(s, \tilde{W}_{ijx}^*) = \dot{s} \dot{s} + \lambda \tilde{W}_{ijx}^* \tilde{W}_{ijx}^* \]

\[ = s \frac{u_x \left( \prod_{i=1}^{j} x_i \dot{W}_{ijx} + \theta_{ijx}^* \right) }{m} - \eta \text{sgn}(s) \]

\[ + \lambda \tilde{W}_{ijx}^* \tilde{W}_{ijx}^*, \]

\[ = \frac{u_x \left( \prod_{i=1}^{j} x_i \dot{W}_{ijx} + \theta_{ijx}^* \right) }{m} - \eta |s| + \lambda \tilde{W}_{ijx}^* \tilde{W}_{ijx}^*, \]

\[ = -\eta |s| + s \frac{u_x \theta_{ijx}^*}{m} + s \tilde{W}_{ijx}^* \left( \frac{u_x \prod_{i=1}^{j} x_i \beta}{m} + \lambda \tilde{W}_{ijx}^* \right). \] (40)
According to equations (13), (31), and (38), $\dot{V}(s, \bar{W}_{ij}^*)$ can be written as follows:

$$
\dot{V}(s, \bar{W}_{ij}^*) = \dot{s} + \lambda \bar{W}_{ij}^* \bar{x}_{ij}^*
$$

$$
\leq -\eta |s| + \sum_{i=1}^m \frac{u_{s}^* \theta_{ijx}^*}{m} + s \bar{W}_{ij}^* \left( \sum_{i=1}^l \frac{\beta_{ijx}^*}{m} + \lambda \bar{W}_{ij}^* \right),
$$

$$
\leq -\eta |s| + \sum_{i=1}^m \frac{u_{s}^* \theta_{ijx}^*}{m},
$$

$$
\leq -\eta |s| + s \bar{\theta}_{ijx}^* \leq 0.
$$

(41)

According to (41), the Lyapunov function $\dot{V}(s, \bar{W}_{ij}^*) \leq 0$. The system is stable.

Using the same method, we can get the adaptive Sigma-Pi neural network control law in the $y$ and $z$ direction.

$$
\dot{\bar{W}}_{ijy} = -\frac{u_{x} \prod_{i=1}^l x_{ijy}^*}{\lambda_{y} m} \bar{W}_{ijy} = -\frac{u_{x} \prod_{i=1}^l x_{ijy}^*}{\lambda_{y} m}.
$$

(42)

5. Numerical Simulation

In this section, several numerical examples will be introduced to demonstrate the effectiveness of the adaptive control law. The simulation parameters of quadrotor are shown in Table 1. In this paper, to solve the problem and simulate all experiments, we all use Matlab Simulink software and the computer we used is ThinkPad E480. The parameter of this computer is shown in Table 2.

5.1. Case A: Simulation with Different Hidden Layers. In this case, the paper will demonstrate that the tracking performance will be better when increasing the hidden layers. So, the paper uses different hidden layers neural network to track the same curve. The desired curve is shown in Table 3. Simulation results are shown in Figures 6–8.

As can be seen in Figure 6, it illustrates the trajectory tracking results using different hidden layers neural network. These three curves are reference curve, the tracking curve with a signal hidden layer, and the tracking curve with 3 hidden layers. It is clear that the quadrotor can soon track the desired trajectory using three hidden layers neural network. To be more specific, it is observed that with increase in hidden layer, the tracking performance becomes better.

The curve of tracking error by using neural network with different hidden layers is shown in Figure 7. It is obvious that the tracking error with three hidden layers neural network is obviously small in each direction. To be more specific, we can also observe that using three hidden layers neural network, the track error will be soon converged to 0 before 4s in each direction. So, we can conclude that with the increase in network layers, the tracking error will be soon converged and the tracking time can also be decreased.
obviously eliminated. It also demonstrates that introducing the neural network into the attitude control loop can improve the stability of the system. At the same time, it can decrease the jitter of the control system.

5.2. Case B: Tracking 3D “8-Shaped” Spiral Curve. In this simulation, the paper will evaluate the lift, sideslip, and forward performances of a quadrotor to track the “8-shaped” spiral curve using three hidden layers neural network. Comparing with reference [24], we increase the tracking difficulty that the initial position error is larger and improve the frequency of sine curves. The desired curve parameters are shown in Table 4. All these simulation results are shown in Figures 9–12.

Figures 9 and 10 show the result of tracking “8-shaped” curve based on the adaptive trajectory tracking method proposed in this paper. From different perspectives, we can observe that the quadrotor can soon track the desired trajectory. It is also demonstrated that the quadrotor has a better performance in lift, sideslip, and forward. Comparing with reference [24], the tracking performance is better, even increasing the tracking difficulty.

Figure 11 shows the trajectory tracking error curve when tracking the “8-shaped” curve. Comparing with reference [28], when tracking the same 8-shaped trajectory, it is obvious that the tracking error in each direction is very small and the initial tracking error will be soon converged to 0 before 3 s.

Figure 12 shows the change of attitude angle. It is obvious that despite the initial attitude angle fluctuates greatly, the attitude will be soon converged. The main reason for this phenomenon is that the difference between the initial position, pointing and the desired position, pointing.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>Desire trajectory ( (x_1) )</td>
<td>( 8 - 8 \cos \left( \pi (t - 4)/6 \right)/m )</td>
</tr>
<tr>
<td>( y_1 )</td>
<td>Desire trajectory ( (y_1) )</td>
<td>( 4 \sin \left( \pi (t - 4)/3 \right)/m )</td>
</tr>
<tr>
<td>( z_1 )</td>
<td>Desire trajectory ( (z_1) )</td>
<td>( -0.1 \left( 1 - \exp \left( -0.3t \right) \right)/m )</td>
</tr>
<tr>
<td>( x_1 )</td>
<td>Init ( (x_1) )</td>
<td>( 8/m )</td>
</tr>
<tr>
<td>( y_1 )</td>
<td>Init ( (y_1) )</td>
<td>( 3/m )</td>
</tr>
<tr>
<td>( z_1 )</td>
<td>Init ( (z_1) )</td>
<td>( 0/m )</td>
</tr>
</tbody>
</table>
5.3. Case C: Anti-Interference Simulation. To verify the disturbance rejection properties of the method, we will add the external disturbance which affects the translation motion in x, y, and z direction. In the simulation, the quadrotor will also track the 3D curve used in Case A and uses three hidden layers neural network to compensate the tracking error. The involved 3D curve parameters are shown in Table 5.

As shown in Table 5, the paper will regard the white noise as the external disturbance force added to the system and the units of this force is N. The corresponding disturbance force curve is shown in Figure 13. As shown in Figure 13, it is clear
that the change of the noise force is so random and the magnitude of the force is bigger than reference [10]. So, this type of the external force is very suitable to verify the disturbance rejection properties of the method proposed in this paper. The trajectory tracking results in disturbance circumstance are shown in Figures 13–17.

As can be seen in Figures 14 and 15, even adding the external disturbance to the system, the quadrotor still tracks the desire curve well. In addition, we can also observe that when disturbances occur, the tracking curve fluctuates slightly. However, the quadrotor will soon be stabilized.

Figure 16 shows the trajectory tracking error curve of the quadrotor. It is obvious that despite adding the strong noise to the system, the tracking error in each direction is very small and the initial tracking error will be soon converged to 0 before 4 s. As a whole, the system is unaffected by the noise force. Comparing with reference [26], it is clear that the method proposed in this paper has a strong anti-interference feature.

Figure 17 shows the change of the attitude angle. It is obvious that when adding the noise force to each direction, the involved Euler angles fluctuate slightly. This illustrates that introducing the neural network to the system can improve the ability of anti-inference. It also demonstrates that the external disturbances can be successfully rejected.
6. Conclusions

This paper proposes a new adaptive trajectory tracking algorithm of the quadrotor by introducing multilayer neural network into the system. First of all, the paper gives the dynamic model of the quadrotor. Then, the principle of Sigma-Pi neural network was introduced. At the same time, the paper designs the corresponding adaptive control law and gives the Lyapunov-type stability analysis. To demonstrate the effectiveness of the method proposed in this paper, several involved simulations are given. By comparison, using the adaptive control method proposed in this paper can decrease the tracking error and improve the stability of the system. We also conclude that with increasing the neural network layer, the trajectory tracking performance is becoming better.

7. Future Recommendations

Though a new adaptive trajectory tracking algorithm is better than traditional process especially in having a high-precision in trajectory tracking, the stability of the system is improved. There remain many problems such as how to transplant the adaptive trajectory tracking algorithm to the platform. In our experience, we cordially recommend the Xilinx-Z7 platform. When transplanting the algorithm, we will face how to simplify the algorithm model to increase computational efficiency. We will continue researching all these problems in the future.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to thank the support of the flight control lab of Beijing University of Technology for this research. This study was supported by the National Natural Science Foundation of China (no. 61673212) and the Natural Science Foundation of Jiangsu Province (no. BK20161490).

References


