Chapter 2
Neural Network-Based Motion Control Algorithm for Perching Nano-Quadrotor on Outdoor Vertical Surface

Sandeep Gupta and Laxmidhar Behera

1 Introduction

Nano-quadrotors are light weighted, compact in size and utilized in wide variety of applications due to which attracted the interest of researchers and industrialists. They have multiple modes of operation like hovering, vertical take-off, aggressive manoeuvrability and landing on different surfaces including inclined and rough surfaces [1]. Moreover, the nano-quadrotors offer satisfactory performance in spite of its complex and coupled dynamics. Due to significant advances in nonlinear control theory and micro-feedback sensors, there is vast opportunity for the new applications of nano-quadrotors perching on walls [2–3] and cable lines for surveillance [4]. The common applications of quadrotors are remote inspection, video shooting and defence services [5]. The effective attitude and position control algorithms are required for quadrotor's autonomous flight operation.

The quadrotor altitude and position control are relatively complicated due to unmodelled dynamics and multivariable input–output system. In this system, there are four inputs to control the roll, pitch, yaw angle and throttle although a quadrotor's dynamics have mainly six outputs as 3-D position (x, y, z) and Euler angles $(\emptyset, \theta, \varphi)$. There are variety of motion control algorithms that exist in literature which are developed for fully actuated systems, but these algorithms cannot be applied directly for underactuated nonlinear systems. The modelling of nano-UAVs usually subjected to unmodelled dynamics [6]. Hence, in development of control law, high nonlinearity and aerodynamics disturbances must to address properly.

In the past few years, different versions of PID and LQR from linear control theory have been developed and implemented for quadrotor system. These control techniques have used linearized model and not effective for such system with coupled

S. Gupta (✉) · L. Behera
Electrical Engineering Department, Indian Institute of Technology, Kanpur, India
e-mail: sngupta@iitk.ac.in

R. Tiwari et al. (eds.), *Proceedings of International Conference on Computational Intelligence*, Algorithms for Intelligent Systems,
https://doi.org/10.1007/978-981-19-2126-1_2

dynamics [7, 8]. These types of control methods are suitable for attitude control and the hovering state of UAV. The other nonlinear control techniques like backstepping control, sliding mode control, feedback linearization, and adaptive control have been developed to control the attitude and position of quadrotor which are interconnected [9, 10]. The hybrid nonlinear control technique is also used for improved trajectory tracking in [11]. The other category of nonlinear control algorithms is intelligent control techniques such as neural networks (NN's) and fuzzy logic-based control methods [12, 13]. Many researchers have used neural networks in the system identification as well as in designing controller for the nonlinear systems with unknown dynamics. Few control algorithms have been used with neural networks which are model reference adaptive control (MRAC), model predictive control (MPC) and feedback linearization (FL).

Generally, in designing of neural network controllers two steps are used such as (a) system identification and (b) controller design. In the first step, a neural network model is designed for the system. The neural network model of the system is trained offline in batch mode. In the second step, controller is designed and its parameters are optimized using neural network for desired performance. The model predictive control has the largest computational time while in feedback linearization control scheme, computational time is comparatively very small as the designed control law is simply a rearrangement of the system model. In recent times, many researchers proposed neural network-based control techniques for autonomous quadrotor flight operation [14]. The main interest of researchers is to develop adaptive neural control law [15, 16] in the presence of parameter variations, external disturbance and unmodelled dynamics of the UAV system. The major benefit of choosing ChNN is its less complex structure in comparison of multilayer feed forward neural networks for efficient computations [17].

In this work, direct adaptive neural control algorithm is presented for unknown dynamics in the system matrix and control matrix of the system model. The stability analysis is presented via Lyapunov theorem and deriving the weight update laws for Chebyshev neural networks (ChNN). The major benefit of choosing ChNN is its simple structure which requires less computational resources as in case of nano-quadrotors. It provides simpler implementation in embedded systems and fast convergence rate. The proposed neural network-based control algorithm is designed to preserve the stability during the perching operation while keeping minimum error in trajectory tracking of nano-quadrotor.

The paper is organized as follows. Section 2 presents the quadcopter dynamic modelling and problem formulation. Design of ChNN-based control algorithm is introduced in Sect. 3. Finally, Sect. 4 provides the experimental results for the quadrotor perching application for surveillance. The conclusion and future scope of this research article are given in Sect. 5.

2 Quadrotor Dynamic Modelling and Problem Formulation

A nano-quadrotor is simply a scaled-down version of a normal quadrotor. Nano-quadrotor is lightweight. Due to its various advantages such as silent operation, agility, safety, and so on, the nano-quadrotor is a rapid developing area of small UAVs. There is limited payload capacity, limited flight time due to small battery and compact size of the nano-quadrotor which create major challenges to flight control development. The quadrotor is an underactuated highly nonlinear system, where its twelve states are controlled by four control inputs. The governing dynamics of the quadrotor in three-dimensional space is taken from [18].

$$\begin{aligned}
\ddot{\emptyset} &= \dot{\theta}\dot{\Psi}\left(\frac{J_y - J_z}{J_x}\right) + \dot{\emptyset}\frac{k_{\emptyset}}{J_x} + \frac{d}{J_x}u_2 \\
\ddot{\theta} &= \dot{\emptyset}\dot{\Psi}\left(\frac{J_z - J_x}{J_y}\right) + \dot{\theta}\frac{k_{\theta}}{J_y} + \frac{d}{J_y}u_3 \\
\ddot{\Psi} &= \dot{\emptyset}\dot{\theta}\left(\frac{J_x - J_y}{J_z}\right) + \dot{\Psi}\frac{k_{\Psi}}{J_y} + \frac{1}{J_z}u_4 \\
\ddot{z} &= -g - \frac{k_z\dot{z}}{m} + \frac{u_1}{m}(\cos\emptyset\cos\theta) \qquad (1)\\
\ddot{x} &= -\frac{k_x\dot{x}}{m} + \frac{u_1}{m}(\sin\theta\cos\Psi\cos\emptyset + \sin\Psi\sin\emptyset) \\
\ddot{y} &= -\frac{k_y\dot{y}}{m} + \frac{u_1}{m}(\sin\theta\cos\emptyset\sin\Psi - \cos\Psi\sin\emptyset)
\end{aligned}$$

where roll, pitch and yaw are represented by $(\emptyset, \theta, \Psi)$, respectively, and (x, y, z) are position coordinates. m is the mass, J_x, J_y, J_y are the inertial values in respective coordinates. The aerodynamics damping/drag coefficients are given as $k_x, k_y, k_z, k_{\emptyset}, k_{\theta}, k_{\Psi}$ and d is moment arm. u_1 is force and u_2,u_2, u_3 torques. These four control inputs [17] are given as $u_1 = F_1+F_2+F_3+F_4$, $u_2 = -F_1-F_2+F_3+F_4$, $u_3 = -F_1 + F_2 + F_3 - F_4$,and $u_4 = -F_1 + F_2 - F_3 + F_4$ where F_1, F_2,F_3, and F_4 are applied forces on the rotors. When the quadrotor aligns in perching mode, the governing dynamics will become [2].

$$\begin{aligned}
\ddot{\theta}_p &= \frac{d}{J_y}u_{\theta p} \\
\ddot{z}_p &= -g - \frac{k_z\dot{z}}{m} + \frac{u_{\text{zp}}\cos\theta_p}{m} \qquad (2)
\end{aligned}$$

$$\ddot{x}_p = -\frac{k_x \dot{x}}{m} + \frac{u_{xp} \sin\theta_p}{m}$$

For the nonlinear model represented in (1) and (2), the objective is to track the desired path. $x_d, y_d, z_d, \theta_d, \emptyset_d, and \Psi_d$ are the desired values of the states of the system. Desired values of pitch and roll are given in equations (3) and (4).

$$\emptyset_d = \sin^{-1}\left(\frac{m}{u_1}\left(u_x \sin\Psi_d - u_y \cos\Psi_d\right)\right) \tag{3}$$

$$\theta_d = \sin^{-1}\left(\frac{u_x \cos\Psi_d - u_y \sin\Psi_d}{\sqrt{\left(1 - \left(u_x \sin\Psi_d - u_y \cos\Psi_d\right)^2\right)}}\right) \tag{4}$$

Two virtual control inputs u_x and u_y are given as $u_x = (\sin\theta \cos\Psi \cos\emptyset + \sin\Psi \sin\emptyset)$ and $u_y = (\sin\theta \cos\emptyset \sin\Psi - \cos\Psi \sin\emptyset)$. In the case of perching (2), the desired states are $\theta_{\text{pd}}, x_{\text{pd}}$, and z_{pd}. The virtual control inputs are $u_{\text{xp}}, u_{\text{zp}}$ and $u_{\theta p}$. The quadrotor dynamics (1) can be divided into two subsystems: translational subsystem $\ddot{T} = f_t(T_r, \dot{T}_r) + g_t(T_r, \dot{T}_r)u_t$ and rotational subsystem $\ddot{R}_t = f_r(R_t, \dot{R}_t) + g_r(R_t, \dot{R}_t)u_r$, where $T_r = [x, y, z]$ and $R_t = [\emptyset, \theta, \Psi]$. The system matrix f_t,f_r are unknown dynamics and the control matrix g_t, g_r are known functions of the quadrotor mathematical model. For perching mode, 2-D system dynamics are represented as $\ddot{T}_{\text{rp}} = f_{\text{tp}}(T_{\text{rp}}, \dot{T}_{\text{rp}}) + g_{\text{tp}}(T_{\text{rp}}, \dot{T}_{\text{rp}})u_{\text{tp}}$ and $\ddot{R}_{\text{tp}} = f_{\text{rp}}(R_{\text{tp}}, \dot{R}_{\text{tp}}) + g_{\text{rp}}(R_{\text{tp}}, \dot{R}_{\text{tp}})u_{\text{rp}}$.

2.1 Problem Statement

The control inputs $u_1, u_2, u_3, u_4, u_x, u_y, \left(u_{xp}, u_{zp}, u_{\theta p}\right)$ and adaptive control law have to be designed for relative degree two type of subsystems assuming that f_t,f_r,f_{tp} and f_{rp} for both the subsystems are unknown dynamics of the system. Here, feedback linearization-based controller is designed and the unknown dynamics of the quadrotor model is estimated using adaptive neural network (ChNN). The error dynamics for which the controller has been designed is discussed below.

3 Design of ChNN-Based Control Algorithm

A general MIMO system is given as follows:

$$\dot{x} = f(x) + g(x)u \tag{5}$$

$$y = \text{cx} \tag{6}$$

where $x \in R^n$, $f(x) \in R^n$,$u \in R^m$, $g(x) \in R^{m\times n}$,$y \in R^p$,$c \in R^{p\times n}$. Each subsystem of quadrotor dynamics (1) denotes a single input single output nonlinear system. The altitude dynamics are considered, where we take $z = z_1$ and $\dot{z} = z_2$ then altitude dynamics will become:

$$\dot{z}_1 = z_2 \tag{7}$$

$$\dot{z}_2 = f(z) + g(z)u_1 \tag{8}$$

$$y = z_1 \tag{9}$$

The control objective is to design a control input u_1 so that quadrotor follows the desired trajectory $z_d(t)$, while the states stay bounded [16–19]. Using the feedback linearization technique, the control input law is provided as

$$u_1 = (g(z))^{-1}\left[-f(z) + P_{\text{dd}}E_z + \rho\dot{e}_z + \dot{z}_{2d}\right] \tag{10}$$

where e is the tracking error, $z \to (z_1, z_2)$, $e_z = z_1 - z_{1d}$, a new variable $E_z = \rho e_z + \dot{e}_z$ and design parameter terms P_{dd} and ρ are selected as positive integers. To approximate the unknown function (z), the Chebyshev neural network as adopted. The structure of ChNN contains input layer, Chebyshev polynomial-based function expansion block, tunable weights of the network, a single output node and an adaptive law for tuning the weights neural network providing least estimation error. This neural network model has two parts in which numerical transformation is first part and learning algorithm is second part. The numerical transformation is the first part where enhanced input pattern is multiplied by weight vector to get the network output. In the second part, an adaptive law is taken for weights improvement to minimize the estimation error.

The numerical transformation as a functional expansion (FE) is to expand input pattern. The two initial Chebyshev polynomials are chosen as $P_0(e) = 1,\ P_1(e) = e$ and $[P_2(e),\ P_3(e), \ldots\ldots,\ P_N(e)]$ will be generated by a recursive function [20, 21] as $P_{N+1}(e) = 2eP_N(e) - P_{N-1}(e)$, where $P_N(e)$ represents nth-order polynomial. The argument is taken as $-1 < \text{e} < 1$, and input pattern is n dimensional such as $e = (e_1, e_2, \ldots., e_n)^T \in R^n$. Then the enhanced pattern will become as

$$\beta(e) = [P_0(e_1), P_1(e_1), \ldots, P_N(e_1), \ldots, P_0(e_n), P_1(e_n), \ldots, P_m(e_n)]^T \tag{11}$$

Hence, ChNN neural network output will be expressed as $O_{ch} = W_{\text{mn}}^T\beta(e)$, where Chebyshev polynomial basis function is $\beta(e)$, order of Chebyshev polynomial is m, number of inputs is n in input layer and W_{mn} is weight vector of network. Figure 1 shows the structure of ChNN model.

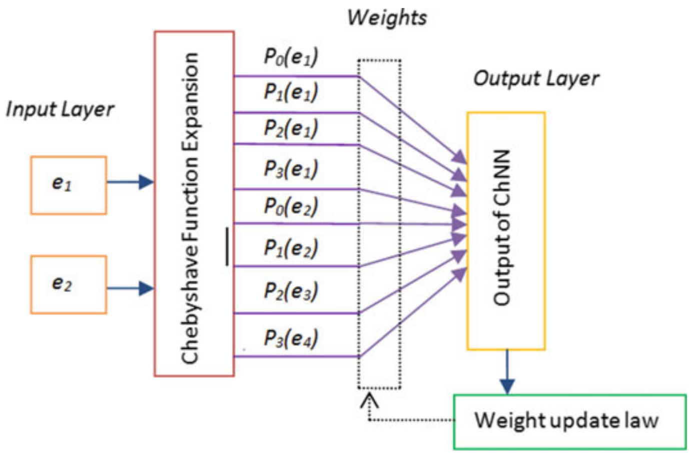


Fig. 1 Model of single layer two-input Chebyshev neural networks

Based on ChNN model, unknown function $f(z)$. can be approximated as $f(z) = W^T_{\mathrm{mm}}\beta(e_{\mathrm{in}}) + \varepsilon$, where $e_{\mathrm{in}} = [e, \dot{e}]^T$ is input vector, $W_{\mathrm{mn}} = [W_{11}, W_{21}, W_{31}, W_{41}, W_{12}, W_{22}, W_{32}, W_{42}]$ is weight vector and ε is approximation error. Now the estimation of $f(z)$ is provided as $\hat{f}(z) = \hat{W}^T_{\mathrm{mn}}\beta(e_{\mathrm{in}})$ where $\hat{W}_{\mathrm{mn}}$ is estimation of W_{mn}. Now putting $\hat{f}(z)$ in place of $f(z)$ in Eq. (10), the control law will become

$$u_1 = (g(z))^{-1}\Big[-\hat{f}(z) + P_{\mathrm{dd}}E_z + \rho\dot{e}_z + \dot{z}_{2d}\Big] \tag{12}$$

Let us assume $\varepsilon = 0$ and putting control law u_1 in system Eq. (8) we get,

$$\begin{aligned}\dot{z}_2 =& f(z) + g(z)\cdot(g(z))^{-1}\Big[-\hat{f}(z) + P_{\mathrm{dd}}E_z + \rho\dot{e}_z + \dot{z}_{2d}\Big]\\ =& W^T_{\mathrm{mn}}\beta - \hat{W}^T_{\mathrm{mn}}\beta + P_{\mathrm{dd}}E_z + \rho\dot{e}_z + \dot{z}_{2d}\end{aligned} \tag{13}$$

Defining $\tilde{W}^T = W^T_{\mathrm{mn}} - \hat{W}^T_{\mathrm{mn}}$, the equation will become

$$\dot{z}_{2d} - \dot{z}_2 = -\tilde{W}^T\beta - P_{\mathrm{dd}}E_z - \rho\dot{e}_z \tag{14}$$

$$\ddot{e}_z = -\tilde{W}^T\beta - P_{dd}E_z - \rho\dot{e}_z \tag{15}$$

We have $E_z = \rho e_z + \dot{e}_z$ and after differentiating this equation, we get $\dot{E}_z = \rho\dot{e}_z + \ddot{e}_z$. Substituting $\ddot{e}_z = \dot{E}_z - \rho\dot{e}_z$ in Eq. (15),

$$\dot{E}_z - \rho\dot{e}_z = -\tilde{W}^T\beta - P_{\mathrm{dd}}E_z - \rho\dot{e}_z \tag{16}$$

$$\dot{E}_z = -\tilde{W}^T\beta - P_{\mathrm{dd}}E_z \tag{17}$$

This is the linear and stable closed-loop error dynamics after putting control law u_1 in Eq. (8). To obtain the weight update law, Lyapunov theory is used. Let us chose a Lyapunov function V_{z} in which D is a positive definite matrix.

$$V_z = \frac{1}{2}E_z^2 + \frac{1}{2}\tilde{W}^T D^{-1}\tilde{W} \tag{18}$$

$$\dot{V}_z = E_z\dot{E}_z + \tilde{W}^T D^{-1}\dot{\tilde{W}} \tag{19}$$

From Eq. (17), putting $\dot{E}_z$ into (19),

$$\dot{V}_z = E_z\left(-\tilde{W}^T\beta - P_{\mathrm{dd}}E_z\right) + \tilde{W}^T D^{-1}\dot{\tilde{W}} \tag{20}$$

$$\dot{V}_z = -P_{\mathrm{dd}}E_z^2 - \tilde{W}^T(\beta E_z + D^{-1}\dot{\hat{W}}_{\mathrm{mn}}) \tag{21}$$

If we chose the second term as 0 in Eq. (21), $\dot{V}_z = -P_{\mathrm{dd}}E_z^2$ which satisfy the condition of stability in sense of Lyapunov ($V_z > 0$ and $\dot{V}_z \leq 0$). The update law from Eq. (21) is given below as

$$\left(\beta E_z + D^{-1}\dot{\hat{W}}_{\mathrm{mn}}\right) = 0 \tag{22}$$

$$\dot{\hat{W}}_{\mathrm{mn}} = -D\beta E_z \tag{23}$$

Finally, the control low for the altitude control will become as,
$u_1 = \frac{m}{\cos\emptyset\cos\theta}\left[-\hat{W}_{\mathrm{mn}}^T\beta(e_{\mathrm{in}}) + P_{\mathrm{dd}}E_z + \rho\dot{e}_z + \ddot{z}_d\right]$,

$$\dot{\hat{W}}_{\mathrm{mn}} = -D\beta E_z \tag{24}$$

where $\ddot{z}_d = \dot{z}_{2d}$. The same procedure is used to obtain different control laws for rest control inputs (roll, pitch, and yaw) including two virtual control for x and y positions. The control inputs are:

$$\begin{aligned} u_x &= \frac{m}{u_1}\left[-\hat{W}_{\mathrm{mnx}}^T\beta(e_{\mathrm{inx}}) + P_{\mathrm{ddx}}E_x + \rho_x\dot{e}_x + \ddot{x}_d\right], \\ \dot{\hat{W}}_{\mathrm{mnx}} &= -D_x\beta E_x \end{aligned} \tag{25}$$

$$\begin{aligned} u_y &= \frac{m}{u_1}\left[-\hat{W}_{\mathrm{mny}}^T\beta\left(e_{\mathrm{iny}}\right) + P_{\mathrm{ddy}}E_y + \rho_y\dot{e}_y + \ddot{y}_d\right], \\ \dot{\hat{W}}_{\mathrm{mny}} &= -D_y\beta E_y \end{aligned} \tag{26}$$

$$u_2 = \frac{J_x}{d}\left[-\hat{W}^T_{\mathrm{mn}\emptyset}\beta(e_{in\emptyset}) + P_{\mathrm{dd}\emptyset}E_\emptyset + \rho_\emptyset\dot{e}_\emptyset + \ddot{\emptyset}_d\right],$$
$$\dot{\hat{W}}_{\mathrm{mn}\emptyset} = -D_\emptyset\beta E_\emptyset. \tag{27}$$

$$u_3 = \frac{J_y}{d}\left[-\hat{W}^T_{\mathrm{mn}\theta}\beta(e_{\mathrm{in}\theta}) + P_{\mathrm{dd}\theta}E_\theta + \rho_\theta\dot{e}_\theta + \ddot{\theta}_d)\right],$$
$$\dot{\hat{W}}_{\mathrm{mn}\theta} = -D_\theta\beta E_\theta \tag{28}$$

$$u_4 = \frac{J_z}{d}\left[-\hat{W}^T_{\mathrm{mn}\Psi}\beta(e_{\mathrm{in}\Psi}) + P_{\mathrm{dd}\theta}E_\Psi + \rho_\Psi\dot{e}_\Psi + \ddot{\Psi}_d\right],$$
$$\dot{\hat{W}}_{\mathrm{mn}\Psi} = -D_\Psi\beta E_\Psi \tag{29}$$

And for perching states, the control laws are:

$$u_{\theta p} = \frac{J_y}{d}\left[-\hat{W}^T_{\mathrm{mn}\theta p}\beta\left(e_{\mathrm{in}\theta p}\right) + P_{\mathrm{dd}\theta p}E_{\theta p} + \rho_{\theta p}\theta p + \ddot{\theta}_{\mathrm{pd}}\right],$$
$$\dot{\hat{W}}_{\mathrm{mn}\theta p} = -D_{\theta p}\beta E_{\theta p}. \tag{30}$$

$$u_{zp} = \frac{m}{\cos\theta_p}\left[-\hat{W}^T_{\mathrm{mnxp}}\beta\left(e_{\mathrm{inzp}}\right) + P_{\mathrm{ddzp}}E_{\mathrm{zp}} + \rho_{\mathrm{zp}}\dot{e}_{\mathrm{zp}} + \ddot{z}_{\mathrm{pd}}\right],$$
$$\dot{\hat{W}}_{\mathrm{mnzp}} = -D_{\mathrm{zp}}\beta E_{\mathrm{zp}} \tag{31}$$

$$u_{\mathrm{xp}} = \frac{m}{\sin\theta_p}\left[-\hat{W}^T_{\mathrm{mnxp}}\beta\left(e_{\mathrm{inxp}}\right) + P_{\mathrm{ddxp}}E_{\mathrm{xp}} + \rho_{\mathrm{xp}}\dot{e}_{\mathrm{xp}} + \ddot{x}_{\mathrm{pd}}\right],$$
$$\dot{\hat{W}}_{\mathrm{mnxp}} = -D_{\mathrm{xp}}\beta E_{\mathrm{xp}} \tag{32}$$

Hence the proposed control input laws will stabilize the quadrotor dynamics as per the Lyapunov theory in which weights are modified using derived update laws.

4 Experimental Results

Parameters for Crazyflie nano-quadrotor are taken as: length of the arm $l = 0.010$ m, mass $m = 0.040$ kg, inertial parameters in 3D are $J_x = 1.112951 * 10^{-5}$, $J_y = 1.114361 * 10^{-5}$ and $J_z = 2.162056 * 10^{-5}$ kg/m^2 respectively. The quadrotor is equipped with an optical flow sensor for measuring movements with reference to ground and ToF flight sensor (laser sensor) for measuring distance from wall. Experiments are conducted for perching task using ROS running on Linux operating system. Nano-quadrotor's start location is $[x = 0 y = 0 z = 0]$ in outdoor environment where x, y, z distances are in metres. The initial Euler angles are at $[\emptyset = 0 \theta = 0 \Psi = 0]$.

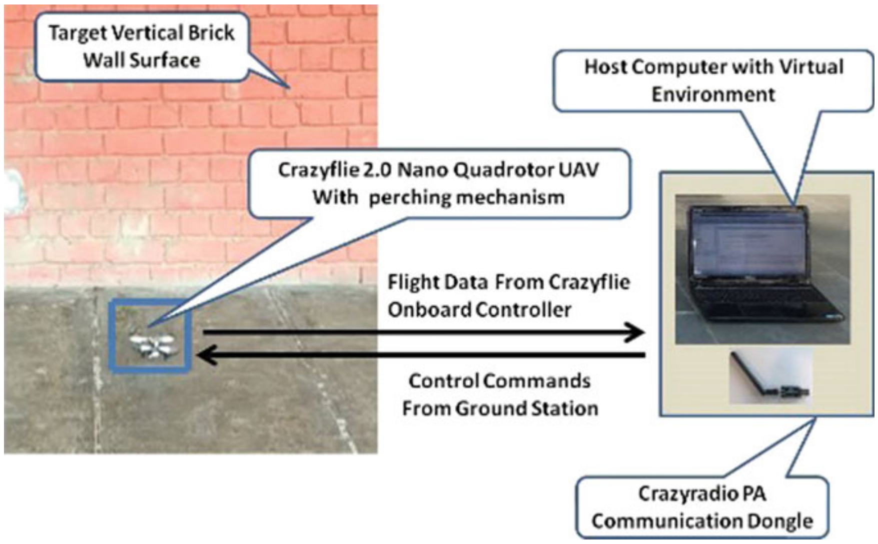

Target Vertical Brick Wall Surface
Host Computer with Virtual Environment
Crazyflie 2.0 Nano Quadrotor UAV With perching mechanism
Flight Data From Crazyflie Onboard Controller
Control Commands From Ground Station
Crazyradio PA Communication Dongle

Fig. 2 Experimental working environment

In Fig. 2, the experimental set-up shows the nano-quadrotor, one host computer for communicating with system and target vertical wall. The host computer is used to receive the necessary flight data for orientation estimation and to send few emergency commands. A communication radio module (nRF24LU1) is on-board of the quadrotor. Two test cases are performed on experimental set-up.

Case 1: For perching in fixed position, $[x = 1 y = 0.5 z = 1.5]$ is the target position and $\theta_d = 1$ radian is the desired angles for pitch. During perching, states of the quadrotor change from 3-D space to 2-D space. Perching in the fixed position of quadrotor can be used for surveillance purpose on high-rise transparent wall. Figure 3 shows the tapped images of fixed position perching. Figure 4 shows the evolution of states using event-trigger strategy.
Case 2: Perching and pose changing in fixed position. In this test, $[x = 1 y = 0.25 z = 1]$ is the desired position and $\theta_d = 1$radian is the desired angle for pitch. Perching and pose changing in the fixed position of quadrotor can be used for capturing the images from a high-rise building. Figure 5 shows the tapped images, and Fig. 6 shows the evolution of states.

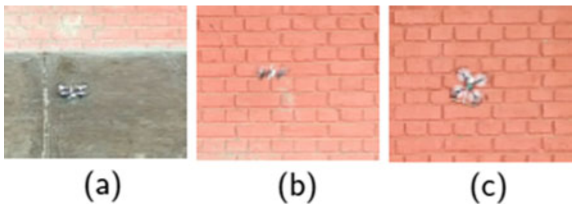


Fig. 3 Tapped images of perching in fixed position

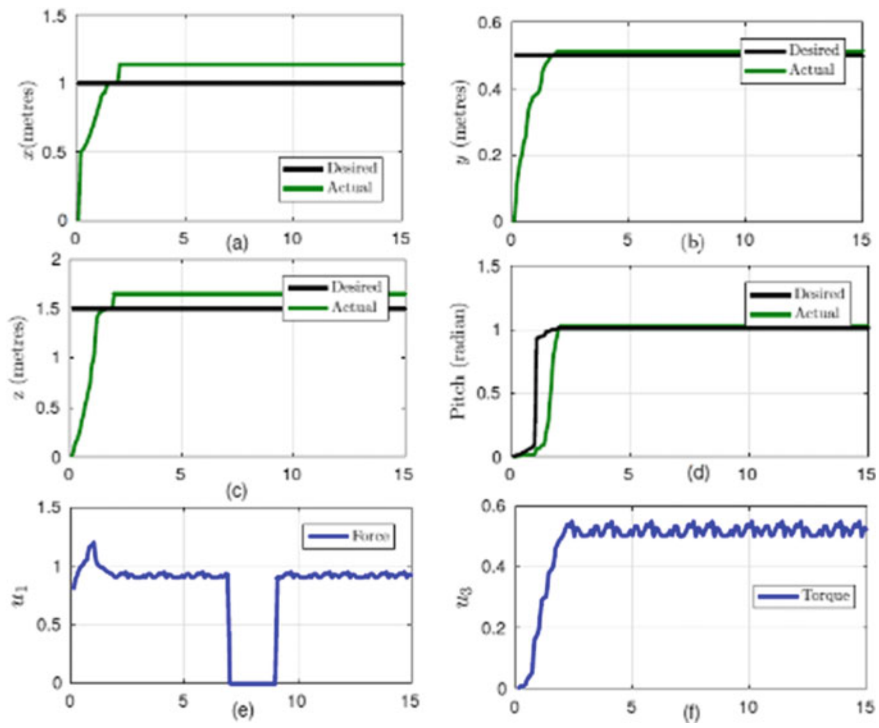
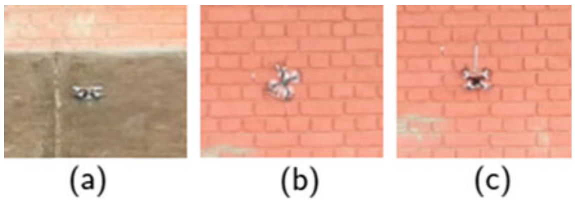


Fig. 4 Evolution of system states, **a** x position, **b** y position, **c** z position, **d** pitch, **e** position force, **f** pitch torque

Fig. 5 Tapped images of perching and pose changing in fixed position



5 Conclusion

The article presents a perching application of nano-quadrotor using direct adaptive Chebyshev neural network-based control algorithm in presence of unmodelled system dynamics. The control laws are designed using feedback linearization technique and update law for the weights of the network are derived using Lyapunov theory. Experiments are performed for two different scenarios to validate the proposed neural controller. The future scope of the present work is to deal with challenges in climbing and taking off of quadrotor for surveillance and image capturing application.

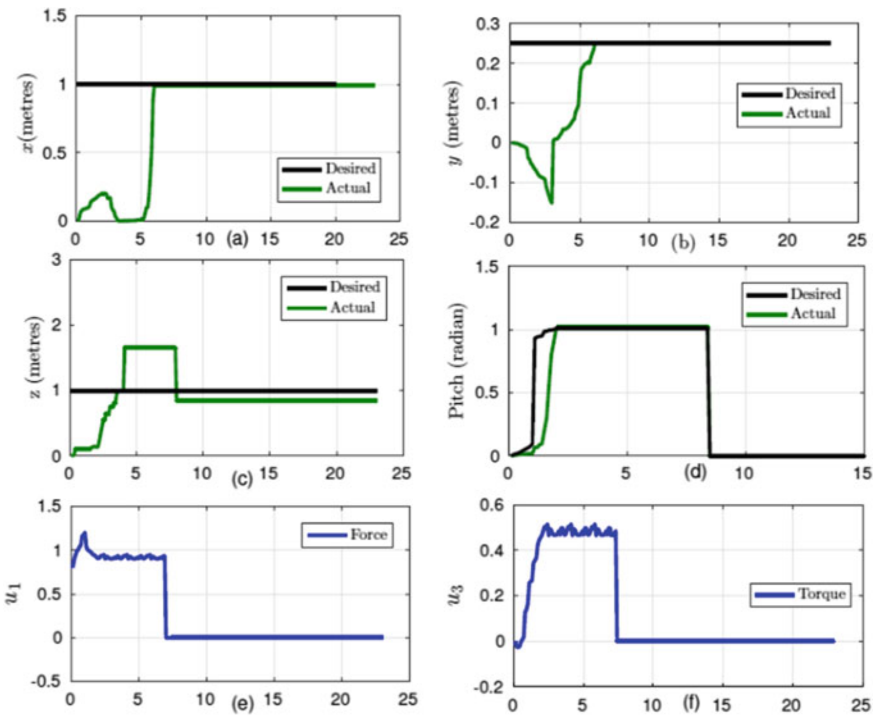


Fig. 6 Evolution of system states, **a** x position, **b** y position, **c** z position, **d** pitch, **e** position force, **f** pitch torque

References

1. Thomas J, Loianno G, Pope M, Hawkes EW, Estrada MA, Jiang H, Cutkosky MR, Kumar V (2015) Planning and control of aggressive maneuvers for perching on ınclined and vertical surfaces. In: Proceedings of the ınternational design engineering technical conferences and computers and ınformation in engineering conference. Volume 5C, 39th Mechanisms and g. ASME
2. Singh P, Gupta S, Behera L, Verma NK, Nahavandi S (2020) Perching of nano-quadrotor using self-trigger finite-time second-order continuous control. IEEE Syst J 1–11
3. Pope MT, Kimes CW, Jiang H, Hawkes EW, Student Member, IEEE, Estrada MA, Kerst CF, Roderick WRT, Han AK, Christensen DL, Cutkosky MR (2017) A multimodal robot for perching and climbing on vertical outdoor surfaces. IEEE Trans Robot 33(1):38–48
4. Mohta K, Kumar V, Daniilidis K (2014) Vision-based control of a quadrotor for perching on lines. In: IEEE ınternational conference on robotics and automation (ICRA), pp 3130–3136
5. Tayebi A, McGilvray S (2006) Attitude stabilization of a vtol quadrotor aircraft. IEEE Trans Control Syst Technol 14(3):562–571
6. Erginer B, Altug E (2007) Modeling and pd control of a quadrotor vtol vehicle. Proc IEEE Intell Vehicles Symp 894–899
7. Bouabdallah S, Siegwart R (2007) Full control of a quadrotor. Proc IEEE/RSJ Int Intell Robots Syst Conf 153–158

8. Madani T, Benallegue A (2006) Control of a quadrotor mini-helicopter via full state backstepping technique. In: Proceedings of the 45th IEEE conference on decision and control, pp 1515–1520
9. Martins L, Cardeira C, Oliveira P (2021) Feedback linearization with zero dynamics stabilization for quadrotor control. J Intell Robot Syst 101
10. Singh P, Gupta S, Behera L, Verma NK (2021) Sum of square based event-triggered control of nano-quadrotor in presence of packet dropouts. In: International conference on unmanned aircraft systems, pp 767–776
11. Tripathi VK, Behera L, Verma N (2016) Disturbance observer based backstepping controller for a quadcopter. In: Proceedings of 42nd annual conference of the IEEE ındustrial electronics society, IECON, pp 108–113
12. Dierks T, Jagannathan S (2010) Output feedback control of a quadrotor UAV using neural networks. IEEE Trans Neural Netw 21(1):50–66
13. Abdollahi T, Salehfard S, Xiong C, Ying J (2018) Simplified fuzzy-Padé controller for attitude control of quadrotor helicopters. IET Control Theory Appl 12(2):310–317
14. Yogi SC, Tripathi VK, Behera L (2021) Adaptive ıntegral sliding mode control using fully connected recurrent neural network for position and attitude control of quadrotor. IEEE Trans Neural Netw Learn Syst
15. Kar I, Behera L (2009) Direct adaptive neural control for affine nonlinear systems. Appl Soft Comput 756–764
16. Zhao B, Xian B, Zhang Y, Zhang X (2015) Nonlinear robust adaptive tracking control of a quadrotor UAV via immersion and invariance methodology. IEEE Trans Ind Electron 62(5):2891–2902
17. Sornam M, Vanitha V (2018) Application of chebyshev neural network for function approximation. Int J Compu Sci Eng 4:201–204
18. Chen F, Jiang R, Zhang K, Jiang B, Tao G (2016) Robust backstepping sliding-mode control and observer-based fault estimation for a quadrotor UAV. IEEE Trans Ind Electron 63(8):5044–5056
19. Pedro JO, Crouse AJ (2015) Direct adaptive neural control of a quadrotor unmanned aerial vehicle.: 10th Asian control conference (ASCC), pp 1–6
20. Bhat RB, Chakravety S (2004) Numerical analysis in engineering. Alpha Science Int Ltd., Oxford
21. Mall S, Chakraverty S (2017) Single layer chebyshev neural network model for solving elliptic partial differential equations. Neural Process Lett 45(3):825–840