IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 January 2024, accepted 5 March 2024, date of publication 7 March 2024, date of current version 13 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3374894

RESEARCH ARTICLE

Robust Tracking Control for Quadrotor UAV With External Disturbances and Uncertainties Using Neural Network Based MRAC

MULUKEN MENEBO MADEBO, CHALA MERGA ABDISSA, LEBSEWORK NEGASH LEMMA, AND DEREJE SHIFERAW NEGASH

School of Electrical and Computer Engineering, Addis Ababa University, Addis Ababa 1000, Ethiopia

Corresponding authors: Chala Merga Abdissa (chala.merga@aait.edu.et) and Lebsework Negash Lemma (lebsework.negash@aait.edu.et)

This work was supported by Addis Ababa Institute of Technology.

ABSTRACT In this paper, an intelligent Model Reference Adaptive Control (MRAC) based on a neural network is proposed for robust tracking control of quadrotor UAV under external disturbances and parameter variations. First, the singularity-free dynamic model of the quadrotor is developed using Newton-Quaternion formalism. Then, conventional MRAC is designed to generate training data. With the generated data, the Feed Forward Neural Network (FFNN) and Recurrent Neural Network (RNN) are trained offline to get an initial set of network parameters for position controller parameters estimation and attitude control of the quadrotor, respectively, and an online learning algorithm is developed to update those network parameters in real-time. Finally, the performance of the designed Neural network-based MRAC has been evaluated using a numerical simulation in a nominal scenario and by introducing parametric variation and external disturbances as matched and unmatched uncertainties into the system. The simulation results show that the proposed controller has a better tracking performance and disturbance rejection capability compared with the Linear Quadratic Regulator (LQR) and conventional MRAC. Furthermore, the utilized control efforts are minimal and smooth proving functional safety and economical use of the controller. Therefore, the suggested controller is feasible for real-time implementation of the quadrotor UAV.

INDEX TERMS Feed forward neural network, LQR, model reference adaptive control, newton-quaternion formalism, online learning algorithm, recurrent neural network, quadrotor UAV.

I. INTRODUCTION

Unmanned Aerial Vehicle (UAV) is a kind of recent generation of autopilot aircraft deployed for various industry, civilian, and military applications. A quadrotor is a type of UAV that is driven by four rotors with the capability of Vertical takeoff and landing and other flight maneuverings. Due to this capability, low operation cost, and maneuverability at low altitudes and dangerous environments for high-risk applications, research on quadrotors has been growing enormously from time to time and currently become a research hotspot in the aviation field [1], [2]. Civilian use of quadrotor UAV includes aerial photography, inspection missions of railways and power lines, surveillance for illegal imports and exports, road detection and tracking, fire detection and control, traffic management, border patrolling, crop monitoring and spraying, and search and rescue operation for missing persons and natural disasters [3], [4]. The aforementioned applications attract control researchers to design robust control algorithms to achieve the assigned tasks precisely. Therefore, this research study follows the model-based high-performance closed-loop flight controller design for autopilot quadrotor UAV.

The Flight Control System (FCS) is the brain of a quadrotor UAV that is responsible for achieving the desired attitude and position while the quadrotor flies. Quadrotor UAV control researchers face many challenges in developing high-performance controllers for the FCS due to time-varying

The associate editor coordinating the review of this manuscript and approving it for publication was Qichun Zhang.

properties of the environment, nonlinear coupled dynamics of the system, uncertainties in modeling, and external disturbances from the wind [5]. These make the dynamics more challenging for control applications. To deal with such dynamics, adaptive control has been a promising technique that can improve the performance of control systems in the presence of uncertainty due to model degradation and uncertainty from the operating environment [6]. The basic working principle of adaptive control laws is the Certainty Equivalence Principle (CEP), which is a principle that most current adaptive controllers are working on. Based on this principle, the parameters of the adaptive controller are derived from a set of uncertain plant parameters whose true values are unknown. According to CEP, the controller parameters are replaced by estimates and taken as the true parameters when the control laws are manipulated. However, this parameter estimation requires sufficiently fast parameter convergence for the adaptive controller to achieve the desired performance specification [7], [8].

The adaptive control technique is an appropriate solution for dealing with parametric uncertainties, and it has better performance than robust control in dealing with uncertainties in constant or slow-varying parameters. However, robust control is advantageous in dealing with disturbances, quickly varying parameters, and unmodeled dynamics [9]. Here, the goal of robust control design is to maintain system performance with acceptable error within a certain range of model variations and inaccuracies, so it's well suited for applications where stability and reliability of the system are top priorities, where process dynamics are known and variation ranges for uncertainties can be estimated. Since robust control system design requires such a high level of expertise that can exactly estimate variation ranges for uncertainties, it has never been a popular solution for control problems in several industries [10]. For this reason, adaptive control gets more attention than robust control in industries among those, NASA has been performing a series of adaptive flight control studies since 70's [11].

Model Reference Adaptive Control (MRAC) is the type of adaptive control that aims to force the dynamic response of the system to asymptotically approach that of a desired reference model despite parametric uncertainties in the system. Nowadays, control engineers classify MRAC into direct and indirect MRAC [12]. In the case of direct MRAC, controller parameters are directly updated online without the need to estimate plant parameters [8]. In this control topology, the purpose of MRAC is to make plant output behave like a reference model in the way that the adaptation rule works to estimate uncertain plant parameters whereas the controller works to achieve acceptable convergence to a desired output [13]. On the other hand, the indirect MRAC technique involves applying some system identification method to obtain a model of a system and its environment from input-output experiments, and the controller is redesigned online based on the estimated model. Self-tuning regulator (STR) is an example of indirect MRAC. In this control topology, uncertain parameters of the plant model are estimated first, and then based on the estimated model controller parameters are computed separately [7].

The control systems that combine AI techniques such as neural networks, fuzzy logic, evolutionary competition, and Bayesian probability; with conventional adaptive control systems are classified under intelligent control systems due to their knowledge-based decision-making capabilities. From stated AI techniques, Artificial Neural Network (ANN) is considered as a complex adaptive system that can change its internal structure based on the information passing through it [14]. With these features, ANN is a widely used effective method to solve estimation and control problems, and its application areas in aviation fields have also increased [15]. Therefore, in this study intelligent flight controller is designed by combining the MRAC technique with the learning power of ANN, which are two equivalent topics under different umbrellas of research such as control engineering and machine learning, respectively [16]. Such systems can achieve sustained behavior in the presence of uncertainty in system dynamics, unpredictable environmental changes, unreliable sensor information, and actuator malfunction. Furthermore, can perform complex tasks as compared to conventional control techniques.

A. RELATED WORKS

Due to the significant increase in quadrotor UAV application areas, various control researchers have published studies on quadrotor UAV control. The Proportional Integral Derivative (PID) controller is the aviation industry's most widely used control algorithm. It is a linear controller with the advantages of being easy to tune and alter parameter gains, simple to construct, and strong resilience [17]. Authors in reference [18] by linearizing the quadrotor model and decoupling the state, designed a PID controller to stabilize and control the quadrotor. The authors demonstrated that the PID controller is effective only within the range of $\pm 10^0$ for roll and pitch angles, and the controller performance is highly affected by external disturbances. Using a PID controller for a quadrotor has the following problems: it's suitable only for linear systems, it loses control performance for wider angles of roll and pitch rotations, which limits quadrotor maneuverability near the specified range, and it can't tolerate external disturbances.

Authors in reference [19] did a comparative study between LQR and PID controller's performance on the quadrotor system. First, the authors drive a linear mathematical model of system dynamics and then simulate the two designed control techniques under the same initial condition. The authors verified that the LQR controller is more stable and robust than the PID controller. However, the LQR control mechanism has the following two drawbacks. First, it is

sensor-intensive. Second, LQR controllers can not handle unmodeled dynamics, which is the most common problem of implementing linear controllers in nonlinear systems. On the other hand, authors in [12] and [20] proposed a nonlinear robust MRAC approach for matched uncertainty in quadrotor UAV dynamics. The authors proved that this control approach loses its robustness for the unmatched uncertainty. Other solutions are proposed to solve the nonlinearity of the quadrotor dynamics and the effect of matched uncertainties.

Intelligent control techniques have been emerged as promising control techniques in tolerating the harmful effects of unmatched uncertainty and complexity of the system dynamics. Authors in reference [21] did a comparative analysis between neural network-based direct MRAC and conventional MRAC, and the author proved that neural network-based direct MRAC has a smaller rising time, steady-state error, and settling time than the conventional MRAC approach. Authors in [22] also developed extended minimum resource allocating network (EMRAN) based MRAC for UAV control and compared controller performance with existing PID controller. The results prove that the controller has better performance in the presence of disturbances compared to the existing one. However, this study is only focused on offline trained neural networks that estimate the parameters of the controller but outdoor quadrotor applications require adaptive control, which can adapt its parameters to changing flight conditions of quadrotor dynamics. In this phenomenon, the idea of online training that continuously tunes network parameters with changing flight dynamics of the system with environmental conditions is crucial for quadrotor control.

Authors in reference [23] designed a neural network-based intelligent adaptive control for nonlinear systems. The authors developed an online trained neural network structure along with a conventional MRAC by obtaining training patterns for the network training from PI control. The authors found that the online tuned network compensates for the nonlinearity of the system that is not considered in conventional MRAC, and the proposed controller significantly improves the system performance by minimizing reference model tracking error, and it shows superior performance than PI-based MRAC and pure MRAC approaches. However, the nonlinearity of adaptation laws in conventional MRAC causes difficulty in tuning PI controller gains, so the author obtained PI controller gains manually by try and error method, and when the controller gains are large the generated training patterns are mostly corrupted by noise and the network unable to fit the patterns. These make the network training process tricky. Generally, in the literature survey, the recently proposed methods have been focused on offline-trained networks, and noisy sensitive data used to train the networks, which motivates us to develop a novel training and control approach for the quadrotor.

B. MAIN CONTRIBUTIONS

In this study, the intelligent flight control technique, which combines ANN from the AI technique and MRAC from the conventional control technique, is developed for quadrotor control. In this approach, ANN is employed to learn and approximate the system dynamics meanwhile MRAC adjusts its parameters based on the error between the actual system output and the desired reference model output. The training pattern for the network is obtained from input-output patterns of the adaptation law of conventional MRAC, and the generalized Back Propagation Algorithm (BPA) is developed to update network parameters in real-time enabling the controller to adapt its parameters to changing flight conditions of the quadrotor dynamics. Furthermore, the proposed control technique is effective in tolerating matched and unmatched uncertainties, requires less processing time, and can perform its control action in all quadrotor maneuvering ranges. This implies that the proposed control technique is an effective control approach for real-time implementation on the quadrotor platform. Therefore, the objective of this study is to develop a high-performance ANN-based MRAC algorithm for quadrotor control with the following major contributions:

1) A novel neural network-based MRAC architecture for robust tracking control based on quaternion modeling of quadrotor UAV has been proposed,
2) To increase the performance of the designed controller, a more precise quaternion-based singularity-free quadrotor dynamic model is developed by applying Newton-Quaternion formalism and considering real-time scenarios of quadrotor maneuvering,
3) The proposed technique has been designed and tested under external disturbances and parameters uncertainties using simulation and proved to have a better performance than other similar controllers that are used for UAV control, and
4) An online learning algorithm has been introduced to enhance controller performance. Artificial neural networks are used to update weights and biases in real- time which in turn increases convergence rate and ensures the stability of the system.

The rest of the paper is organized as follows. In section II, the detailed quaternion-based mathematical model of quadrotor flight dynamics is presented. In section III, ANN-based MRAC design is described. In section IV, simulation results are presented. Finally, concluding remarks and possible further works are presented in section V.

II. QUATERNION-BASED MATHEMATICAL MODEL OF QUADROTOR UAV

A. PRELIMINARY NOTIONS OF QUADROTOR UAV

To model quadrotor dynamics, the coordinate frames of the quadrotor must be defined first. There are two coordinate frames, as shown in Figure 1, that the quadrotor will operate in: inertial frame (x_i, y_i, z_i) and body frame

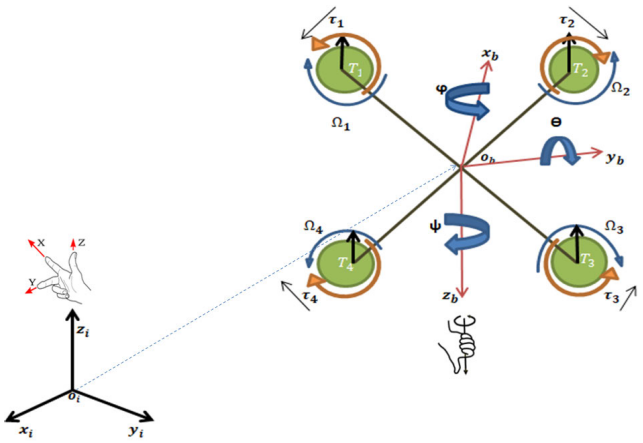

FIGURE 1. Quadrotor free body diagram and its reference frames.

(x_b, y_b, z_b) [24]. Inertial Frame (IF) is defined with respect to the ground with a positive z-axis pointing in the opposite direction of gravity, whereas Body Frame (BF) is defined with respect to the Center of Gravity (CoG) of the quadrotor with its axes fixed to the body. The necessity of defining those reference frames is because of

1) aerodynamics forces and torques are manipulated with respect to BF,
2) on-board builtin sensors measure quantities with their respective reference frame, for instance, Inertial Measurement Unit (IMU) gives readings with respect to BF whereas GPS gives readings with respect to IF, and
3) Newton laws are valid in IF only.

Therefore, projection matrices are required to transform quantities between the two frames and derive a system governing equations according to a single reference frame [25].

Coordinate frames transformation is used to transform quantities between the two reference frames. To avoid the nonlinearity, geometric singularity, and computational cost usually associated with the Euler angles [26], [27], the quaternion approach is applied for coordinate frames transformation. A quaternion is a hyper-complex number with four dimensions that include a scalar part q_o, and a three-dimensional vector part $\vec{q} = [q_i, q_j, q_k]$ with orthonormal base $[i, j, k]$ [28]. Thus, the quaternion can be represented as follows:

$$q = q_o + q_1\hat{i} + q_2\hat{j} + q_3\hat{k} = [q_o, q_1, q_2, q_3] = (q_o, \vec{q}) \quad (1)$$

To represent a valid 3D orientation or to compute valid vector rotations, the quaternion must be normalized. The normalized quaternion is ideally suitable for representing and transforming the orientation of coordinates between any two reference frames. Quaternion normalization can be done as follows:

$$q = \frac{q_o + q_1\hat{i} + q_2\hat{j} + q_3\hat{k}}{\sqrt{{q_o}^2 + {q_1}^2 + {q_2}^2 + {q_3}^2}} \quad (2)$$

A unit quaternion provides a convenient mathematical notation for representing orientations in 3D space. Hence, to derive the quadrotor dynamic model, the unit quaternion notion is applied. The product between two quaternion $q_a = (q_{oa}, \vec{q_a})$ and $q_b = (q_{ob}, \vec{q_b})$, is manipulated as follows:

$$q_a \otimes q_b = (q_{oa}q_{ob} - \vec{q_a}.\vec{q_b}, q_{oa}\vec{q_b} + q_{ob}\vec{q_a} + \vec{q_a} \times \vec{q_b}) \quad (3)$$

Coordinate transformation between two different reference frames by using quaternion rotation is computed as [29]

$$X = q \otimes X' \otimes q^{-1} \quad (4)$$

where $\mathbf{X}$ is rotated vector and X' is vector to be rotated. Equation 4 implies that multiplying two or more rotation quaternions produces another rotation quaternion that represents the total rotation obtained by performing each individual rotation for each quaternion. Vectors can be transformed from one axis system to another if first transformed into quaternions with a zero scalar value, also this is known as a pure quaternion. Thus, a quaternion-based rotation matrix that transforms quantities from IF to the BF is obtained by rewriting equation 4 as follows:

$$\begin{bmatrix} 0 \\ r_b \end{bmatrix} = q \otimes \begin{bmatrix} 0 \\ r_i \end{bmatrix} \otimes q^* \quad (5)$$

where $r_b = [x_b, y_b, z_b]'$ and $r_i = [x_i, y_i, z_i]'$ are position quantities with respect to BF and IF, respectively. By using the quaternion product in equation 3, a 3D rotation matrix $R_I^B(q)$ [30] that projects position quantities from BF into IF can be obtained by rearranging equation 5 as follows:

$$\begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} = \begin{bmatrix} 2({q_o}^2 + {q_1}^2) - 1 & 2(q_1q_2 + q_oq_3) & 2(q_1q_3 - q_oq_2) \\ 2(q_1q_2 - q_oq_3) & 2({q_o}^2 + {q_2}^2) - 1 & 2(q_2q_3 + q_oq_1) \\ 2(q_oq_2 + q_1q_3) & 2(q_2q_3 - q_oq_1) & 2({q_o}^2 + {q_3}^2) - 1 \end{bmatrix} \begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} \quad (6)$$

Transformation matrix that projects BF rates $\omega = [p, q, r]'$ into IF quaternion rates $\dot{q}$ can be obtained as [31], [32]:

$$\begin{bmatrix} \dot{q}_o \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2} q \otimes \omega = \frac{1}{2} \begin{bmatrix} q_o & -q_1 & -q_2 & -q_3 \\ q_1 & q_o & -q_3 & q_2 \\ q_2 & q_3 & q_o & -q_1 \\ q_3 & -q_2 & q_1 & q_o \end{bmatrix} \begin{bmatrix} 0 \\ p \\ q \\ r \end{bmatrix} \quad (7)$$

B. QUADROTOR FLIGHT DYNAMICS MODELING

Almost all systems in real life are nonlinear in nature, so control researchers design model-based either linear controllers by linearizing those nonlinear systems at some operating point, or design nonlinear controllers based on a simplified nonlinear model of the system by ignoring some features that are assumed to have less effect on the system. During simulation, the designed controller may show

promising results by effectively driving tracking error toward zero. However, during the implementation of the developed control algorithm in the real system, the error may not converge to zero. This is due to the designed control law, which is designed based on a linearized model not based on the real plant. So, unmodeled dynamics, which are neglected during modeling, may result in poor controller performance and may also cause the system unstable. To solve such harmful effects of unmodeled dynamics, in this section more realistic behavior of the quadrotor is enforced by considering every possible effect which has a significant change on the quadrotor dynamics. This point is raised from the general fact of control engineering: when the accuracy of the model improves, the controller performance will also improves though this increases the complexity of the model. Taking this general fact into consideration, the following assumptions have been made to model the quadrotor flight dynamics [33], [34]:

1) structure of the quadrotor is rigid with constant mass and uniform symmetry,
2) gravitational force doesn't change with altitude,
3) the resistance and inductance of Brushless Direct Current (BLDC) electric motor are negligible, and
4) at the hovering position, velocity of air and propellers are equal.

Parameter values of the system dynamics are presented in Table.1.

TABLE 1. Specifications for the quadrotor model design [35], [36].

Symbol	Description	Values
g	Gravitational acceleration	9.81 m/s^2
m	Mass of the quadrotor	0.468 Kg
I_{xx}	Body moment of inertia around the x-axis	4.856×10^{-3} N·m· s^2
I_{yy}	Body moment of inertia around the y-axis	4.856×10^{-3} N·m· s^2
I_{zz}	Body moment of inertia around the z-axis	8.801×10^{-3} N·m· s^2
ℓ	Length of the quadrotor arm	0.225 m
b	Thrust coefficient	2.98×10^{-3} N/m/s
d	Drag factor	1.140×10^{-7} N·m·s^2
J_p	Rotor or propeller inertia	3.357×10^{5} N·m·s^2
$C_{a_p} = C_{a_q} = C_{a_r}$	Rotational drag coefficients	0.25 N/m/s
$C_{a_x} = C_{a_y} = C_{a_z}$	Translational drag coefficients	0.2 N/m/s
R	Rotor or propeller Radius	0.25 m
γ	Correction factor	0.083 unit

1) QUADROTOR ROTATIONAL MOTION

Using Newton's 2^{nd} law of rotational motion for fully actuated quadrotor rotational dynamics, the net moment derived from BF as follows:

$$\sum \tau_i = J\alpha = [U_2, U_3, U_4]' - M_{gyro} - M_{aero} \tag{8}$$

$$\dot{\omega} = J^{-1}\left[[U_2, U_3, U_4]' - \omega \times (J\omega) - M_{gyro} - M_{aero}\right] \tag{9}$$

where J is inertia matrix, $\alpha = \dot{\omega} = [\dot{p}, \dot{q}, \dot{r}]'$ are angular accelerations in BF; U_2, U_3, and U_4 are moments on BF which are used to control roll, pitch, and yaw, respectively; $M_{gyro} = J_p\Omega \times \omega$ is a gyroscopic moment that the spinning propellers induce when the quadrotor rotates in the space [37]; $M_{aero} = C_a\omega$ is aerodynamic frictional moment [38]; C_a is drag coefficient; and Ω is net propeller angular speed. By rearranging equation 9, rotational dynamics of the quadrotor is obtained as

$$\begin{cases} \dot{p} = \dfrac{1}{J_{xx}}U_2 + \dfrac{J_{yy} - J_{zz}}{J_{xx}}qr - \dfrac{J_p q}{J_{xx}}\Omega_r - \dfrac{C_{a_p}}{J_{xx}}p \\ \dot{q} = \dfrac{1}{J_{yy}}U_3 + \dfrac{J_{zz} - J_{xx}}{J_{yy}}pr + \dfrac{J_p p}{J_{yy}}\Omega_r - \dfrac{C_{a_q}}{J_{yy}}q \\ \dot{r} = \dfrac{1}{J_{zz}}U_4 + \dfrac{J_{xx} - J_{yy}}{J_{zz}}pq - \dfrac{C_{a_r}}{J_{zz}}rw \end{cases} \tag{10}$$

2) QUADROTOR TRANSLATIONAL MOTION

The translational dynamics of the quadrotor is derived based on Newton's 2^{nd} law of translational motion. Since Newton's laws are valid only in IF, the rotation matrix R_I^B is used to project the thrust force that is generated in BF onto IF. Thus, the translational dynamics of the quadrotor in IF is derived as

$$\sum F_i = ma = R_I^B(q)U_1 - F_g - F_d \tag{11}$$

$$a = \frac{1}{m}[R_I^B(q)U_1 - F_g - F_d] \tag{12}$$

where U_1 is the total thrust force used to control the altitude, $a = [\ddot{x}, \ddot{y}, \ddot{z}]'$ is the linear acceleration of quadrotor in IF, $F_g = mg$ is a gravitational force along the z direction, $F_d = C_d[\dot{x}, \dot{y}, \dot{z}]'$ [39] is air drag or friction force in IF which is proportional to linear velocity, C_d is drag coefficient, g is gravity, and m is mass of quadrotor. Since the quadrotor translation is the underactuated system, only the altitude of the quadrotor is controlled directly, and any non-zero roll or pitch angle results in the quadrotor maneuvering along the x or y direction. Therefore, equation 12 can be rewritten as

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \frac{1}{m}\left[R_I^B(q) \begin{bmatrix} 0 \\ 0 \\ U_1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix} - \begin{bmatrix} C_{d_x}\dot{x} \\ C_{d_y}\dot{y} \\ C_{d_z}\dot{z} \end{bmatrix} \right] \tag{13}$$

$$\begin{cases} \ddot{x} = 2(q_1q_3 - q_oq_2)\dfrac{U_1}{m} - \dfrac{C_{d_x}}{m}\dot{x} \\ \ddot{y} = 2(q_2q_3 + q_oq_1)\dfrac{U_1}{m} - \dfrac{C_{d_y}}{m}\dot{y} \\ \ddot{z} = \left[2({q_o}^2 + {q_3}^2) - 1\right]\dfrac{U_1}{m} - g - \dfrac{C_{d_z}}{m}\dot{z} \end{cases} \tag{14}$$

3) AERODYNAMIC GROUND EFFECT

Ground Effect (GE) is an aerodynamic phenomenon that the ground pushes the quadrotor up when the propeller

approaches the ground surface [40]. For the same transmitted power to the actuator, the propeller generates more thrust caused only by the presence of the ground as shown in Figure 2.

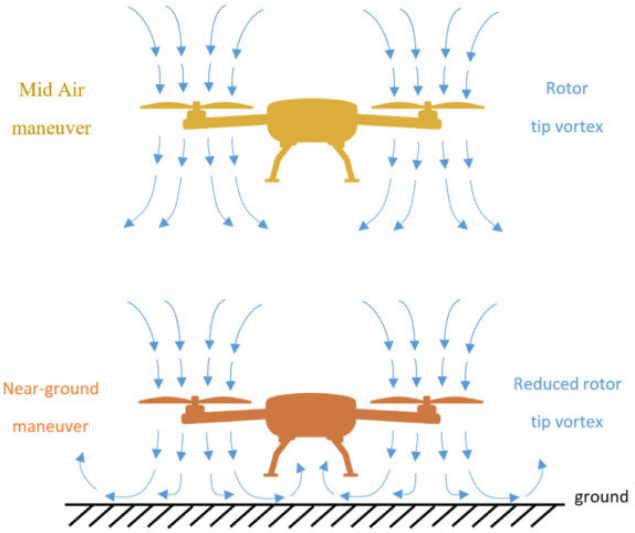

FIGURE 2. Variation of thrust generated at Mid Air maneuvering and Near-ground maneuvering due to GE in the quadrotor [33].

Quadrotor has a stronger GE up to the altitude of $z = 6R$, so GE is considered for the altitude up to $z = 6R$. Hence, the region where $z \geq 0.5R$ and $z \leq 6R$ is considered as In Ground Effect (IGE), and the region where $z < 0.5R$ and $z > 6R$ is considered as Out of Ground Effect (OGE) where GE is typically doesn't exist. Experiments show that at constant power, GE is mathematically expressed as [27], [35]

$$f_{GE}(z) = \begin{cases} \dfrac{1}{1-\gamma(\frac{R}{4z})^2} & 0.5R \leq z \leq 6R \\ 1 & otherwise \end{cases} \tag{15}$$

where $f_{GE}(z)$ is the ground effect factor which is decreased as altitude z increases. γ is the correction coefficient, which is estimated from the experiment, and it's used to adjust unpredictable airflow influence between the rotors of the quadrotor, and R is the radius of the propellers. To account the effect of GE, the thrust force model is modified [27] as

$$T_i = b\Omega^2 f_{GE}(z) \tag{16}$$

The rate multiplication of the Coriolis term which is the product of quaternion rates is very small and has insignificant effects in the quadrotor governing equation. In addition to this, the net residues speed Ω_r in equation 10, has a nonzero value for yaw rotation but yaw rotation doesn't have a gyroscopic effect. Therefore, both Coriolis term and net residues speed Ω_r are ignored to simplify the rotational dynamics of the quadrotor. Hence, by combining equations 10 and 14, and considering the GE, the overall quadrotor flight dynamics is summarized as

$$\begin{cases} \ddot{x} = 2(q_1q_3 - q_oq_2)\dfrac{U_1}{m}f_{GE} - \dfrac{C_{d_x}}{m}\dot{x} & (17a) \\ \ddot{y} = 2(q_2q_3 + q_oq_1)\dfrac{U_1}{m}f_{GE} - \dfrac{C_{d_y}}{m}\dot{y} & (17b) \\ \ddot{z} = \left[2({q_o}^2 + {q_3}^2) - 1\right]\dfrac{U_1}{m}f_{GE} - g - \dfrac{C_{d_z}}{m}\dot{z} & (17c) \\ \dot{p} = \dfrac{1}{J_{xx}}U_2 f_{GE}(z) - \dfrac{C_{a_p}}{J_{xx}}p & (17d) \\ \dot{q} = \dfrac{1}{J_{yy}}U_3 f_{GE}(z) - \dfrac{C_{a_q}}{J_{yy}}q & (17e) \\ \dot{r} = \dfrac{1}{J_{zz}}U_4 f_{GE}(z) - \dfrac{C_{a_r}}{J_{zz}}r & (17f) \\ \dot{q_o} = -\dfrac{1}{2}(pq_1 + qq_2 + rq_3) & (17g) \\ \dot{q_1} = \dfrac{1}{2}(pq_o + rq_2 - qq_3) & (17h) \\ \dot{q_2} = \dfrac{1}{2}(qq_o + pq_3 - rq_1) & (17i) \\ \dot{q_3} = \dfrac{1}{2}(rq_o + qq_1 - pq_2) & (17j) \end{cases}$$

Based on the free body diagram of the quadrotor in Figure 1 and the relation between propeller angular speed Ω and induced torque τ, control allocation equations are obtained as follows

$$\begin{cases} \Omega_1 = \sqrt{\dfrac{1}{f_{GE}(z)}\left(\dfrac{U_1}{4b} + \dfrac{U_2}{4b\ell} + \dfrac{U_3}{4b\ell} - \dfrac{U_4}{4d}\right)} \\ \Omega_2 = \sqrt{\dfrac{1}{f_{GE}(z)}\left(\dfrac{U_1}{4b} - \dfrac{U_2}{4b\ell} + \dfrac{U_3}{4b\ell} + \dfrac{U_4}{4d}\right)} \\ \Omega_3 = \sqrt{\dfrac{1}{f_{GE}(z)}\left(\dfrac{U_1}{4b} - \dfrac{U_2}{4b\ell} - \dfrac{U_3}{4b\ell} - \dfrac{U_4}{4d}\right)} \\ \Omega_4 = \sqrt{\dfrac{1}{f_{GE}(z)}\left(\dfrac{U_1}{4b} + \dfrac{U_2}{4b\ell} - \dfrac{U_3}{4b\ell} + \dfrac{U_4}{4d}\right)} \end{cases} \tag{18}$$

and the overall residual or net propeller angular speed Ω_r is obtained as

$$\Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \tag{19}$$

Conversion from quaternion to Euler angle can be performed by using a canonical set of Euler angles as [41]

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \arctan 2(2(q_oq_1 + q_2q_3), {q_o}^2 - {q_1}^2 - {q_2}^2 + {q_3}^2) \\ \arcsin(2(q_oq_2 - q_3q_1)) \\ \arctan 2(2(q_oq_3 + q_1q_2), {q_o}^2 + {q_1}^2 - {q_2}^2 - {q_3}^2) \end{bmatrix} \tag{20}$$

III. INTELLIGENT FLIGHT CONTROL DESIGN OF QUADROTOR

A. QUADROTOR CONTROLLER DESIGN STRATEGY

Due to the underactuation of the quadrotor translational dynamics, there is no way to directly control the lateral motion of the quadrotor. Therefore, a nested control approach has to be deployed in which desired attitude commands are generated by the outer position control loop based on the targeted response, and the inner attitude control loop stabilizes and controls the attitudes of the quadrotor based on the generated desired command. To ensure that every quadrotor output follows a specified trajectory, virtual control inputs are introduced to the system [42]. The physical interpretation of this control law is that control of the translation motion of the quadrotor depends on desired quaternion rotation q_d and total thrust force U_1. In the case of no external disturbance acts on the system, the virtual control inputs can be obtained as

$$U = [U_x, U_y, U_z]' = R_I^B(q_d)U_1 f_{GE} - mg \tag{21}$$

$$\begin{cases} U_x = 2(q_{1_d}q_{3_d} - q_{o_d}q_{2_d})U_1 f_{GE} & \text{(22a)} \\ U_y = 2(q_{2_d}q_{3_d} + q_{o_d}q_{1_d})U_1 f_{GE} & \text{(22b)} \\ U_z = \left[2({q_{o_d}}^2 + {q_{3_d}}^2) - 1\right] U_1 f_{GE} - mg & \text{(22c)} \end{cases}$$

where U_x, U_y and U_z are virtual position control inputs along x, y and z directions, respectively. This method of controlling a nonlinear system as if it is linear is called Nonlinear Dynamic Inversion (NDI). NDI is used to linearize the system, decouple the controlled variables, and separate the main model from the dynamic inversion model to compute a solution in closed form. By substituting equation 22(a, b & c) into equation 17(a,b&,c), the translational quadrotor dynamics can be rewritten in terms of virtual control inputs as follows:

$$\begin{cases} \ddot{x} = \dfrac{U_x}{m} - \dfrac{C_{d_x}}{m}\dot{x} \\ \ddot{y} = \dfrac{U_y}{m} - \dfrac{C_{d_y}}{m}\dot{y} \\ \ddot{z} = \dfrac{U_z}{m} - \dfrac{C_{d_z}}{m}\dot{z} \end{cases} \tag{23}$$

To design a controller for 2^{nd} order system, a small quaternion approach is implemented to rewrite quadrotor rotational dynamics in quaternion. This assumption is true for stabilizing the quadrotor near to hovering state. Since the unit quaternion approach is utilized to represent the orientation of the quadrotor, the following conditions are always true for near hovering state maneuverings such as $q_o \approx 1$, and $q_1 = q_2 = q_3 \approx 0$. In this scenario, the relationship between BF rates and quaternion rates, which is expressed in equation 7, becomes

$$\begin{bmatrix} \dot{q}_o \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ p \\ q \\ r \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 0 \\ p \\ q \\ r \end{bmatrix} \tag{24}$$

Hence,

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = 2\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} \tag{25}$$

and then taking time derivative on both sides result in

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = 2\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \end{bmatrix} \tag{26}$$

Then, substitute equations 25 and 26 into equation 17(d, e & f). Finally, 2^{nd} order rotational dynamics of the quadrotor in terms of a quaternion can be obtained as

$$\begin{cases} \ddot{q}_1 = \dfrac{1}{J_{xx}}\left(\dfrac{1}{2}U_2 - C_{a_p}\dot{q}_1\right) \\ \ddot{q}_2 = \dfrac{1}{J_{yy}}\left(\dfrac{1}{2}U_3 - C_{a_q}\dot{q}_2\right) \\ \ddot{q}_3 = \dfrac{1}{J_{zz}}\left(\dfrac{1}{2}U_4 - C_{a_r}\dot{q}_3\right) \end{cases} \tag{27}$$

In the suggested control strategy, the attitude controller takes control action based on the deviation of the actual quaternion from the desired quaternion. Even though the attitude controller is designed to track only the vector part of quaternion, from the property of unit quaternion, the actual scalar part of quaternion q_o will be converged to its desired q_{o_d} by the following relation:

$$q_o = \sqrt{1 - \left({q_1}^2 + {q_2}^2 + {q_3}^2\right)} \tag{28}$$

By assigning those virtual control inputs, now quadrotor dynamics become fully actuated since there are six control inputs such as U_x, U_y, U_z, U_2, U_3, and U_4 for controlling six degrees of freedom quadrotor maneuvering along $x, y, z, q_1, q_2, \&q_3$ direction, respectively. Therefore, each trajectory of the quadrotor can be controlled separately by corresponding control input.

To design flight controller, state variables can be defined as follows, $x = [x, \dot{x}, y, \dot{y}, z, \dot{z}, q_1, \dot{q}_1, q_2, \dot{q}_2, q_3, \dot{q}_3] = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, x_{11}, x_{12}]'$ where x, y, and z are linear position and $\dot{x}$, $\dot{y}$ and $\dot{z}$ are speeds along x, y, and z axis, respectively; whereas q_1, q_2, and q_3 are orientation in quaternion and $\dot{q}_1$, $\dot{q}_2$, and $\dot{q}_3$ are quaternion rates along roll, pitch, and yaw direction, respectively. Thus, this state vector has two parts namely the position control represented by the vector $[x, \dot{x}, y, \dot{y}, z, \dot{z}]$, and orientation control represented by vector $[q_1, \dot{q}_1, q_2, \dot{q}_2, q_3, \dot{q}_3]$. By rewriting equation 23 and 27 in state space equation form, the final

quadrotor dynamics model, which is defined from fixed IF, is obtained as

$$\dot{x} = f(x, U) = f(x) + g(x)U \tag{29}$$

$$\dot{x} = \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \frac{1}{m}\left(U_x - C_{d_x}x_2\right) \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \frac{1}{m}\left(U_y - C_{d_y}x_4\right) \\ \dot{x}_5 = x_6 \\ \dot{x}_6 = \frac{1}{m}\left(U_z - C_{d_z}x_6\right) \\ \dot{x}_7 = x_8 \\ \dot{x}_8 = a_1U_2 - d_1x_8 \\ \dot{x}_9 = x_{10} \\ \dot{x}_{10} = a_2U_3 - d_2x_{10} \\ \dot{x}_{11} = x_{12} \\ \dot{x}_{12} = a_3U_4 - d_3x_{12} \end{cases} \tag{30}$$

where

- $a_1 = \frac{1}{2J_{xx}}$, and $d_1 = \frac{C_{ap}}{J_{xx}}$,
- $a_2 = \frac{1}{2J_{yy}}$, and $d_2 = \frac{C_{aq}}{J_{yy}}$,
- $a_3 = \frac{1}{2J_{yy}}$, and $d_3 = \frac{C_{ar}}{J_{zz}}$.

Based on this setup, the novel quadrotor UAV control approach that this research study contributed is shown in Figure 3.

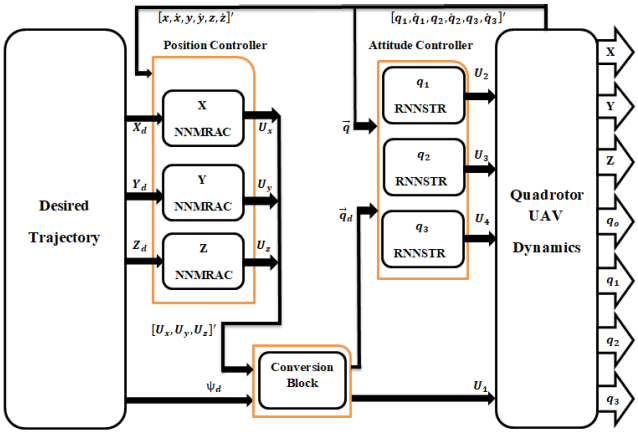

FIGURE 3. Hierarchical hybrid control architecture of the quadrotor.

To tackle the aforementioned problem of the absence of direct actuation control for the position of the quadrotor, the linearized model is established in equation 23 by using the NDI method. Thus, Feedforward ANN-based MRAC (NNMRAC) is designed for position control of the quadrotor as a control architecture in Figure 4. To obtain input-output data for ANN training, conventional MRAC for linearized systems is designed, first. Then, by exporting input-output data of adaptation law of MRAC system to MATLAB® workspace, FFNN is trained to estimate position controller parameters.

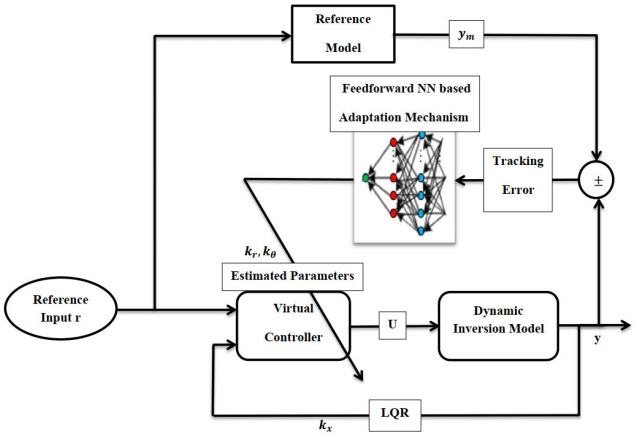

FIGURE 4. NN-based MRAC of quadrotor position control architecture.

On the other hand, the attitude controller has one to one relationship between control inputs and corresponding outputs i.e., there are direct actuation controls for the attitude that drive the quadrotor in the desired direction. Hence, RNN-based STR (RNNSTR) is designed to control the attitude of the quadrotor as shown in Figure 5. In this control architecture, the RNN plant model acts as a Black-box identifier which is trained first to generate one step ahead prediction of plant output, and the process is known as system identification. Then, by inserting trained plant model network parameters into the appropriate locations in the overall control architecture, the controller network will be trained. Here, the controller network is trained to generate control action that will drive estimated plant output toward reference input by following the desired reference model response [43]. When the controller network is trained, the plant model network is considered as a fixed network. To stabilize the system, control input must be one step delayed to separate the two processes since system identification must be accomplished before the controller takes control action without overlapping.

Since both direct and indirect adaptive control approaches are implemented to design closed-loop controllers for the quadrotor, the overall proposed quadrotor control architecture becomes a hybrid control system as shown in Figure 3.

B. QUADROTOR POSITION CONTROLLER DESIGN

To design a Neural Network-based MRAC, the following two steps have been followed. The first step is designing conventional MRAC for second-order systems, which is described in equation 23. The second step is training FFNN

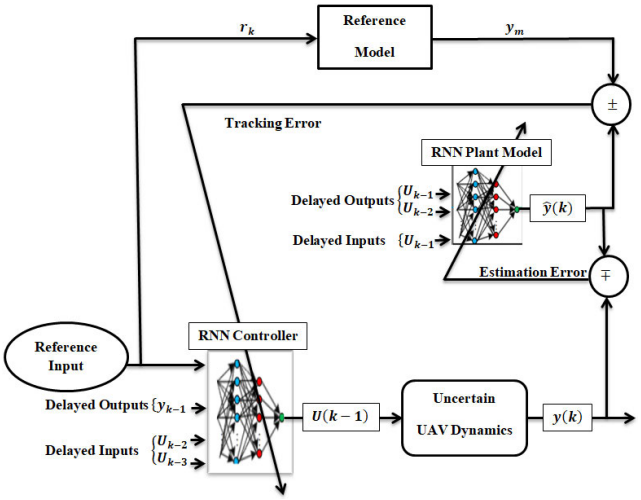

FIGURE 5. RNNSTR of quadrotor attitude control architecture.

offline first to estimate the controller parameters and then implementing online learning algorithm to update network parameters in real-time.

1) CONVENTIONAL MRAC DESIGN OF 2^{ND} ORDER SYSTEM

The implicit assumption in the design of the MRAC system is that the control designer is sufficiently familiar with the plant and its operating environment under consideration so that the designer can specify the desired behavior of the plant in terms of the output of a reference model. In the MRAC technique, the reference model describes system performance; and the adaptive controller forces the system to behave like the reference model. Here, the parameters of the adaptive controller are adjusted based on the deviation of actual plant output from the desired reference model output. As shown in Figure 4, MRAC system has two loops. The first loop is a normal or inner loop, which is an ordinary loop consisting of a plant and nominal nonadaptive controller, that is used to track the set point and stabilize the system [44]. The second loop is parameter adjustment or outer loop that adjusts controller parameters in such a way that driving reference model response tracking error toward zero.

Mathematical techniques like MIT rule and Lyapunov direct method can be used to develop adaptation rules. Even though MIT rule is a widely used straightforward technique to design MRAC to achieve satisfactory results during controller performance evaluation, it is very sensitive to the changes in the amplitude of the reference input. Hence, it can't guarantee global stability in MRAC system [45]. On the other hand, Lyapunov direct method is a well-known method that can be applied to ensure the overall stability in the MRAC system [46]. It is based on establishing some positive definite or Lyapunov function of tracking and estimation error, first, and then adaptation laws are derived in a way that makes the gradient of the candidate Lyapunov function negative. In this way, the candidate Lyapunov function will decrease as time goes forward and derived adaptation laws give bounded asymptotic convergence of control parameters so that the controller can ensure global stability of the overall system.

Since the translational dynamics of the quadrotor in equation 23 is a 2^{nd} order system, MRAC is designed for 2^{nd} order system. To design MRAC, the plant dynamics with matched uncertainty is given as

$$\dot{x} = Ax + B(U - \theta^{*T}\phi(x)) \tag{31}$$

where $f(x) = \theta^{*T}\phi(x)$ is matched uncertainty, $\theta = [\theta_1, \theta_2 \ldots]$ are unknown constant vector, and $[\phi_1(x), \phi_2(x) \ldots]$ are vector of known bounded basis functions. The reference model for 2^{nd} order system is given as

$$\dot{x}_m = A_m x_m + B_m r \tag{32}$$

The ideal control law is defined as

$$U^* = k_r{}^* r - k_x{}^* x + \theta^{*T}\phi(x) \tag{33}$$

where U^* is the ideal control law, and $k_r{}^*$, $k_x{}^*$, and θ^* are final constant estimated values of controller gains when the plant model output perfectly tracks the reference model response. The actual control law that perfectly cancels out the matched uncertainty is defined as

$$U = k_r r(t) - k_x x(t) + \theta^T \phi(x) \tag{34}$$

where k_r, k_x, and θ are actual adaptation parameters. Estimation errors is defined as: $\tilde{k_x} = k_x - k_x{}^*$, $\tilde{k_r} = k_r - k_r{}^*$ and $\tilde{\theta} = \theta - \theta^*$. To drive the adaptation law by using the Lyapunov direct method, let's choose the positive definite Lyapunov candidate function of tracking and estimation error as [9]

$$V(e, \tilde{k_r}, \tilde{k_x}, \tilde{\theta}) = e^T P e + |b| \left(\frac{\tilde{k_r}^2}{\gamma_r} + \tilde{k_x} \gamma_x{}^{-1} \tilde{k_x}^T + \tilde{\theta}^T \gamma_\theta{}^{-1} \tilde{\theta} \right) \tag{35}$$

The gradient of the Lyapunov candidate function can be obtained as

$$\begin{aligned} \dot{V}(e, \tilde{k_r}, \tilde{k_x}, \tilde{\theta}) = & -e^T Q e + 2|b|\tilde{k_r} \left(-r e^T \bar{P} sgn(b) + \gamma_r{}^{-1} \dot{\tilde{k_r}} \right) \\ & + 2|b|\tilde{k_x} \left(x e^T \bar{P} sgn(b) + \gamma_x{}^{-1} \dot{\tilde{k_x}} \right) \\ & + 2|b|\tilde{\theta}^T \left(-\phi e^T \bar{P} sgn(b) + \gamma_\theta{}^{-1} \dot{\tilde{\theta}} \right) \end{aligned} \tag{36}$$

The adaptation laws, which make closed-loop error dynamics uniformly stable, for 2^{nd} system can be obtained by making the gradient of the Lyapunov function negative as follows:

$$\begin{cases} \dot{k}_r = \gamma_r r e^T \bar{P} sgn(b) \\ \dot{k}_x = -\gamma_x x e^T \bar{P} sgn(b) \\ \dot{\theta} = \gamma_\theta \phi e^T \bar{P} sgn(b) \end{cases} \tag{37}$$

In most cases, if A and B are known, then k_x and/or k_r can be designed by any standard non-adaptive control techniques to stabilize the closed-loop system and enable plant output to follow a reference input [9]. Since all the states of the quadrotor can be obtained with the help of sensors onboard platforms, all the states of the quadrotor dynamics can be assumed to be available. Therefore, by considering the adaptation parameter k_x as a full-state feedback gain, k_x will be designed from robust non-adaptive servo LQR control as a control architecture in Figure 4.

- Reference model for 2^{nd} order system is designed based on desired transient response of $t_s = 3$ second settling time and 5% maximum overshoot:

$$x_m = \frac{4}{s^2 + 2.76s + 4} r \tag{38}$$

- Translation quadrotor dynamics in state space form rewritten as

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \frac{1}{m}\left(U_x - C_{d_x} x_2\right) \end{cases} \tag{39}$$

- Since the objective control design is minimizing tracking error, error dynamics to design LQR becomes

$$\begin{cases} \dot{z}_1 = z_2 \\ \dot{z}_2 = -\frac{C_{d_x}}{m} z_2 - \frac{U_x}{m} \end{cases} \tag{40}$$

$$U_x = U_{LQR} + U_{adaptive} \tag{41}$$

where U_{LQR} is non adaptive controller with fixed gain k_x, and $U_{adaptive}$ is adaptive control with adaptive parameters k_r and θ. By using a similar approach, U_y and U_z are designed to control the position of the quadrotor along y and z directions.

2) FFNN TRAINING TO ESTIMATE POSITION CONTROLLER PARAMETERS

ANN with a sufficient number of network parameters has well-proved properties to universally approximate any smooth arbitrary functions to an acceptable level of accuracy. This is the core property of ANN for its superiority over other approximation methods. Furthermore, once trained, ANN requires less computation time and memory storage to perform its tasks [47]. However, choosing the right number of ANN parameters is the main problem in training ANN since there is no defined rule to select it. Therefore, most of the time ANN parameters are chosen by trial and error, keeping in mind that the smaller the numbers are, the better the network is in terms of memory storage and processing time. To estimate position controller parameters, a three-layer ANN with 10 neurons in the hidden layer is trained and tested with 5000 samples of simulation data, and its performance is shown in Figure 6.

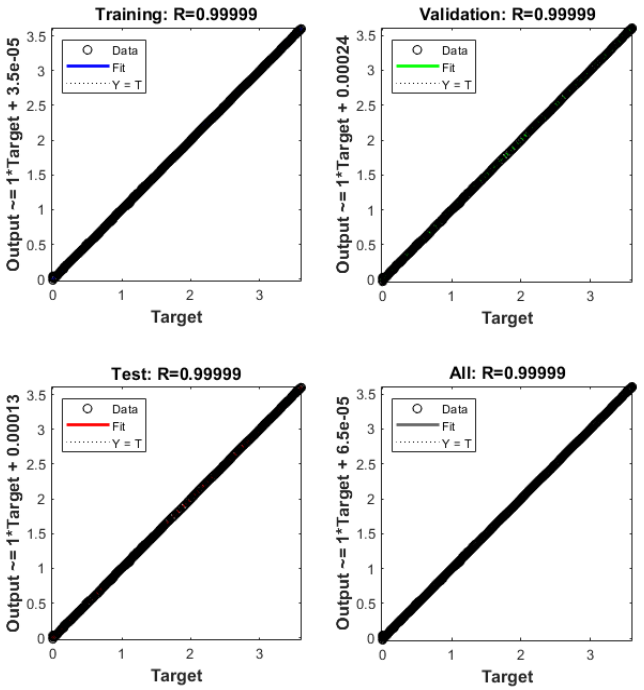

FIGURE 6. Trained FFNN training, validation and test performances.

C. DESIRED ALTITUDE AND QUATERNION COMPUTATION

Desired U_1 and q_d are derived by rewriting equation 22 (a,b&c) and using unit quaternion relation, as follows:

$$\begin{cases} \frac{U_x}{U_1 f_{GE}} = 2(q_{1_d} q_{3_d} - q_{o_d} q_{2_d}) & \text{(42a)} \\ \frac{U_y}{U_1 f_{GE}} = 2(q_{2_d} q_{3_d} + q_{o_d} q_{1_d}) & \text{(42b)} \\ \frac{U_z + mg}{U_1 f_{GE}} = {q_{o_d}}^2 - {q_{1_d}}^2 - {q_{2_d}}^2 + {q_{3_d}}^2 & \text{(42c)} \\ 1 = {q_{o_d}}^2 + {q_{1_d}}^2 + {q_{2_d}}^2 + {q_{3_d}}^2 & \text{(42d)} \end{cases}$$

To derive U_1, square both side of equations 42(a-c), and then take sum as follows:

$$\begin{aligned} &\left[\frac{U_x}{U_1 f_{GE}}\right]^2 + \left[\frac{U_y}{U_1 f_{GE}}\right]^2 + \left[\frac{U_z + mg}{U_1 f_{GE}}\right]^2 \\ &= [2(q_{1_d} q_{3_d} - q_{o_d} q_{2_d})]^2 + [2(q_{2_d} q_{3_d} + q_{o_d} q_{1_d})]^2 \\ &\quad + [{q_{o_d}}^2 - {q_{1_d}}^2 - {q_{2_d}}^2 + {q_{3_d}}^2]^2 \\ &= \left[{q_{o_d}}^2 + {q_{1_d}}^2 + {q_{2_d}}^2 + {q_{3_d}}^2\right]^2 \end{aligned}$$

For unit quaternion rotation, ${q_{o_d}}^2 + {q_{1_d}}^2 + {q_{2_d}}^2 + {q_{3_d}}^2 = 1$. Hence,

$$\left[{U_x}^2 + {U_y}^2 + (U_z + mg)^2\right] \frac{1}{{U_1}^2 {f_{GE}}^2} = 1 \tag{43}$$

$$U_1 = \frac{1}{f_{GE}} \sqrt{{U_x}^2 + {U_y}^2 + (U_z + mg)^2} \tag{44}$$

From desired rotation around z-axis ψ_d, corresponding desired quaternion is obtained by using axis-angle relationship in [29] as follows:

$$q_{3_d} = sin(\frac{\psi_d}{2}) \tag{45}$$

To obtain the desired scalar quaternion q_{o_d}, take the sum of equation 42(c & d), first as

$$\frac{U_z + mg}{U_1 f_{GE}} + 1 = 2{q_{o_d}}^2 + 2{q_{3_d}}^2 \tag{46}$$

$$2{q_{o_d}}^2 = \frac{U_z + mg}{U_1 f_{GE}} - 2{q_{3_d}}^2 + 1 \tag{47}$$

Then, substitute equation 44 & 45 into equation 47. Finally, q_{o_d} can be obtained as

$$q_{o_d} = \frac{1}{\sqrt{2}} \sqrt{\frac{U_z + mg}{\sqrt{{U_x}^2 + {U_y}^2 + (U_z + mg)^2}} - 2sin^2(\frac{\psi_d}{2}) + 1} \tag{48}$$

To derive q_{1_d} and q_{2_d}, first, rewrite equation 42(a & b) as follows:

$$\frac{1}{U_1 f_{GE}} \begin{bmatrix} U_x \\ U_y \end{bmatrix} = 2 \begin{bmatrix} q_{3_d} & -q_{o_d} \\ q_{o_d} & q_{3_d} \end{bmatrix} \begin{bmatrix} q_{1_d} \\ q_{2_d} \end{bmatrix} \tag{49}$$

From the 2×2 matrix inverse property, the inverse is obtained, first as

$$\begin{bmatrix} q_{1_d} \\ q_{2_d} \end{bmatrix} = \frac{1}{2U_1 f_{GE}} * \frac{1}{{q_{o_d}}^2 + {q_{3_d}}^2} \begin{bmatrix} q_{3_d} & q_{o_d} \\ -q_{o_d} & q_{3_d} \end{bmatrix} \begin{bmatrix} U_x \\ U_y \end{bmatrix} \tag{50}$$

Then, substitute equations 44, 45 & 48 into equation 50. Finally, q_{1_d} and q_{2_d} can be obtained as follows:

$$q_{1_d} = \frac{sin(\frac{\psi_d}{2})U_x + \frac{1}{\sqrt{2}} U_y \sqrt{\frac{U_z + mg}{\sqrt{{U_x}^2 + {U_y}^2 + (U_z + mg)^2}} - 2sin^2(\frac{\psi_d}{2}) + 1}}{U_z + mg + \sqrt{{U_x}^2 + {U_y}^2 + (U_z + mg)^2}} \tag{51}$$

and

$$q_{2_d} = \frac{sin(\frac{\psi_d}{2})U_y - \frac{1}{\sqrt{2}} U_x \sqrt{\frac{U_z + mg}{\sqrt{{U_x}^2 + {U_y}^2 + (U_z + mg)^2}} - 2sin^2(\frac{\psi_d}{2}) + 1}}{U_z + mg + \sqrt{{U_x}^2 + {U_y}^2 + (U_z + mg)^2}} \tag{52}$$

Equations 44, 45, 48, 51, and 52 are dealt with inside the conversion block as shown in Figure 3.

D. QUADROTOR ATTITUDE CONTROLLER DESIGN

To make the response of the attitude controller fast relative to the position controller response, a reference model with a settling time of 2 seconds and zero overshoot is selected as

$$y_m = \frac{4}{s^2 + 4s + 4} r \tag{53}$$

As expressed earlier in Figure 5, the overall RNNSTR architecture consists of two separate networks, which are trained separately to perform different tasks. One is trained to estimate the model, and the other is trained to generate control actions that drive the system toward reference input. In this scenario, the first step is estimating the plant model by generating input and output data of the system model in equation 27. To train the plant model network, first, the maximum-minimum input of the system is obtained from equation 18 by considering 75% of the maximum speed of the BLDC motor running at 10000 RPM that yields 1.1/−1.1 Nm torque; and the desired output is seated to lay in the range of [−1 1] quaternion unit. Then, 5000 samples of input-output data are generated with a sampling time of 0.05 seconds. Finally, a three-layer ANN with 10 neurons in the hidden layer is configured and trained. The performance of the trained plant model network on training, validation, and testing datasets is shown in Figure 7, 8, and 9. As can be seen from these figures, the training, validation, and testing data errors are minimal i.e., the plant model network estimates the quadrotor rotation model precisely.

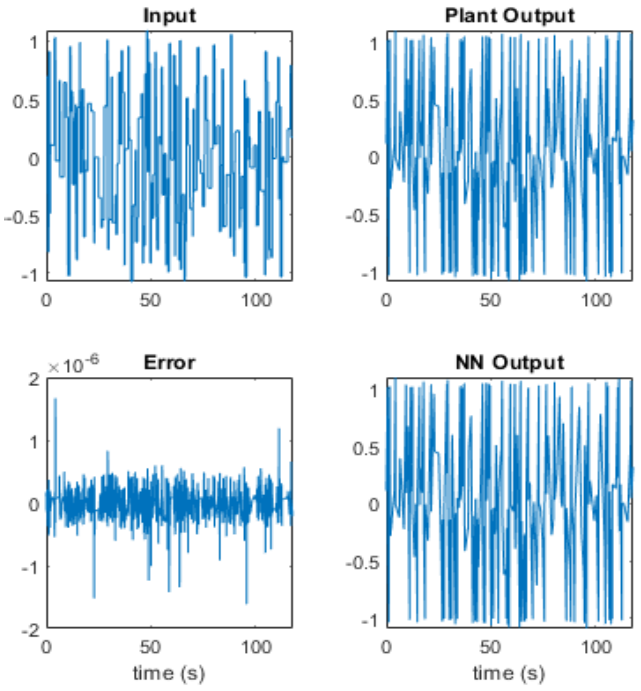

FIGURE 7. Plant model network training performance.

The second step is training the controller network by generating input-out data of the reference model in equation 53. To train the controller network, 4000 data are generated by considering the maximum/minimum size of the desired quaternion unit 1/−1. Here, before training the controller network, plant network parameters are set into appropriate places of the overall network configuration. In this step, only controller network parameters are updated whereas plant network parameters remain fixed. The desired reference

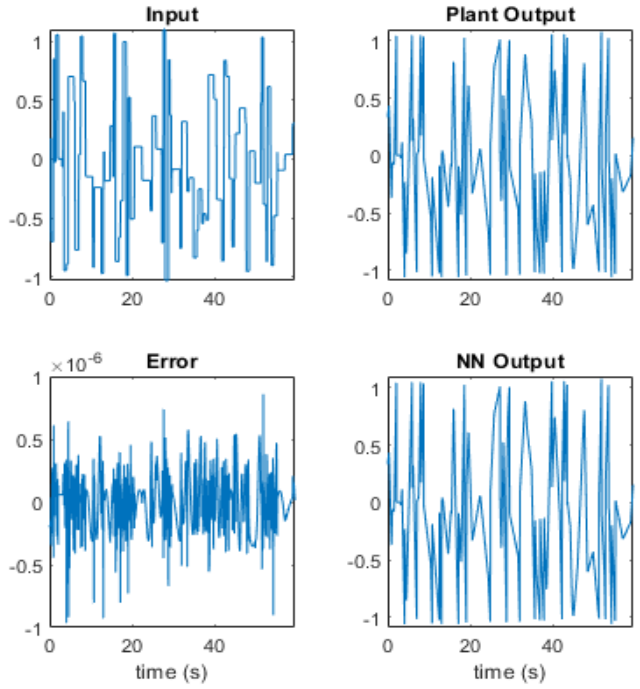

FIGURE 8. Plant model network validation performance.

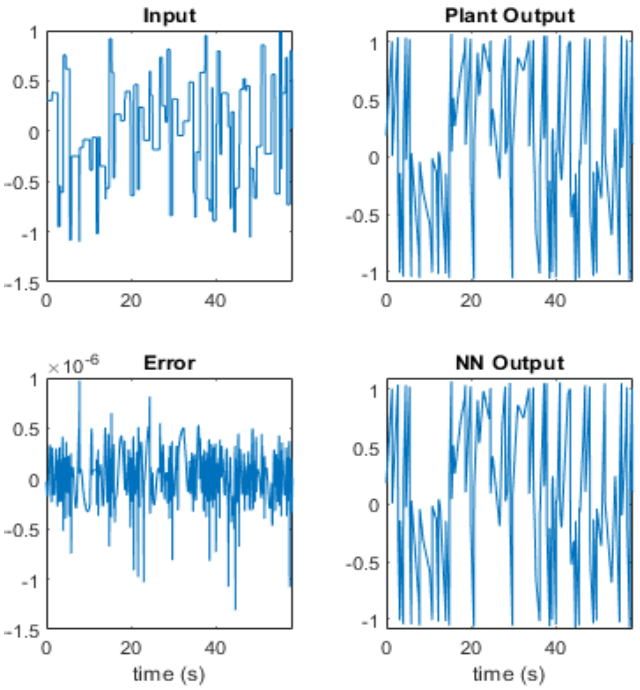

FIGURE 9. Plant model network testing performance.

model response and tracking performance of RNNSTR for uniform random reference input signal on the training dataset is shown in Figure 10. The result shows that the plant network output(green) precisely tracks the reference model output(blue).

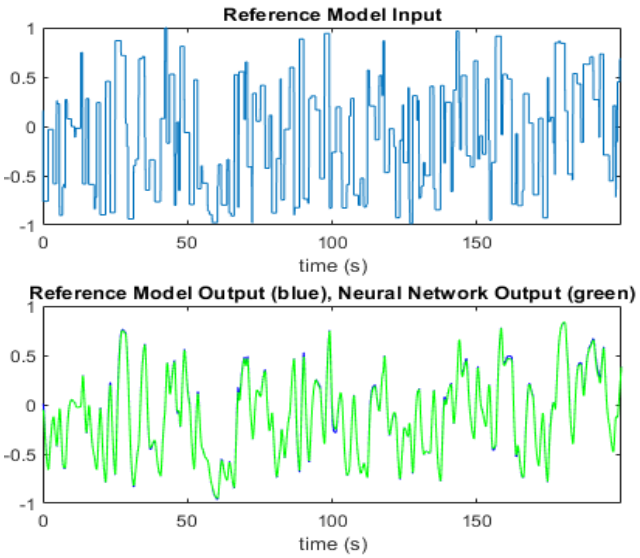

FIGURE 10. Tracking performance of RNNSTR on generated dataset.

These results, which are shown in Figures 7 to 10, are RNNSTR attitude controller training to control q_1 rotation only. By applying the same procedure for q_2 and q_3, RNNSTR attitude controllers are designed as well.

E. ONLINE LEARNING ALGORITHM

Offline learning is not always applicable for real-time implementation. In some scenarios, online learning is needed to update the parameters of the network in real-time. However, using the initial set of network parameters from offline training increases the convergence rate and ensures the stability of the system [48]. Therefore, to increase the performance of the trained network in position and attitude control architectures in Figure 4 and 5, both learning methodologies are applied in this study. Here, offline training is deployed to get the initial set of weights and biases of the network and then, the online learning algorithm is developed to update those weights and biases in real-time. These weights and biases are updated only when the output error exceeds a certain predefined threshold value. To update weight and biases online, a generalized BPA is utilized as follows:

- Once the network output y_i is obtained, output layer delta δ_j is computed, first. Then, output layer weights w_{ji} and biases b_j are updated as

$$\begin{cases} \delta_j = y_i(1-y_i)e_o \\ w_{ji}(k+1) = w_{ji}(k) + \mu\delta_j(k)x_i(k) \\ b_j(k+1) = b_j(k) + \mu\delta_j(k) \end{cases} \tag{54}$$

where delta δ in ANN represents the rate at which the error changes with respect to the output of a particular neuron.

- Then, the error at the hidden layer is obtained as

$$e_h = \sum_{j=1}^{N_o} \sum_{h=1}^{M_o} w_{kh} \delta_j \tag{55}$$

where μ is the learning rate, N_o is the maximum number of output nodes, M_o is the maximum number of hidden layer neurons, w_{kh} is the weights of hidden to the output layer, and e_h is the error at each node in the hidden layer. Once hidden layer error and delta are computed, the delta rule in equation 54 is applied similarly to update the hidden layer weights and biases. In this study, the network is deployed to retrain online by BPA when the error exceeds ± 0.0005, and the flow chart for the online learning algorithm is shown in Figure 11.

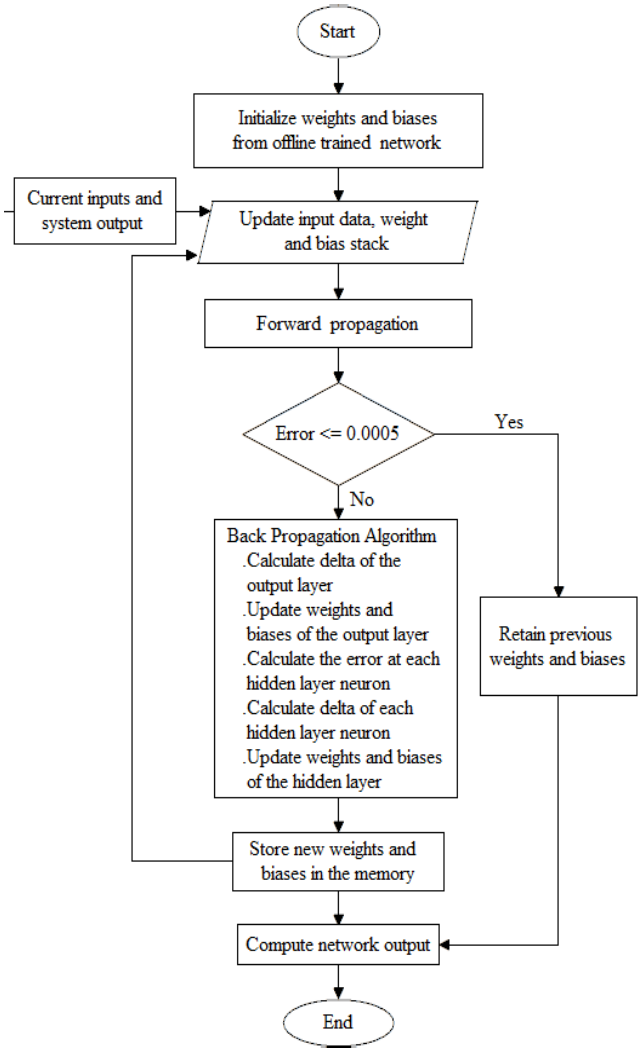

FIGURE 11. Online learning algorithm flowchart.

F. LQR CONTROLLER DESIGN FOR QUADROTOR UAV

LQR is an optimal linear control algorithm that operates in a dynamic system by minimizing a suitable cost function. It is often used when information about the system is completely known and undesirable disturbances like wind that can affect the system are absent. LQR computes the maximum gains to minimize the cost function, which enhances the performance of the system. Generally, high control effort is required to achieve maximum state regulation so that the system states are driven strongly to a stable state [37]. To design LQR, Jacobian approximation is utilized to linearize the nonlinear quadrotor dynamics that can be written as

$$\dot{Z} = AZ + BU \tag{56}$$

where $Z \epsilon R^6$ and $U \epsilon R$ are error state variable and input, respectively and the cost function J for which the controller seeks to minimize is given as [49]

$$J = \int_0^\infty (Z^T QZ + U^T RU) \tag{57}$$

where $Q = Q^T \geq 0$ and $R = R^T > 0$ are positive definite symmetric matrices that weigh the relative importance of the existing error as well as the energy expenditure to stabilize the system [50]. By applying the optimality principle to the optimal control problem, the goal of LQR is to minimize the cost function given in equation 57, and the equation is reduced to a simple Riccatti equation as

$$PA + A^T P - PBR^{-1}B^T P + Q = 0 \tag{58}$$

where $P = P^T \geq 0$ is a positive definite symmetric matrix obtained from equation 58. The optimal state feedback gain matrix is obtained as

$$k = R^{-1}B^T P \tag{59}$$

and the closed-loop LQR controller that drives the cost function towards zero as time goes to infinity is manipulated as

$$U = r - kX \tag{60}$$

where r is the reference input, and X is state vector of the system. The closed-loop control architecture that is deployed to design the LQR controller is shown in Figure 12

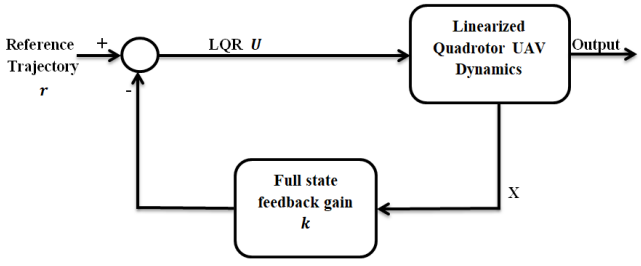

FIGURE 12. LQR control architecture of the quadrotor UAV.

To design LQR, the quadrotor dynamics is linearized at a hovering position. Thus, the linearized model of the quadrotor

translation dynamics from equation 23 is obtained as

$$\begin{cases} \ddot{x} = \dfrac{U_x}{m} \\ \ddot{y} = \dfrac{U_y}{m} \\ \ddot{z} = \dfrac{U_z}{m} \end{cases} \tag{61}$$

Position error dynamics in state space form are obtained as

$$\dot{Z}_p = A_p Z_p + B_p U_p \tag{62}$$

$$\underbrace{\begin{bmatrix} \dot{e}_x \\ \ddot{e}_x \\ \dot{e}_y \\ \ddot{e}_y \\ \dot{e}_z \\ \ddot{e}_z \end{bmatrix}}_{\dot{Z}_p} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{A_p} \underbrace{\begin{bmatrix} e_x \\ \dot{e}_x \\ e_y \\ \dot{e}_y \\ e_z \\ \dot{e}_z \end{bmatrix}}_{Z_p} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ -2.1368 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -2.1368 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -2.1368 \end{bmatrix}}_{B_p} \underbrace{\begin{bmatrix} 0 \\ U_x \\ 0 \\ U_y \\ 0 \\ U_z \end{bmatrix}}_{U_p} \tag{63}$$

where Z_p is a position error vector, which is the difference between the desired and actual state of the system. By using try and error finally, the weighting matrix Q and R are selected for position controller as

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{64}$$

The optimal state feedback gain matrix for position controller can be obtained as

$$k_p = \begin{bmatrix} 1 & 1.3914 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1.3914 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1.3914 \end{bmatrix} \tag{65}$$

Similarly, the attitude controller is designed by linearizing attitude dynamics, which is given in equation 66. At the hovering position, the linear model for rotational dynamics can be obtained as

$$\begin{cases} \ddot{q}_1 = \dfrac{1}{2J_{xx}} U_2 \\ \ddot{q}_2 = \dfrac{1}{2J_{yy}} U_3 \\ \ddot{q}_3 = \dfrac{1}{2J_{zz}} U_4 \end{cases} \tag{66}$$

Attitude error dynamics in state space form are obtained as

$$\dot{Z}_a = A_a Z_a + B_a U_a \tag{67}$$

$$\underbrace{\begin{bmatrix} \dot{e}_{q_1} \\ \ddot{e}_{q_1} \\ \dot{e}_{q_2} \\ \ddot{e}_{q_2} \\ \dot{e}_{q_3} \\ \ddot{e}_{q_3} \end{bmatrix}}_{\dot{Z}_a} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{A_a} \underbrace{\begin{bmatrix} e_{q_1} \\ \dot{e}_{q_1} \\ e_{q_2} \\ \dot{e}_{q_2} \\ e_{q_3} \\ \dot{e}_{q_3} \end{bmatrix}}_{Z_a} + \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ -102.9654 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & -102.9654 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -56.8117 \end{bmatrix}}_{B_a} \underbrace{\begin{bmatrix} 0 \\ U_2 \\ 0 \\ U_3 \\ 0 \\ U_4 \end{bmatrix}}_{U_a} \tag{68}$$

where Z_a is the attitude error vector. The Weighting matrix Q and R for attitude control design are selected as the same as that of position controller, which is described in equation 64. The optimal state feedback gain matrix for the attitude controller can be obtained as

$$k_a = \begin{bmatrix} 1 & 1.0097 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1.0097 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1.0174 \end{bmatrix} \tag{69}$$

By using the values of the optimal state feedback gain matrix in the equation 65 and 69, LQR in equation 60 is designed to stabilize and control the position and attitude of the quadrotor.

IV. SIMULATION RESULTS AND DISCUSSION

A. HELICAL TRAJECTORY TRACKING

A helical trajectory is obtained by using the following desired trajectories: $x_d = \cos t$, $y_d = \sin t$, $z_d = t$, and $\psi_d = -0.5$ rad. The tracking performance of the controller is shown as

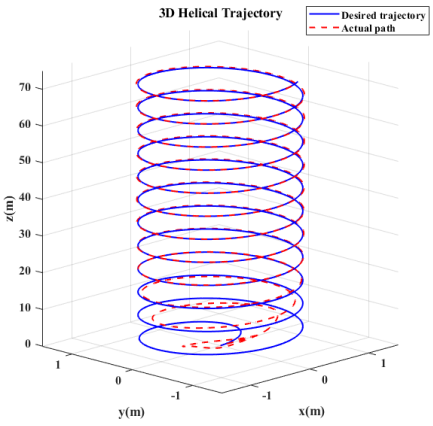

FIGURE 13. 3D helical trajectory tracking performance.

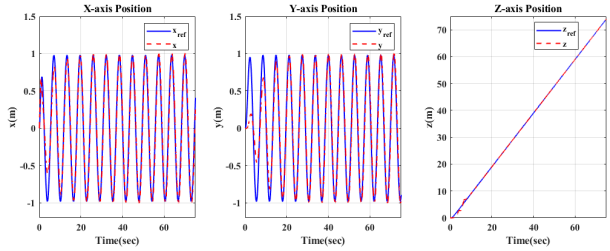

FIGURE 14. Translational trajectory tracking for helical path.

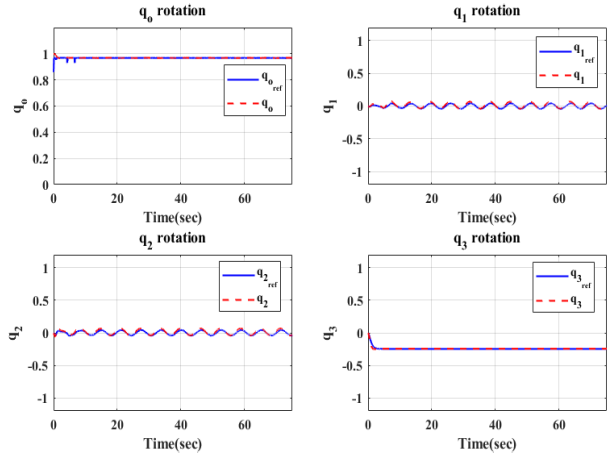

FIGURE 15. Desired quaternion tracking for helical path.

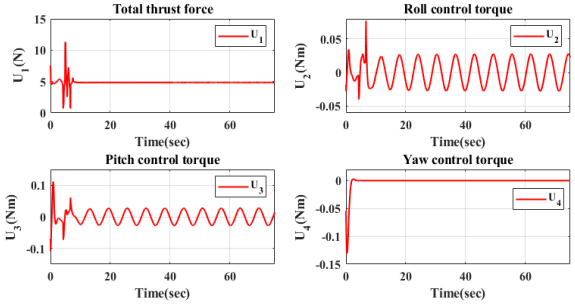

FIGURE 16. Required control efforts to track helical path.

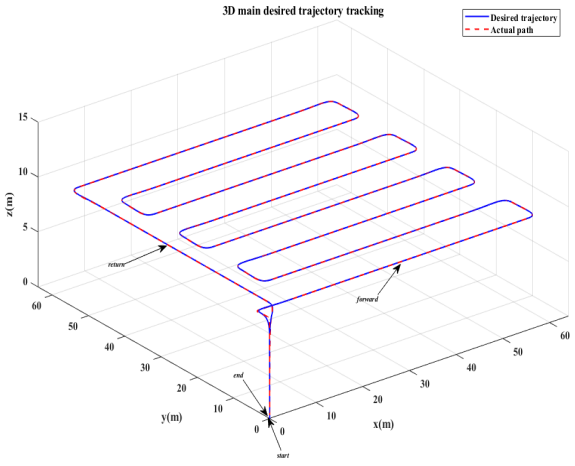

FIGURE 17. 3D main trajectory tracking.

Figure 13, 14 & 15 illustrate that within a finite time, both translational and rotational trajectories converged to

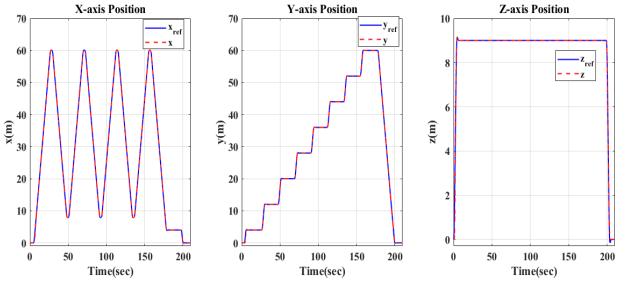

FIGURE 18. Translational main trajectory tracking in nominal scenario.

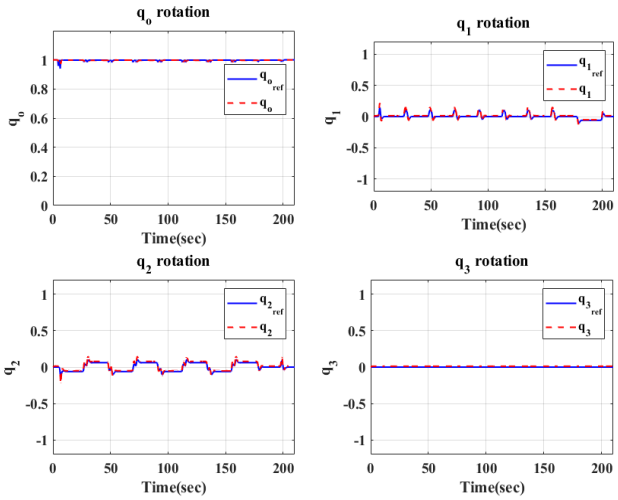

FIGURE 19. Quaternion-based rotational trajectory tracking in nominal scenario.

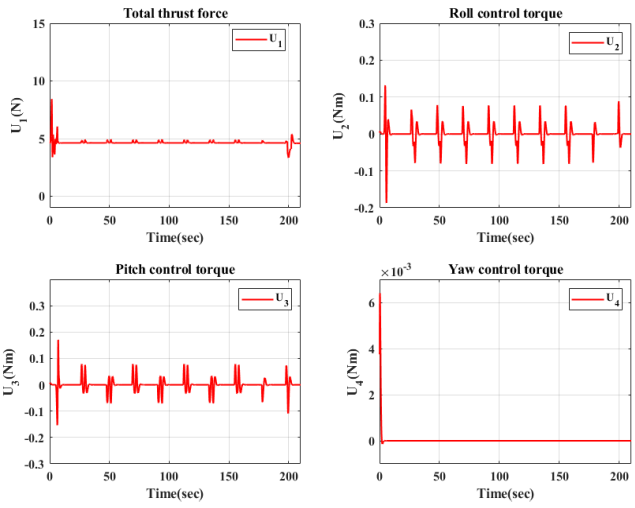

FIGURE 20. Control efforts in nominal scenario.

a desired reference model response, so the suggested controller effectively stabilize the quadrotor system with a high tracking precision. Furthermore, the control efforts that enabled the quadrotor to achieve this task successfully are shown in Figure 16. By manipulating equation 18 with the rated speed of the BLDC motor, the maximum

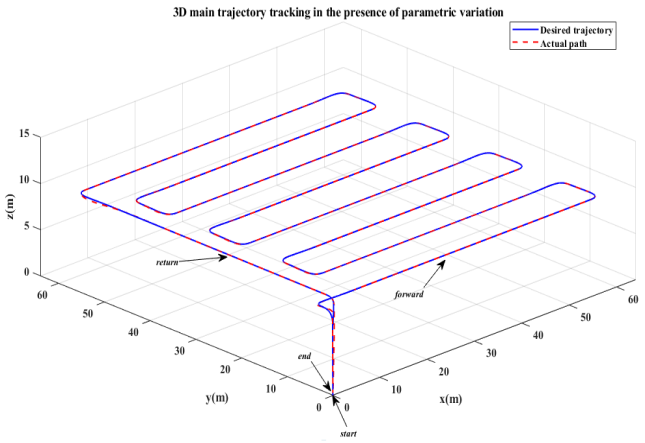

FIGURE 21. 3D main trajectory tracking in the presence of Matched uncertainty.

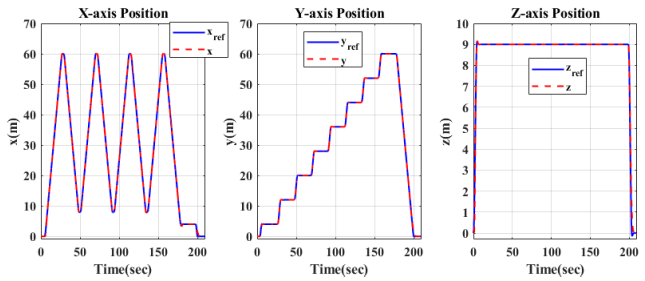

FIGURE 22. Translation tracking in the presence of matched uncertainties.

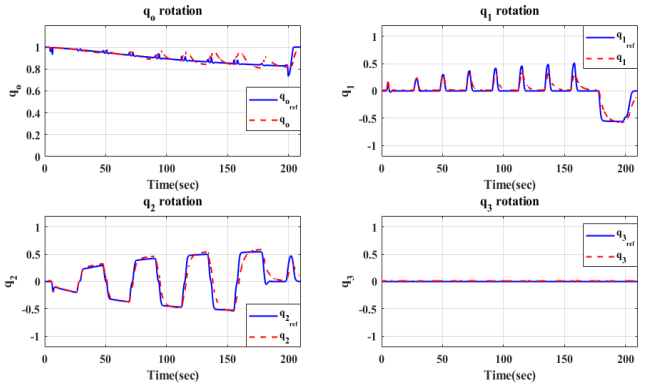

FIGURE 23. Rotation tracking in the presence of matched uncertainties.

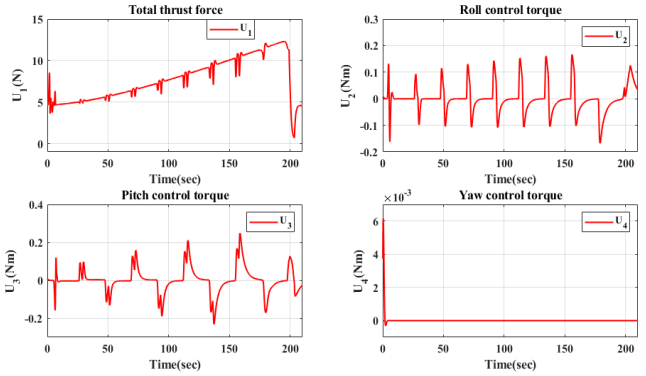

FIGURE 24. Control efforts in the presence of matched uncertainties.

control effort for U_1, U_2, U_3, and U_4 can be obtained as 13.07 N, 1.47 Nm, 1.47 Nm, and 0.25 Nm, respectively. Compared with these, the utilized control efforts to take control action are minimal and smooth.[1] These prove the feasibility of the derived control algorithm. Here, minimal control effort is explicitly stated, in comparison with the control effort that can be obtained when the actuator of the quadrotor rotates at its rated speed. Hence, the proposed control algorithm guarantees accurate desired reference model trajectory tracking by stabilizing the quadrotor system at the expense of appropriate control efforts.

B. MAIN TRAJECTORY TRACKING

The main trajectory is developed specifically for aerial photography from the quadrotor by considering altitude = 9 m, quadrotor speed = 2 m/s, and lateral overlap i.e., the area of one image includes the area already captured in another image = 20%. The generated 3D trajectory and controller tracking performance are shown in Figure 17.

[1]Minimal and smooth control effort indicate better functional safety and economical use of the control system [51], [52]. Relatively large control effort requires large power from an electronic speed controller (ESC) that limits flight time since the quadrotor has a limited energy source, this makes the control system more expensive related to battery size; and oscillation rate in the control signal determines the intensity of wear of moving mechanical parts that probably damage and reduce aging of mechanical parts of the actuator.

As can be depicted in Figure 18 & 19, the rotational and translational quadrotor trajectory precisely track the desired reference model response. Moreover, 3D trajectory tracking in Figure 17 proves that the controller guarantees accurate desired reference model response tracking by perfectly stabilizing the quadrotor when it flies on the top of the field as a desired stated path. Furthermore, the control inputs results in Figure 20 show that the control efforts required to drive the quadrotor are smooth and minimal which proves the feasibility of the proposed control algorithm.

C. MAIN TRAJECTORY TRACKING PERFORMANCE IN THE PRESENCE OF MATCHED UNCERTAINTY

Aerodynamic drag coefficients and inertia depend on the speed of the quadrotor, the density of air, and the altitude of flight, so it is crucial to design a controller that can tolerate these parametric uncertainties in the quadrotor model. Therefore, to validate the robustness of the proposed control algorithm in the presence of parametric variations that can be considered as a matched uncertainty, parametric variations with time are conducted as follows: $J_{xx} = 2J_{xx_o}$, $J_{yy} = 2J_{yy_o}$, $J_{zz} = 2J_{zz_o}$, $c_{d_x} = c_{d_y} = c_{d_z} = (1 + 0.1t)c_{d_o}$, and $c_{a_p} = c_{a_q} = c_{a_r} = (1 + 0.01t)c_{a_o}$ where subscript 'o' is included to indicate parameters at the nominal scenarios.

In Figure 21-23, the tracking errors are acceptable, this implies that the proposed controller tolerates parametric

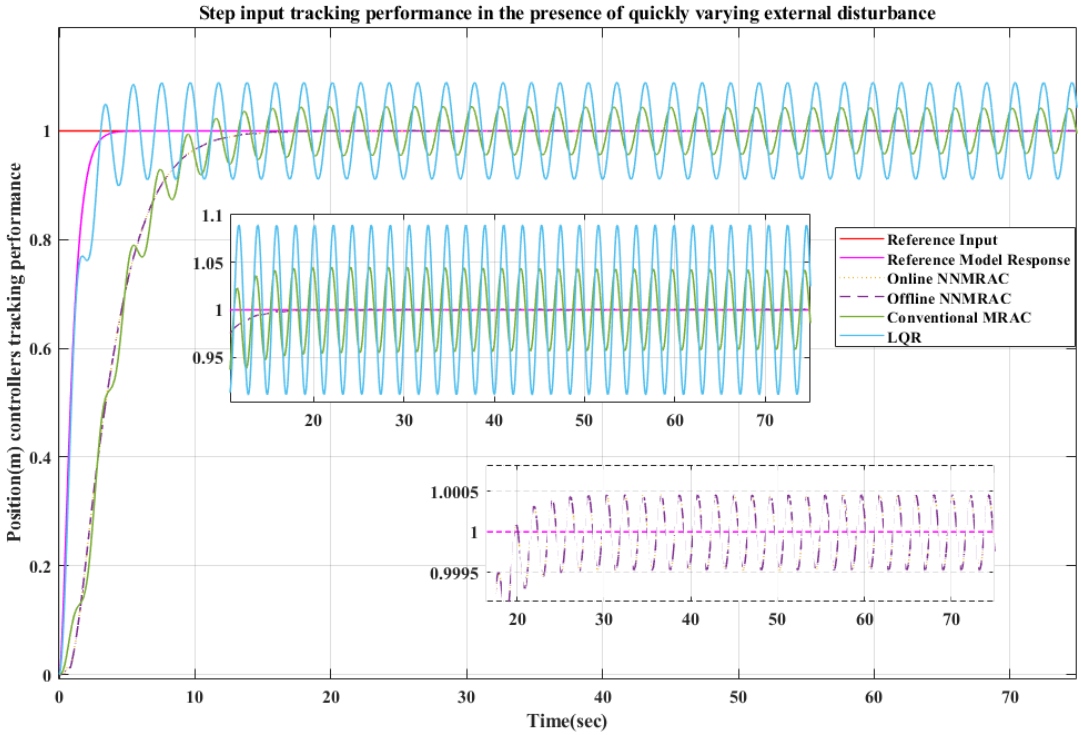

FIGURE 25. Tracking performance for quickly varying disturbance for case-1.

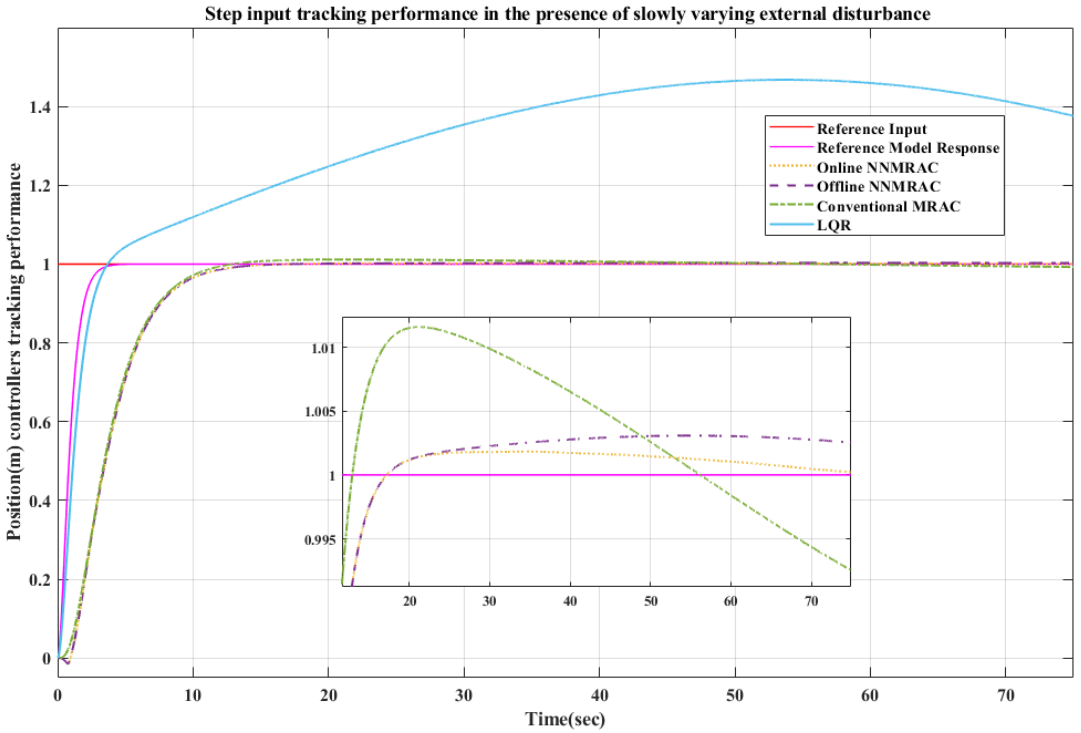

FIGURE 26. Tracking performance for slowly varying disturbance for case-2.

variations in the system by guaranteeing accurate desired trajectory tracking in the presence of parametric variations. This proves the robustness of the proposed control technique since the controller keeps the tracking error within the acceptable range. However, to track the given trajectory, the pose of the quadrotor in Figure 23 is different than that of the nominal scenario in Figure 19. Moreover, as time elapse increases, parametric variations become large and large also

demanding larger control efforts at the turning point as shown in Figure 24. Hence, the proposed controller guarantees accurate desired trajectory tracking with the expense of additional control effort relative to the nominal operation in Figure 20.

D. PERFORMANCE ANALYSIS IN THE PRESENCE UNMATCHED UNCERTAINTY

Quadrotor maneuvering in the field is exposed to external disturbance from the wind, so performance analysis is done by applying unknown external disturbances as unmatched uncertainties for case-1: $d = \sin(3t)$ and case-2: $d = \sin(0.03t)$. Comparison is done based on the tracking performance of online ANN-based MRAC, offline ANN-based MRAC, conventional MRAC, and LQR control approaches. Notice that the disturbances that have been considered here are arbitrary functions since the uncertainty is assumed as unmatched, the controller doesn't have knowledge about it.

In Figure 25, for quickly varying disturbance in case-1, the error is very large for LQR and conventional MRAC; and in the case of online ANN-based MRAC and offline ANN-based MRAC the tracking errors are within a specified range i.e., $|e| \leq 0.0005$, which shows the robustness of the ANN-based control system. Furthermore, from Figure 26, for slowly varying disturbance in case-2, the proposed controller with online tuned ANN parameters, outperforms other presented control strategies. Moreover, in both scenarios, LQR has poor external disturbance rejection capability compared to the presented adaptive control strategies. Therefore, these results prove that online learning algorithms enhance the external disturbance rejection capability of the control system.

V. CONCLUSION AND FUTURE WORKS

A neural network-based MRAC is proposed in this article for the position and attitude control of the quadrotor in the presence of external disturbances and uncertainties. To design the suggested controller, first, quadrotor flight dynamics is modeled by considering every phenomenon that has a significant effect on the quadrotor UAV maneuvering. Then, the proposed controller is designed, and for performance comparison, the LQR controller is designed as well. Finally, to validate the effectiveness of the suggested control approach, numerical simulations have been carried out in nominal scenarios and the presence of matched and unmatched uncertainties. The simulation results validate the effectiveness and superiority of the proposed control technique in achieving high tracking precision and disturbance rejection capability in the nominal scenarios and the presence of matched and unmatched uncertainties. Furthermore, the utilized control efforts to achieve the desired tasks are minimal and smooth. These prove functional safety and economical use of the overall control architecture, so the proposed controller is feasible for real-time implementation of the quadrotor platform. Therefore, the future works of this study will be implementing a fully autonomous quadrotor prototype using the proposed intelligent flight control technique.

REFERENCES

[1] K. Chen, C. Ye, C. Wu, H. Wang, L. Jin, F. Zhu, and H. Hong, "A novel open-closed-loop control strategy for quadrotor trajectory tracking on real-time control and acquisition platform," *Appl. Sci.*, vol. 13, no. 5, p. 3251, Mar. 2023.

[2] E. Okyere, A. Bousbaine, G. T. Poyi, A. K. Joseph, and J. M. Andrade, "LQR controller design for quad-rotor helicopters," *J. Eng.*, vol. 2019, no. 17, pp. 4003–4007, Jun. 2019.

[3] A. Eltayeb, M. F. Rahmat, M. A. M. Basri, M. A. M. Eltoum, and M. S. Mahmoud, "Integral adaptive sliding mode control for quadcopter UAV under variable payload and disturbance," *IEEE Access*, vol. 10, pp. 94754–94764, 2022.

[4] P. K. Reddy Maddikunta, S. Hakak, M. Alazab, S. Bhattacharya, T. R. Gadekallu, W. Z. Khan, and Q.-V. Pham, "Unmanned aerial vehicles in smart agriculture: Applications, requirements, and challenges," *IEEE Sensors J.*, vol. 21, no. 16, pp. 17608–17619, Aug. 2021.

[5] N. P. Nguyen, N. X. Mung, H. L. N. N. Thanh, T. T. Huynh, N. T. Lam, and S. K. Hong, "Adaptive sliding mode control for attitude and altitude system of a quadcopter UAV via neural network," *IEEE Access*, vol. 9, pp. 40076–40085, 2021.

[6] V. R. Puttige, "Neural network based adaptive control for autonomous flight of fixed wing unmanned aerial vehicles," Ph.D. dissertation, UNSW Sydney, 2009.

[7] S. Bhattacharyya, D. Cofer, D. J. Musliner, J. Mueller, and E. Engstrom, "Certification considerations for adaptive systems: Technical report," NASA, Washington, DC, USA, Tech. Rep., 218702, 2015.

[8] G. Pin, Y. Wang, A. Serrani, and T. Parisini, "Dynamic certainty equivalence adaptive control by nonlinear parameter filtering," in *Proc. 59th IEEE Conf. Decis. Control (CDC)*, Dec. 2020, pp. 1454–1459.

[9] N. T. Nguyen and N. T. Nguyen, *Model-Reference Adaptive Control*. Berlin, Germany: Springer, 2018.

[10] G. Cheng, D. Li, and M. He, "Model-free coking furnace adaptive control," *Hydrocarbon Process.*, vol. 78, no. 12, p. 73, 1999.

[11] S. S. Haider, "Simplified neural networks algorithms for function approximation and regression boosting on discrete input spaces," Ph.D. dissertation, Univ. Manchester, 2011.

[12] B. Whitehead and S. Bieniawski, "Model reference adaptive control of a quadrotor UAV," in *Proc. AIAA Guid., Navigat., Control Conf.*, Aug. 2010, p. 8148.

[13] J. C. Lopez-Hoyos, J. S. Cervantes-Rojas, P. Ordaz, and O. Sandre-Hernandez, "Model reference adaptive control for an unmanned aerial vehicle with variable-mass payloads," in *Proc. 18th Int. Conf. Electr. Eng., Comput. Sci. Autom. Control (CCE)*, 2021, pp. 1–6.

[14] R. Burns, *Advanced Control Engineering*. Amsterdam, The Netherlands: Elsevier, 2001.

[15] T. Oktay, S. Arik, I. Turkmen, M. Uzun, and H. Celik, "Neural network based redesign of morphing UAV for simultaneous improvement of roll stability and maximum lift/drag ratio," *Aircr. Eng. Aerosp. Technol.*, vol. 90, no. 8, pp. 1203–1212, Nov. 2018.

[16] N. A. Bakshi, "Model reference adaptive control of quadrotor UAVs: A neural network perspective," *Adapt. Robust Control Syst.*, p. 135, Mar. 2018.

[17] P. Priya and S. S. Kamlu, "Robust control algorithm for drones," in *Aeronautics-New Advances*. IntechOpen, 2022.

[18] L. E. Romero, D. F. Pozo, and J. A. Rosales, "Quadcopter stabilization by using PID controllers," *Maskana*, vol. 5, pp. 175–186, Jan. 2014.

[19] P. Saraf, M. Gupta, and A. M. Parimi, "A comparative study between a classical and optimal controller for a quadrotor," in *Proc. IEEE 17th India Council Int. Conf. (INDICON)*, Dec. 2020, pp. 1–6.

[20] T. Bierling, "Comparative analysis of adaptive control techniques for improved robust performance," Ph.D. dissertation, Technische Univ. Munchen, 2014.

[21] A. Assefa, "Neural network based direct MRAC tech-nique for improving tracking performance for nonlinear pendulum system," *J. Informat. Electr. Electron. Eng. (JIEEE)*, vol. 1, no. 2, pp. 1–15, Nov. 2020.

[22] D. Noble and S. Bhandari, "Neural network based nonlinear model reference adaptive controller for an unmanned aerial vehicle," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2017, pp. 94–103.

[23] R. Prakash and R. Anita, "Neuro- PI controller based model reference adaptive control for nonlinear systems," *Int. J. Eng., Sci. Technol.*, vol. 3, no. 6, pp. 44–60, Jan. 1970.
[24] Y. Y. Lv, W. Huang, J. Liu, and Z. F. Peng, "A sliding mode controller of quadrotor based on unit quaternion," *Appl. Mech. Mater.*, vols. 536–537, pp. 1087–1092, Apr. 2014.
[25] A. Hussein and R. Abdallah, "Autopilot design for a quadcopter," Univ. Khartoum, Khartoum, Sudan, Tech. Rep., 2017.
[26] B. J. Emran and H. Najjaran, "A review of quadrotor: An underactuated mechanical system," *Annu. Rev. Control*, vol. 46, pp. 165–180, Jan. 2018.
[27] F. Ahmadinejad, J. Bahrami, M. B. Menhaj, and S. S. Ghidary, "Autonomous flight of quadcopters in the presence of ground effect," in *Proc. 4th Iranian Conf. Signal Process. Intell. Syst. (ICSPIS)*, 2018, pp. 217–223.
[28] H. A. Gonzalez, "Robust tracking of dynamic targets with aerial vehicles using quaternion-based techniques," Ph.D. dissertation, Univ. de Technologie de Compiegne, 2019.
[29] M. E. Guerrero-Sánchez, H. Abaunza, P. Castillo, R. Lozano, and C. D. García-Beltrán, "Quadrotor energy-based control laws: A unit-quaternion approach," *J. Intell. Robotic Syst.*, vol. 88, nos. 2–4, pp. 347–377, Dec. 2017.
[30] J. Sanwale, P. Trivedi, M. Kothari, and A. Malagaudanavar, "Quaternion-based position control of a quadrotor unmanned aerial vehicle using robust nonlinear third-order sliding mode control with disturbance cancellation," *Proc. Inst. Mech. Eng., G, J. Aerosp. Eng.*, vol. 234, no. 4, pp. 997–1013, Mar. 2020.
[31] J. Cariño, H. Abaunza, and P. Castillo, "A fully-actuated quadcopter representation using quaternions," *Int. J. Control*, vol. 96, no. 12, pp. 3132–3154, Dec. 2023.
[32] S. R. Nekoo, J. Á. Acosta, and A. Ollero, "Quaternion-based state-dependent differential Riccati equation for quadrotor drones: Regulation control problem in aerobatic flight," *Robotica*, vol. 40, no. 9, pp. 3120–3135, Sep. 2022.
[33] L. Baoying, L. Mingqiu, and Y. Junwei, "Research on pose control of quadrotor UAV based on sliding mode active disturbance rejection," in *Proc. 4th Int. Conf. Electron. Inf. Technol. Comput. Eng.*, Nov. 2020, pp. 663–668.
[34] O. Doukhi, A. R. Fayjie, and D. J. Lee, "Intelligent controller design for quad-rotor stabilization in presence of parameter variations," *J. Adv. Transp.*, vol. 2017, pp. 1–10, Jan. 2017.
[35] D. D. C. Bernard, F. Riccardi, M. Giurato, and M. Lovera, "A dynamic analysis of ground effect for a quadrotor platform," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 10311–10316, Jul. 2017.
[36] N. Ahmed and M. Chen, "Sliding mode control for quadrotor with disturbance observer," *Adv. Mech. Eng.*, vol. 10, no. 7, Jul. 2018, Art. no. 168781401878233.
[37] J. M. Selfridge and G. Tao, "A multivariable adaptive controller for a quadrotor with guaranteed matching conditions," *Syst. Sci. Control Eng.*, vol. 2, no. 1, pp. 24–33, Dec. 2014.
[38] C. Bensalah, N. K. M'Sirdi, and A. Naamane, "Full modelling and sliding mode control for a quadrotor UAV in visual servoing task," in *Proc. IMAACA*, 2019, pp. 1–11.
[39] R. Beard, "Quadrotor dynamics and control rev 0.1," Tech. Rep., 2008.
[40] P. Sanchez-Cuevas, G. Heredia, and A. Ollero, "Characterization of the aerodynamic ground effect and its influence in multirotor control," *Int. J. Aerosp. Eng.*, vol. 2017, pp. 1–17, Aug. 2017.
[41] P. J. Sanchez-Cuevas, V. Martin, G. Heredia, and A. Ollero, "Aerodynamic effects in multirotors flying close to obstacles: Modelling and mapping," in *Proc. 4th Iberian Robot. Conf.*, vol. 1. Springer, 2020, pp. 63–74.
[42] Z. Li, X. Ma, and Y. Li, "Model-free control of a quadrotor using adaptive proportional derivative-sliding mode control and robust integral of the signum of the error," *Int. J. Adv. Robotic Syst.*, vol. 15, no. 5, Sep. 2018, Art. no. 172988141880088.
[43] M. T. Matthews and S. Yi, "Model reference adaptive control and neural network based control of altitude of unmanned aerial vehicles," in *Proc. SoutheastCon*, 2019, pp. 1–8.
[44] H. Alimohammadi, B. B. Alagoz, A. Tepljakov, K. Vassiljeva, and E. Petlenkov, "A NARX model reference adaptive control scheme: Improved disturbance rejection fractional-order PID control of an experimental magnetic levitation system," *Algorithms*, vol. 13, no. 8, p. 201, Aug. 2020.
[45] S. Pankaj, J. S. Kumar, and R. K. Nema, "Comparative analysis of MIT rule and Lyapunov rule in model reference adaptive control scheme," *Innov. Syst. Des. Eng.*, vol. 2, no. 4, pp. 154–162, 2011.
[46] P. Jain and M. J. Nigam, "Design of a model reference adaptive controller using modified MIT rule for a second order system," *Advance Electron. Electric Eng.*, vol. 3, no. 4, pp. 477–484, 2013.
[47] Y. Shin, "Neural network based adaptive control for nonlinear dynamic regimes," Ph.D. dissertation, Georgia Inst. Technol., Atlanta, Georgia, 2005.
[48] W. R. De Mel, "Artificial neural network based adaptive controller for DC motors," Tech. Rep., 2004.
[49] H. Purnawan and E. B. Purwanto, "Design of linear quadratic regulator (LQR) control system for flight stability of LSU-05," *J. Phys., Conf.*, vol. 890, Sep. 2017, Art. no. 012056.
[50] K. Ogata, *Modern Control Engineering*, 5th ed. 2010.
[51] M. R. Naseh and M. Haeri, "Robust synchronization of chaotic systems using active sliding mode control with minimum control effort," *Int. J. Modern Phys. B*, vol. 25, no. 17, pp. 2271–2288, Jul. 2011.
[52] S. Skogestad, "Tuning for smooth PID control with acceptable disturbance rejection," *Ind. Eng. Chem. Res.*, vol. 45, no. 23, pp. 7817–7822, Nov. 2006.

MULUKEN MENEBO MADEBO received the B.Sc. degree in electrical and computer engineering and the M.Sc. degree in control engineering from Addis Ababa University, Addis Ababa, Ethiopia, in 2019 and 2023, respectively. He is currently an Aerospace Guidance Navigation and Control Researcher with Information Network Security Administration (INSA), Ethiopia. His research interests include UAV (fixed wing, quadcopter, and hexacopter), cruise missile, precision agriculture, neural networks, fuzzy logic control, robust control, adaptive control, intelligent control, and deep learning.

CHALA MERGA ABDISSA received the B.Sc. degree in electrical engineering and the M.Tech. degree in microelectronic engineering from Addis Ababa University, Addis Ababa, Ethiopia, in 2009 and 2011, respectively, and the Ph.D. degree in electronics engineering from Jeonbuk National University, Jeonju, Republic of Korea, in 2018. He is currently an Assistant Professor with the School of Electrical and Computer Engineering, Addis Ababa University. His main research interests include robotics, automation, and control.

LEBSEWORK NEGASH LEMMA received the B.Sc. degree in electrical and computer engineering and the M.Sc. degree in control engineering from Addis Ababa University, Addis Ababa, Ethiopia, in 1997 and 2000, respectively, and the Ph.D. degree in aerospace, aeronautical and astronautical engineering from Korea Advanced Institute of Science and Technology, Daejeon, South Korea, in 2018. He is currently an Assistant Professor and the Chair of Control Engineering, School of Electrical and Computer Engineering, Addis Ababa University. His main research interests include HVAC, agriculture, autonomous aerial vehicles, climate mitigation, control engineering computing, energy conservation, industrial robots, predictive control, and scheduling.

DEREJE SHIFERAW NEGASH received the M.Tech. and Ph.D. degrees from the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, India, in 2009 and 2011, respectively. He is currently an Assistant Professor with the School of Electrical and Computer Engineering, Addis Ababa University. His research interests include neural networks, fuzzy logic systems, genetic algorithms, and application of AI in nonlinear control and robotics.