Self-Organizing BFBEL Control System for a UAV Under Wind Disturbance

Praveen Kumar Muthusamy, Bhivraj Suthar, Rajkumar Muthusamy, Matthew Garratt, Hemanshu Pota, Lakmal Seneviratne, and Yahya Zweiri, *Member, IEEE*

***Abstract*—A self-organizing bidirectional fuzzy brain emotional learning (SO-BFBEL) controller is developed to control a quadcopter UAV in an uncertain environment. The proposed SO-BFBEL controller improves the performance of the existing BFBEL controller by generating more accurate fuzzy layers in real-time and removes the need to depend on expert knowledge to set the fuzzy layers. The proposed SO-BFBEL controller is applied to control the position of a quadcopter UAV for accurate 3-D eight shape trajectory tracking for three different speed settings under extreme wind disturbances up to 5 m/s in real-time experimentation. Two industrial fans are used to create the wind disturbance for the experiments. The performance is compared to the DNN-MRFT based PID controller and to the BFBEL controller. The experimental results show that the proposed SO-BFBEL controller achieves robust trajectory tracking for both circle and 3-D eight shaped trajectory under extreme wind disturbance and with lower computational cost. The proposed self-organizing algorithm can be applied to optimize other controllers with fuzzy neural network structure.**

***Note to Practitioners*—The learning rate alpha and beta are set manually at the beginning (if no simulations are done to test) to check for the appropriate magnitude of the control signal. If the learning rates are too low, then the controller output will be too low to control the system (too slow to adapt) and if it is too high, then the system will become too sensitive or unstable (if the control signal is too high). Once the magnitude of the learning rate is achieved, then it will be trivial to adjust it further. For example, if alpha and beta values set to 0.0001 is too low and 1 is too high, then around 0.01 will give the most appropriate response. It can then be increased or decreased based on the system response. No other parameters require any tuning and they can all be set to the default values as mentioned in the article.**

***Index Terms*—Brain emotional learning based intelligent control (BELBIC), deep neural network with the modified relay feedback test (DNN-MRFT), flight control, proportional–integral–derivative (PID), quadrotor, reinforcement learning, wind disturbance.**

I. Introduction

ADVANCES in unmanned aerial vehicles (UAVs), also known as drones, have opened up many new applications. Drones include fixed wings, rotary wings, flapping wings, and hybrid configurations. A rotary wing configuration is considered in this article. It is the most suitable for many applications due to its simpler structure with vertical take-off-landing and hovering capabilities. Some recent rotary wing drone applications include automatic power line inspection [1], managing COVID-19 in healthcare [2], [3], firefighting [4], photography [5], and long distance drone delivery [6], [7]. Multirotor drones are under-actuated systems and have nonlinear system dynamics with couplings [8], [9]. Drones are affected by parametric uncertainties and wind disturbance. One of the fundamental parts of drone robotics is its control system especially, under uncertainties and there is a need to improve the performance of the controllers to extend the envelope of the drone applications [10].

A. Related Work

There is extensive literature available in drone control systems with numerous techniques and approaches [11], [12], [13], [14], [15], [16], [17], [18], [19]. The proportional–integral–derivative (PID) controller is popular for drone control due to its simplicity but it needs proper tuning to achieve control which is time consuming. Some of the recent controllers include, a backstepping and sliding mode control combination [11] and a fuzzy negative imaginary controller [12].

In [20], a Hammerstein model-based model predictive controller was used to control a UAV with a three axis gimbal

Manuscript received 7 September 2022; revised 30 December 2022, 1 March 2023, and 27 April 2023; accepted 6 June 2023. Date of publication 19 June 2023; date of current version 18 December 2023. This work was supported by Khalifa University under Grant RC1-2018-KUCARS and Grant CIRA-2020-082. *(Corresponding author: Praveen Kumar Muthusamy.)*

Praveen Kumar Muthusamy, Bhivraj Suthar, and Yahya Zweiri are with the Khalifa University Center for Autonomous Robotic Systems (KUCARS), Khalifa University of Science and Technology, Abu Dhabi 127788, UAE, and also with the Department of Aerospace Engineering, Khalifa University of Science and Technology, Abu Dhabi 127788, UAE (e-mail: praveen.muthusamy@ku.ac.ae; bhivraj.suthar@ku.ac.ae; yahya.zweiri@ku.ac.ae).

Rajkumar Muthusamy is with the Dubai Future Labs-Robotics Lab, Dubai 555509, UAE (e-mail: rajkumar.muthusamy@dubaifuture.gov.ae).

Matthew Garratt and Hemanshu Pota are with the School of Engineering and Information Technology, University of New South Wales, Canberra, NSW 2052, Australia (e-mail: m.garratt@adfa.edu.au; h.pota@adfa.edu.au).

Lakmal Seneviratne is with the Khalifa University Center for Autonomous Robotic Systems (KUCARS), Khalifa University of Science and Technology, Abu Dhabi 127788, UAE, and also with the Department of Mechanical Engineering, Khalifa University of Science and Technology, Abu Dhabi 127788, UAE (e-mail: lakmal.seneviratne@ku.ac.ae).

Video link: https://www.youtube.com/watch?v=nf_aFB-yBmg

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIE.2023.3285922.

Digital Object Identifier 10.1109/TIE.2023.3285922

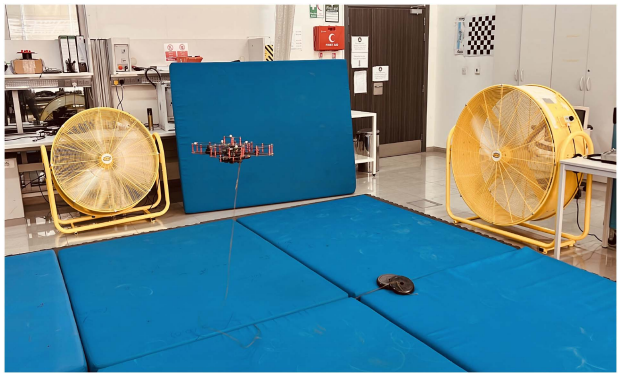

Fig. 1. Qdrone UAV flight under wind disturbance from two industrial fans.

mounted on it. In this approach, the model of the system is first identified using the flight data and it is used to predict the future response based on the input–output relation to produce and optimize a control response appropriate for the situation. By this method, they improve the wind disturbance handling capability compared to a conventional PID but this method relies on the availability of the system model where the flight data for uncertain environments may not be available all the time. In [21], the Harris Hawks Optimization technique was used to optimize the PID controller to improve the controller performance. In [22], a nonlinear auto regressive eXogenous (NARX) model of the UAV was obtained using the flight data and it was used to achieve better control performance of an UAV with payload delivery. In [23], a hybrid metaheuristic optimization algorithm was used to optimize the path planning and its trajectory and along with a PID controller to get an optimal flight control response.

In [13], an intelligent control system based on a radial basis function adaptive NN was used to control the UAV. Here, a system identifier is used to identify the system while it is controlled with a predefined controller, and in the mean time the NN is trained using the identified model in parallel. The trained NN takes over the predefined controller once it is ready, and the system can always switch back to the predefined controller if the NN fails, as a fail safe mechanism. Though this scheme is quite useful, it needs a predefined controller that already works and this may not be suitable for situations where a working controller is not available. Another approach is based on data engineering where the system uses flight data, machine learning, and autotuning algorithms to pretune the control parameters in simulation before experimentation [14].

Many existing control techniques are complex to design and depend on the availability of the system model. Modeling uncertainties such as wind disturbance and other unknown factors are not possible at all times so it becomes challenging to design with the model-based approach in those situations. This is the drawback of the model-based controller or controllers that depend on the availability of the system model and their capability to handle uncertainties. A detailed survey analyzing the model-based controllers and the NN controllers that depend on the training and system identification are detailed in the recent survey paper [24].

There are parametric uncertainties in the UAVs which include modelling uncertainty, rotors, motors, and battery degradation. A detailed analysis of the effects of parametric uncertainties on the UAV was given in [25]. These are handled by modeling the system with multiple flight data in the required environment. A deep neural network with the modified relay feedback test (DNN-MRFT) algorithm follows this method and it is used to identify the system along with the uncertainties based on the data and tune the PID controller for immediate flight [15], [16], [17]. In this approach, the DNN is trained using the data from multiple simulated flight tests. Then, the MRFT technique is used when the drone is in flight using a tuned PID controller. When the MRFT technique is turned-ON, the drone performs oscillatory cycles for all three attitudes and altitude for 2–3 s each. The data from the MRFT is then processed through the trained DNN to get the optimal controller gain values. This procedure identifies the best control gain values for the PID controller for each controlled output and it is done for all three positions and attitude control. This technique provides good performance as it was trained and identified for this environment. In this article, we choose this method to get the best PID controller performance for comparison as it is one of the effective PID controller tuning technique in the recent literature. A brain emotional learning based intelligent control (BELBIC) system inspired by the emotional and logical mechanism of the brain was first introduced in [26] and it was applied to numerous applications including a UAV [27] without much change in the structure of the system. This architecture had ambiguity as parts of the controller were not clearly defined and it needed to be defined for every application. This was addressed in [28], [29] to a large extent and the later version of the controller was given in [30]. In [30], a recurrent NN and a wavelet membership function were used to achieve control of a system. The control system uses a sliding surface which has arbitrary gain values that need to be manually chosen based on the system and this introduces ambiguity in designing the controller. The adaptation algorithm is unchanged and it can only adapt monotonously which means the network weights will keep increasing to infinity even after convergence.

These gaps are addressed in a recent paper where a nonmodel-based bidirectional fuzzy brain emotional controller (BFBEL) that does not depend on the system model, flight data, or require any pretraining of NNs was introduced in [18], [19]. The BFBEL controller is used to achieve six degrees of freedom (6DOF) control for a quadcopter UAV [18] and it was also applied to achieve 1DOF control for a flapping wing platform [31]. Since this is a nonmodel-based controller it is applied to control a magnetic sphere levitation system [29] as well. The advantage of the BFBEL control systems over other NN controllers that are based on the traditional backpropagation learning mechanism is that the BFBEL controller can start from zero initial weights whereas other controllers need to have their initial NN weight values to be close to the converged weight values. This increases the practical applicability of the proposed controller for a wide range of applications with minimal effort. Though the performance of the BFBEL control and its applicability to different systems are

established, the literature on further studies, development, and analysis of the BFBEL controller are quite lacking.

In [18], the fuzzy layers were predefined by using an expected error range and the operation of the controller is limited to that specific range even if the control system can handle more. This is a problem if an external disturbance causes the system to go beyond the expected range where it may not be able to handle it. The capability of the controller is not experimentally tested for wind disturbance. So a mechanism where the operating range of the control system can adapt is missing in the literature along with its experimental performance under wind disturbance.

In this paper, we introduce a novel self-organizing (SO) algorithm for the bidirectional fuzzy brain emotional learning (BFBEL) control system and it is named the Self-Organising BFBEL (SO-BFBEL) controller. The concept of self-organization has been developed in different ways in the literature to modify the structure of the control system by both generating and deleting neural connections to improve the performance and computation time. It is observed that not much attention is given to development of a self-organizing mechanism that is simple enough to be easily extended to other control systems and improve the computational efficiency without sacrificing the performance. The structure of an artificial neural network (ANN) can be developed with a variety of hidden layers and hierarchies depending on the application and the system. It is not practical to develop a simple mechanism that can be used for all ANN designs. This research focuses on controllers based on fuzzy neural network (FNN) structure as they generally utilize the fuzzy membership function on the inputs and connect to the relevant neurons for weight adaptation. This structure is easy and straight forward to use as the activity of neural networks depend on the firing strength of the fuzzy membership functions. In this work, an efficient SO algorithm for FNN based intelligent control system (ICS) is presented. The contributions of this article are as follows.

1) A novel self-organizing BFBEL (SO-BFBEL) control system is introduced that is based on the BFBEL control system. The SO algorithm introduced in this controller can be applied to any FNN-based controllers.
2) The main contribution of the proposed SO algorithm is that it removes the requirement of expert knowledge in the system to predefine the fuzzy layers and removes the limitation of the predefined fuzzy range in the existing BFBEL controller. This improves the tracking performance by creating the most appropriate fuzzy membership functions (fuzzy layers) in realtime and it also lowers the computation cost by deleting the underused or unused fuzzy layers in real-time.
3) The capability of the proposed SO-BFBEL controller is tested by initializing the controller with zero initial weights and with only one fuzzy layer (membership function) with zero as the mean value to test its ability to create its own fuzzy layers online. The proposed SO-BFBEL controller is applied to a Quanser Qdrone QUAV for position tracking control under extreme wind disturbance.
4) Efficiency and performance of the proposed SO-BFBEL controller is tested and compared with the existing BFBEL controller and the DNN-MRFT based PID controller for QUAV control under different scenarios of wind disturbance for multiple flight trajectories.

The rest of this article is organized as follows. Section II gives the problem formulation, and Section III discusses the control structure. Section IV introduces the SO algorithm. The proposed SO-BFBEL controller is simulated and compared with the BFBEL control system in Section V. Experimentation and their results are described in Section VI. The findings of the research are then discussed in Section VII. Finally, Section VIII concludes the article, and some remarks and note to practitioners are given in Sections IX and X.

II. Problem Formulation

Consider a nonlinear system described by the following equation:

$$\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t)) + g(\boldsymbol{x}(t))\boldsymbol{u}(t) + \boldsymbol{d}(t) \tag{1}$$

where $\boldsymbol{u}(t) \in \Re^m$, $\boldsymbol{x}(t) \in \Re^m$, m, and $\boldsymbol{d}(t) \in \Re^m$, denote the control signal, system output, the number of input states, and the external disturbance, respectively. The notations $f(\boldsymbol{x}(t))$ and $g(\boldsymbol{x}(t))$ represent smooth nonlinear continuous functions and they are assumed to be bounded within known limits. The nominal system is defined as

$$\dot{\boldsymbol{x}}(t) = f_0(\boldsymbol{x}(t)) + g_0\boldsymbol{u}(t) + \boldsymbol{d}(t) \tag{2}$$

where $f_0(\boldsymbol{x}(t))$ and g_0 represent nominal $f(\boldsymbol{x}(t))$ and $g(\boldsymbol{x}(t))$, respectively, and $\boldsymbol{d}(t)$ is the lumped uncertainty. It is assumed that $g_0 > 0$ and it is also assumed that the nonlinear system of (2) is controllable and that g_o^{-1} exists. With modeling uncertainties and external disturbances, the nonlinear system (1) can be written as

$$\dot{\boldsymbol{x}}(t) = f_0(\boldsymbol{x}(t)) + g_0\boldsymbol{u}(t) + \boldsymbol{l}(\boldsymbol{x}(t), t) + \boldsymbol{d}(t)$$
$$\text{where} \quad l(\boldsymbol{x}(t), t) = \Delta f(\boldsymbol{x}(t)) + \Delta g(\boldsymbol{x}(t)) \tag{3}$$

where $\Delta f(\boldsymbol{x}(t))$ and $\Delta g(\boldsymbol{x}(t))$ denote the modeling uncertainties of $f(\boldsymbol{x}(t))$ and $g(\boldsymbol{x}(t))$. The control problem is to design a control system where the system output $\boldsymbol{x}(t)$ tracks the desired trajectory $\boldsymbol{x}_d(t)$. The tracking error is defined as

$$\dot{\boldsymbol{e}}(t) = \dot{\boldsymbol{x}}(t) - \dot{\boldsymbol{x}}_d(t) \in \Re^m. \tag{4}$$

The system tracking error vector is defined as

$$\boldsymbol{e}_i \triangleq [e_i(t), \dot{e}_i(t)]^T. \tag{5}$$

Substituting (3) in (4), we get

$$\dot{e}(t) = f_0(\boldsymbol{x}(t)) + g_0\boldsymbol{u} + \boldsymbol{l}(\boldsymbol{x}(t), t) + \boldsymbol{d}(t) - \dot{\boldsymbol{x}}_d(t). \tag{6}$$

If $\boldsymbol{l}(\boldsymbol{x}(t), t)$ is zero or exactly known, an ideal controller can be designed as

$$\boldsymbol{u}^*(t) = g_0^{-1}[\dot{\boldsymbol{x}}_d(t) - f_0(\boldsymbol{x}(t)) - \boldsymbol{l}(\boldsymbol{x}(t), t) - \boldsymbol{d}(t) - \boldsymbol{h}^T\boldsymbol{e} - U_r(t)] \tag{7}$$

where $\boldsymbol{h} \in \Re^{m\times n}$ is the feedback gain matrix and its values are chosen to correspond to the coefficients of a Hurwitz polynomial, i.e., a polynomial whose roots lie in the open left half of the complex plane, so that $\lim_{t\to\infty} ||\boldsymbol{e}|| = 0$. Here, d is assumed

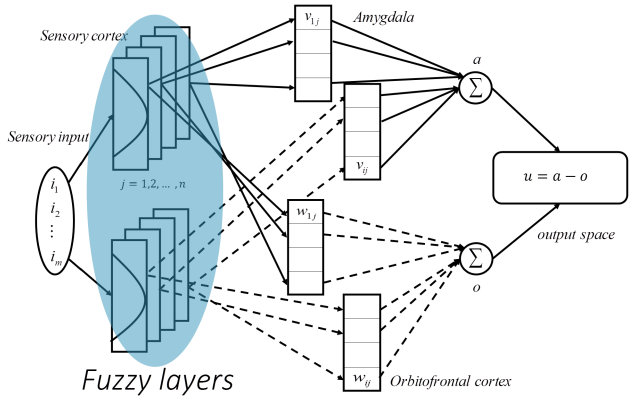

Fig. 2. Self organizing BFBEL control structure used for self-organizing mechanism (highlighted).

to be $||d||_1 < U_r$, where U_r is the approximation controller that can compensate for any approximation error and it is calculated as $U_r(t) = (2r_c^2)^{-1}(r_c^2+1)e(t)$, where r_c is the prescribed positive attenuation constant. However, uncertainty $\boldsymbol{l}(\boldsymbol{x}(t), t)$, is nonzero and generally unknown, so an ideal controller is not practical. A practical BFBEL control is proposed to achieve the desired control for nonlinear systems with uncertainties.

III. BFBEL Control System

The BFBEL control system developed in [18] could adapt rapidly to control a nonlinear system. The BFBEL control system is based on the FNN structure and it is used in this work for further research. The number of neurons in the controller is based on the number of fuzzy membership functions assigned for an input. The controller structure is designed in a way that the adaptation of an individual neuron weight depends on the firing strength of its connected fuzzy layer as shown in Fig. 2. Each fuzzy layer is assigned a range using the Gaussian membership function and it is called the fuzzy range. The fuzzy range of each fuzzy layer is predefined based on the expected operating range of the system. The controller can handle the system if the range of the controller input is within the boundary of the fuzzy range, but cannot work if the input is outside the defined fuzzy range. So designing the fuzzy layer with a specific range is important but unexpected error values may occur due to external factors or noise so a self-organizing mechanism that could generate a new fuzzy layer with an appropriate fuzzy range would greatly improve the capability of the BFBEL controller and its ease of use.

The BFBEL control system used in this research is based on the work in [18] and the structure of the BFBEL control system that is affected by the proposed self-organizing mechanism is shown in Fig. 2.

The structure of the BFBEL control system has four stages as shown in Fig. 2. Each stage of the BFBEL control system, namely, sensory input, sensory cortex, amygdala, and orbitofrontal cortex network, and output as well as the reward signal and the weight adaptation laws are defined below.

$$\underline{\mathbf{i}} = [i_1, i_2, \ldots, i_i, \ldots, i_m]^T \in \Re^m \tag{8}$$

$$b_{ij} = \exp\left[-\frac{1}{2}\left(\frac{i_i - \varsigma_{ij}}{\sigma_{ij}}\right)^2\right] \begin{cases} \text{for} & i = 1, 2, \ldots, m \\ & j = 1, 2, \ldots, n \end{cases} \tag{9}$$

$$a = \sum_{i=1}^{m}\sum_{j=1}^{n} b_{ij} v_{ij};\ o = \sum_{i=1}^{m}\sum_{j=1}^{n} b_{ij} w_{ij} \tag{10}$$

$$U_{\text{BFBEL}} = a - o;\ R = \sum_{i=1}^{m} (q_i i_i) + (c\, U_{\text{BFBEL}}) \tag{11}$$

$$\Delta v_{ij} = \alpha\left[b_{ij}\,(R - a)\right];\ \Delta w_{ij} = \beta\left[b_{ij}\,(U_{\text{BFBEL}} - R)\right] \tag{12}$$

where m is the number of input states, $\underline{\mathbf{i}}$ is input state vector, ς_{ij} is the mean, and σ_{ij} is the variance of the ith input and the jth layer. The notations m and n denote the number of inputs and layers, respectively. The notation v_{ij} and w_{ij} denotes the amygdala and the orbitofrontal cortex network weights. The output of the BFBEL controller is denoted by U_{BFBEL} and the adaptation laws for amygdala and orbitofrontal cortex weights are denoted by Δv_{ij} and Δw_{ij}. The learning rates of the amygdala and orbitofrontal cortex adaptation algorithm, are denoted by α and β, respectively. The notation R denotes the reward signal, q_i is the gain vector, and c is the gain value of the reward signal. The weights are updated as $v_{ij}(t+1) = v_{ij}(t) + \Delta v_{ij}$ and $w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}$.

IV. SO Algorithm

The purpose of the SO algorithm presented in this work is to make the control system perform better trajectory tracking under disturbance with lower computation cost. The proposed SO algorithm is designed to maximize the capability of the controller by allowing the controller to freely alter the operating fuzzy range whereas it is limited by predefined fuzzy range in the existing controller in [18]. The proposed SO algorithm creates new fuzzy layers when the range of existing fuzzy layers does not cover the system input and deletes the existing fuzzy layers that are not used. The activation values of the membership functions are used to determine whether to generate a new layer or delete an existing layer. The generation of a new layer is based on the condition given in (13), where gen_l is the predefined generating threshold.

$$\text{if } \max_i(b_{ij}) < gen_l \text{ then } n_{ij}(t+1) = n_{ij}(t) + 1. \tag{13}$$

If the condition in (13) is met, then the total number of fuzzy layers is increased, where $n_{ij}(t)$ is the number of the existing fuzzy layers for the ith input at time t and $(t+1)$ denotes the next computation-cycle/iteration. When a new layer is created, the initial mean, variance and their associated weight values are assigned as in (14).

$$\begin{aligned} \varsigma_{ij}(t+1) &= I_i \quad \sigma_{ij}(t+1) = \sigma_i \\ v_{ij}(t+1) &= v_n \quad w_{ij}(t+1) = w_n. \end{aligned} \tag{14}$$

Here I_i is the value of ith input, σ_i is the predefined variance value used for the ith input, v_n and w_n are the predefined initial weight values which can be set as either zero, random, or

predefined values based on the system. The generation of new fuzzy layers expands the boundary of operation of the BFBEL controller as required to maintain and improve the control of the system. On the other hand, unwanted fuzzy layers that are not active can be deleted to reduce unnecessary computation. The condition to delete an existing layer is given in (15), where del_l is a predefined deleting threshold. The maximum number of fuzzy layers are restricted to appropriate numbers so that it does not create too many layers and the minimum number of fuzzy layers are set to maintain the smooth control after convergence. Both these limits are set based on the system.

$$\text{if } \min_i(b_{ij}) < del_l \text{ then } n_{ij}(t+1) = n_{ij}(t) - 1. \tag{15}$$

The layer that satisfies the condition in (15) is deleted along with its associated neural network weights and the total number of fuzzy layers are reduced.

The problem formulation and stability of the control system are similar to the one described in [18] and comply with the boundary conditions for achieving stability.

A. Intelligent Control and Stability

There exists an optimal BFBEL control U^*_{BFBEL} to learn the ideal controller U_I such that

$$\begin{aligned} U_I &= U^*_{\text{BFBEL}}(i, \boldsymbol{f}_i, \boldsymbol{W}^*_a, \boldsymbol{W}^*_o) + \epsilon \\ &= a^* - o^* + \epsilon \\ U_I &= f_1 \boldsymbol{l}_a \boldsymbol{W}^*_a - f_2 \boldsymbol{l}_o \boldsymbol{W}^*_o + \epsilon \end{aligned} \tag{16}$$

$$\begin{aligned} f_1 &= R - a; \quad & \boldsymbol{l}_a &= [b_{11}, \ldots, b_{ij}]; \quad & \boldsymbol{W}_a &= [v_{11}, \ldots, v_{ij}] \\ f_2 &= U_{\text{BFBEL}} - R; \quad & \boldsymbol{l}_o &= [b_{11}, \ldots, b_{ij}]; \quad & \boldsymbol{W}_o &= [w_{11}, \ldots, w_{ij}] \end{aligned}$$

where ϵ is a minimum reconstructed error, $\boldsymbol{W}^*_a$ and $\boldsymbol{W}^*_o$ are the optimal parameters of $\boldsymbol{W}_a$ $\boldsymbol{W}_o$. The control law of the BFBEL control scheme is assumed to take the following form:

$$\hat{U} = \hat{U}_{\text{BFBEL}}((i, \boldsymbol{f}_i, \hat{\boldsymbol{W}}_a, \hat{\boldsymbol{W}}_o)) = f_1 \boldsymbol{l}_a \hat{\boldsymbol{W}}_a - f_2 \boldsymbol{l}_o \hat{\boldsymbol{W}}_o \tag{17}$$

where $\hat{\boldsymbol{W}}_a$ and $\hat{\boldsymbol{W}}_o$ are estimates of the optimal parameters obtained from the tuning algorithm introduced later. Subtracting (17) from (16), an approximation error $\tilde{U}$ is defined as

$$\begin{aligned} \tilde{U} &= U_I - \hat{U} = f_1 \boldsymbol{l}_a \tilde{\boldsymbol{W}}_a - f_2 \boldsymbol{l}_o \tilde{\boldsymbol{W}}_o + \epsilon \\ \tilde{U} &= F_1 - F_2 + \epsilon \end{aligned} \tag{18}$$

where $F_1 = f_1 \boldsymbol{l}_a \tilde{\boldsymbol{W}}_a$ and $F_2 = f_2 \boldsymbol{l}_o \tilde{\boldsymbol{W}}_o$.

Theorem 1: Consider the nonlinear system represented by (2) and if the BFBEL controller is designed as in (17) with the adaptation laws as in (19a), (19b), (20a), and (20b), then the convergence of the network parameters and the tracking error of the proposed BFBEL control system can be assured.

$$\dot{\hat{\boldsymbol{W}}}_a = -\alpha (e^T f_1 \boldsymbol{l}_a)^T \tag{19a}$$

$$\text{if } (||\hat{\boldsymbol{W}}_a|| < b_a) \text{ or } (||\hat{\boldsymbol{W}}_a|| = b_a \text{ and } e^T f_1 \boldsymbol{l}_a \hat{\boldsymbol{W}}_a \geq 0)$$

$$\dot{\hat{\boldsymbol{W}}}_a = -\alpha (e^T f_1 \boldsymbol{l}_a)^T + \alpha [e^T f_1 \boldsymbol{l}_a (\hat{\boldsymbol{W}}^T_a \hat{\boldsymbol{W}}_a / ||\hat{\boldsymbol{W}}_a||^2)]^T$$

$$\text{if } (||\hat{\boldsymbol{W}}_a|| = b_a) \text{ and } (e^T f_1 \boldsymbol{l}_a \hat{\boldsymbol{W}}_a < 0) \tag{19b}$$

$$\dot{\hat{\boldsymbol{W}}}_o = \beta (e^T f_2 \boldsymbol{l}_o)^T \tag{20a}$$

$$\text{if } (||\hat{\boldsymbol{W}}_o|| < b_o) \text{ or } (||\hat{\boldsymbol{W}}_o|| = b_o \text{ and } e^T f_2 \boldsymbol{l}_o \hat{\boldsymbol{W}}_o \geq 0)$$

$$\dot{\hat{\boldsymbol{W}}}_o = \beta (e^T f_2 \boldsymbol{l}_o)^T - \beta [e^T f_2 \boldsymbol{l}_o (\hat{\boldsymbol{W}}^T_o \hat{\boldsymbol{W}}_o / ||\hat{\boldsymbol{W}}_o||^2)]^T$$

$$\text{if } (||\hat{\boldsymbol{W}}_o|| = b_o) \text{ and } (e^T f_2 \boldsymbol{l}_o \hat{\boldsymbol{W}}_o < 0) \tag{20b}$$

where $||.||$ denotes the Euclidean norm; α and β are the learning rates; f_1 and f_2 are the amygdala and orbitofrontal cortex processing functions which incorporates positive and negative rewards; b_a and b_o are the given parameter bounds. From (19a) and (19b), if the corresponding situations $||\hat{\boldsymbol{W}}_a|| = b_a$ and $e^T f_1 l_a \hat{\boldsymbol{W}}_a < 0$ are met, then the following condition $(\boldsymbol{W}^*_a - \hat{\boldsymbol{W}}_a)\hat{\boldsymbol{W}}^T_a = 0.5(||\boldsymbol{W}^*_a||^2 - ||\hat{\boldsymbol{W}}_a||^2 - ||\hat{\boldsymbol{W}}_a - \boldsymbol{W}^*_a||) < 0$ holds because $||\boldsymbol{W}^*_a|| < b_a$. This concludes that $V_a \geq 0$. Similarly, from (20a) and (20b), if the corresponding situations $||\hat{\boldsymbol{W}}_o|| = b_o$ and $e^T f_2 l_o \hat{\boldsymbol{W}}_o < 0$ are met then the following condition $(\boldsymbol{W}^*_o - \hat{\boldsymbol{W}}_o)\hat{\boldsymbol{W}}^T_o = 0.5(||\boldsymbol{W}^*_o||^2 - ||\hat{\boldsymbol{W}}_o||^2 - ||\hat{\boldsymbol{W}}_o - \boldsymbol{W}^*_o||) < 0$ holds because $||\boldsymbol{W}^*_o|| < b_o$. This concludes that $V_o \geq 0$.

Proof: Denoting $V(i(t), \boldsymbol{f}_i, \tilde{\boldsymbol{W}}_a, \tilde{\boldsymbol{W}}_o)$ as V, define a Lyapunov function candidate as

$$V = \frac{1}{2} \boldsymbol{e}^T g_0^{-1} \boldsymbol{e} + \frac{1}{2\alpha} tr(\tilde{\boldsymbol{W}}^T_a \tilde{\boldsymbol{W}}_a) + \frac{1}{2\beta} tr(\tilde{\boldsymbol{W}}^T_o \tilde{\boldsymbol{W}}_o)$$

$$\dot{V} = \boldsymbol{e}^T g_0^{-1} \dot{\boldsymbol{e}} - \frac{1}{\alpha} tr(\dot{\hat{\boldsymbol{W}}}^T_a \tilde{\boldsymbol{W}}_a) - \frac{1}{\beta} tr(\dot{\hat{\boldsymbol{W}}}^T_o \tilde{\boldsymbol{W}}_o)$$

$$\dot{V} = \boldsymbol{e}^T g_0^{-1} \dot{\boldsymbol{e}} - A - O$$

where $A = \frac{1}{\alpha} tr(\dot{\hat{\boldsymbol{W}}}^T_a \tilde{\boldsymbol{W}}_a)$ and $O = \frac{1}{\beta} tr(\dot{\hat{\boldsymbol{W}}}^T_o \tilde{\boldsymbol{W}}_o)$.

By subtracting the system dynamic model in (3) from the control law in (7), we get $\dot{e} = -g_0 \tilde{U} - g_0 \boldsymbol{h} \boldsymbol{e} - g_0 U_r(t)$ and by using (18), we can obtain

$$\begin{aligned} \dot{V} &= \boldsymbol{e}^T g_0^{-1} (-g_0 \tilde{U} - g_0 \boldsymbol{h}^T \boldsymbol{e} - g_0 U_r(t)) - A - O \\ &= -\boldsymbol{e}^T \tilde{U} - \boldsymbol{e}^T \boldsymbol{h} \boldsymbol{e} - \boldsymbol{e}^T U_r(t) - A - O \\ &= -\boldsymbol{e}^T (F_1 - F_2 + \epsilon) - \boldsymbol{e}^T \boldsymbol{h} \boldsymbol{e} - \boldsymbol{e}^T U_r(t) - A - O \\ &= -\boldsymbol{e}^T F_1 + \boldsymbol{e}^T F_2 - \boldsymbol{e}^T \epsilon - \boldsymbol{e}^T \boldsymbol{h} \boldsymbol{e} - \boldsymbol{e}^T U_r(t) - A - O \\ &= -\underbrace{(A + \boldsymbol{e}^T F_1)}_{\boldsymbol{V}_a} - \underbrace{(O - \boldsymbol{e}^T F_2)}_{\boldsymbol{V}_o} - \boldsymbol{e}^T \epsilon - \boldsymbol{e}^T \boldsymbol{h} \boldsymbol{e} - \boldsymbol{e}^T U_r(t) \\ \dot{V} &= -\boldsymbol{V}_a - \boldsymbol{V}_o - \boldsymbol{e}^T \epsilon - \boldsymbol{e}^T \boldsymbol{h} \boldsymbol{e} - \boldsymbol{e}^T U_r(t). \end{aligned}$$

Here, the uncertain term is assumed to be bounded as $||\epsilon||_1 < U_r$ and $\dot{V}(i(t), \boldsymbol{f}_i, \tilde{\boldsymbol{W}}_a, \tilde{\boldsymbol{W}}_o) \leq 0$, which is a negative semidefinite function and

$$V(i(t), \boldsymbol{f}_i, \tilde{\boldsymbol{W}}_a, \tilde{\boldsymbol{W}}_o) < V(i(0), \boldsymbol{f}_i, \hat{\boldsymbol{W}}_a, \hat{\boldsymbol{W}}_o).$$

This implies $i, f, \tilde{\boldsymbol{W}}_a$, and $\tilde{\boldsymbol{W}}_o$ are bounded. Let function $B_b(t) \equiv -\boldsymbol{h} \boldsymbol{e}^2 \leq \dot{V}$, so

$$\int_0^t B_b(x) dx \leq V(i(0)) - V(i(t)).$$

Because $V(i(0), \boldsymbol{f}_i, \tilde{\boldsymbol{W}}_a, \tilde{\boldsymbol{W}}_o)$ is bounded functions, $V(i(t), \boldsymbol{f}_i, \tilde{\boldsymbol{W}}_a, \tilde{\boldsymbol{W}}_o)$ is a nonincreasing and bounded function and $\lim_{t\to\infty} \int_0^t B_b(x)dx < \infty$. According to Lyapunov's stability theorem and Barbalat's lemma, it implies that error will converge to zero as $t \to \infty$. Moreover, the parameter estimation errors $\tilde{\boldsymbol{W}}_a, \tilde{\boldsymbol{W}}_o$ can be guaranteed to be bounded in the sense of projection algorithm [32], [33]. This guarantees the stability of the system.

V. Simulation

The simulation is done on a QUAV with nonlinear system dynamics and the control setup as described in [18]. The QUAV dynamics equations are written in state space format as [34]

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{x})\boldsymbol{U} \tag{21}$$

with $\boldsymbol{x} = [x\ y\ z\ \psi\ \theta\ \phi\ \dot{x}\ \dot{y}\ \dot{z}\ p\ q\ r]^T \in \Re^{12}$

where $\boldsymbol{x}$ is the state vector of the system and $\boldsymbol{U}$ is the control signal vector. They are defined as

$$\boldsymbol{f}(\boldsymbol{x}) = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ q\frac{s(\phi)}{c(\theta)} + r\frac{c(\phi)}{c(\theta)} \\ q(c(\phi)) - r(s(\phi)) \\ p + q(s(\phi)t(\theta)) + r(c(\phi)t(\theta)) \\ 0 \\ 0 \\ g \\ qr(I_y - I_z)/I_x \\ rp(I_z - I_x/I_y \\ pq(I_x - I_y)/I_z \end{pmatrix} \qquad \boldsymbol{U} = \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \end{pmatrix}$$

$$\boldsymbol{g}_1(\boldsymbol{x}) = [0\ 0\ 0\ 0\ 0\ 0\ g_1\ g_2\ g_3\ 0\ 0\ 0]^T \in \Re^{12}$$

$$\boldsymbol{g}_2(\boldsymbol{x}) = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ (1/I_x)\ 0\ 0]^T \in \Re^{12}$$

$$\boldsymbol{g}_3(\boldsymbol{x}) = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ (1/I_y)\ 0]^T \in \Re^{12}$$

$$\boldsymbol{g}_4(\boldsymbol{x}) = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ (1/I_z)]^T \in \Re^{12}$$

$$\text{with} \begin{cases} g_1 = -\left(s(\phi)s(\psi) + c(\phi)c(\psi)s(\theta)\right)/m \\ g_2 = -\left(c(\psi)s(\phi) - c(\phi)s(\psi)s(\theta)\right)/m \\ g_3 = -\left(c(\phi)c(\theta)\right)/m \end{cases}$$

where m is the mass of the drone, c, s and t denote the sin, cos, and tan, respectively. The g denotes the acceleration due to gravity and I_x, I_y, I_z are the moments of inertia with respect to x, y, z axes, and p, q, r are the angular rates of roll ϕ, pitch θ and yaw ψ angles, respectively. The control signals are given as

$$U_1 = (T_1 + T_2 + T_3 + T_4) \qquad U_2 = T_l(T_1 - T_2 + T_3 - T_4)$$

$$U_3 = T_l(T_1 + T_2 - T_3 - T_4) \qquad U_4 = (-Q_1 + Q_2 + Q_3 - Q_4)$$

where $T_l = l\ \sin(45^\circ)$, l is the distance from the centre of the quadcopter frame to the centre of the rotor, U_1, U_2, U_3, and U_4 denotes the controller output for altitude (Z-position), roll (ϕ), pitch (θ), and yaw (ψ), respectively. The attitude update is done using the quaternion method as described in [35], [36]. The nonlinear dynamic equations for the quadrotor dynamics, aerodynamics, wind disturbances, and the BFBEL control system are implemented in Simulink using S-function blocks.

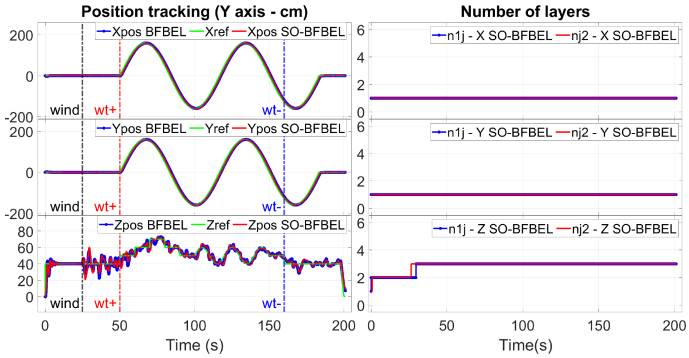

Fig. 3. Position control performance simulation results.

To compare the performance and the efficiency, the parameters of both the BFBEL and the SO-BFBEL controller are set to be the same. Both the SO-BFBEL controller and the BFBEL controller learning rates are set the same but the number of fuzzy layers and fuzzy range are different.

To show the capabilities of the SO algorithm, all the SO-BFBEL controllers are initialized with zero weights and start with only one fuzzy layer ($n_{ij} = 1$) with zero mean values ($\varsigma_{ij} = 0$). Also, for the newly generated layers, the initial weight values are set to zero, i.e., $v_n = 0$ and $w_n = 0$. The generating threshold (gen_l) is set to 0.5 and the deleting threshold (del_l) is set to 0.3. The maximum number of layers that can be generated is limited to 5 and the minimum number of layers to maintain after generation is set to 2.

A combination of cosine and constant signals are set as the trajectories for $X - Y$ position and a combination of step, cosine and constant signals are set for the altitude. The output of the $X - Y$ position controller is set as the reference for roll and pitch attitudes and yaw is set as 0°.

The position of the drone is controlled with changing payload of 225 g under wind disturbance. Wind gusts and turbulence are generated using Dryden spectra model [37] and a gust generation block in Simulink. The wind gust is generated with a wind velocity of 1 m/s with gust amplitude of 5 m/s. The turbulence is generated with a mean wind velocity of 12 m/s. The generated wind gusts and turbulence are combined as wind disturbance and they are applied from 25 s for the simulations. The simulation results for position and attitude control using the BFBEL and SO-BFBEL control system are shown in Fig. 3. The results show that both the BFBEL and the proposed SO-BFBEL controller can adapt swiftly to handle the varying payload under wind disturbance to obtain precise trajectory tracking. These results also show that the high wind disturbances create intense vertical motion along the Z axis and it is effectively controlled by both the BFBEL and the proposed SO-BFBEL controller where both the controller adapts to reduce the effects of the wind disturbance over time. The horizontal drift along the $X - Y$ axes is also effectively controlled by both controllers.

The number of layers used by the SO-BFBEL controller for position and attitude control is shown in Fig. 3. From these results, it can be seen that the number of layers used for X and Y position control is just 1 and the layers used for the altitude control is 3, which is less than the 5 used in the BFBEL controller.

The benefit of this SO algorithm is that it reduces the computation cost without sacrificing the performance and makes the controller more capable of handling uncertainties and disturbances. Computation time is lower as the value of n_{ij} increases and decreases as required for the SO-BFBEL controller whereas the number of layers is fixed to $n_{ij} = 5$ for the BFBEL controller. Another benefit is that unlike the BFBEL controller in [18] where the fuzzy range needs to be predefined, the SO-BFBEL controller can be implemented with zero fuzzy range, and it can generate the range as required, which makes it much more effective and easy to use.

VI. Experiments

To evaluate the performance of the proposed control system, a quadcopter UAV is made to multiple trajectories under different scenarios where measurable wind disturbance is applied during flight. In this section, first the experimental setup used for experimentation is described, then the experimentation process is described followed by the experimental results and their analysis.

A. Experimental Setup

A Quanser Qdrone UAV [38] is used as the test platform and its total weight is 1.2 kg. An OptiTrack Motion Capture System (MCS) is used to measure the position and velocity of the drone. An inertial measurement unit sensor is used to measure the attitude of the drone. The drone has an onboard computer with Intel Aero Compute Board Quad-core processor. All the computation for control is done onboard and the position data is transmitted through WiFi from the MCS. The experiment is done using MATLAB Simulink software interface provided by Quanser.

Two IGMA heavy duty industrial fans [39] are used to generate wind disturbance as shown in Fig. 1. The average wind velocity of both fans is measured using an anemometer. The graph in Fig. 5 shows the average wind velocity measured by the anemometer at various distances along the (x_1, x_2, x_3), (y_1, y_2, y_3), and (z_1, z_2, z_3) coordinates from both the fans. The wind velocity was measured from up to 4.5 m from the fan. The maximum wind velocity was recorded at the nearest location to the fan. The wind velocity decreases as the distance from the fan increases. The maximum wind velocity of the Fan-1 and Fan-2 are 6.8 m/s and 6.5 m/s, respectively. The minimum wind velocity for Fan-1 and Fan-2 was measured as 3.1 m/s and 1.7 m/s, respectively, at 4.5 m along the (x_2, z_3) direction. Both Fan-1 and Fan-2 are placed at 2.5 m from the drone's hover position (red cross) as shown in Fig. 4. The circle trajectory is made with 0.5-m radius and it starts from the hovering point. The closest and farthest distances between the drone and Fan-2 and Fan-1 are (1.5 m and 2 m) and (2.5 m and 3 m), respectively, for the circle trajectory. The eight shaped trajectory is designed

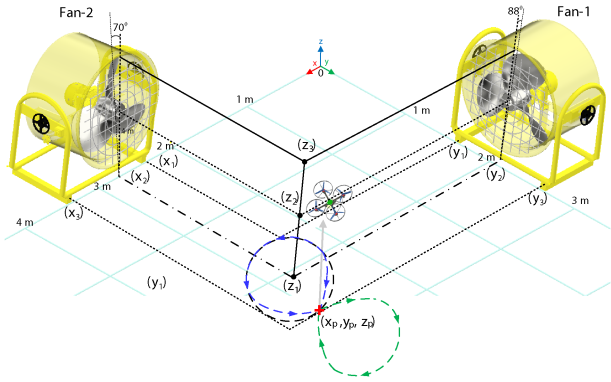

Fig. 4. Representation of wind disturbance using two fans on the UAV.

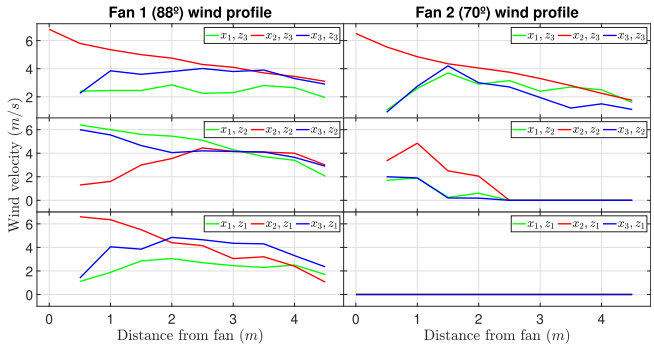

Fig. 5. Wind profile of both the industrial fans where $(z_1, z_2, z_3) = (0, 0.5, 1)$m.

with 0.5-m amplitude on the X axis and 1-m amplitude on the Y axis. The closest and the farthest distance between the drone and Fan-2 and Fan-1 are (1.5 m and 2 m) and (3.5 m and 3 m), respectively, for the eight shaped trajectories.

Fan-1 is tilted at 88° and it is positioned on the Y axis to create wind disturbance on the horizontal plane along the X axis. Fan-2 is tilted at 70° and it is positioned on the X axis to create wind disturbance on an upward slope along the Y axis. The Fan-2 is tilted so that it can produce upward wind to simulate a gust. Each fan creates enough wind to imitate turbulence and two fans combined create cross wind. The fans are placed in such a way that the wind disturbance can be applied in both horizontal and vertical planes as shown in Fig. 6.

B. Experimentation

For experimentation, the proposed SO-BFBEL and the BFBEL controllers are used to control the position of the drone and a tuned PID controller is used to control the attitude of the drone. The drone is made to fly under extreme environment to evaluate the proposed control scheme. We created the two most common wind disturbance situations for drone flight which are: 1) wind disturbance from one direction or one axis; and 2) wind disturbance from two directions or two axes, also known as cross wind. The experiments are done for five scenarios. First is to fly without any wind disturbance, second is to fly with wind disturbance on Y axis (only Fan-2), third is to fly with wind

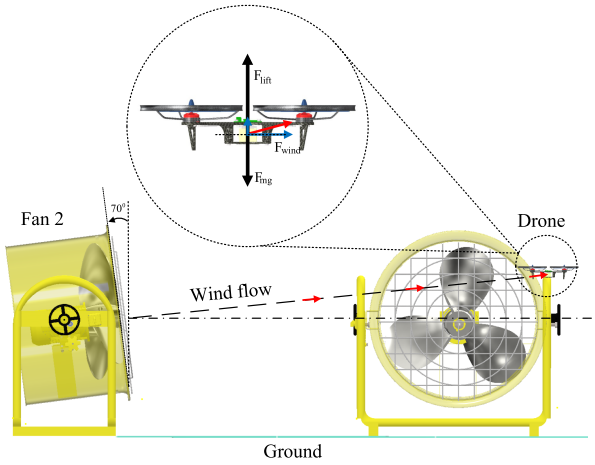

Fig. 6. Free body diagram showing the direction of the wind disturbance applied on the drone.

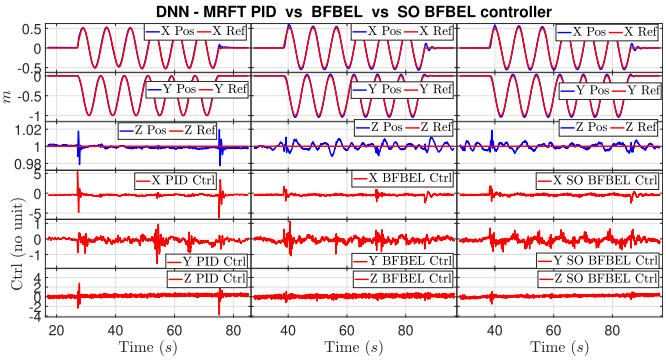

Fig. 7. Position control comparison for case 1.

disturbance on X axis (only Fan-1), and the last two are to fly with wind disturbance on both X and Y axis (both fans) for different flight trajectories.

C. Experimental Results

All the flight test scenarios are done using three controllers, namely, DNN-MRFT based PID controller [15], [16], [17], BFBEL controller [18], and the proposed SO-BFBEL controller.

1) Case 1: No Wind Disturbance: In this case, the drone is made to hover, follow multiple circle trajectories, and hover again without any wind disturbance. The flight control performance of the drone using all three controllers along with their respective control signals are shown in Fig. 7.

2) Case 2: With Wind Disturbance From Y-Axis (Only Fan-2): In this case, the drone is made to hover, multiple circle trajectories, and hover again while wind disturbance is applied on Y axis using Fan-2. The flight control performance of the drone using all three controllers along with their respective control signals are shown in Fig. 8.

3) Case 3: With Wind Disturbance From X-Axis (Only Fan-1): In this case, the drone is made to hover, follow multiple circle trajectories and hover again while wind disturbance is applied on the X axis using Fan-1. The flight control performance

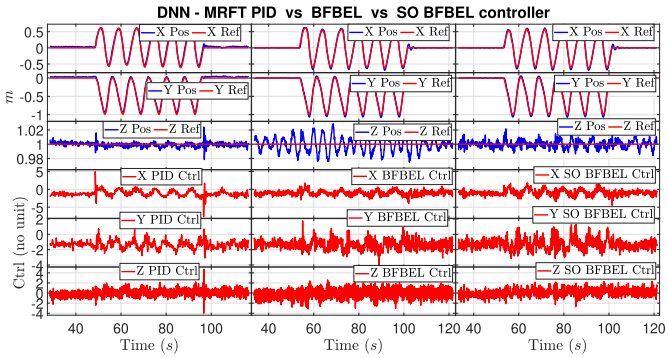

Fig. 8. Position control comparison for case 2.

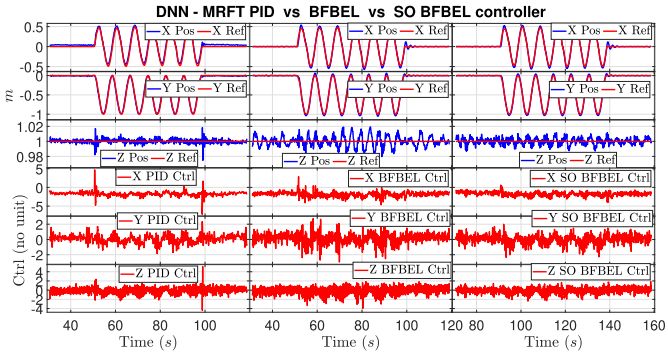

Fig. 9. Position control comparison for case 3.

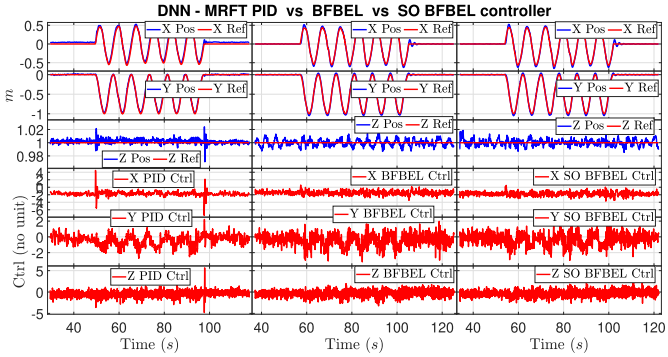

Fig. 10. Position control comparison for case 4.

of the drone using all three controllers along with their respective control signals are shown in Fig. 9.

4) Case 4: With Wind Disturbance From Both X and Y Axes (Both Fan-1 and Fan-2): In this case, the drone is made to hover, follow multiple circle trajectories, and hover again while wind disturbance is applied on both the X and Y axes using both Fan-1 and Fan-2 at the same time. The flight control performance of the drone using all three controllers along with their respective control signals are shown in Fig. 10.

5) Case 5: 3-D—Eight Shape Trajectory Under Wind Disturbance From Both X and Y Axes (Both Fan-1 and Fan-2): In this case, the drone is made to follow an eight trajectory at three different speeds on all three axes under wind disturbance

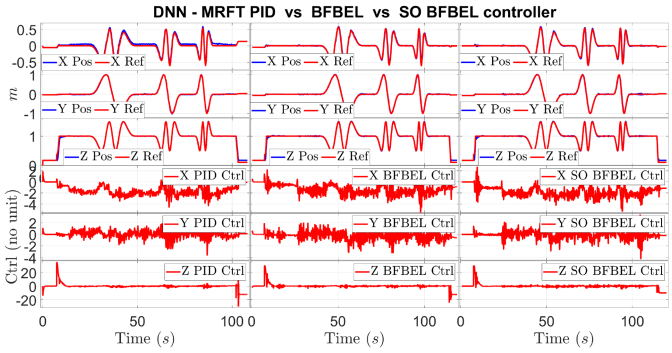

Fig. 11. Position control comparison for case 5.

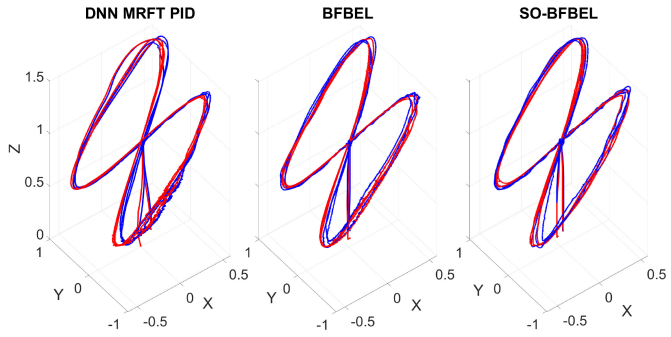

Fig. 12. 3-D plot of position control for case 5 where red is reference and blue is actual trajectory.

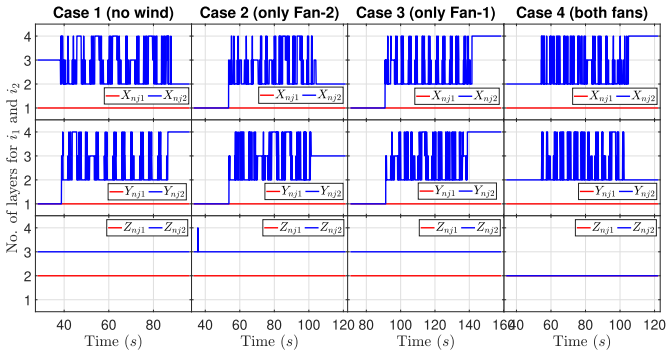

Fig. 13. Number of fuzzy layers used for cases 1–4.

from both the fans. The eight trajectories are designed to be completed in a specific time. The first eight trajectory is designed to be completed in 25 s, the second in 15 s, and the last one in 10 s. The flight control performance of the drone using all three controllers along with their respective control signals are shown in Fig. 11. The 3-D plot of the position control for case 5 results is shown in Fig. 12 for a 3-D representation of the results.

6) Number of Fuzzy Layers Used in SO-BFBEL for All Cases: The number of fuzzy layers and neural connection for SO-BFBEL during the flight for cases 1–4 are shown in Fig. 13 and case 5 is shown in Fig. 14.

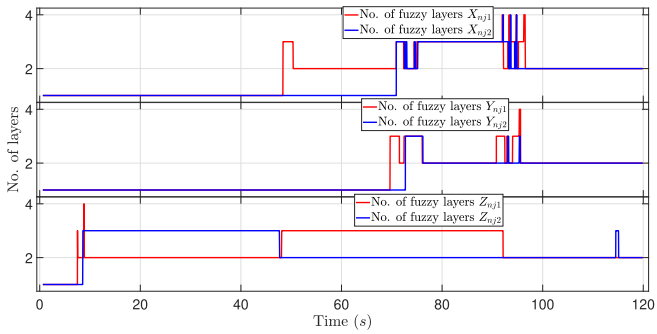

Fig. 14. Number of fuzzy layers used for case 5.

TABLE I
RMSE AND EXECUTION TIME COMPARISON

Cases / controllers	DNN-MRFT PID	BFBEL	SO-BFBEL
X pos case 1-RMSE	2.7442 cm	2.8404 cm	2.8324 cm
Y pos case 1-RMSE	2.2422 cm	2.3437 cm	2.3574 cm
Z pos case 1-RMSE	0.2625 cm	0.3836 cm	0.3162 cm
Comp time case 1	7.11E-04 s	6.66E-04 s	6.66E-04 s
X pos case 2-RMSE	4.6519 cm	2.7199 cm	2.6079 cm
Y pos case 2-RMSE	2.3743 cm	2.1755 cm	2.0951 cm
Z pos case 2-RMSE	0.2897 cm	0.7205 cm	0.4323 cm
Comp time case 2	7.50E-04 s	6.89E-04 s	6.62E-04 s
X pos case 3-RMSE	4.3888 cm	2.5366 cm	2.5117 cm
Y pos case 3-RMSE	3.7183 cm	2.5527 cm	2.5613 cm
Z pos case 3-RMSE	0.3169 cm	1.0722 cm	0.5254 cm
Comp time case 3	7.36E-04 s	7.65E-04 s	5.64E-04 s
X pos case 4-RMSE	4.827 cm	2.5728 cm	2.623 cm
Y pos case 4-RMSE	2.1376 cm	2.2212 cm	2.2459 cm
Z pos case 4-RMSE	0.3803 cm	0.5053 cm	0.4775 cm
Comp time case 4	7.94E-04 s	7.52E-04 s	6.94E-04 s
X pos case 5-RMSE	4.3608 cm	2.0998 cm	2.0561 cm
Y pos case 5-RMSE	1.3021 cm	1.4749 cm	1.2764 cm
Z pos case 5-RMSE	3.0918 cm	2.4274 cm	2.4135 cm
Comp time case 5	7.01E-04 s	6.83E-04 s	6.01E-04 s

D. Analysis of the Experimental Results for All Five Cases

The performance of all three controllers is quantified in terms of root-mean-square-error (RMSE) and it is shown in Table I. The flight time is clipped exactly for all the controllers for the first four cases to calculate the absolute RMSE values for comparison. For case 1, the flight time is clipped at 10 s before and after the multiple circle trajectories to highlight the hovering and trajectory tracking performance. As for cases 2,3, and 4, the flight time is clipped at 20 s before and after the multiple circle trajectories to highlight the hovering and trajectory tracking performance for various scenarios of wind disturbance. The computation time is measured for the specified flight time and its average is used for comparison. For case 5, the flight time for all three eight-shaped trajectories are manually clipped for calculating the RMSE and computation time for all three controllers as the hovering time between each speed setting is different for each controller due to manual switching.

In case 1, all three controllers are applied to control the drone to follow a trajectory. The trajectory for X and Y positions are set to hover and follow multiple circle trajectories and hover again, whereas the Z position (altitude) is set to hover constantly. This flight test is done with no external wind disturbance and this scenario is made to see the difference in the performance of the controllers without any disturbance and to use it as a benchmark to compare with other scenarios. It can be seen from Table I that the performance of the DNN-MRFT based PID is slightly better than both BFBEL and the proposed SO-BFBEL by very tiny margin. This is because the DNN was trained and the MRFT was used to calibrate the controller gain values for this situation before the flight test. On the other hand, both BFBEL and the proposed SO-BFBEL were not pretrained for the test and both the controllers showed remarkable real-time adaptation and achieved convergence within a short span of time, and performed the trajectory tracking continuously. The most notable part is that the performance of both BFBEL and SO-BFBEL are very close to a DNN-MRFT based PID even though they were adapting from zero initial weights and the difference in performance is roughly $1\ \mathrm{mm}$ for all axes.

The control signal of all three controllers follow a similar pattern since they track the same trajectory, but there is a noticeable difference in the magnitude of the control signals. The PID control signal is twice the value of the proposed SO-BFBEL controller when there is a sudden change in the trajectory and it can be seen at 75 s compared to the SO-BFBEL shown in Fig. 7 at 85 s. Those are the exact time when the circle trajectory ends and the impact of sudden stops affect the Z position (altitude). The shift in drone motion using the PID controller was noticeably sudden compared to the other two controllers. The PID values obtained using the DNN-MRFT were designed to give the best performance leading to somewhat aggressive controller response. On the other hand, both BFBEL and SO-BFBEL controllers have much smoother responses without compromising performance.

In case 2, Fan-2 is used to apply wind disturbance along the Y axis. This fan is tilted at 70° to apply wind in an upward slope (bottom to top) along the Y axis. The area of wind coverage for Fan-2 is quite large and it circulates the wind inside the entire lab measuring $5 \times 4.5\ \mathrm{m}$. The direction of the wind disturbance affects the $X-Y$ position and causes drift along the $X-Y$ axes and since the wind has an upward direction it affects the drone along the Z axis as well. From case 2 results, it can be seen that both BFBEL and SO-BFBEL controllers have much better performance under wind disturbance for both X and Y positions, and the PID controller edges slightly better in the Z position. The performance of both BFBEL and SO-BFBEL controllers are better than the PID controller by a wide margin of $2\ \mathrm{cm}$ for X position and more than $1\ \mathrm{cm}$ for Y position whereas PID leads in the Z axis by $1.5\ \mathrm{mm}$ compared to the SO-BFBEL and $4.3\ \mathrm{mm}$ compared to the BFBEL controller.

The impact of both the BFBEL and SO-BFBEL controller's adaptation can be seen on both X and Y position since they are not affected by a larger disturbance and their ability to adapt quickly to the disturbance achieves position tracking performance better than case 1 (no wind disturbance) for both X and Y position. As for the performance of the altitude control, it can be seen from Fig. 13 that the number of layers used for Z position control by SO-BFBEL did not change much and remained almost the same for both the Z control inputs. This is because the error margin of the Z position did not exceed the predefined threshold to create or delete a new layer thus remaining the same. This effectively means that the altitude control is done using two fuzzy layers but it still performs better than the BFBEL controller which is operating with five fuzzy layers for each input. This is because the SO-BFBEL chooses a better fuzzy range that will be active all the time whereas the manually predefined five fuzzy layers of the BFBEL controller may not be active all the time which highlights the practical advantage of the self-organizing mechanism.

In case 3, Fan-1 is used to apply wind disturbance along the X axis. Fan-1 is slightly bigger and more powerful than Fan-2 and its area of wind coverage is greater than Fan-2. From case 3 results, it can be seen that the proposed SO-BFBEL and the BFBEL controller can effectively maintain the $X-Y$ position trajectory performance even under the severe wind disturbance compared to the PID controller. Here, both the SO-BFBEL and BFBEL controller perform better than the PID controller by a wide margin of $1.8\ \mathrm{cm}$ for X position and $1.15\ \mathrm{cm}$ for Y position. Here, the Z position performance of BFBEL controller is worse by a bigger margin than case 2 due to a wider predefined fuzzy range where more fuzzy layers are inactive and the active ones are not producing enough firing strength for the networks to adapt to smaller error. This is due to the lower error sensitivity that was predefined, but this problem does not affect the SO-BFBEL controller to such an extent since it gives consistent performance despite having the same issue from the predefined threshold. Similar to case 2, the number of active fuzzy layers for the SO-BFBEL controller shown in Fig. 13 is not changing as the error threshold for creating or deleting a new layer is not met, but this can be adjusted by narrowing the error threshold to create or delete fuzzy layers.

In case 4, both Fan-1 and Fan-2 are used to apply wind disturbance along both $X-Y$ axes. Here the entire lab is quite windy and there was a large amount of cross wind from all directions as the wind was circulating inside the lab. This scenario replicates the windy outdoor environment more appropriately where the wind direction and speed are entirely random. From case 4 results, it is can be seen that both the BFBEL and the SO-BFBEL controllers perform consistently better than the DNN-MRFT based PID controller for both $X-Y$ axes and the PID controller edging slightly better at Z position as explained in case 2 and 3.

In case 5, it can be seen that the drone performs in three eight-shaped trajectories in all three axes with extreme wind disturbance from both Fan-1 and Fan-2. In this case, both the fuzzy range of the BFBEL controller and the fuzzy layer generation and deletion threshold is modified to analyze the performance of these two controllers further. The drone is set to hover while both the fans are turned-ON to apply the wind disturbance. It is then made to perform its first, second and third 3-D eight-shaped trajectory with trajectory time of 25 s, 15 s, and 10 s, respectively. This flight test is different from the cases 2–4 as the drone is set to continuously move between extreme wind

(turbulent) and low wind (gust) as it moves towards and away from both the fans. In addition to that, the altitude of the drone is set to vary continuously to test the performance of the altitude controller even further. In this case, for all three controllers, the reference trajectory vary slightly as the reference trajectory is created based on the current position of the drone and this is done to smoothen the trajectory and for safety.

From the results of case 5, it can be seen in Fig. 11 that there is a noticeable difference in the performance of *X* position control between the PID and the SO-BFBEL controller whereas the difference in both $Y - Z$ axes are not too noticeable. To give a holistic perspective of the results, it is plotted in a 3-D plot as is shown in Fig. 12. In the 3-D plot, it has to be noted that Fan-2 and Fan-1 are placed exactly in the position where X and Y are labelled in the $X - Y$ axis ((0,0) on the axes scale). The drone moves closer to both the fans when the Y-axis trajectory goes below 0. The drone flies right above Fan-2 when the X-axis trajectory is at 0 while the *Y*-axis trajectory is at -1, i.e., at $(0, -1)$.

In the 3-D plot, it can be seen that the performance of the DNN-MRFT based PID is significantly affected when it passes through the $(0, -1)$ coordinates as the effect of wind disturbance is quite high (turbulent) at that point whereas the performance of both the BFBEL and the SO-BFBEL controller is way better. Between these two, it can be seen that the trajectory tracking of the SO-BFBEL controller is more smooth and close to the reference trajectory (accurate) at all times compared to the BFBEL controller. Likewise, the performance of the DNN MRFT based PID controller gets significantly affected again by the wind disturbance from both fans around the (0.5,1) coordinate where a significant gust effect occurs due to the angle of Fan-2 (upward projection) and due to chaotic wind circulation inside the flight test area. It can be clearly seen that the BFBEL controller has little effect and the proposed SO-BFBEL controller has almost no effect in the same region compared to the DNN-MRFT based PID controller. This demonstrates the rapid adaptation capability of the SO-BFBEL controller due to the SO algorithm that is creating the most accurate fuzzy layers required to perform accurate trajectory tracking response, even if the drone transitions between both highly turbulent and gusty wind environment.

From the results of all the five cases above, it can be observed that when the wind disturbance is applied on either X or Y or both axes, there is a slight steady-state error with the DNN-MRFT based PID in *X* position while hovering whereas there is no steady-state error for both the BFBEL controller and the SO-BFBEL controller. The steady-state error in PID can easily be fixed by increasing the integral gain, but it was left unchanged since it was not present without the disturbance and to show the difference when it is under wind disturbance. The control signal response from all three controllers is consistent for all the cases and it can be seen that the PID controller produces a much greater control signal than the proposed SO-BFBEL controller for all the cases. The proposed SO-BFBEL controller also demonstrates effectively less computation time than the BFBEL controller for all the cases and improves the overall performance as well.

The effects of the model-based offline tuning can be seen in the performance of the PID controller in case 1 where it performs well in an ideal scenario without wind disturbance, but that is not the case when wind disturbance is applied. Whereas, the SO-BFBEL controller adapts to handle both the uncertainties and the wind disturbance in real-time without depending on the model which highlights its practical usability for all the cases.

From Fig. 13, it can be seen that the number of fuzzy layers and neural network keep changing during flight to reduce the computation cost and uses less neural networks weights to get similar or better performance than the BFBEL controller for case $1 - 4$. From Fig. 14, it can be seen that the frequency of the fuzzy layer generation-deletion is decreased for both $X - Y$ controllers as the generation-deletion threshold value was increased slightly. Likewise, the frequency of the fuzzy layer generation-deletion is increased for the Z controller as the generation-deletion threshold value was decreased slightly. The BFBEL controller uses five fuzzy layers ($n_{1j} = n_{2j} = 5$) for all three position controllers for all the scenarios whereas the proposed SO-BFBEL controller starts with only one fuzzy layer ($n_{1j} = n_{2j} = 1$) and it starts to change during flight. In all these tests, the minimum number of fuzzy layers is set to two (after it increases) and the maximum is set to five and the SO algorithm adapts accordingly.

VII. Discussion

It is observed that the PID controller gain values obtained using the DNN-MRFT approach performed better than the trial and error approach. This approach lacks real-time adaptation to changes in the system or environment since the DNN-MRFT is an offline approach. This can be addressed if there is a mechanism to update the PID gain values to changes or disturbances in real-time. On the other, both the BFBEL and the SO-BFBEL controller are nonmodel based and they can be applied without any data based offline training or requiring the system model which makes it more suitable and practical for applications where uncertainties are common as in the case of UAV flight tracking. The overall performance of the BFBEL controller can be improved by modifying the fuzzy range to be more precise and appropriate for the situation but this needs to be done by carefully analyzing the input states (error and velocity) for those situations which require expert knowledge. This is addressed by the proposed SO mechanism where the exact fuzzy range is created in real-time and they are also deleted if they get redundant or useless in real-time. Thus, the proposed mechanism is improving the performance of the controller without manual intervention.

VIII. Conclusion

In this work, a novel SO-BFBEL control system was developed to control a drone (QUAV). The developed SO-BFBEL controller is not limited by a predefined operating range and it effectively lowers the computational time during operation. Simulation results demonstrate the superior performance of the proposed SO-BFBEL controller with payload change during flight under severe wind disturbance of 12 m/s with lower

computation time. The experimental analysis under wind disturbance of 5 m/s using two industrial fans demonstrates the better performance and computation time of the proposed SO-BFBEL controller compared to the BFBEL and the DNN-MRFT based PID controller. The self-organizing feature increases the operating range of the system in real-time and it will be beneficial if the input error range goes beyond the expected/predefined range due to uncertainties. In addition, it also creates the most accurate fuzzy layer in real-time to improve the performance of the drone under extreme wind disturbance. The self-organizing feature is demonstrated to generate and delete layers as required in real-time to maintain optimal performance and computation cost. The developed SO algorithm can be extended to other fuzzy neural network-based control systems.

IX. Remarks

The DNN-MRFT based PID controller performs well under ideal circumstances without wind disturbance since it was trained and tuned for that environment but its performance deteriorates when it is subjected to wind disturbance. On the other hand, both the BFBEL and the SO-BFBEL controllers get better over time especially under disturbances since they actively adapt in real-time, which makes them ideal for use in uncertain environments. This can be further improved by creating a mechanism to adapt the learning rates in real-time or using an offline optimization technique to get the optimal learning rates before the flight.

Acknowledgment

The authors would like to thank Eng. Anees Peringal for his support during experiments.

References

[1] H.-A. Langaker et al., "An autonomous drone-based system for inspection of electrical substations," *Int. J. Adv. Robotic System*, vol. 18, no. 2, 2021, Art. no. 17298814211002973.

[2] A. Restas, "Drone applications fighting COVID-19 pandemic-towards good practices," *Drones*, vol. 6, 2022, Art. no. 15.

[3] M. D. Mohanty, P. K. Mallick, and M. N. Mohanty, "Drone applications fighting COVID-19 pandemic—towards good practices," *IEEE Internet Things Mag.*, vol. 4, pp. 24–29, 2021.

[4] M. S. S. Tanjim, S. A. Rafi, S. Barua, A. N. Oishi, and M. I. Hossain, "FRIQ 1.0: A guided quadcopter to inject retardant fluid or gas aerially into the fire affected zone," in *Proc. 2nd Int. Conf. Sustain. Technol. Ind. 4.0*, 2020.

[5] B. Chamberlain and W. Sheikh, "Design and implementation of a quadcopter drone control system for photography applications," in *Proc. Intermountain Eng., Technol. Comput.*, 2022, pp. 1–7.

[6] H. Huang and A. V. Savkin, "Deployment of charging stations for drone delivery assisted by public transportation vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 15043–15054, Sep. 2022.

[7] M. Perreault and K. Behdinan, "Delivery drone driving cycle," *IEEE Trans. Veh. Technol.*, vol. 70, no. 2, pp. 1146–1156, Feb. 2021.

[8] A. Noormohammadi-As, O. Esrafilian, M. A. Arzati, and H. D. Taghirad, "System identification and H_∞-based control of quadrotor attitude," *Mech. Syst. Signal Process.*, vol. 135, pp. 1–16, 2020.

[9] B. J. Emran and H. Najjaran, "A review of quadrotor: An underactuated mechanical system," *Annu. Rev. Control*, vol. 46, pp. 165–180, 2018.

[10] V. Kangunde, R. S. Jamisola, and E. K. Theophilus, "A review on drones controlled in real-time," *Int. J. Dyn. Control*, vol. 46, pp. 1832–1846, 2021.

[11] D. J. Almakhles, "Robust backstepping sliding mode control for a quadrotor trajectory tracking application," *IEEE Access*, vol. 8, pp. 5515–5525, 2020.

[12] V. P. Tran, F. Santoso, M. A. Garratt, and I. R. Petersen, "Fuzzy self-tuning of strictly negative-imaginary controllers for trajectory tracking of a quadcopter unmanned aerial vehicle," *IEEE Trans. Ind. Electron.*, vol. 68, no. 6, pp. 5036–5045, Jun. 2020.

[13] M. Jafari and H. Xu, "Intelligent control for unmanned aerial systems with system uncertainties and disturbances using artificial neural network," *Drones*, vol. 2, pp. 1–13, 2018.

[14] P. T. Jardine, S. N. Givigi, and S. Yousefi, "Leveraging data engineering to improve unmanned aerial vehicle control design," *IEEE Syst. J.*, vol. 15, no. 2, pp. 2595–2606, Jun. 2021.

[15] A. Ayyad, M. Chehadeh, M. I. Awad, and Y. Zweiri, "Real-time system identification using deep learning for linear processes with application to unmanned aerial vehicles," *IEEE Access*, vol. 8, pp. 122539–122553, 2020.

[16] A. Ayyad et al., "Multirotors from takeoff to real-time full identification using the modified relay feedback test and deep neural networks," *IEEE Trans. control Syst. Technol.*, vol. 30, no. 4, pp. 1561–1577, Jul. 2022.

[17] A. Y. Alkayas, M. Chehadeh, A. Ayyad, and Y. Zweiri, "Systematic online tuning of multirotor UAVs for accurate trajectory tracking under wind disturbances and in-flight dynamics changes," *IEEE Access*, vol. 10, pp. 6798–6813, 2022.

[18] P. K. Muthusamy, M. Garratt, H. Pota, and R. Muthusamy, "Realtime adaptive intelligent control system for quadcopter UAV with payload uncertainties," *IEEE Transaction Ind. Electron.*, vol. 16, no. 2, pp. 1641–1653, Feb. 2022.

[19] P. K. Muthusamy, "Intelligent control systems for unmanned aerial vehicle," Ph.D. dissertation, School of Engineering and IT, University of New South Wales, Australia, Jul. 2021.

[20] A. Aytaç and R. Hacıoğlu, "Model predictive control of three-axis gimbal system mounted on UAV for real-time target tracking under external disturbances," *Mech. Syst. Signal Process.*, vol. 138, 2020, Art. no. 106548.

[21] A. Aytaç, "Performance of metaheuristic optimization algorithms based on swarm intelligence in attitude and altitude control of unmanned aerial vehicle for path following," in *Proc. IEEE 4th Int. Symp. Multidisciplinary Stud. Innov. Technol.*, 2020, pp. 1–6.

[22] A. Aytaç, A. Özgür, and H. Rıfat, "Real-time control based on NARX neural network of hexarotor UAV with load transporting system for path tracking," in *Proc. IEEE 6th Int. Conf. Control Eng. Inf. Technol.*, 2018, pp. 1–6.

[23] E. Belge, A. Altan, and R. Hacıoğlu, "Metaheuristic optimization-based path planning and tracking of quadcopter for payload hold-release mission," *Electronics*, vol. 11, no. 8, 2022, Art. no. 1208.

[24] W. Gu, K. P. Valavanis, M. J. Rutherford, and A. Rizzo, "UAV model-based flight control with artificial neural networks: A survey," *J. Intell. Robotic Syst.*, vol. 100, pp. 1469–1491, 2020.

[25] D. Lim, H. Kim, and K. Yee, "Uncertainty propagation in flight performance of multirotor with parametric and model uncertainties," *Aerosp. Sci. Technol.*, vol. 122, 2022, Art. no. 107398.

[26] J. MorÉn and C. Balkenius, "Emotional learning: A computational model of the amygdala," *Cybern. Syst.*, vol. 32, no. 6, pp. 611–636, 2001.

[27] M. Jafari, H. Xu, and L. R. G. Carrillo, "A neurobiologically-inspired intelligent trajectory tracking control for unmanned aircraft systems with uncertain system dynamics and disturbance," *Trans. Inst. Meas. Control*, vol. 41, no. 2, pp. 417–432, 2019.

[28] C. M. Lin and C. C. Chung, "Fuzzy brain emotional learning control system design for nonlinear systems," *Int. J. Fuzzy Syst.*, vol. 17, no. 2, pp. 117–128, Jun. 2015.

[29] C. M. Lin and P. K. Muthusamy, "Intelligent brain emotional learning control system design for nonlinear systems," in *Proc. IEEE 11th Asian Control Conf.*, 2017, pp. 958–963.

[30] J. Zhang, F. Chao, H. Zeng, C.-M. Lin, and L. Yang, "A recurrent wavelet-based brain emotional learning network controller for nonlinear systems," *Soft Comput.*, vol. 26, pp. 3013–3028, 2021.

[31] P. K. Muthusamy, M. Garratt, H. Pota, J. Wang, and J. M. Kok, "Bidirectional fuzzy brain emotional learning control for aerial robots," in *Proc. IEEE Symp. Ser. Comput. Intell.*, 2018, pp. 146–153.

[32] C. T. Lin and C. S. G. Lee, *Neural Fuzzy Systems: A. Neuro-Fuzzy Synergism to Intelligent Systems*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1996.

[33] E. Lavretsky and K. A. Wise, *Robust and Adaptive Control With Aerospace Applications*. Berlin, Germany: Springer, 2013.

[34] F. Sabatino, "Quadrotor control: Modeling, nonlinear control design, and simulation," Ph.D. dissertation, KTH, School of Electrical Engineering (EES), Automatic Control, 2015.

[35] B. L. Stevens and F. L. Lewis, *Aircraft Control and Simulation*. Hoboken, NJ, USA: Wiley, 2016.
[36] E. Fresk and G. Nikolakopoulos, "Full quaternion based attitude control for a quadrotor," in *Proc. Eur. Control Conf.*, 2013, pp. 3864–3869.
[37] T. R. Beal, "Digital simulation of atmospheric turbulence for Dryden and von Kármán models," *J. Guid., Control Dyn.*, vol. 16, no. 1, pp. 132–138, 1993.
[38] "Quanser QDrone," Accessed: Dec. 1, 2021. [Online]. Available: https://www.quanser.com/products/qdrone/
[39] "IGMA ventilation industrial fans," Accessed: Dec. 1, 2021. [Online]. Available: http://igmaventilation.com

Praveen Kumar Muthusamy received the bachelor of engineering degree in electrical and electronics engineering from Anna University, Chennai, India, the M.Sc. degree in electrical engineering from Yuan Ze University, Taoyuan City, Taiwan, and the Ph.D. degree in aerospace engineering from UNSW Canberra, Australia.

He is currently working as a Postdoctoral Fellow with Khalifa University of Science and Technology, Abu Dhabi, UAE. His research interests include UAV, autonomous systems, robotics, and intelligent control systems.

Bhivraj Suthar received the B.E. degree from the College of Technology and Engineering, Rajasthan, India, the M.S. degree from the Indian Institute of Technology Delhi, Delhi, India, and the Ph.D. degree from the Korea University of Technology and Education, Cheonan, South Korea, in 2010, 2015, and 2020, respectively.

He is currently a Postdoctoral Researcher with the Department of Aerospace Engineering, Khalifa University of Science and Technology, Abu Dhabi, UAE. His research interests include robot design and control.

Rajkumar Muthusamy received the B.E. degree in electrical electronics engineering from Anna University, Chennai, India, the M.S. degree in electrical engineering with a gold medal for academic excellence from Yuan Ze University, Taoyuan City, Taiwan, and the D.Sc.(Tech) degree in automation, systems, and control engineering from Aalto University, Espoo, Finland, in 2009, 2013, and 2018, respectively.

He is a Senior Robotics Scientist with Dubai Future Labs, Dubai, UAE, currently focusing on the development of autonomous robotic systems and solutions for mobile manipulation and industrial applications.

Mr. Muthusamy was a recipient of the Finnish Engineering Award in 2020.

Matthew Garratt received the B.E. degree in aeronautical engineering from Sydney University, Sydney, Australia, in 1990, a graduate diploma in applied computer science from Central Queensland University, Rockhampton, Australia, in 1997, and the Ph.D. degree in robotics from the Australian National University, Canberra, Australia, in 2008.

He is a Professor and the Deputy Director (Defence & Security) of the UNSW AI Institute, UNSW Sydney, Australia. His research interests include sensing, guidance, and control for autonomous systems with particular emphasis on biologically inspired and computational intelligence approaches.

Hemanshu Pota received the B.E. degree from Sardar Vallabhbhai Regional College of Engineering and Technology (SVRCET), Gujarat, India, the M.E. degree from the Indian Institute of Science (IISc) Bangalore, India, and the Ph.D. degree from the University of Newcastle, Callaghan, Australia, in 1979, 1981, and 1985, respectively, all in electrical engineering.

He is currently an Associate Professor with the University of New South Wales, Canberra, Australia. His research interests include power system dynamics and control, and modeling and control of mechanical systems such as flexible structures, acoustical systems, and UAVs.

Lakmal Seneviratne received the Ph.D. degree in mechanical engineering from King's College London, London, U.K., in 1986.

He is a Professor of mechanical engineering and the Founding Director of the Centre for Autonomous Robotic Systems (KUCARS), Khalifa University of Science and Technology, Abu Dhabi, UAE. Prior to joining Khalifa University, he was a Professor of mechatronics, the Founding Director of the Centre for Robotics Research, and the Head of the Division of Engineering, King's College London, London, U.K. He has authored or coauthored more than 400 peer reviewed publications.

Prof. Seneviratne is a Member of the Mohammed Bin Rashid Academy of Scientists in the UAE.

Yahya Zweiri (Member, IEEE) received the Ph.D. degree in mechanical engineering from King's College London, London, U.K., in 2003.

He is currently a Professor with the Department of Aerospace Engineering and the Director of the Advanced Research and Innovation Center, Khalifa University, UAE. Over the past two decades, he has actively participated in defense and security research projects at institutions such as the Defense Science and Technology Laboratory, King's College London, and the King Abdullah II Design and Development Bureau in Jordan. Dr. Zweiri has a prolific publication record, with more than 130 refereed journals and conference papers, as well as ten filed patents in the USA and U.K. His primary research focus centers around robotic systems for challenging environments, with a specific emphasis on applied AI and neuromorphic vision systems.