Hindawi
Complexity
Volume 2019, Article ID 6460156, 20 pages
https://doi.org/10.1155/2019/6460156

Research Article

Wind and Payload Disturbance Rejection Control Based on Adaptive Neural Estimators: Application on Quadrotors

Jesús Enrique Sierra and Matilde Santos

Computer Science Faculty, Complutense University of Madrid, 28040 Madrid, Spain

Correspondence should be addressed to Jesús Enrique Sierra; jesier01@ucm.es and Matilde Santos; msantos@ucm.es

Received 29 July 2018; Revised 7 December 2018; Accepted 12 December 2018; Published 3 January 2019

Academic Editor: Dan Selişteanu

In this work, a new intelligent control strategy based on neural networks is proposed to cope with some external disturbances that can affect quadrotor unmanned aerial vehicles (UAV) dynamics. Specifically, the variation of the system mass during logistic tasks and the influence of the wind are considered. An adaptive neuromass estimator and an adaptive neural disturbance estimator complement the action of a set of PID controllers, stabilizing the UAV and improving the system performance. The control strategy has been extensively tested with different trajectories: linear, helical, circular, and even a lemniscate one. During the experiments, the mass of the UAV is triplicated and winds of 6 and 9 in Beaufort's scale are introduced. Simulation results show how the online learning of the estimator increases the robustness of the controller, reducing the effects of the changes in the mass and of the wind on the quadrotor.

1. Introduction

In recent years, new and valuable applications of unmanned aerial vehicles (UAV) have emerged in different sectors such as defense, security, construction, agriculture, entertainment, and shipping [1–3]. These and other applications demand the design of efficient and robust controllers for those autonomous vehicles. That is why the modelling and control of these complex and unstable systems still motivate the research and interest of the scientific community [4–11].

Nevertheless, the modelling and control of quadrotor vehicles are not an easy task. The complexity comes from the randomness of the airstreams and of the exogenous forces, the high nonlinearity dynamics, the coupling between the internal variables, the uncertainty of the measurements, etc. These factors make the techniques based on artificial intelligence a promising approach for the identification and control of these systems [12].

Moreover, these intelligent strategies are especially interesting when the model parameters change while the system is working [13]. For example, the total mass will undergo variations when the vehicle is performing logistic tasks, since the mass depends on the loads that are shipped.

There are few works that study the effect of the payload variation on the quadrotor dynamics and that take it into account. In [14], an adaptive control is used to mitigate the impact of the parameter variation by estimating them under guaranteed performance. In [15], least square and gradient methods are implemented for adaptive parameter estimation, which are used to update the control output to the current UAV mass and inertia moment. Recently, Wang [16] applied again the same strategy to estimate the variations in the payload and the effect of the wind gusts.

We propose to use estimators based on neural networks. They offer three main advantages with respect to other techniques: the fact that knowledge about the internal structure of the system to be estimated is not necessary, unlike other estimation techniques such as PEM (parametric error models), Hammerstein-Weiner, and Volterra; its online learning ability; and the easy parallelization. Even more, using adaptive neural networks allows the online learning of the estimation not to be so biased by the selection of the training data. This is a relevant advantage against other offline estimation methods which are very sensitive to the dataset used during the training.

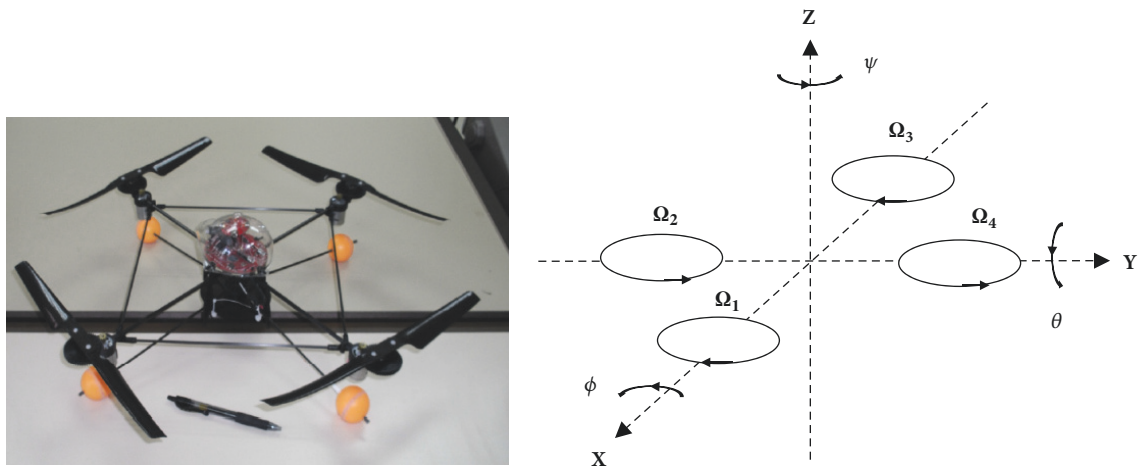

Figure 1: A quadrotor vehicle (left) and UAV's coordinate system (right).

There are other papers more focused on a closely related problem, the UAV stabilization and path tracking when its center of gravity changes, and what happens by the manipulation of suspended loads by cables. In [17], using the flatness property, a trajectory generation method is presented that enables finding nominal trajectories with various constraints that result in minimal load swing. Also, the same authors present a very interesting cooperative control strategy to manipulate suspended loads by several quadrotors at the same time [18]. In [19], a controller based on a set of connected PD regulators is used to tackle this issue. An adaptive tracking controller based on output feedback linearization is used in [20]; this controller compensates for dynamical changes in the center of gravity of the quadrotor. In [21], this issue is again addressed using the dynamic programming approach. In [22], the complexity of the aerial vehicle is incremented by considering the elasticity of the cable in the system equations.

Other studies are focused on the rejection of wind disturbances. In [23], Lyapunov-based observers are used to estimate the external force disturbances. In [24], a control strategy based on sliding mode and adaptive control techniques is proposed to deal with slow and fast time-varying wind conditions. In [25], a switching model predictive attitude controller for an unmanned aerial vehicle subject to atmospheric disturbances is presented. In [26], a nonlinear adaptive state feedback controller for thrust and torque actuation is designed, so that it guarantees global convergence of the closed-loop path following in the presence of constant wind disturbances. In [27], a sliding mode control driven by sliding mode disturbance observer (SMC-SMDO) approach is used to design a robust flight controller for a small quadrotor.

So far, the studies are mainly focused on payload variations and on wind disturbances rejection. Only few recent papers have been found that address both problems at the same time, such as in [16]. This challenging issue demands further research. Furthermore, although there are some papers where neural networks are applied to model quadrotors [28, 29], and to control them [30–33], these techniques have not been explored to solve this specific research problem.

Therefore, in this work, we propose the design of an intelligent control strategy based on neural networks to cope with these external disturbances, payload changes and wind, that can affect quadrotor dynamics. The final goal is to stabilize the UAV and to improve the system performance. The control strategy has been extensively tested by simulation with different trajectories. Indeed, the online learning estimator that has been implemented increases the robustness of the controller, reducing the effects produced for these variations.

The paper is organized as follows. In Section 2, the dynamic behavior of the system is described. The design of the controller and the adaptive neural estimators are presented in Section 3. Simulation results are discussed in Section 4. The document ends with the conclusions and future works.

2. System Model

A quadrotor vehicle is composed by four perpendicular arms, each one with a motor and a propeller (Figure 1, left). The four motors drive the lift and direction control.

The UAV absolute position is described by three coordinates, (x, y, z), and the attitude is given by the three Euler's angles (ϕ, θ, ψ), under conditions: $(-\pi \leq \psi < \pi)$ for the yaw angle, $(-\pi/2 \leq \phi < \pi/2)$ for the roll, and $(-\pi/2 \leq \theta < \pi/2)$ for the pitch, all angles in radians.

The system is based on two couples of propellers opposed to each other, (1, 3) and (2, 4) (Figure 1, right). To keep the balance of the system, one pair of motors turns clockwise while the other one spins counterclockwise. The increment of the speed of rotor 3 with respect to rotor 1 produces a positive pitch ($\theta > 0$), while increasing the speed of rotor 4 regarding rotor 2 produces a positive roll ($\phi > 0$). The increment of the speeds of rotors 1 + 3 with respect to rotors 2 + 4 produces a positive yaw ($\psi > 0$).

Regarding the UAV modelling, in the related literature, there are typically two approaches to obtain the mathematical model of the quadrotor: the Lagrangian method and the one based on the representation of the translational and angular dynamics. As we need to develop a control-oriented model, we have used the latter, that is, the Newton-Euler method, which describes the dynamic systems in terms of force and momentum.

The Newton dynamic equation states that the sum of forces applied in a system is equal to the variation of the lineal momentum:

$$\sum F_i = \frac{d}{dt}(mv). \tag{1}$$

While the mass is constant, this equation is equivalent to

$$\sum F_i = m\dot{v}. \tag{2}$$

In our case, the forces F_i are the vector of forces T produced by the rotors and by the gravity, considering an Earth reference system; thus, the previous equations give the translational dynamic (3) considering the assumption $m\dot{v} \gg \dot{m}v$, implicitly assumed in other papers [33]:

$$m\dot{v} = RT - mge_3, \tag{3}$$

where m is the mass of the quadrotor in Kg, R is the rotation matrix which is dimensionless, g is the gravitational acceleration in $\mathrm{m{\cdot}s^{-2}}$, T is a vector of forces in N, and $e_3 = [0, 0, 1]^T$ is a unit vector which describes the rotor orientation.

The Euler dynamic equation expresses that the sum of torques is equivalent to the variation of the angular moment:

$$\sum \mu_i = \frac{d}{dt}(J \times w) = \frac{d}{dt}J \times w + J \times \frac{d}{dt}w. \tag{4}$$

This sum of torques represents the vector of torques τ (N·m) in the three axes produced by the rotors, that is, the angular dynamic of system (5) [34], considering the assumption $J \times (d/dt)w \gg (d/dt)J \times w$, also implicitly assumed in other papers [33]:

$$\tau = J\dot{\omega} + \omega \times J\omega, \tag{5}$$

where J is the inertia tensor in $\mathrm{Kg{\cdot}m^2}$ (6), ω is the angular velocities vector in rad/s, and $\times$ represents the vector product:

$$J = \begin{pmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{pmatrix}. \tag{6}$$

The vectors τ (7) and T (8) are a function of the velocities of the propellers:

$$\tau = \begin{pmatrix} bl\left(\Omega_4^2 - \Omega_2^2\right) \\ bl\left(\Omega_3^2 - \Omega_1^2\right) \\ d\left(\Omega_2^2 + \Omega_4^2 - \Omega_1^2 - \Omega_3^2\right) \end{pmatrix} \tag{7}$$

$$T = \begin{pmatrix} 0 \\ 0 \\ b\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \end{pmatrix}, \tag{8}$$

where b is the thrust coefficient in $\mathrm{N{\cdot}s^2}$, d is the drag coefficient, l is the longitude of each arm in m, and $\Omega_1, \ldots, \Omega_4$ are the velocities in rad/s of the rotors 1 to 4, respectively.

To simplify the calculations, instead of using the speed of the rotors, it is possible to define a set of control signals u_1, u_2, u_3 y u_4 as follows (9):

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & 0 & 1 \\ -1 & 0 & 1 & 0 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix}. \tag{9}$$

This matrix is invertible, so it is possible to generate speed references for the rotors from a set of control signals.

Finally, from (1) to (9), the following system of equations is derived:

$$\ddot{\phi} = \frac{\dot{\theta}\dot{\psi}\left(I_y - I_z\right)}{I_x} + \left(\frac{lb}{I_x}\right)u_2 \tag{10}$$

$$\ddot{\theta} = \frac{\dot{\phi}\dot{\psi}\left(I_z - I_x\right)}{I_y} + \left(\frac{lb}{I_y}\right)u_3 \tag{11}$$

$$\ddot{\psi} = \frac{\dot{\phi}\dot{\theta}\left(I_x - I_y\right)}{I_z} + \left(\frac{d}{I_z}\right)u_4 \tag{12}$$

$$\ddot{X} = -\left(\sin\theta\cos\phi\right)\left(\frac{b}{m}\right)u_1 \tag{13}$$

$$\ddot{Y} = \left(\sin\phi\right)\left(\frac{b}{m}\right)u_1 \tag{14}$$

$$\ddot{Z} = -g + \left(\cos\theta\cos\phi\right)\left(\frac{b}{m}\right)u_1. \tag{15}$$

The constants of (10) to (15) that are used during the simulations are listed in Table 1. The values have been extracted from [35].

3. Control Strategy Design

3.1. First Approach: Adaptive Inverse Controller. There are different control strategies based on neural networks [29–34]. In a first approach [32], a variant of the generalized learning algorithm (GLA) was used to control this system (Figure 2). The procedure was as follows.

The first step is the application of the GLA algorithm to offline train the neural network in order to identify the inverse dynamic of the plant (Figure 2). Once the network has been offline trained, it is placed in cascade connection with the plant and a PID controller. In Figure 3, this control strategy is shown for the altitude variable of the UAV. Then, the configuration of the network is online refined. In order to do this, during each control interval, two processes are sequentially applied to the network (first the simulation, later the online learning):

(1) **Simulation:** The output of the PID, $PID_{OUT}(t_i)$, feeds one of the inputs of the artificial neural network; the

Table 1: Parameter values of the model.

Parameter	Description	Value/Units
l	Longitude of an arm	0.232 m
m	Mass of the quadrotor	0.52 Kg
d	Drag coefficient	$7.5e^{-7}$ N.m.s^2
b	Thrust coefficient	$3.13e^{-5}$ N.s^2
I_x	Inertia in X	$6.228e^{-3}$ Kg.m^2
I_y	Inertia in Y	$6.225e^{-3}$ Kg.m^2
I_z	Inertia in Z	$1.121e^{-2}$ Kg.m^2
ρ_{air}	Density of the air	1.2 Kg/m^3
A	Area in the direction of the wind	0.0186 m^2
Cd	Wind drag coefficient	1

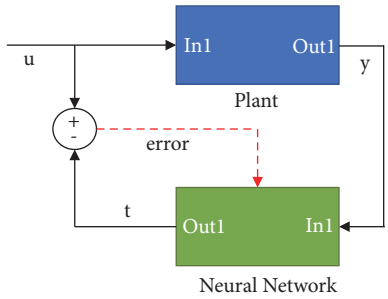

Figure 2: Offline training to identify the plant inverse dynamic.

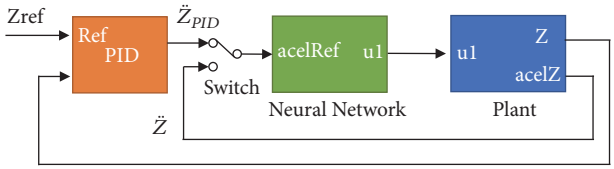

Figure 3: Adaptive altitude neurocontrol strategy.

rest of the inputs are past values of the plant output, $PLANT_{OUT}(t_i - j * Ts)$. The network generates the control input, u_1 (16), which is the input of the plant (Figure 3, switch in the upper position):

$$u_1(t_i) = f_{NET}(PID_{OUT}(t_i), PLANT_{OUT}(t_i - j * Ts), par_{NET}(t_i - Ts)) \quad j = 1 \ldots (N_{inputs} - 1), \tag{16}$$

where Ts is the sampling time in seconds, par_{NET} denotes the configuration parameters of the network, and N_{inputs} means the number of inputs of the neural network.

(2) **Online learning:** The neural network is trained again with the current and previous outputs of the plant, in order to generate the control output, u_1, obtaining the new configuration parameters par_{NET} (17). The network input dataset is made up of the past values of the plant output, $PLANT_{OUT}(t_i - j * Ts)$. The output dataset is the current value of the plant input $u_1(t_i)$ (Figure 3, switch in the lower position):

$$par_{NET}(t_i) = f(PLANT_{OUT}(t_i - j * Ts), u_1(t_i), par_{NET}(t_i - Ts)), \quad j = 0 \ldots (N_{inputs} - 1). \tag{17}$$

In order to test the validity of this first approach, we firstly focused only on the altitude control. UAVs are normally provided with accelerometers, so it is assumed that the acceleration in the z-axis ($\ddot{Z}$) is available. The network must be able to simulate the control signal u_1 by using acceleration measurements.

In this example, PID_{OUT} (16) is the reference of the acceleration in the z-axis, $\ddot{Z}_{PID}$; $PLANT_{OUT}$ is the acceleration in the z-axis in m·s^{-2}, and u_1 is the control signal. Thanks to the artificial neural network, the PID does not need to include the plant gain. The network is able to learn the plant gain and work with it. In other words, with this approach, it is not necessary to know the system parameters to control it [32].

3.2. New Advanced Strategy: Controller with Adaptive Neural Estimators. The generalized learning algorithm proposed in the previous section is especially useful when there is not any knowledge about the real dynamic of the plant (black-box system). However, when some knowledge about the system dynamics is available, even if it is not complete or accurate, it is positive to include it in some way in the controller. In this section, the system equations ((10) to (15)) are introduced in the controller, and the neural networks are focused on the uncertain terms: the mass, the wind disturbance, and the nonmodelled dynamics.

The control system has been designed to track trajectories defined by tuples of three coordinates $(Xref, Yref, Zref)$.

The model of the whole system with the controllers is shown in Figure 4. For the sake of simplicity, several ports that route the signals haven been introduced to reduce the number of lines in the diagram.

The UAV model control inputs are the four control signals, u_i, $i = 1, .., 4$, that represent the power of the rotors. Four main controllers are defined to obtain these model inputs, the controllers of the X, Y, and Z coordinates, and

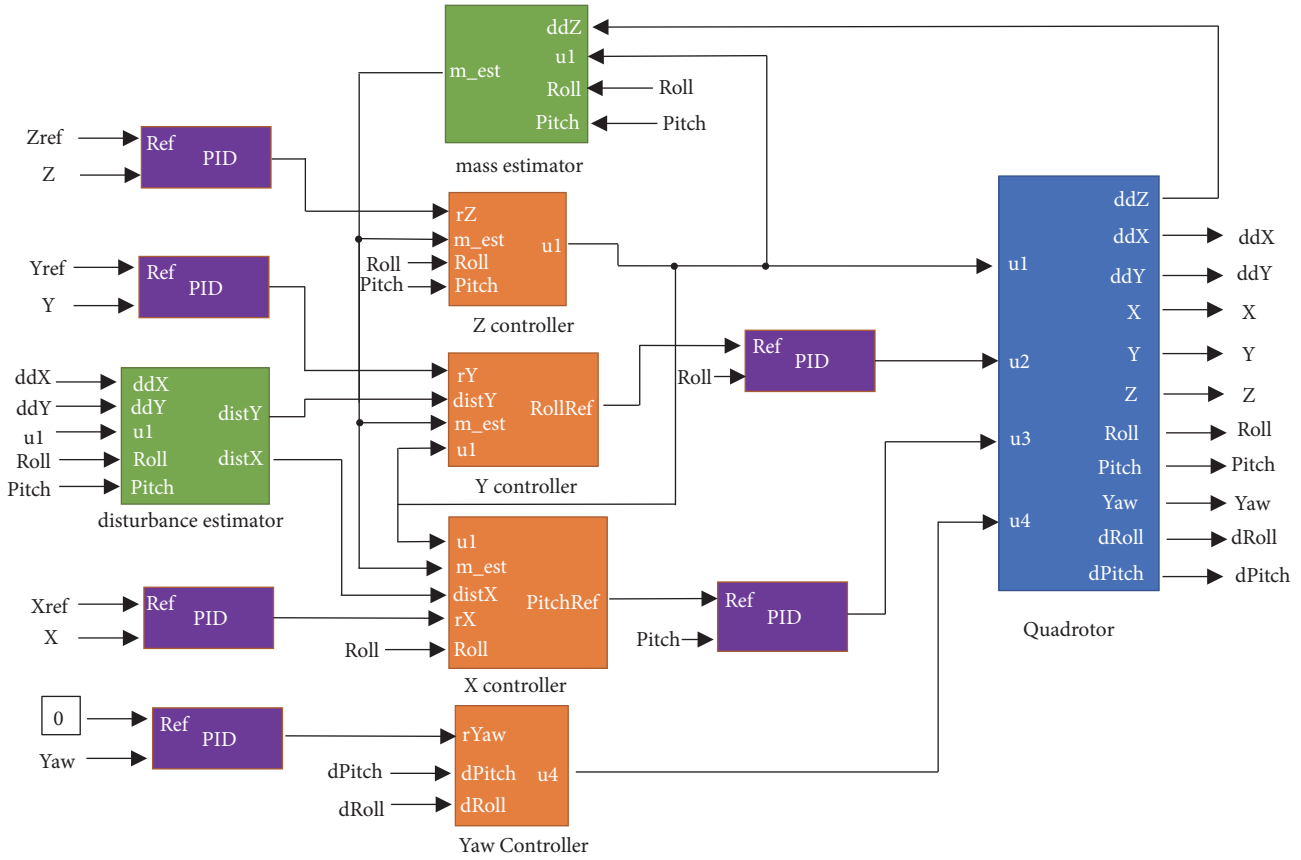

Figure 4: Control system with mass and disturbance estimators.

another for the yaw angle. This is because the pitch and roll angles are used to track the *Xref* and *Yref* reference coordinates. The control of Z is carried out by the control signal u_1. The Y controller generates the roll reference, and the tracking of the roll angle is performed by the control signal u_2. In the same way, the controller of X generates the pitch angle reference value, and control signal u_3 will be in charge of getting this value. The control signal u_4 is used to stabilize the yaw angle around zero.

The aim of the PIDs controllers that appear in Figure 4 is to generate the acceleration references to make the attitude error $(\phi_r - \phi, \theta_r - \theta, \psi_r - \psi)$ and the tracking error converge to zero. The rest of the controllers are used to compensate the nonlinearities of the system. The PID tuning parameters have been set by trial and error and the values of the gains are shown in Table 2.

A mass estimator and a wind disturbance estimator have been added (Figure 4). They are implemented by neural networks. These estimators are used to compensate the variations of the system's mass and the influence of the wind disturbances. These adaptive neural estimators feed the inputs of the X, Y, and Z controllers.

The controllers are defined as follows.

The performance of the Z controller is given by

$$u_1(t_i) = \begin{cases} (rZ(t_i) + g)\left(\dfrac{m_{est}(t_{i-1})}{b}\right) & \cos\theta_{i-1}\cos\phi_{i-1} = 0 \\ (rZ(t_i) + g)\left(\dfrac{m_{est}(t_{i-1})}{b}\right)\left(\dfrac{1}{\cos\theta_{i-1}\cos\phi_{i-1}}\right) & \cos\theta_{i-1}\cos\phi_{i-1} \neq 0, \end{cases} \tag{18}$$

where rZ is the output of the Z PID controller, m_{est} is the estimation of the mass in Kg, $(\phi_{i-1}, \theta_{i-1})$ represents the roll and pitch signals at time t_{i-1}, and the rest of the parameters and variables have been previously defined. In (18), it is possible to see how there is a discontinuity at $\cos\theta\cos\phi = 0$ that has been taken into account.

Table 2: PID parameters for each variable.

Signal	Kp	Kd	Ki
X	8	8	0
Y	8	8	0
Z	2	2	0.9
Roll	8	8	0
Pitch	8	8	0
Yaw	8	8	0

The Y controller is defined by

$$\phi_R(t_i) = \begin{cases} \phi_R(t_{i-1}) & u_1(t_i) = 0 \\ \operatorname{asin}\left(\left(rY(t_i) + distY_{est}(t_{i-1}) \right) \left(\frac{m_{est}(t_{i-1})}{bu_1(t_i)} \right) \right) & u_1(t_i) \neq 0, \end{cases} \tag{19}$$

where ϕ_R is the reference of the roll PID controller in rad, rY is the output of the Y PID controller in $\mathrm{m{\cdot}s^{-2}}$, $distY_{est}$ is the estimation of the wind disturbance in the y-axis in $\mathrm{m{\cdot}s^{-2}}$, and the rest of the parameters and variables have been already defined. The discontinuity at $u_1(t_i) = 0$ has been considered.

The control of the X coordinate is given by the expression:

$$\theta_R(t_i) = \begin{cases} \theta_R(t_{i-1}) & u_1(t_i)\cos\phi_{i-1} = 0 \\ -\operatorname{asin}\left(\left(rX(t_i) + distX_{est}(t_{i-1}) \right) \left(\frac{m_{est}(t_{i-1})}{bu_1(t_i)\cos\phi_{i-1}} \right) \right) & u_1(t_i)\cos\phi_{i-1} \neq 0, \end{cases} \tag{20}$$

where θ_R is the reference of the pitch PID controller, rX is the output of the X PID controller in $\mathrm{m{\cdot}s^{-2}}$, $distX_{est}$ is the estimation of the wind disturbance in the X coordinate in $\mathrm{m{\cdot}s^{-2}}$, and the rest of the parameters and variables have been already cited. Again, there is a discontinuity at $u_1(t_i)\cos\phi_{i-1} = 0$.

3.3. Adaptive Neural Estimator for Disturbances. The control scheme proposed in Figure 4 of this paper uses three different estimations: the estimation of the total mass of the system, the estimation of the disturbances in the X coordinate, and the estimation of the disturbances in the Y coordinate. These approximations, as may be observed in (18)-(20), feed the inputs of the different nonlinear controllers in order to reject the effect of the changes in the mass (m_{est}) and the wind external disturbances ($distX_{est}$ and $distY_{est}$).

These parameters that affect the dynamic of the UAV must be estimated if there are no sensors that could measure these disturbances, as it is the case. Although the estimation of each one of these variables (m_{est}, $distX_{est}$, and $distY_{est}$) has been implemented in a different function block, the inner structure of all of them is the same. They differ in some parameters of the configuration and the input and output signals. Thus, they can be jointly explained.

The **estimator** is based on an artificial neural network with online learning. Thus, there is one neural network to, let say, model each of the three parameters or disturbances considered. The following figure represents its generic structure (Figure 5). It is based on the diagram shown in [28], but in this case a new parameter model has been included. Furthermore, in this case, the inputs of the neural network are the outputs of the UAV, and its output are the output of the parameter model (when in [28], the output of the neural network is fed by the outputs of the UAV).

The parameter model receives the inputs and outputs of the UAV. The output of the parameter model is used as target output of the neural network during the training. The output of the proposed estimator is always the output of the neural network. The parameter model is needed because we are using supervised artificial networks and the inputs, and their corresponding outputs must be known.

We will use one parameter model for the mass, one parameter model for the wind disturbance in the x-axis, and another for the y-axis. The complete process is further explained below.

Each element of the dataset used to offline train the network is composed of the following:

(i) Target: Parameter (disturbance) value calculated with the model at instant t_i, param(t_i)

```
Net= configureNet ()
Dataset= ∅
For i=1 to toff        #Generating the dataset for off-line learning
     inputNet =[out1(i-1),... out1(i-n1), out2(i-1),..., outM (i-1).... outM (i- nM )]
     ParM (i)=model(out1(i-1),....., outM (i-1)])
     Element = { parM (i), inputNet }
     Dataset=Dataset ∪ Element
endFor
Net= offlineTraining (Net, Dataset)
For i= toff to tend      #on-line learning
     inputNet =[out1(i-1),... out1(i-n1), out2(i-1),..., outM (i-1).... outM (i- nM )]
     ParS (i)=simulate(Net, inputNet )
     [out1(i),....., outM (i)]= executeUAV ( ParS (i))          #Execute UAV + Controller
     ParM (i)=model(out1(i-1),....., outM (i-1)])
     If ∃ ParM(i) then
          Element= { parM (i), inputNet }
          Net= onlineTraining (Net, Element)
     endIf
endFor
```

Pseudocode 1

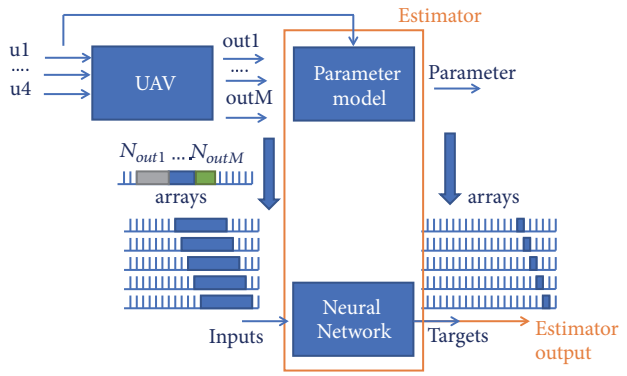

Figure 5: Neural estimator structure and configuration.

(ii) Network inputs: for each output signal i, the previous N_i values to t_i are collected and structured as an array. A theoretical example would be the following: At t_i=10 and for M = 3 outputs with a configuration Nout1=3, Nout2=2, Nout3=1, the network inputs are [out1(7), out1(8), out1(9), out2(8), out2(9), out3(9)] and the target is: param(10).

If the parameter cannot be calculated (division by zero, squared root of negative numbers, or any other singularity), that element is not included into the training dataset.

In this offline learning, the training dataset has as many elements as previous instants of time are considered. Longer time will normally produce better accuracy but will require more computational effort. The selection of the data for training is a delicate task. Indeed, the accuracy of the model depends on the data used to train the network. Another disadvantage of exclusively using offline approaches is that they do not capture the dynamics when it is changing over time [28]. For these reasons, in our proposal, we use adaptive learning for the mass and wind disturbance estimators' calculation.

Once the offline learning has finished, at each instant of time, a new training element is added {target, network inputs}, as has been previously explained. The target is obtained as the output of the model and the network inputs the output signals of the UAV. This new element is used to teach the networks how to adapt its parameters according to the new input. That is, the function to estimate the parameter is continuously changing over time.

Pseudocode 1 details the algorithm which relates to offline learning, the simulation, the online learning, and how the parameters are updated.

As it has been commented before, when it is not possible to measure or to calculate the parameter with the model, the input values are not defined, or there is a singularity in the calculation, the artificial neural network of our approach is very useful to estimate the disturbance. In these situations, we could say roughly speaking that the network generates new knowledge.

Once the inner structure of the estimator has been introduced, we explain the specific configurations for the mass and the wind disturbances.

Mass Estimation

(i) Target output: The mass is approximated with the model:

$$\widetilde{m}(t_i) = \frac{\ddot{Z}(t_{i-1}) + g}{u_1(t_{i-1}) \cos\theta_{i-1} \cos\phi_{i-1}}. \quad (21)$$

(ii) Network inputs:

(a) Acceleration in the z-axis $\ddot{Z}(t_{i-1})$

(b) Roll and pitch cosine angles multiplied $(\cos\theta_{i-1} \cos\phi_{i-1})$.

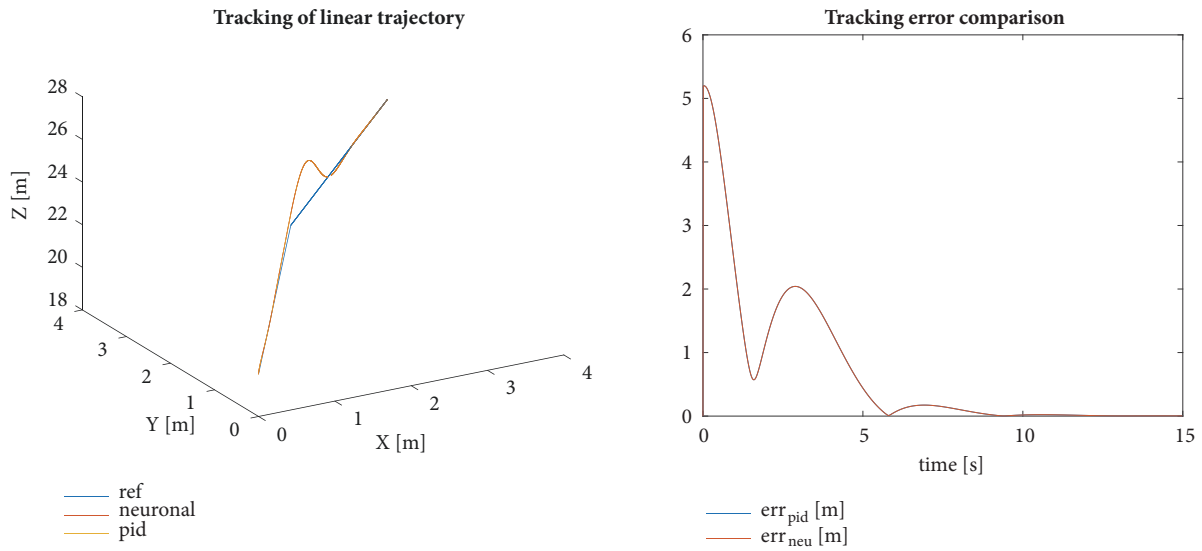

Figure 6: UAV tracking of a linear trajectory (left) and its tracking error (right).

Disturbance in X-Axis Estimation

(i) Target output: The disturbance is approximated with the model:

$$\widetilde{distX}(t_i) = -(\sin\theta_{i-1}\cos\phi_{i-1})\frac{bu_1(t_{i-1})}{m_{est}(t_{i-1})} - \ddot{X}(t_{i-1}). \quad (22)$$

(ii) Network inputs:

(a) Acceleration in the x-axis $\ddot{X}(t_{i-1})$
(b) Roll angle θ_{i-1}
(c) Pitch angle ϕ_{i-1}.

Disturbance in Y-Axis Estimation

(i) Target output: The disturbance is approximated with the model:

$$\widetilde{distY}(t_i) = \sin\phi_{i-1}\frac{bu_1(t_{i-1})}{m_{est}(t_{i-1})} - \ddot{Y}(t_{i-1}). \quad (23)$$

(ii) Network inputs:

(a) Acceleration in the y-axis $\ddot{Y}(t_{i-1})$
(b) Roll angle θ_{i-1}
(c) Pitch angle ϕ_{i-1}.

For the three different estimators, the artificial network implemented is a multilayer perceptron (MLP) with a hidden layer. The number of neurons of the hidden layer has been set to 20. The Levenberg-Marquardt algorithm with μ=0.001 has been used for the training. The network is offline trained for the first 2 seconds and then the online learning is applied for the remaining 13 seconds.

4. Results and Discussion

Simulation results have been obtained with Matlab/Simulink software. The duration of each simulation is 15 s. The controller is offline trained during the first 2 s. Then, the online learning algorithms are applied for the remaining 13 s. The sample time Ts is set to 10 ms.

In the experiments, in order to simplify the system, the yaw angle is set to 0, but it could be set to any other value by the user.

4.1. Trajectory Tracking without Disturbances. The control system has been first tested and validated with several trajectories without considering any disturbances. For each trajectory, the path followed by the UAV with adaptive neural estimators (red line), without them (yellow line), the reference (blue line), and the tracking error, are shown (Figures 6–11). The tracking error is calculated by the following equation:

$$tERR(t_i) = \sqrt{(X_i - Xref_i)^2 + (Y_i - Yref_i)^2 + (Z_i - Zref_i)^2}, \quad (24)$$

where $Xref_i, Yref_i, Zref_i$ are the references for X, Y, and Z in t_i, respectively.

The trajectories used to validate the controller are described by the following equations, with the corresponding parameters:

Linear:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.2t + 1 \\ 0.2t + 1 \\ 5 * step(t) + Z_0 \end{bmatrix}. \quad (25)$$

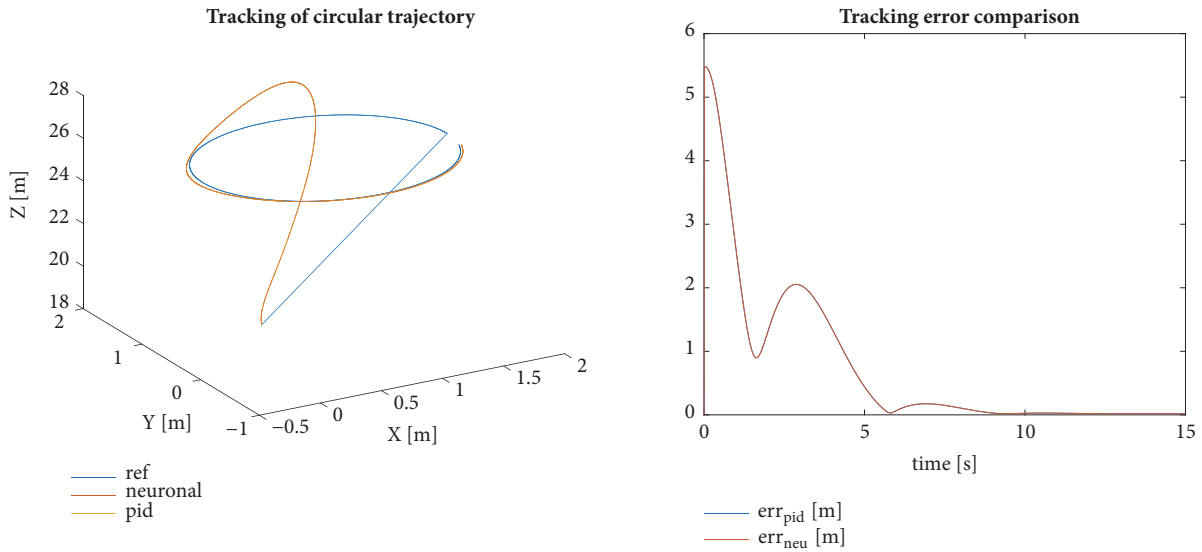

Figure 7: UAV tracking of a circular trajectory (left) and its tracking error (right).

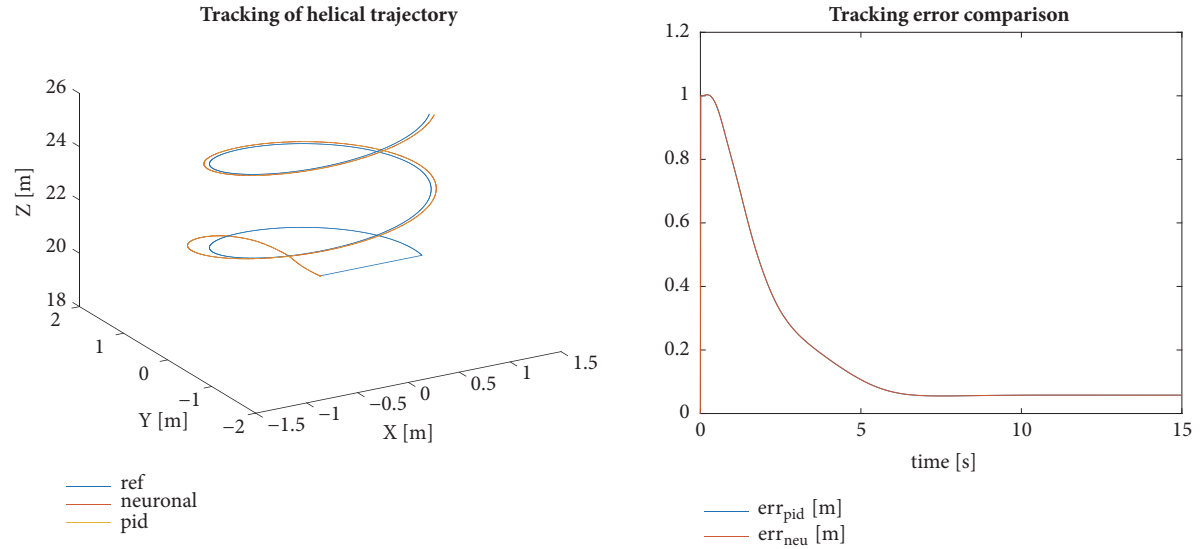

Figure 8: UAV tracking of a helical trajectory (left) and its tracking error (right).

Circular:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos(0.4t) + 1 \\ \sin(0.4t) + 1 \\ 5 * step(t) + Z_0 \end{bmatrix}. \tag{26}$$

Helical:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos(0.8t) \\ \sin(0.8t) \\ 0.4t + Z_0 \end{bmatrix}. \tag{27}$$

Cyclic helical:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos(1.6t) \\ \sin(1.6t) \\ \sin(0.4t) + Z_0 \end{bmatrix}. \tag{28}$$

Lemniscate:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \dfrac{\cos(0.4t)}{1 + \sin(0.4t)^2} \\ \dfrac{\cos(0.4t)\sin(0.4t)}{1 + \sin(0.4t)^2} \\ Z_0 \end{bmatrix}. \tag{29}$$

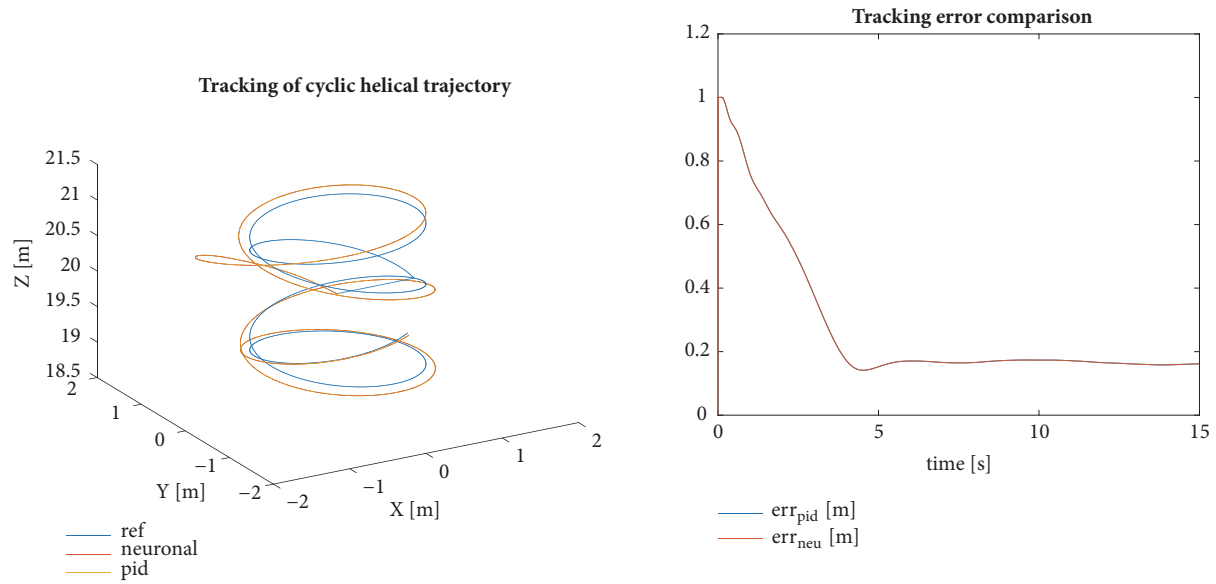

Figure 9: UAV tracking of a cyclic helical trajectory (left) and its tracking error (right).

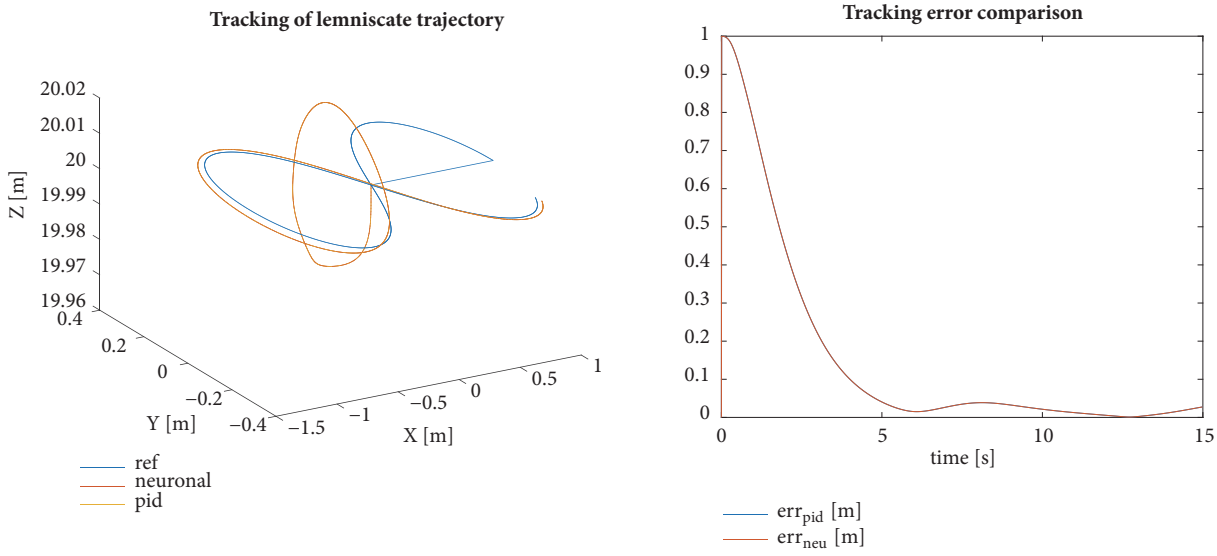

Figure 10: UAV tracking of a lemniscate trajectory (left) and its tracking error (right).

Helical lemniscate:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \dfrac{\cos(0.8t)}{1+\sin(0.8t)^2} \\ \dfrac{\cos(0.8t)\sin(0.8t)}{1+\sin(0.8t)^2} \\ 0.1t + Z_0 \end{bmatrix}. \quad (30)$$

In Figures 6–11, it is possible to observe how the tracking error decreases along the time. In t = 0, the error is high because the value of the reference starts in a very different value than the initial position of the system, that is, $(X_0, Y_0, Z_0) = (0, 0, 20)$. Then, the controller starts to work, and the tracking error is reduced.

For example, in the linear trajectory (Figure 6), the reference in t_0 is (1, 1, 25); thus, the initial tracking error at t=0 is $\sqrt{(1+1+25)}$ =5.196.

For the same reason, the initial error of the helical and lemniscate trajectories is much lower. The initial value reference is (1, 0, 20), and thus the initial error is 1.

In these figures, there is not any significant visual difference between the results with and without the neural network due to the fact that if the PIDs are well tuned and there are no changes either in the mass or in the external disturbances, as it is the case in this section, the neural networks do not provide relevant advantages. Nevertheless, the networks in the control scheme of Figure 4 show their full potential, when they must

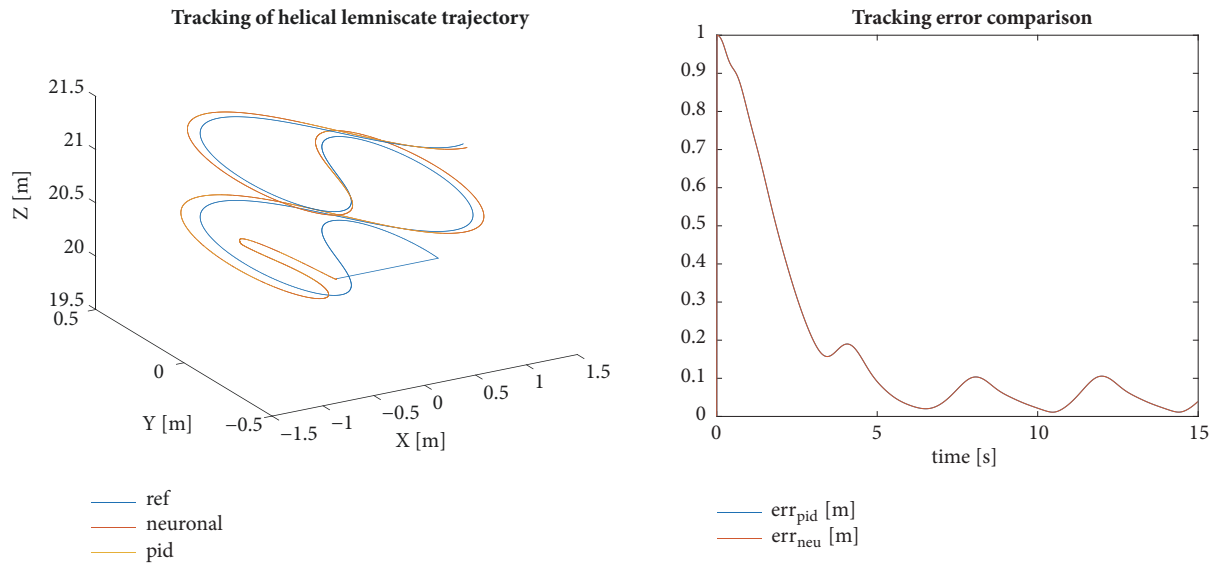

Figure 11: UAV tracking of a helical lemniscate trajectory (left) and its tracking error (right).

tackle disturbances, as it can be seen in the following sections. Consequently, the red and yellow lines seem to be overlapped.

Other appreciable result is that the stationary tracking error is higher when the trajectory has helical component. Predictably, the circular and helical trajectories show a cyclic nature.

As a conclusion, the controller performs a good tracking for a wide range of different trajectories. Thus, it can be said that the control strategy has been validated.

4.2. Control Robustness with Mass Variations

4.2.1. Mass Disturbance Model. Adding a payload in the quadrotor typically has three effects: the total mass of the system is increased, and the centre of the gravity can be modified and therefore also the inertia. In this work, we assume that the payload is an isotropic symmetric rigid solid attached to the UAV, not suspended, with dimensions much smaller than the dimensions of the quadrotor. The distance between the centre of gravity of the UAV and the centre of gravity of the load depends on the shape and the weight distribution of the manipulator and the shape and weight distribution of the load. In our experiment, we assume that this distance is zero. Under these circumstances, the effect of the inertia and centre of gravity variation can be neglected. Therefore, our paper is only addressing the effect of the mass variation.

When the UAV is performing a logistic task, there are two possible stable states regarding the mass: one is before the load is in contact with the quadrotor, and then only the mass of the UAV is considered; the second one is when the payload mass is part of the system, and the sum of both masses is then considered as an only system. Between these two states, there may be several profiles of mass variation depending on the grasping technology and the properties of the surface of the load.

We assume that the grasping and the load are nondeformable; thus, one step profile may be applied regarding the mass disturbance, as in other papers [16].

The mass variation is simulated by adding a new term $dist_m$ (34) to (13) to (15), resulting in (31) to (34). The modelling of the mass disturbance is a step function. The total mass then is triplicated at t = 4s (34) meaning that a payload has been attached to the UAV. After 4 sec, the total mass of the system is $m + m_L = 3m$, with m being the mass of the quadrotor and the mass of the load $m_L = 2m$. Even if it can be considered a simple approach of dealing with this disturbance, the final effect is well represented:

$$\ddot{X} = -(\sin\theta\cos\phi)\left(\frac{b}{(m + dist_m)}\right)u_1 \tag{31}$$

$$\ddot{Y} = (\sin\phi)\left(\frac{b}{(m + dist_m)}\right)u_1 \tag{32}$$

$$\ddot{Z} = -g + (\cos\theta\cos\phi)\left(\frac{b}{(m + dist_m)}\right)u_1 \tag{33}$$

$$dist_m = m_L \cdot step\,(t - 4)\,. \tag{34}$$

This experiment represents a possible situation while performing a logistic task, where the payload is heavier than the quadrotor itself.

4.2.2. Simulation Results with Mass Variation. Now, we test the control proposal with mass disturbances for the helical lemniscate trajectory because it is the most challenging one.

Figure 12, left, shows the reference trajectory in blue, the trajectory obtained by the controller with the neuromass estimator in red, and without the mass estimation in yellow (only PID). It is possible to observe how the trajectory obtained without the adaptive neural estimators (yellow line) moves away the reference even if it later comes closer to it.

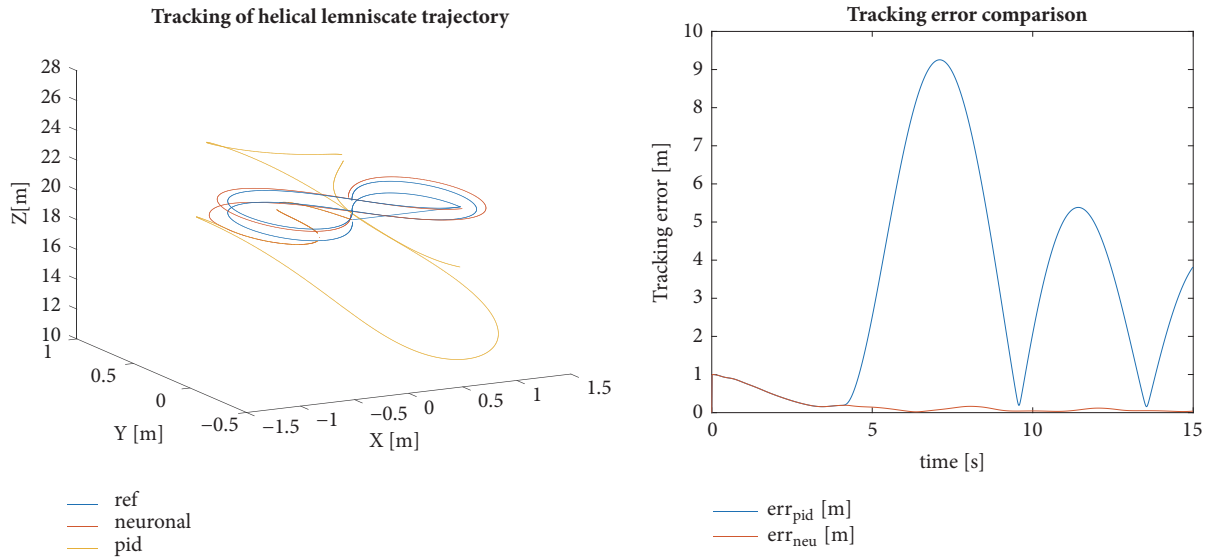

FIGURE 12: UAV tracking of a helical lemniscate trajectory with mass variation (left) and its tracking error (right).

At the right side of Figure 12, the tracking error is presented (red line with mass estimation and blue line without mass estimation). In both figures, it is possible to observe how the performance with the mass estimator is much better. The tracking error is mostly the same until the mass changes at t = 4 s, but then the error without mass estimation increases significantly.

The trajectory tracking of each coordinate has been represented in a separated figure to study the performance of the controller. Figure 13, top left, shows the tracking of the X coordinate, at the top, right, the tracking of the Y coordinate, and at the bottom the tracking of the Z coordinate, with reference (magenta line), the trajectory with mass estimator (red line), and the trajectory without mass estimator (blue line). For every coordinate, it is possible to see how the trajectory obtained by the controller with the mass estimator fits better the reference. Even though, the deviation mainly comes from the Z coordinate due to the fact that the acceleration in the z-axis is more sensible to changes in the mass.

Figure 14 shows the mass obtained by the neuromass estimator in this experiment (real mass in blue and the estimated mass in red). The estimate of the mass is very similar to the real one but slightly noisy. Clearly, it can be appreciated that the mass is triplicated following a step variation profile.

The controller is robust even with other types of variations in the mass like a sine function, for instance, if the mass changes according to

$$\begin{aligned} m(t) &= m_{model}\left(step(t) - step(t-4)\right) \\ &\quad + step(t-4)\left(2 \cdot m_{model} + m_{model} \cdot \sin\left(\frac{2\pi}{8}t\right)\right). \end{aligned} \tag{35}$$

The results of considering this mass variation profile are shown in Figure 15. The tracking error of each component is like the previous ones, but the system experiences a stronger change with the sinusoidal reference profile. In this case, the tracking error of the system without the adaptive neural estimators tends to increase; meanwhile, in the case of the step change, the tracking error decreased over the time.

Therefore, it can be seen how the controller with the neural estimator also works well with different mass change profiles.

Figure 16 shows the mass obtained by the neuromass estimator (red line) in this case and the corresponding real mass (blue line). The estimate of the mass is again very similar to the real one but slightly noisy but now the sinusoidal nature of the profile can be clearly observed.

The controller has been extensively tested for different trajectories and the MSE numerical results are summarized in Table 3 for the different trajectories and for each coordinate, with the neural estimator (neuro) and without it (PID). The last column shows the absolute error. The best result for each component and trajectory is boldfaced.

Table 4 compares the values of the maximum error in this experiment for the same cases with before.

In Tables 3 and 4, it is possible to observe that the controller with adaptive neural estimators provides less or equal tracking error for every tested trajectory. A general trend in the "PID column" (without mass disturbance estimation) is that the worst tracking error is obtained for the Z component. One of the reasons of this may be the fact that the maximum amplitude of the reference signal is in the z-axis.

4.3. Control Robustness with Wind Disturbances

4.3.1. Wind Disturbance Model. The effect of the wind in an UAV in flight can be considered as a drag force and torque

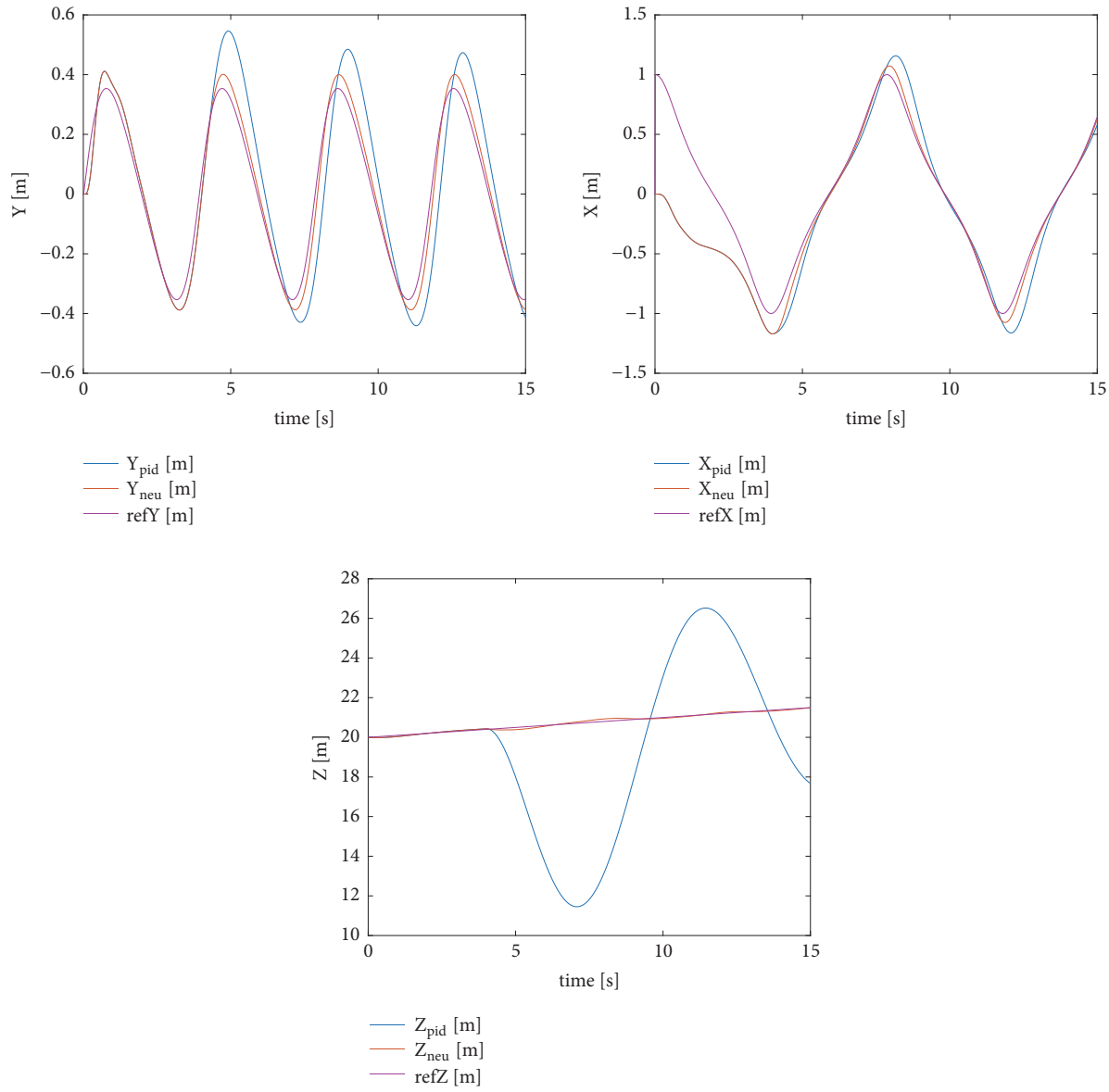

Figure 13: UAV tracking trajectory of X, Y, and Z coordinates, respectively, with mass variation.

depending on the aerodynamics properties of the object and the characteristics of the air flow.

The wind speed is a vector field; that is, its value may be different in each coordinate (x, y, z) of the space. For small UAVs like the one we are using in the experiments, we can assume a planar air flow, so the wind is the same in the region of the space where the UAV is flying. Under these circumstances, the wind influence in the torque can be neglected.

On the other hand, the following equation represents the drag force [36]:

$$\overline{F_D} = \frac{\overline{A} \cdot C_d \cdot \rho_{air} \cdot \overline{V}^2}{2}, \tag{36}$$

where C_d is the drag coefficient, which is usually determined experimentally and collects the complex dependency, A is the area exposed to the wind, ρ_{air} is the density of the air, and V is the relative velocity of the flying object with respect to the wind. Since the air flow is planar, it can be assumed that the drag force is fully exerted in the centre of the gravity of the vehicle.

In our work, this equation is divided by the mass. This way the expression is transformed to an acceleration ((40) to (43)) that can be easily introduced in the equations of the translational dynamics ((37) to (39)), where wind disturbances have been represented as $dist_wX$, $dist_wY$, and $dist_wZ$.

As the wind speed is a vectorial magnitude, in our experiment, we suppose the wind components X and Y are

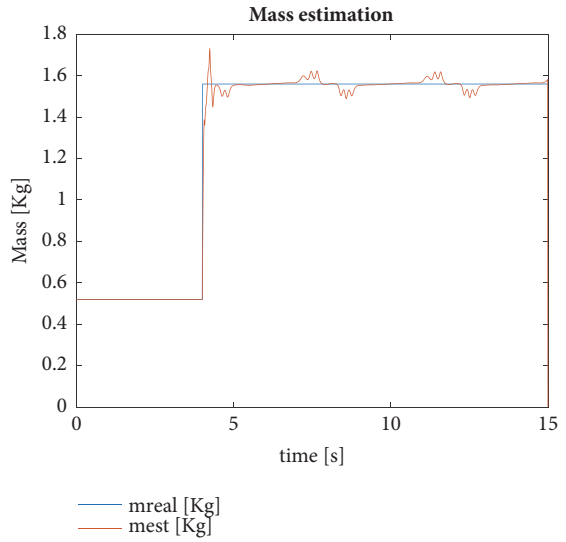

Figure 14: Mass obtained by the neural estimator.

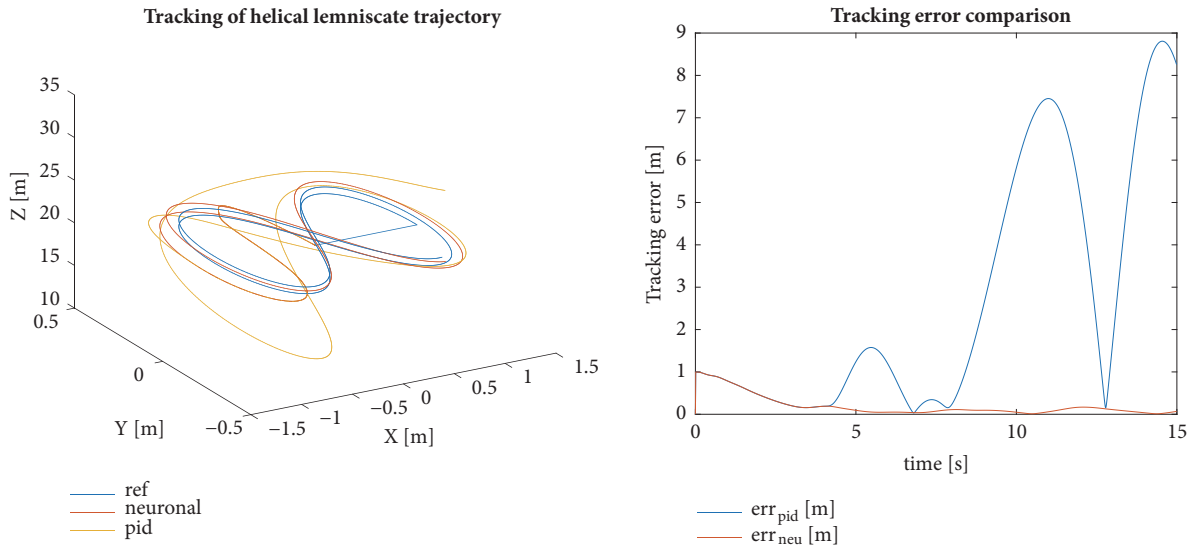

Figure 15: UAV tracking of a helical lemniscate trajectory with sinusoidal mass variation (left) and its tracking error (right).

twice the Z one. Due to this assumption, there appears a factor of 2 multiplying the wind speed in (41) and (42):

$$\ddot{X} = -dist_w X - (\sin\theta\cos\phi)\left(\frac{b}{m}\right)u_1 \tag{37}$$

$$\ddot{Y} = -dist_w Y + (\sin\phi)\left(\frac{b}{m}\right)u_1 \tag{38}$$

$$\ddot{Z} = -dist_w Z - g + (\cos\theta\cos\phi)\left(\frac{b}{m}\right)u_1. \tag{39}$$

In the free atmosphere, the wind is a balance between the Coriolis, centrifugal, and pressure forces acting on the air mass. But in the boundary layer, the mean wind velocity is also a function of the height [37], and it can be expressed as a logarithmic function. This fact is shown in the following:

$$v_w(Z) = v_{w(Z=20)} \cdot \frac{\log(Z/C)}{\log(20/C)} \tag{40}$$

$$dist_w X = \operatorname{sgn}(v_w(Z)) \cdot \rho_{air} \cdot A_x \cdot Cd \cdot \frac{\left(\dot{X} - 2 * v_w(Z)\right)^2}{(2m)} \tag{41}$$

$$dist_w Y = \operatorname{sgn}(v_w(Z)) \cdot \rho_{air} \cdot A_y \cdot Cd \cdot \frac{\left(\dot{Y} - 2 * v_w(Z)\right)^2}{(2m)} \tag{42}$$

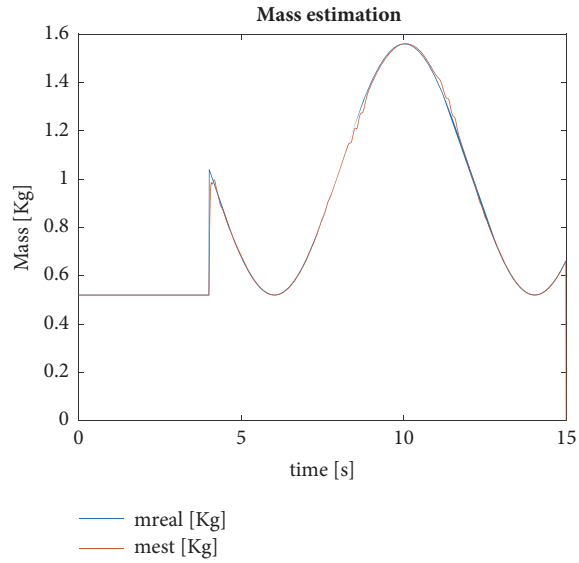

Figure 16: Mass obtained by the neural estimator with a sinusoidal mass change profile.

Table 3: Comparison of the MSE of the tracking error for different trajectories with mass variation.

Trajectory	MSE_X		MSE_Y		MSE_Z		MSE_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Linear	0.0707	0.0707	0.0708	0.0708	**1.6099**	21.9021	**0.6635**	3.8267
Circular	**0.2766**	0.2775	**0.0720**	0.0729	**1.6101**	21.9026	**0.7076**	3.8657
Helical	**0.0934**	0.1053	**0.0018**	0.0148	**0.0029**	18.2737	**0.1899**	3.2131
Cyclic helical	**0.1067**	0.2145	**0.0126**	0.1212	**0.0046**	17.5445	**0.2715**	3.1897
Lemniscate	**0.0902**	0.0916	**0.003**	0.0020	**0.0010**	18.2297	**0.1589**	3.1981
Helical lemniscate	**0.0923**	0.1060	**0.0018**	0.0166	**0.0034**	18.2335	**0.1956**	3.2087

$$dist_w Z = \operatorname{sgn}\left(v_w(Z)\right) \cdot \rho_{air} \cdot A_z \cdot Cd \cdot \frac{\left(\dot{Z} - v_w(Z)\right)^2}{(2m)}, \tag{43}$$

where $v_{w(Z=20)}$ is the wind speed at 20 m of altitude in m/s, $v_w(Z)$ is the wind speed at Z altitude in m/s, C is a constant related to the flight (in this experiment, the value is set to 1.5) which is dimensionless, ρ_{air} is the air density in Kg·m^3, A_x to A_z are the effective area of the quadrotor exposed to each component of the wind in m^2, Cd is the drag coefficient with respect to the wind which is dimensionless, $\dot{X}$, $\dot{Y}$, and $\dot{Z}$ are the velocities in the x-axis, y-axis, and z-axis in m/s, and sgn denotes the sign function.

The mean wind velocity can be considered constant during the experiments, but not its instant value. The most common assumption is to consider the wind turbulence as a stationary Gaussian random process [37]. Considering this fact, in our experiment, the wind speed is simulated by a step with Gaussian noise at t = 4 s. The SNR between the average wind and the noise is 10 dB. The average wind speed is 12 m/s in the z-axis and 24 m/s in the x-axis and y-axis. These values match numbers 6 and 9 in Beaufort's scale (strong breeze and strong gale) [38].

4.3.2. Simulation Results with Wind Variation. Figure 17, left, shows the tracking of the trajectory, the reference in blue, the trajectory with adaptive neural estimators in red, and without estimators in yellow. At the right side of Figure 17, the tracking error is shown with estimators (red line) and without estimators (blue line). In both figures, it is possible to see how the performance of the control strategy with the neural estimators is much better. Like in the case of the mass variation, the tracking error is the same until t = 4 s because before there is no wind disturbance, but from that moment on, the tracking error of the controller without the estimators increases significantly.

If the tracking according to each coordinate, X, Y, and Z, is represented (Figure 18), the contribution to the tracking error seems to be more balanced (reference, magenta; trajectory with adaptive neural estimators, red; trajectory without neural estimators, blue). For every coordinate, it is possible to see how the trajectory obtained by the controller with the neural estimators better fits the reference. Figure 18 also shows how the disturbance produces an important deviation around t=7s due to the big peak of disturbance in the y-axis which cannot be compensated without the adaptive neural estimator and it becomes a stationary error. This deviation can be also easily observed in Figure 17 in the PID line.

Table 4: Comparison of the MAX of the tracking error for different trajectories with mass variation.

Trajectory	MAX_X		MAX_Y		MAX_Z		MAX_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Linear	1.0270	1.0270	1.0270	1.0270	**5.0010**	9.7835	**5.1986**	9.7835
Circular	2	2	1.0562	1.0562	**5.0010**	9.7836	**5.4796**	9.7837
Helical	1	1	**0.1207**	0.2074	**0.1582**	9.2655	**1.0033**	9.2675
Cyclic helical	1	1	**0.2060**	0.6163	**0.1591**	9.1763	**1.0006**	9.1937
Lemniscate	1	1	**0.0614**	0.0962	**0.0913**	9.2541	**1**	9.2544
Helical lemniscate	1	1	**0.1128**	0.2703	**0.1322**	9.2549	**1**	9.2553

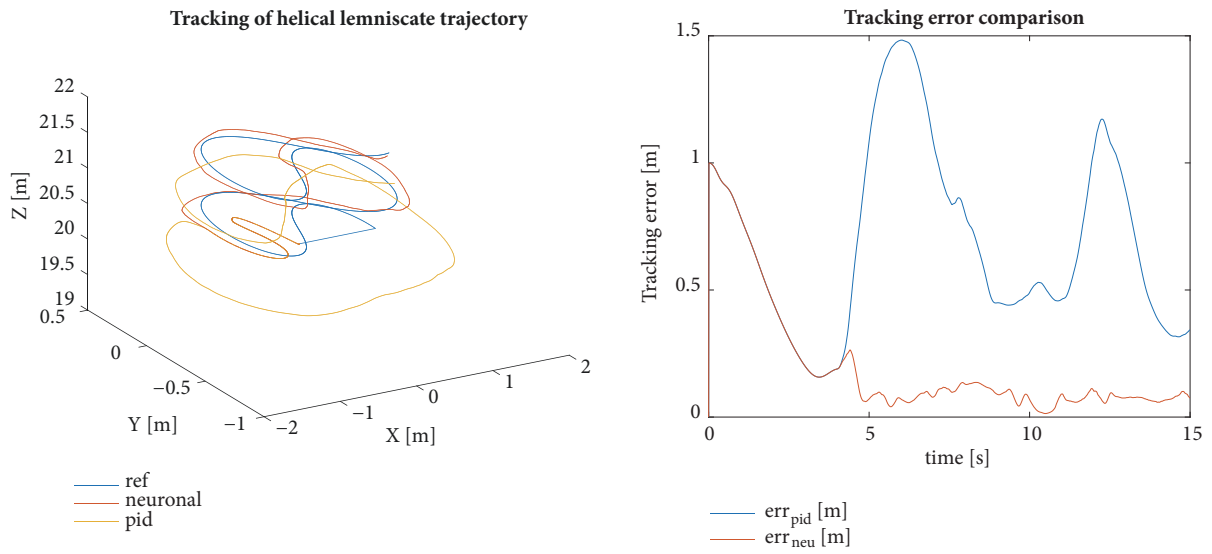

Figure 17: UAV tracking of a helical lemniscate trajectory with wind disturbance (left) and its tracking error (right).

Another interesting result which can be drawn from the previous figures is that we are not using a wind disturbance estimator for the Z coordinate (see Figure 4) but the tracking is still good enough. The reason is that adaptive neuromass estimator senses the wind disturbances in the z-axis as a virtual mass variation and it can compensate it. This is shown in Figure 19, where the estimate of the wind disturbance in the x-, y-, and z-axis is interpreted by the mass estimator as a mass variation. Indeed, the neural estimator (red line) fits reasonably well the real disturbances (blue line).

The controller has been extensively tested for different trajectories and the MSE numerical results are summarized in Table 5 for the different trajectories and for each coordinate, with the neural estimator (neuro) and without it (PID). The last column shows the absolute error. The best result for each component and trajectory is boldfaced.

Table 6 compares the values of the maximum error in this experiment for the same cases with before.

According to Tables 5 and 6, the controller with adaptive neural estimators provides less or equal tracking error for every tested trajectory. The minimal tracking error is achieved for the lemniscate trajectory in the z-axis, with a very small error value of 0.0005, 600 times less than the error value without the neural estimator. The maximum tracking error is obtained with the circular trajectory due to the high error of the initial conditions.

The controller with the adaptive neural estimators works also well when the wind follows other different profiles, such as a sinusoidal function. In the next experiment, the wind average speed at Z = 20 m has been set to

$$v_{w(Z=20)} = 12 + 6 * \sin\left(\frac{2\pi}{4}t\right). \quad (44)$$

Figure 20 shows the results. It can be seen how the performance of the controller without estimators is much worse than with the step wind profile. Nevertheless, the performance of the neurocontroller is like the previous ones. It is also possible to see the sinusoidal shape of the tracking error in Figure 20, right.

5. Conclusions

Intelligent control strategies are especially useful when the parameters change while the system is performing some tasks, and when the external disturbance are relevant, due to

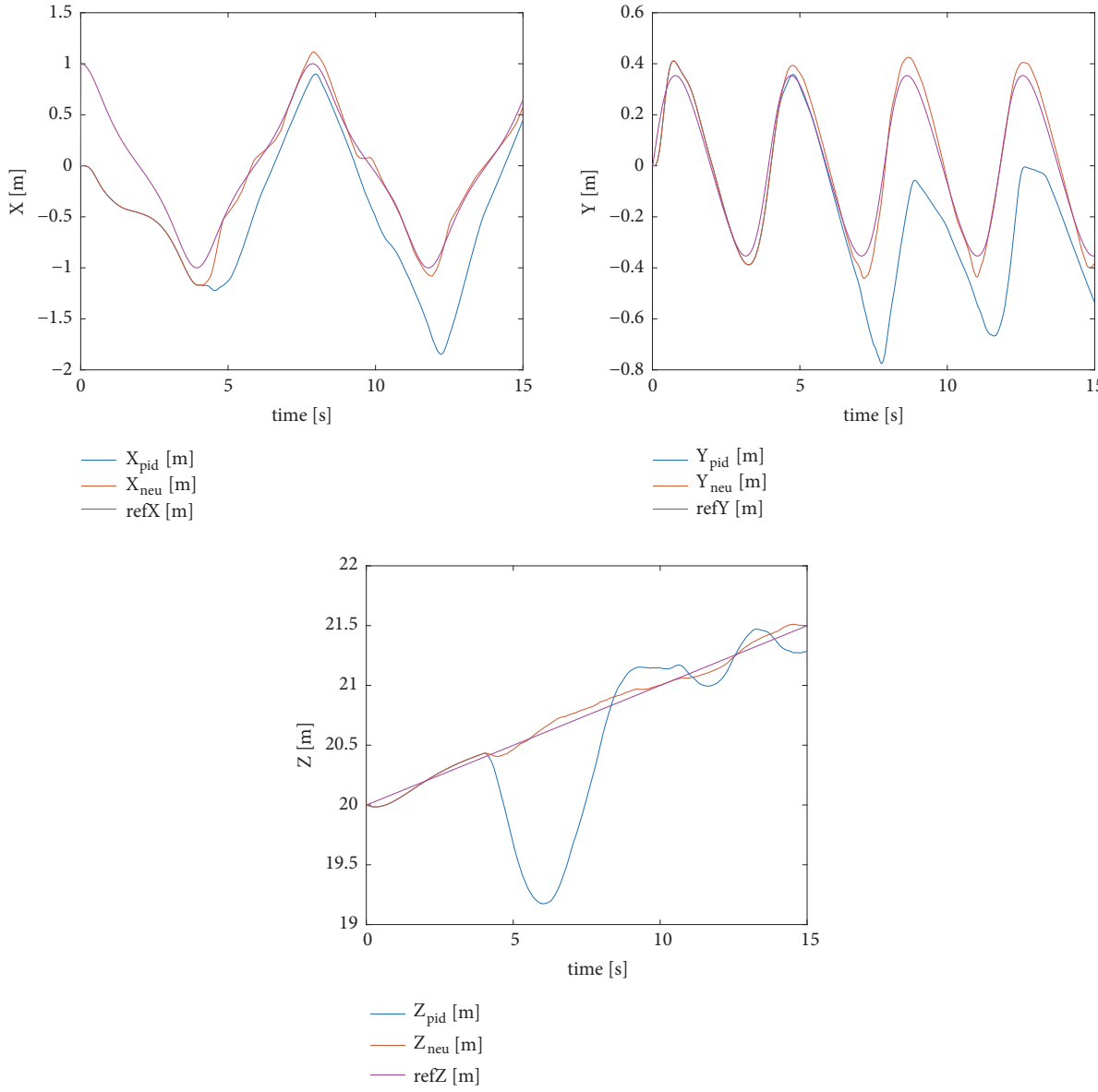

Figure 18: UAV tracking of X, Y, and Z coordinates, respectively, with wind disturbances.

Table 5: Comparison of the MSE of the tracking error for different trajectories with wind disturbances.

Trajectory	MSE_X		MSE_Y		MSE_Z		MSE_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Linear	0.0707	0.0707	0.0708	0.0708	**1.6163**	2.1214	**0.6730**	1.0130
Circular	**0.2805**	0.8612	**0.0739**	0.2596	**1.6186**	2.2711	**0.7591**	1.6014
Helical	**0.0951**	0.4850	**0.0042**	0.2009	**0.0048**	0.2847	**0.2181**	0.8666
Cyclic helical	**0.1053**	0.3435	**0.0132**	0.2000	**0.0069**	0.2596	**0.2746**	0.8084
Lemniscate	**0.0911**	0.1928	**0.0011**	0.0335	**0.0005**	0.3036	**0.1748**	0.6008
Helical lemniscate	**0.0940**	0.2614	**0.0032**	0.0695	**0.0010**	0.2460	**0.2015**	0.6903

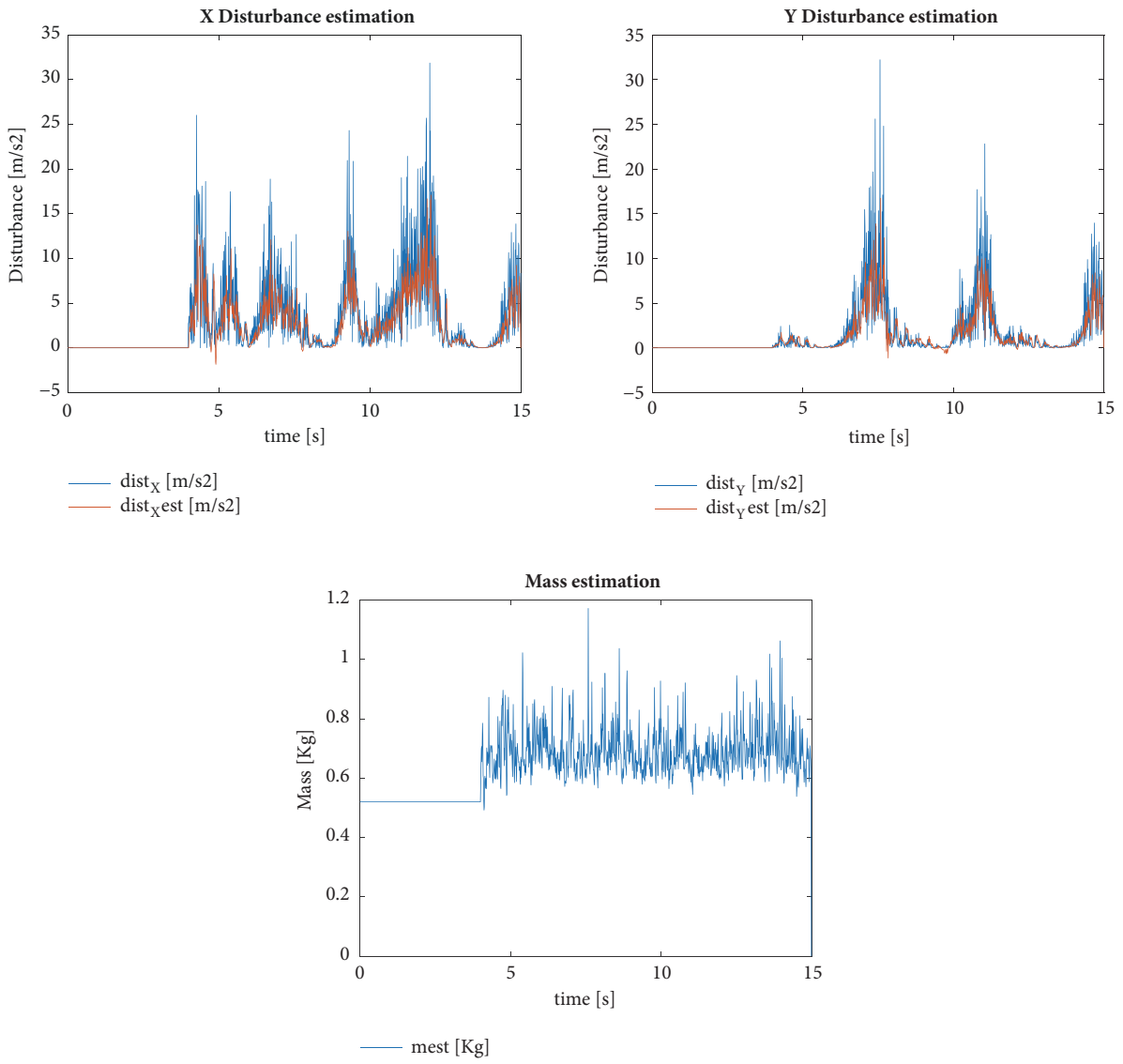

Figure 19: Wind disturbance estimation in the x- and y-axis, top, and mass estimation (bottom).

Table 6: Comparison of the MAX of the tracking error for different trajectories with wind disturbances.

Trajectory	MAX_X		MAX_Y		MAX_Z		MAX_T	
	Neuro	PID	Neuro	PID	Neuro	PID	Neuro	PID
Linear	1.0270	1.0270	1.0270	1.0270	5.0010	5.0010	5.1986	5.1986
Circular	2	2	**1.0562**	1.1864	5.0010	5.0010	5.4796	5.4796
Helical	**1**	1.4086	**0.1834**	0.9557	**0.1582**	1.4093	**1.0033**	1.6068
Cyclic helical	**1**	1.0948	**0.2060**	1.0506	**0.1591**	1.5356	**1.0006**	1.6874
Lemniscate	1	1	**0.1035**	0.5375	**0.0710**	1.6160	**1**	1.6166
Helical lemniscate	**1**	1.0582	**0.1228**	0.7613	**0.0771**	1.3190	**1**	1.3371

the ability to learn and adapt to the changing conditions. This is a common situation in many UAV applications.

In this work, a new intelligent control strategy based on neural networks has been proposed. It includes the design of neural networks that estimate the system parameter variations. They allow the UAV to follow different trajectories with small tracking error when disturbances due to mass changes and wind are included.

Simulation results show how the online learning of the neural estimators increases the robustness of the controllers,

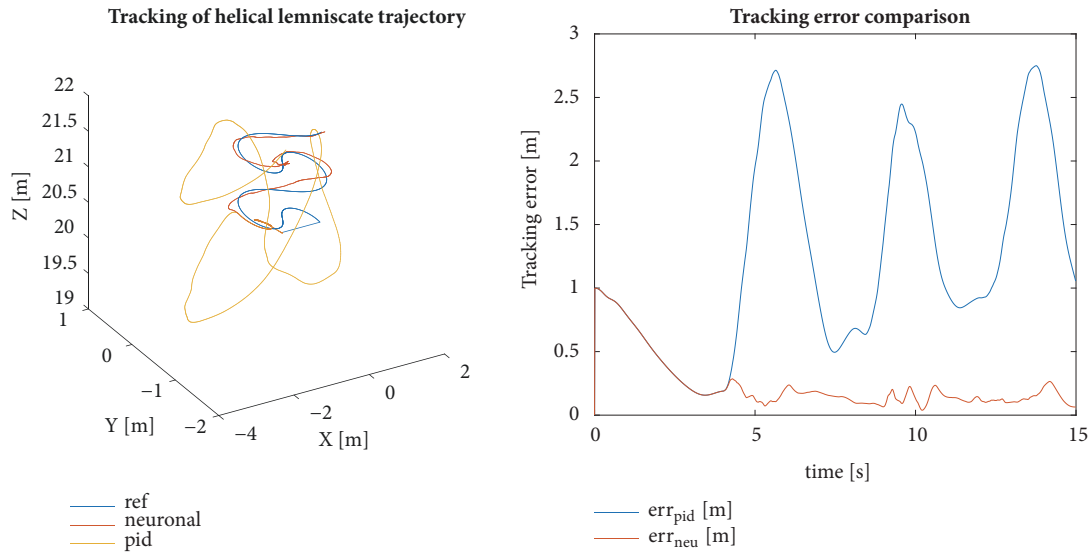

Figure 20: UAV tracking of a helical lemniscate trajectory with sinusoidal wind profile disturbance (left) and its tracking error (right).

reducing the effects of the mass variation and the wind on the UAV.

Among other possible future works, we may highlight the study of the influence of other disturbances such as the ones generated by the engines. In addition, the analysis of the parallelization of this approach for real-time application could be another interesting research line and help to deal with the high computational demand of these systems.

Data Availability

The data used to support the findings of this study are included within the article.

Disclosure

This is an extended version of an earlier version of this work presented in (13th Int. Conf. on Soft Computing Models in Industrial and Environmental Applications, 2018), [32].

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] V. San Juan, M. Santos, J. M. And, and J. M. Andújar, "Intelligent UAV Map Generation and Discrete Path Planning for Search and Rescue Operations," *Complexity*, vol. 2018, Article ID 6879419, 17 pages, 2018.

[2] R. Szabolcsi, "The Quadrotor-Based Night Watchbird UAV System Used In The Force Protection Tasks," *International conference Knowledge-Based Organization*, vol. 21, no. 3, pp. 749–755, 2015.

[3] R. Grassi, P. Rea, E. Ottaviano, and P. Maggiore, "Application of an Inspection Robot Composed by Collaborative Terrestrial and Aerial Modules for an Operation in Agriculture," in *Proceedings of the In International Conference on Robotics in Alpe-Adria Danube Region*, pp. 539–546, Cham, 2017.

[4] A. G. E. Ruiz, H. Alazki, J. E. V. Rubio, and O. G. Salazar, "Embedded super twisting control for the attitude of a Quadrotor," *IEEE Latin America Transactions*, vol. 14, no. 9, pp. 3974–3979, 2016.

[5] M. Santos, V. López, and F. Morata, "Intelligent fuzzy controller of a quadrotor," in *Proceedings of the IEEE International Conference on Intelligent Systems and Knowledge Engineering (ISKE '10)*, pp. 141–146, IEEE, Hangzhou, China, November 2010.

[6] Z. Wu, Z. Guan, C. Yang, and J. Li, "Terminal guidance law for UAV based on receding horizon control strategy," *Complexity*, vol. 2017, Article ID 2750172, 19 pages, 2017.

[7] P. Alfredo Toriz, B. Modesto Raygoza, and N. Daniel Martínez, "UAV technology inclusion model for preventing high-risk jobs in construction industries based on the IVAS methodology," *RIAI - Revista Iberoamericana de Automatica e Informatica Industrial*, vol. 14, no. 1, pp. 94–103, 2017.

[8] A. A. Fahmy and A. M. Abdel Ghany, "Adaptive functional-based neuro-fuzzy PID incremental controller structure," *Neural Computing and Applications*, vol. 26, no. 6, pp. 1423–1438, 2015.

[9] P. Garcia-Aunon, M. Santos Peñas, and J. M. García, "A new UAV ship-tracking algorithm," in *Proceedings of the Preprints of the 20th IFAC World Congress*, pp. 13090–13095.

[10] Z. Zhou, H. Wang, and Z. Hu, "Event-Based Time Varying Formation Control for Multiple Quadrotor UAVs with Markovian Switching Topologies," *Complexity*, vol. 2018, Article ID 8124861, 15 pages, 2018.

[11] X. Zhang, X. Li, K. Wang, and Y. Lu, "A survey of modelling and identification of quadrotor robot," *Abstract and Applied Analysis*, vol. 2014, Article ID 320526, 16 pages, 2014.

[12] M. Santos, "Un Enfoque Aplicado del Control Inteligente," *Revista Iberoamericana de Automática e Informática Industrial RIAI*, vol. 8, no. 4, pp. 283–296, 2011.

[13] M. Santos, R. López, and J. M. De La Cruz, "A neuro-fuzzy approach to fast ferry vertical motion modelling," *Engineering*

Applications of Artificial Intelligence, vol. 19, no. 3, pp. 313–321, 2006.

[14] B.-C. Min, J.-H. Hong, and E. T. Matson, "Adaptive robust control (ARC) for an altitude control of a quadrotor type UAV carrying an unknown payloads," in *Proceedings of the 11th International conference on control, automation and systems Korea*, pp. 26–29, 2011.

[15] C. Wang, M. Nahon, and M. Trentini, "Controller development and validation for a small quadrotor with compensation for model variation," in *Proceedings of the 2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014*, pp. 902–909, May 2014.

[16] C. Wang, B. Song, P. Huang, and C. Tang, "Trajectory Tracking Control for Quadrotor Robot Subject to Payload Variation and Wind Gust Disturbance," *Journal of Intelligent & Robotic Systems*, vol. 83, no. 2, pp. 315–333, 2016.

[17] K. Sreenath, N. Michael, and V. Kumar, "Trajectory generation and control of a quadrotor with a cable-suspended load-a differentially-flat hybrid system," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '13)*, pp. 4888–4895, May 2013.

[18] K. Sreenath and V. Kumar, "Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots," in *Robotics, Science and Systems (RSS)*, Berlin, Germany, 2013.

[19] S. Lee, D. K. Giri, and H. Son, "Modeling and control of quadrotor UAV subject to variations in center of gravity and mass," in *Proceedings of the 14th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2017*, pp. 85–90, Republic of Korea, July 2017.

[20] I. Palunko and R. Fierro, "Adaptive control of a quadrotor with dynamic changes in the center of gravity," in *Proceedings of the In Proceedings 18th IFAC World Congress*, vol. 18, pp. 2626–2631, 2011.

[21] I. Palunko, R. Fierro, and P. Cruz, "Trajectory generation for swing-free maneuvers of a quadrotor with suspended payload: a dynamic programming approach," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '12)*, pp. 2691–2697, 2012.

[22] P. Kotaru, G. Wu, and K. Sreenath, "Dynamics and control of a quadrotor with a payload suspended through an elastic cable," in *Proceedings of the 2017 American Control Conference, ACC 2017*, pp. 3906–3913, USA, May 2017.

[23] B. Yuksel, C. Secchi, H. H. Bülthoff, and A. Franchi, "A nonlinear force observer for quadrotors and application to physical interactive tasks," in *Proceedings of the 2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM 2014*, pp. 433–440, July 2014.

[24] J. Escareño, S. Salazar, H. Romero, and R. Lozano, "Trajectory control of a quadrotor subject to 2D wind disturbances: Robust-adaptive approach," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1-4, pp. 51–63, 2013.

[25] K. Alexis, G. Nikolakopoulos, and A. Tzes, "Switching model predictive attitude control for a quadrotor helicopter subject to atmospheric disturbances," *Control Engineering Practice*, vol. 19, no. 10, pp. 1195–1207, 2011.

[26] D. Cabecinhas, R. Cunha, and C. Silvestre, "A globally stabilizing path following controller for rotorcraft with wind disturbance rejection," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 2, pp. 708–714, 2015.

[27] L. Besnard, Y. B. Shtessel, and B. Landrum, "Quadrotor vehicle control via sliding mode controller driven by sliding mode disturbance observer," *Journal of The Franklin Institute*, vol. 349, no. 2, pp. 658–684, 2012.

[28] J. E. Sierra and M. Santos, "Modelling engineering systems using analytical and neural techniques: Hybridization," *Neurocomputing*, vol. 271, pp. 70–83, 2018.

[29] S. Bansal, A. K. Akametalu, F. J. Jiang, F. Laine, and C. J. Tomlin, "Learning quadrotor dynamics using neural network for flight control," in *Proceedings of the 55th IEEE Conference on Decision and Control, CDC 2016*, pp. 4653–4660, USA, December 2016.

[30] H. Boudjedir, O. Bouhali, and N. Rizoug, "Adaptive neural network control based on neural observer for quadrotor unmanned aerial vehicle," *Advanced Robotics*, vol. 28, no. 17, pp. 1151–1164, 2014.

[31] N. A. Bakshi and R. Ramachandran, "Indirect model reference adaptive control of quadrotor UAVs using neural networks," in *Proceedings of the 10th International Conference on Intelligent Systems and Control, ISCO 2016*, India, January 2016.

[32] J. E. Sierra and M. Santos, "Disturbances Based Adaptive Neuro-Control for UAVs: A First Approach," in *Proceedings of the In The 13th Int. Conf. on Soft Computing Models in Industrial and Environmental Applications, AISC*, pp. 293–302, Springer, 2018.

[33] A. K. Shastry, A. Pattanaik, and M. Kothari, "Neuro-adaptive augmented dynamic inversion controller for quadrotors," *IFAC-PapersOnLine*, vol. 49, no. 1, pp. 302–307, 2016.

[34] H. Yañez-Badillo, R. Tapia-Olvera, O. Aguilar-Mejía, and F. Beltran-Carbajal, "On Line Adaptive Neurocontroller for Regulating Angular Position and Trajectory of Quadrotor System," *RIAI - Revista Iberoamericana de Automatica e Informatica Industrial*, vol. 14, no. 2, pp. 141–151, 2017.

[35] C. Nicol, C. J. B. Macnab, and A. Ramirez-Serrano, "Robust neural network control of a quadrotor helicopter," in *Proceedings of the Canadian Conferenceof Electrical and Computer Engineering, CCECE*, pp. 001233–001238, IEEE, 2008.

[36] NASA Glenn Research Center. https://www.grc.nasa.gov/WWW/K-12/airplane/ldrat.html.

[37] J. Etele, *Overview of wind gust modelling with application to autonomous low-level UAV control*, Mechanical and Aerospace Engineering Department, Carelton University, Ottawa, Canada, 2006.

[38] Royal Meteorological Society, https://www.rmets.org/weather-and-climate/observing/beaufort-scale.