International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 04 | Apr 2020 www.irjet.net p-ISSN: 2395-0072

Intelligent Control of Unmanned Aerial Vehicle

Andrew Hany Fahim[1], Dr. Abdulla Ismail[2]

[1]Graduate Student, Dept. of Electrical Engineering, Rochester Institute of Technology, Dubai, UAE
[2] 2Professor, Dept. of Electrical Engineering, Rochester Institute of Technology, Dubai, UAE

Abstract - *The purpose of this research paper is to address and design an intelligent adaptive controller that solves the current flight challenges found in an Unmanned Aerial Vehicle (UAV). Over the course of the few recent years unmanned aerial vehicles have massively developed technologically that have brought to it a higher reliance in many industries. With this immense interest in UAV's and drones came along unforeseen challenges such as stabilizations and trajectory control difficulties due to the system's coupled, time-varying and non-linear dynamics. This was a call to reevaluate and determine if the classical control methods are still adequate for the sophisticated control required in our present-day. With the conclusive belief that with the technology in hand today control techniques could be advanced to achieve enhanced performance, and one of those technological advancements was Artificial Intelligence (AI). Therefore, a comparative study between the traditional Proportional Integral Derivative (PID) control technique and Artificial Neural Network (ANN) based methods are performed to prove that intelligent algorithms (IA) certainly offer enhanced control performances. After design, modeling and simulation, both controllers were exposed to complex real-stimulated environmental uncertainties/disturbances to assess their performance and reliability. Where the ANN control method performed much better on all measurable factors as its enhanced stability, suppressed disturbance effects and governed a more precise trajectory control over the classical PID controller.*

Key Words: **Quadrotor, UAV, PID, Artificial Intelligence, Neural Networks, Backpropagation.**

1. INTRODUCTION

As Unmanned Aerial Vehicles (UAVs) are in rapid growth, they are becoming of great aid in many technological advancements, civilian and military applications including photography, agricultural support, natural disaster support, rescue missions, , earth science research assistance, hostile zone reconnaissance, border security detection, fire detection/fighting, hazardous biological or chemical agent detection, , etc. These diversified applications have created the need for a single aerial vehicle with the ability to efficiently perform multiple tasks.

Where implementing a fully capable UAV the need of both intelligent and autonomous systems is a mandatory requirement, where the American psychologist Dr. Linda Gottfredson defines intelligence and autonomy as follows:

"Intelligence is a very general mental capability that, among other things, involves the ability to reason, plan, solve problems, think abstractly, comprehend complex ideas, learn quickly and learn from experience." [1]

"Autonomy refers to systems capable of operating in the real-world environment without any form of external control for extended periods of time."

The flight control techniques of UAV platforms have witnessed technological advancements to tackle not only the autonomy concept but to also overcome the trajectory related problems [2]. In this context, there was a need to also develop a new era of controller designs where several applied methods have been employed to solve the attitude stabilization and trajectory tracking problems [3].

1.1 Quadrotor

The UAV used in this study is a quadrotor which is formed from four propellers (helicopter rotors) placed on the edge of a "plus" (+) like shaped structure as shown in Figure 1.1. Where these propellers collaborate and complement each other resulting in a 6-degree of freedom (DOF) flight. Quadrotors come in different sizes and designs depending on the job it is built to accomplish.

Fig -1: Quadrotor Structure

The dynamics of this airframe are like any physical system that are governed by fundamental mathematical and physical equations of motion within the parameters that fall into place for the UAV aerodynamics of propulsion, trajectory control and flight stability. Where actuators are used to covert the commands from the controller into physical motion such as rotation and blade speeds accordingly to achieve the required flight settings. These actuators are electro-mechanical, for early design and analysis, the actuator dynamics are modeled with a second-order transfer function as presented further. where the UAV mathematical model based on the mechanical variables and constraints was derived from the physical components.

1.2 Problem Statement

The control of a multi-input multi-output (MIMO) UAV system is complex due to its intensive exposure to un-known environment disturbances especially under stimulated, volatile and highly integrated system. Where over the years the advancement of controllers has been just focused on traditional control techniques which were just implemented and tested to ensure stability, robust and a controllable UAV, these strategies where headed with the most optimal technique of linear quadratic regulator (LQR) [4]. Robust control methods have also taken a sharp technological advancement turn to assure robust control in addition simultaneously stabilize sophisticated systems when challenged with disturbances that deviate it from the required designed operation [5].

Therefore, there was need to design what we can call a universal controller that is able to address these issues without the need to be programmed to every specific situation the UAV could counter on the field, two different control approaches have been studied with the aim of assessing the quadcopter behavior to track complex trajectories and reject disturbances, where approach I was a classical control method and approach II was an advanced control method that could introduce a whole new set of control concepts.

I. The first control method was to design and implement the traditional PID controller.

II. The second control method proposed is an adaptive control algorithm based on artificial intelligence (AI) mythologies of Neural Networks (NN) was developed.

Approach II is an arising control method that is able to eliminate the drawbacks of non-linear control systems, which is a developed intelligence that is assured to offer stability, optimality, robustness to any system it is being integrated within.

2. QUADROTOR MODEL

The quadrotor system consists six control parameter outputs, the positions variables over the 3D axis (x, y, z) and the angled flight outputs represented as roll (Φ), pitch (Θ), yaw (ψ) which are dynamical motions about the x, y and z axis. After a thorough study of the quadrotor dynamic model, it was concluded that the flight motion dynamics are mainly controlled by the change of the rotational speed of propellers F1 to F4 shown in Figure 2 and are limited to the following motions.

1- Vertical z-axis motion induced by all four propellers rotating at constant speed
2- Pitch (Θ) rotation induced by propellers F1 and F2 rotating at a higher speed than the other two propellers.
3- Roll (Φ) rotation induced by propellers F3 and F4 rotating at a higher speed than the other two propellers.
4- Yaw (ψ) rotation induced by the difference in the counter-torque between each pair of propellers.

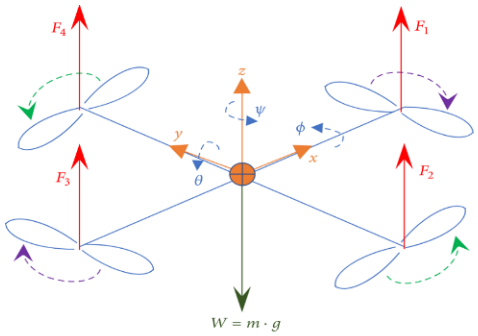


Fig -2: Dynamic model of a quadrotor

Moving forward with the analysis of the quadrotor a few assumptions had to be made to assure certain simplifications to the system to make its modeling possible. The assumptions mainly targeted the following aspects of the quadrotor.

- The quadrotor structure is symmetrical and rigid
- The propellers are also considered rigid
- Thrust and drag of the quadrotor are directly proportional the square of the propeller's speed.

Where the equations of motion are derived from Newton-Euler formalism for fixed frame/rigid body under external forces as in equation (1).

$$\begin{pmatrix} mI_{3x3} & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} \dot{V} \\ \dot{\omega} \end{pmatrix} + \begin{pmatrix} \omega * mV \\ \omega * I\omega \end{pmatrix} = \begin{pmatrix} F \\ \tau \end{pmatrix} \quad (1)$$

Resulting in the set of equations of motion (2), that are derived from the forces, inertia, and moments applied on the quadrotor.

$$\begin{aligned} \ddot{x} &= (\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi)\frac{u_1}{m} \\ \ddot{y} &= (\cos\phi \sin\theta \sin\psi - \sin\phi \cos\psi)\frac{u_1}{m} \\ \ddot{z} &= -g + \cos\phi \cos\theta \frac{u_1}{m} + d_1 \\ \ddot{\phi} &= \dot{\theta}\dot{\psi}\frac{I_{yy} - I_{zz}}{I_{xx}} - \frac{J_r}{I_{xx}}\dot{\theta}\Omega_r + \frac{u_2}{I_{xx}} + d_2 \\ \ddot{\theta} &= \dot{\psi}\dot{\phi}\frac{I_{zz} - I_{xx}}{I_{yy}} + \frac{J_r}{I_{yy}}\dot{\phi}\Omega_r + \frac{u_3}{I_{yy}} + d_3 \\ \ddot{\psi} &= \dot{\phi}\dot{\theta}\frac{I_{xx} - I_{yy}}{I_{zz}} + \frac{u_4}{I_{zz}} + d_4 \end{aligned} \quad (2)$$

Which also develops the control inputs, total trust (U1), roll control (U2), pitch control (U3) and yaw control (U4) represented as the following set of equations (3).

$$\begin{aligned} U_1 &= b(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2) \\ U_2 &= bl(-\Omega_2^2 + \Omega_4^2) \\ U_3 &= bl(-\Omega_1^2 + \Omega_3^2) \\ U_4 &= d(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2) \\ \Omega_r &= (-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4) \end{aligned} \quad (3)$$

Quadrotor model parameters are as described in Table 1.

Table -1: Quadrotor model parameters

Parameter	*Description*	*Unit*
Φ	*Roll angle about the x-axis*	*degree*
Θ	*Pitch angle about the y-axis*	*degree*
Ψ	*Yaw angle about the z-axis*	*degree*
I_{xx}, I_{yy}, I_{zz}	*Inertia on the axis*	*kg.m²*
G	*Gravitational force*	*m.s⁻²*
M	*Mass of the UAV*	*kg*
J_r	*Rotor inertia*	*kg.m²*
B	*Thrust coefficient*	*N. s²*
D	*Drag coefficient*	*N.m. s²*
L	*Arm length*	*m*
U	*Control inputs*	

Bearing state-space vectors equations (4) and (5).

$$X = [\emptyset\ \dot{\emptyset}\ \theta\ \dot{\theta}\ \psi\ \dot{\psi}\ x\ \dot{x}\ y\ \dot{y}\ z\ \dot{z}]^T \quad (4)$$

$$\begin{aligned} x_1 &= \emptyset & x_7 &= x \\ x_2 &= \dot{x_1} = \dot{\emptyset} & x_8 &= \dot{x_7} = \dot{x} \\ x_3 &= \theta & x_9 &= y \\ x_4 &= \dot{x_3} = \dot{\theta} & x_{10} &= \dot{x_9} = \dot{y} \\ x_5 &= \psi & x_{11} &= z \\ x_6 &= \dot{x_5} = \dot{\psi} & x_{12} &= \dot{x_{11}} = \dot{z} \end{aligned} \quad (5)$$

Obtaining the following transitional control matrix (6).

$$f(X,U) = \begin{pmatrix} \dot{\emptyset} \\ \dot{\theta}\dot{\emptyset}a_1 + \dot{\theta}a_2\Omega_r + b_1U_2 \\ \dot{\theta} \\ \dot{\emptyset}\dot{\psi}a_3 + \dot{\emptyset}a_4\Omega_r + b_2U_3 \\ \dot{\psi} \\ \dot{\emptyset}\dot{\theta}a_5 + b_3U_4 \\ \dot{z} \\ g - (cos\theta cos\varphi)\frac{U_1}{m} \\ \dot{x} \\ u_x\frac{U_1}{m} \\ \dot{y} \\ u_y\frac{U_1}{m} \end{pmatrix} \quad (6)$$

Out of the six quadrotor system parameters (x, y, z, Φ, Θ, ψ), only four of those variables will be used in the control system/operation of the quadrotor. This is because the x & y outputs/variables are not decoupled and is not possible to control them directly using the four laws of U1 to U4. As the propellers can't induce direct motion wither the x or y axis due to the orientation limitation. The control of x & y will be done indirectly through the control of the roll and pitch angles introducing the following equation 'Ux' in equation (7) for the x-axis motion control and 'Uy' in equation (8) for the y-axis motion control.

$$u_x = (\cos\psi \sin\theta \cos\varphi + \sin\psi \sin\varphi)\, u_1 \quad (7)$$

$$u_y = (\sin\psi \sin\theta \cos\varphi - \cos\psi \sin\varphi)\, u_1 \quad (8)$$

The quadrotor position control schematic in Figure 3 represents the fundamental control logic, where the quadrotor controller would need to be first fed with inputs of the exact positional coordinates which are referred to as desired x, y and z. where the system through a feedback loop is constantly calculating the error (actual positions against desired positions). On the secondary control step, the roll, pitch and yaw angles are determined to best bring the quadrotor to the desired position also constantly trying to minimize the error there through a feedback loop

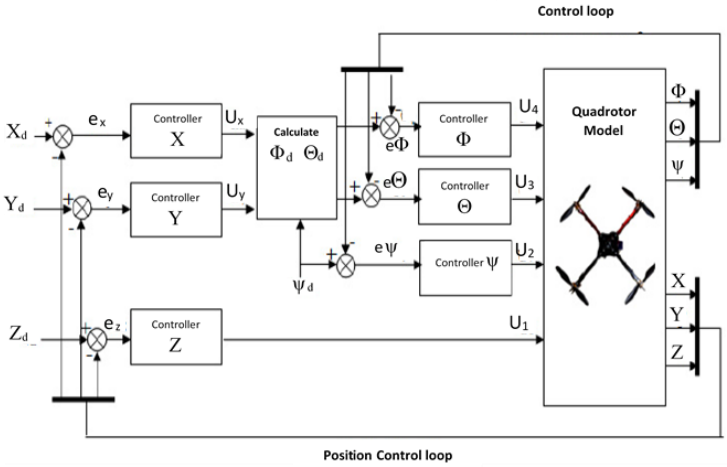


Fig -3: Quadrotor position control schematic

3. PROPORTIONAL-INTEGRAL-DERIVATIVE (PID) CONTROL OF A QUADROTOR

The key aspect of a control system falls within its feedback controller, as most advanced systems are considered complex and faced with uncertainties/disturbances. This is why a Proportional-Integral-Derivative (PID) controller was introduced and proved as an efficient method of control for those systems. As systems became more agile and inconsistent with the increase in their complexity it wouldn't be fair to use custom designed controllers for every case/situation facing the system as it would yield a very high design cost and time. Therefore, PID controllers like the one shown in Figure 4, in equation seemed to be the only logical solution. As the name Proportional-Integral-Derivative (PID) implies, these controllers consist of 3 sub-controllers.

To better understand these sub controllers effects on any controlled system, it should be clear how every controller effects the system response, bearing in mind that any combination of the PID elements could be used together and not necessarily all three elements of the PID, for example a

combination of only a PD or PI controller could be implemented which depends on the system to be controller and the variables that are incorporated into that system considering the required results range and constraints.

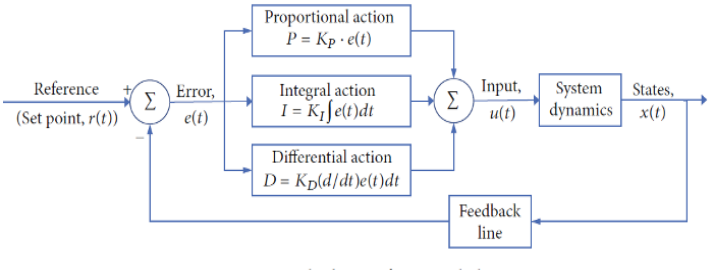


Fig -4: PID controller design

Formalized as in equation (9)

$$u_{PID} = k_p e(t) + k_i \int e(t) + k_d \frac{de(t)}{dt} \qquad (9)$$

With K_p, K_i *and* K_d being the proportional, integral and derivative gain respectively.

3.1 Measures of Control of a Quadrotor

The quadrotor system consists of six outputs, the positions variables (x, y, z) and the flight outputs angles (Φ, Θ, ψ). Where only four of those variables will be used in the control system/operation of the quadrotor.

- Altitude control; is a non-linear input therefore, it is worked out in the Z domain to clear of any non-linearity in the system. U1 being the control input for the altitude is being controller through a PD controller as follows;

$$U_1 = \frac{1}{\cos\Phi \cos\Theta} . K_{zp}(z_d - z) + K_{zd}(\dot{z}_d - \dot{z}) + mg \qquad (10)$$

- Roll control system input equation is represented as follows;

$$U_2 = K_p(\Theta_d - \Theta) + K_d(\dot{\Theta}_d - \dot{\Theta}) \qquad (11)$$

- Pitch control system input equation is represented as follows;

$$U_3 = K_p(\Phi_d - \Phi) + K_d(\dot{\Phi}_d - \dot{\Phi}) \qquad (12)$$

- Yaw control system input equation is represented as follows;

$$U_4 = K_p(\psi_d - \psi) + K_d(\dot{\psi}_d - \dot{\psi}) \qquad (13)$$

Where Z_d Θ_d, ψ_d are desired roll, pitch and yaw respectively.

3.2 PID Controlled Quadrotor Simulation and Results

Enhancement of the UAV quadrotor was performed through the development of four controllers to control the earlier mentioned outputs. A Simulink model was built to simulate the quadrotor system as shown in Figure 5.

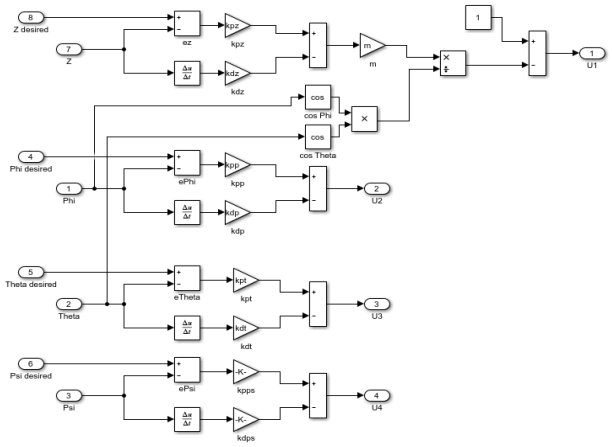


Fig -5: PID Simulink Model

PD implementation of the U1 to U4 equations for Altitude, Roll, Pitch and Yaw control to obtain the following stability graphs, shown in Figure 6. Where the PID controller did demonstrate an acceptable overall stability, with results shown in Table 2. The roll angle experienced oscillations with lower frequency than the pitch angle, which could possibly be because the quadrotor's initial orientation was in favor of the roll (angle on the x-axis). But nonetheless the roll and pitch were bounded within the same amplitude and stabilized over the same time which is an expected response with any decent PID controller. Whereas the yaw angle experiences the least jitters and stabilized quicker also as theoretically expected, this is due to the nature of flight of the quadrotor over the altitude (z-axis).

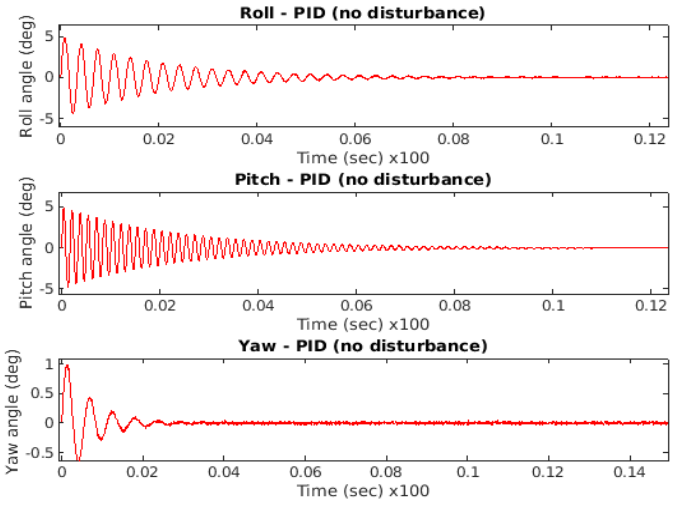


Fig -6: Quadrotor PID controlled flight results

Table -2: Quadrotor model parameters

	Roll Φ	*Pitch Θ*	*Yaw ψ*
Settling time (secs)	5.4	5.7	2.8
Steady state error (secs)	0.3	0.2	0.4

4. INTELLIGENT CONTROL TECHNIQUES

How do we define intelligence?
Can the intelligence be possibly measured and improved?
The answer is YES, and that can be done through understanding the levels of intelligence of controllers. As per Krishna Kumar, from NASA Ames Research Center who defines an intelligent system is one that demonstrates the following traits [6]:

- Learning
- Adaptability
- Robustness across problem domains
- Improving efficiency (over time and/or space)
- Information compression (data to knowledge)
- Extrapolated reasoning

The level of intelligence implemented on a system is determined by the required outputs of every system and its constraints. Table 3 explains in brief the level of intelligence.

Table -3: Levels of intelligence

Level	*Self-improvement*	*Description*
0	Tracking Error (TE)	Robust Feedback control (Error tends to zero).
1	TE + Control Parameters (CP)	Robust feedback control with adaptive control parameters (error tends to zero for non-nominal operations; feedback control is self-improving).
2	TE + CP + Performance Measure (PM)	Robust, adaptive feedback control that minimizes or maximizes a utility function over time (error tends to zero and a measure of performance is optimized).
3	TE + CP + PM + Planning Function	Level 2 + the ability to plan ahead of time for uncertain situations, simulate, and model uncertainties.

There is also a number of methods to achieve an intelligent control system utilizing artificial intelligence (AI) algorithms and methods, such as heuristics, support vector machines, artificial neural networks and Markov decision process. Artificial neural networks will be discussed in details and its implementation and performance on the UAV quadrotor will evaluated.

4.1 Neural Networks (NN)

Adaptive control techniques are classified to be the most intelligent as a result to their knowledge-based decision-making capabilities. Where the evolutionary computing of AI is entirely invading the world and magnifying the idea of being able to perform highly complex tasks easier and smarter in comparison to traditional control methods [7,8]. Where neural networks have got an advantage over any other smart technique which is the ability to approximate and model complex systems, demonstrating spatial skills from human logic.

Artificial Neural Network (ANN) mainly mimics the working strategy of a human brain using mathematical models (in form of algorithms) to resemble the biological neuron, many forms/architectures of neural networks are introduced, although most of them are software oriented some are also implemented on hardware [9]. Where an optimal NN would require advancement in both hardware and software capabilities simultaneously.

Neurons are the simplest processing element of an ANN operating in parallel, where any multi-layer feed forward neural network architecture consists of the following three layers in the network called (1) input layer, (2) hidden layer, and (3) output layer respectively as shown in Figure 7. Where these layers are interconnected with each other but there are no interconnections in the units of the same layer. The nodes represent computational units and need to obtain inputs which should be processed in neurons to produce the outputs.

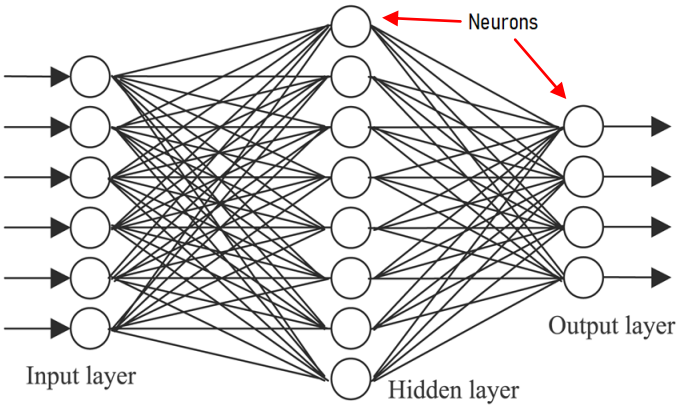


Fig -7: Neural Network Architecture

A basic neuron contains several elements: input, weights, activation function, threshold, and output. Weights are multiplied with inputs and then added in summing function and the sum is processed in activation function and finally, the model gives an output [10], as shown in Figure 8 and the set of equations below.

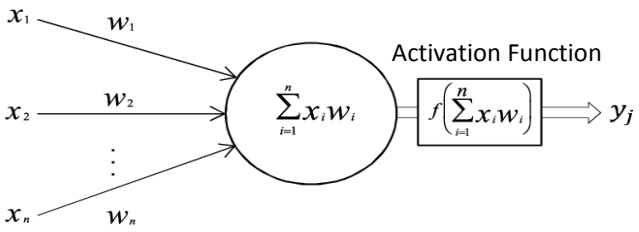


Fig -8: Neuron mathematical functions

The output can be calculated by applying the activation function over the net input, as each one of those simple neurons has an activation function that consist of the output value/decision of the neuron. The activation function is mainly working between the network's input and output defining any non-linearity of the model capabilities.

The hidden neural network layers could be configured in many ways depending on the nature of system where the NN is to work within. Where the weights on the branched hidden layers are usually calibrated thought a very sophisticated algorithm called backpropagation.

Neural networks require to be trained first before it can be deployed for controlling, this training/learning process is basically the adaptation of the network to better handle a task by considering sample (question-answer pairs) observations. This happens through feeding datasets into the NN to analyze and learn from, the larger the dataset the better the NN performs when tested. Where the datasets are usually divided into a training section which is around 70-80% that allows the NN to start sensing the nature of the operations it is to control and start setting weights accordingly, the rest of the dataset is left for the testing which is very important for the network to start validating and developing an initial value of the network's accuracy[11].

While the learning rate determines the speed that the neural network arrives at the minimum acceptable solution. Encouraging the development of a reducing cost function which is a combination of the weight and input that is able to determine the variance between the output and real input value at the terminal stage of the network.

A few different types of learning/trainings exist and are chosen based on the operation or system where the neural network will be utilized, the most well-known and commonly used learning techniques are.

1. Supervised learning – analyzing knowledge-based decisions from pre-fed dataset where the rational decision is also fed into the network for it to start tuning itself to find the correct decision when it is at the test stage, the more of this training/learning process the more the accuracy.

2. Unsupervised learning – the NN is fed with unlabeled data sets/situations (containing only the input data) and left to find patterns in the data and build a new model from it. where the ANN is able to categorize the data by assessing the distance among clusters and it is left to find a cost function.

3. Reinforcement learning - kind of learning that involves interaction with the environment, getting the state of such environment, choosing an action to change this state, sending the action to a simulator or critic agent and being assessed numerically on a pre-defined scale or through any other reward or a penalty form that the network is able to understand with the aim to always be rewarded positively as it gets closer to the required rational goal.

Neural network-based control has demonstrated advanced effectiveness with high dynamic systems where the system dynamics could at times not be fully transparent to the network. This leads into the study of two different identification methods, where in general adaptive neural networks architectures could be categorized in one of the following two categories:

1. Direct Control: When the dynamic model is almost fully known and the system directly converges to the required output value, by directly adjusting to the controlled parameters to reduce the system errors.

2. Indirect Control: {Also known as Direct Inversion Control (DIC)} When the dynamic model is partially known or unknown, this requires the development of a control methodology that is able to visualize/identify the system dynamics first. This happens through a recursively updated identifier model as shown in Figure 9 in which the network is to adapt to plant parameters of the control laws. This self-tuning regulator basically assigns every output being obtained to a correlated specific input and process.

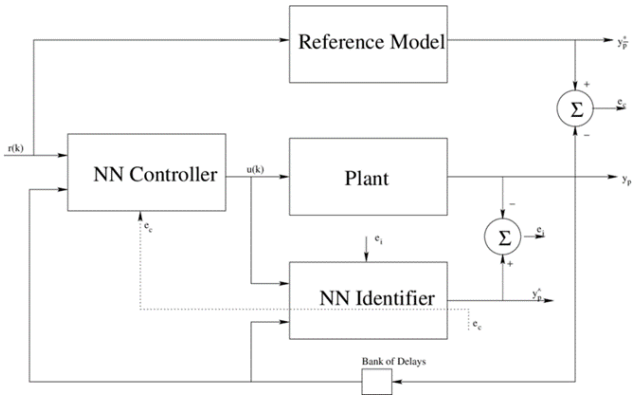


Fig -9: Indirect control model

4.2 Backpropagation (BP)

The backpropagation algorithm is a very efficient approach to compute the derivatives of multi-dimensional problems and optimize the model goal approach though learning and

adjusting the weight coefficient for parametrizing the network nodes [12].

Which is basically going backward through the back end of the network as shown in Figure 10, adjusting the weights between input and the neuron, by reiterating this process and re-assigning more accurate weights that reduces the cost function bearing an optimized rational decision of the model.

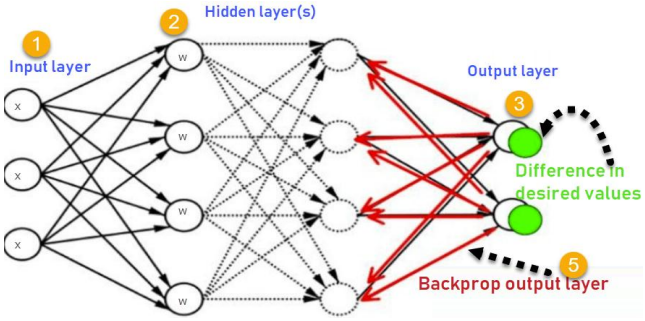


Fig -10: Backpropagation

Backpropagation mainly computes gradient and loss functions with respect to each assigned weight, this allows for a gradient analysis for training and updating the model for the maximum accuracy. This takes place within a structure shown in Figure 11, where the desired and actual control variable values are being fed to the backpropagation algorithm allowing it to set more accurate weights as it iterates, passing the most instantiations accurate weights to the NN.

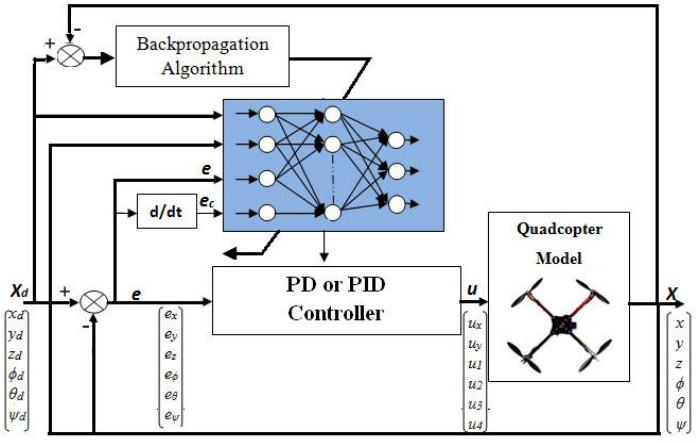


Fig -11: NN controlled quadrotor schematic

Were a gradient decent algorithm is used to analyze and diminish the error function, through the re-assignment of the weight vector (W).

4.3 Gradient Decent (GD)

Gradient decent simply is another algorithm that works by trying to reach to the optimal point assigned on the graphical representation.

So, if the slope is negative, as shown in Figure 4.6 from red arrow side that means you must go downhill from there. This minimizes the count of incorrect weights on the negative slope also reducing the processing time and power to finding a better weight for this processed neuron masking this method consistent and reliable [12].

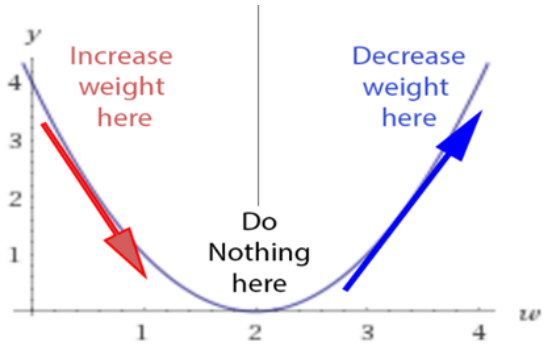


Fig -12: Gradient decent identification

The gradient decent is done through integrating the samples of the data set multiple times to get the most exact weights. In deep learning this is done through what is called an epoch, which is basically a hyperparameter (that is usually defined before the start of the training process) that controls the number of complete passes the dataset has through the network during the training phase.

As the epoch ensures that every sample in the dataset is passed forward and then backward through the NN at least once. Due to the complexity of this algorithm and its operations especially with large datasets, it is simplified by breaking down the datasets into batches.

So, for example if the dataset contains 6000 data entries, it could be divided into 12 batches (500 entries for every batch). And it would take 12 iterations to finish of 1 epoch. This is done to save on processing time and power.

This does guarantee a high level of accuracy as we increase the number of epochs during the training of the NN, but it is witnessed after a while, that even with the increase of the number epochs that the model accuracy doesn't improve remarkably, this could be due to many factors like the number of entries in the dataset (the more entries the more training and accurate the system gets) or it could be due to the number of hidden layers. So, choosing the optimal number of epochs would save in processing time and power.

5. INTELLIGENT CONTROLLER DESIGN FOR QUADROTOR

To accomplish an ideal framework execution, an intelligent control system dependent on backpropagation neural system (BPNN) is designed and implemented for quadcopter control. The attitude vector X1 and position vector X2 are defined in equation (14).

$$Out_i^{(1)} = x(i) \quad i = 1...4 \qquad (14)$$

The controller is designed using the incremental control algorithm and multilayer neural network. The NN is made up of an input layer with four control inputs, a hidden layer and the six output parameters (x, y, z, Φ, Θ, ψ)

The 4 neurons at the input layer:

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \\ \mu_4 \end{pmatrix} = \begin{pmatrix} X_{id} \\ X_i \\ e_i \\ e_{ci} \end{pmatrix}_{i=\{1;2\}} \quad (15)$$

where the input vector (μ), error vector (e) and changing rate error (ec) of the system are defined as follows.

$$e_{ci}(k) = e_i(k) - e_i(k-1) \quad (16)$$

The neuron outputs in the input layer are.

$$X = \begin{bmatrix} X_1 & X_2 \end{bmatrix} = \begin{bmatrix} \varphi & \theta & \psi & x & y & z \end{bmatrix} \quad (17)$$

Hidden layer has n=10 neurons. The hidden layer neurons are calculated respectively as

$$in_j^{(2)}(k) = \sum_{i=1}^{4} v_{ij}\ Out_i^{(1)} \quad j = 1 \ldots n \quad (17)$$

$$Out_j^{(2)} = f\left(in_j^{(2)}(k)\right) \quad (18)$$

Output layer has m=6 neurons. The neurons placed on this layer correspond to the controller gains. The neuron here expressed as

$$in_l^{(3)}(k) = \sum_{j=1}^{n} w_{lj}\ Out_j^{(3)}(k) \quad l = 1 \ldots m \quad (19)$$

$$Out_l^{(3)} = g\left(in_l^{(3)}(k)\right) \quad (20)$$

Where $\mathbf{V}_i$, are Weight connecting the input layer to the hidden layer neurons and $\mathbf{W}_{lj}$ is the Weight connecting the hidden layer to the output layer neurons

The activation functions of the hidden and output layers are defined as f(x) and g(x) respectively as follows;

$$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (21)$$

$$g(x) = \frac{1}{2}(1 + \tanh(x)) = \frac{e^x}{e^x + e^{-x}} \quad (22)$$

The backpropagation algorithm is based on the minimization of a sum-squared error (MSE) utilizing the optimization gradient descent method. MSE is used as the cost function which is a function of error defined as follows:

$$E(k) = \frac{1}{2}e(k)^2 \quad (23)$$

The reassignment weights on hidden and output layers is;

$$\Delta w_{ij}(k) = -\eta \frac{\partial E(k)}{\partial \Delta w_{ij}} \quad (24)$$

This implies that the weight changing in every iteration is not only depending on the current measured error, but also depending on the previous incurred changes. So, the branch weights on the layers are updated by the following equations:

$$\Delta w_{jl}(k) = -\eta \frac{\partial E(k)}{\partial \Delta w_{jl}} + \alpha \Delta w_{jl}(k-1) \quad (25)$$

$$\Delta v_{ij}(k) = -\eta \frac{\partial E(k)}{\partial \Delta v_{ij}} + \alpha \Delta v_{ij}(k-1) \quad (26)$$

where adjustments had to be made to the weight assignments to avoid the issue of the local minima which is a very well-known problem associated with the backpropagation algorithm, therefore the necessary adjustments and Finally, the above analysis can lead to the adjustment of the weights of the output layer by using the following formulas:

$$\Delta w_{jl}(k) = \alpha \Delta w_{jl}(k-1) + \eta \delta_l^{(3)}\ Out_j^{(2)}(k) \quad (26)$$

$$\delta_l^{(3)} = e(k)\,\mathrm{sgn}\left(\frac{\partial y(k)}{\partial \Delta u(k)}\right) \frac{\partial \Delta u(k)}{\partial Out_l^{(3)}(k)}\, g'\left(in_l^{(3)}(k)\right) \quad (27)$$

The learning algorithm of the weighted hidden layer can also be led to:

$$\Delta v_{jl}(k) = \alpha \Delta v_{jl}(k-1) + \eta \delta_j^{(2)}\ Out_i^{(1)}(k) \quad (28)$$

$$\delta_j^{(2)} = f'\left(in_j^{(2)}(k)\right) \sum_{l=1}^{m} \delta_l^{(3)}\ \Delta w_{ij}(k) \quad (29)$$

To envision the control equations of the PID and the NN that is used to tune the controller it can be put together as follows.

$$u_i^{ctrl} = K_{i,P} e + K_{i,I} \int e.dt + K_{i,D} \frac{de}{dt} \quad (30)$$

$$u_i^{NN} = \sum_{j=1}^{N} a_j e^{-\beta_j \|\mathbf{x} - \mathbf{c}_j\|^2} \quad (31)$$

5.1 Neural network Controlled Quadrotor Simulation and Results

The NN-based controller was to generate the control signals {U1, U2, U3, U4} where the quadrotor stability,

$$e_{ci}(k) = e_i(k) - e_i(k-1)$$

performance, overshoot, and settling time was assessed. The NN's performance and accuracy can be visualized through the "Input-Output Data for NN Model Reference Control" run, which falls with the options of model reference control. This basically trains the NN and also predicts how the input-output relation is going to be, the more training and more data the NN gets exposed to the more accurate it becomes. Our training samples for the quadrotor model worked on 6000 samples, which could result in a training time of around 45 minutes or more depending on the CPU used. Which overall did demonstrate and acceptable rate of convergence to the most accurate and rational control decision.

The results of the NN controller shown in Figure 13, were very satisfactory and did demonstrate a level of intelligence in the way it adapted the flight parameters to the environment, for instance no jitters or high frequency oscillations are visible at the take of stage. Where the amplitude is limited within a 1-degree angle of error which will not be at all sensed in actual flight performance.

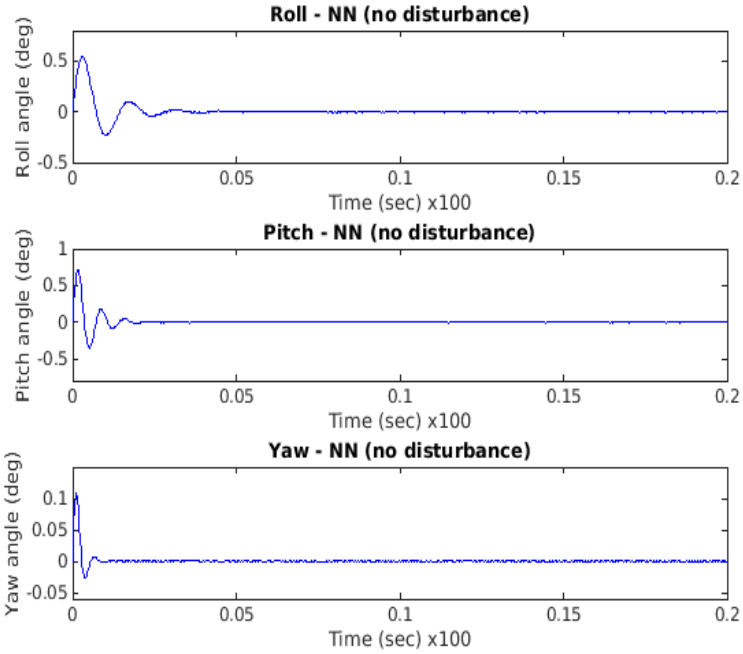


Fig -13: Quadrotor NN controlled quadrotor flight results

Results obtained graphically in Figure 13 and represented numerically in Table 4, are more than satisfactory. These results were easily achieved by the NN controller which are considered optimal flight response settling time with very minimal overshoot and no steady state error. If these results are achieved on an experimental analysis it could lay new control foundations for quadrotors/drones' performance

Table -4: NN controlled quadrotor settling time and steady state error

	Roll Φ	Pitch Θ	Yaw ψ
Settling time (secs)	2.9	2.6	0.7
Overshoot (%)	0.53	0.78	0.13
Steady state error (secs)	0	0	0

6. COMPARISON OF CONTROLLERS PERFORMANCE

Throughout the following comparison factors such as desired position and quadrotor parameters were kept the same to ensure a coherent result assessment. In addition to an ideal condition assessment for both controllers, a wind disturbance was introduced to the experiment to evaluate how both the controllers responded. This wind disturbance was introduced to the system model in the form of a signal intended to disturb the quadrotor flight with a value equivalent to 1Nm of force at t=25s. The following comparison experiment where conducted respectively.

6.2 Comparision of Flight Performance Between a PID and a NN Controlled Quadrotor – No disturbance

The following comparison is of the quadrotor flight performance mainly focused on settling time, overshoot and steady state error, between a PID and a NN controlled quadrotor with no disturbances of any sort to the system. Where it is obvious that the NN improved the system response factors. The graphs shown in Figure 14, (PID in red and NN in blue) are both brough to almost the same scale (x-axis and y-axis) to give a sense of how much of an improvement was achieved through the NN controller. The NN controlled quadrotor almost seems jitter-less and stable from the moment of take-off.

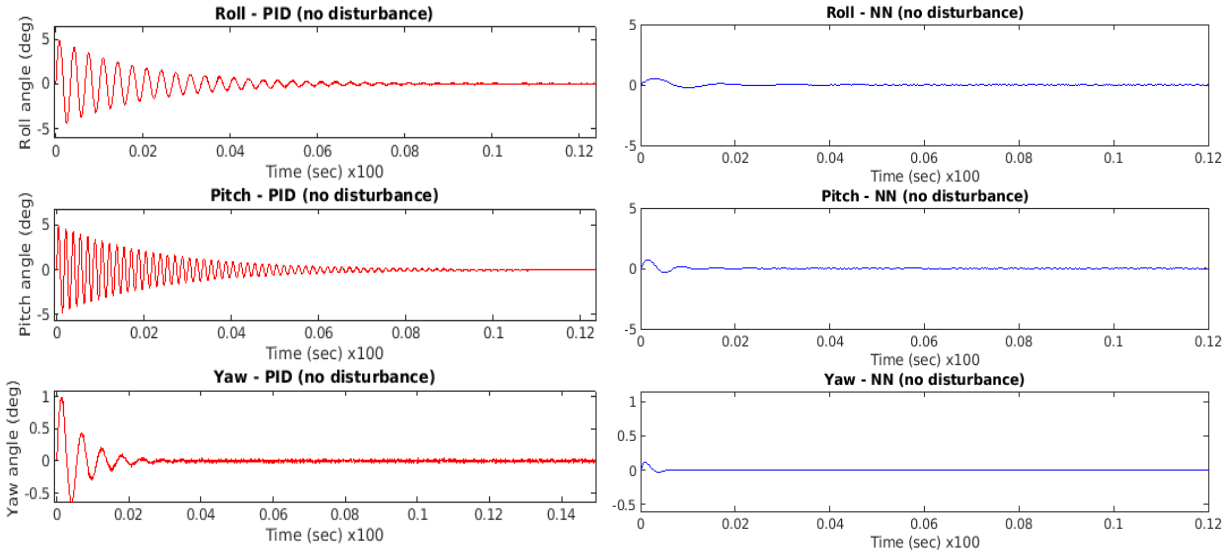


Fig -14: Comparision between PID and NN Controlled Quadrotor Flight (NO Disturbance)

6.3 Comparision of Flight Performance between a PID and a NN Controlled Quadrotor – No disturbance

The following comparison is of the PID and the NN controllers with wind disturbances introduced to the system at t=25s with a magnitude of 1 Nm over a total period of t=100s

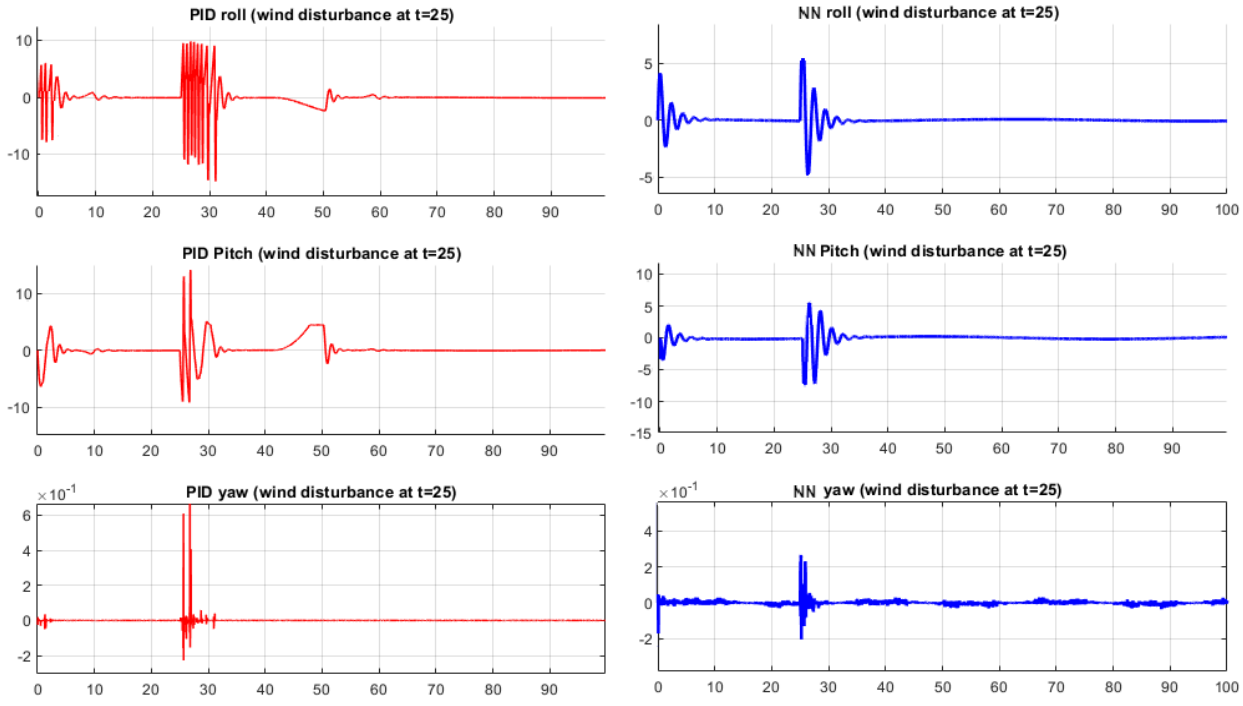


Fig -15: Comparision Between PID and NN Controlled Quadrotor Flight (with Wind Disturbance)

Results are numerically represented in Table 5 comparing the settling time and overshoot amplitude of every controller performance on the roll, pitch and yaw parameters.

Table -5: Quadrotor System Response Comparision Between PID and NN Performance

	Roll Φ		Pitch Θ		Yaw ψ	
Controller	PID	NN	PID	NN	PID	NN
Settling time (secs)	27	6	26	7.5	6	3
Overshoot (oscillations in degrees)	26	10	25	7	8	4.5

The PID did demonstrate its ability to stabilize the quadrotor even after its exposure to the wind disturbance, which took an average of 26 seconds to stabilize the roll and the pitch angles and an average of 8 seconds to stabilize the yaw angle. The PID indeed ensured the controllability over the quadrotor system and would serve as a possible solution for quadrotors not intended to carry out accuracy sensitive tasks.

Whereas the NN performed much better and obtained stability after being exposed to the wind disturbance on an average of 5.5 seconds for all roll, pitch and yaw angles. Oscillating over an amplitude (angles degree error) less than half of that of the PID performance, which enhances the flight trajectory accuracy bounding the quadrotor to the minimal stabilization period.

These comparison results were as expected, the artificial neural network that utilized intelligent algorithms such as backpropagation and gradient decent was able to demonstrate an enormous improvement over the best possible values of a classical PID. It is also worth to mention that not only was that the ANN brought the system to stability over a shorter time, the ANN was also able to sense disturbances and deal with it faster than the PID. This could be due to the integrated intelligence and training experience of the ANN which has made it more responsive even to unknown dynamics of the system.

Although the NN did achieve much better results, some of its drawbacks is the requirement of a more sophisticated hardware to process the edge computing control operations. Edge computing is the capability to undergo mobile computing that requires an enormous decentralized processing power as the data is to be processed by the device/controller itself rather being sent to a sever for its processing. Which does add to the cost burden of the controller design.

7. CONCLUSIONS

In this paper the study, analysis and control of an unmanned aerial vehicle (UAV) quadrotor has been thoroughly conducted with the aim of comparison of different control methods, which were the PID and adaptive neural network controllers.

The motivation to carry out this comparison was the high demand of technological advancements of UAV uses and how there's a global aim to implement them in many of the crucial fields like firefighting, border safety, land exploration, etc.

Over the course of work on this project and as the development of controllers evolved from a PID which was established through into an adaptive intelligent neural network technique to stabilize and output a rational knowledge-based decision as was seen with the wind disturbance.

At first Newton–Euler formalization, was incorporated in a complete control design for the six tilt-rotor's output signals that were used to govern the quadrotors mechanical movement.

Where the PID controller required the exact and specific understanding of the quadrotor's equations of motion for it to be able to stabilize it, which is obviously not very possible to pre-program controllers in our current days into each and every situation that these controllers would have to respond to, and therefore it was necessary to develop a smart/intelligent controller to adapt to those uncertain situations/conditions. This is where the introduction of an artificial intelligent controller comes into light, to cover up the leap of traditional controllers that are being used in machines/devices that are required to be smart and act rationally in uncertain conditions. The AI control techniques are various and are implemented usually depending on the operation or nature of machine/device to be controlled.

The NN controlled quadrotor not only demonstrated a stable flight, but also a much faster response time with a better disturbance suppression. Which was a results of a good backpropagation algorithm that was able to adjust weights on the neural network. All this without being programmed to any specific flight situation and without being fully aware of the dynamics of the system (mathematically or physically). Which makes NN a much more suitable industrial solution to applications that are constantly being exposed to uncertain conditions and environmental disturbances.

My contributions of the paper

1- Modification and simulation of the UAVs for experimenting different control approaches and algorithms to improve quadrotor flight.
2- The ability to integrate the mathematical quadrotor models into the neural network structure.

3- Successful demonstration of NN adaptive control technique for autonomous flight control with the exposure to environmental disturbance.

Future works

There are yet various tools to be developed such as physical robust controllers that are able to withstand harsh shocks/physical conditions during the flight, and many other aspect that will ensure the harmony of work combining physical/dynamic capabilities that certainly need to cope with the code advancements of AI. This will not only result in better UAV performance but in also in low cost ones that would then be considered to be used in many daily activities. Conclusion content comes here

REFERENCES

[1] Gottfredson, L., "Mainstream Science on Intelligence: An Editorial With 52 Signatories, History, and Bibliography," Intelligence, Vol. 24, No. 1, 1997, pp. 13–23, also appeared in Wall Street Jorunal, Dec. 13, 1994.

[2] Salehfard, S, Abdollahi, T, Xiong, C. H & Ai, Y. H. (2018). An Optimized Fuzzy-Padé Controller Applied to Attitude Stabilization of a Quadrotor, International Journal of Control, Automation and Systems.

[3] Basri, M. (2018). Design and application of an adaptive backstepping sliding mode controller for a six DOF quadrotor aerial robot, Robotica.

[4] S. Franko, "Lqr based trajectory control of full envelope, autonomous helicopter," in *Proceedings of theWorld Congress on Engineering*, vol. 1, London, UK, July 2009.

[5] V. G. Nair, M. V. Dileep, and V. I. George, "Aircraft yaw control system using lqr and fuzzy logic controller," *International Journal of Computer Applications*, vol. 45, no. 9, pp. 25–30, 2012.

[6] K., Krishna Kumar. "Intelligent Control Approaches for Intelligent Control Approaches for UAVs." NASA Ames Research Center.

[7] Lakhmi C. Jain and Clarence W. De Silva. Intelligent Adaptive Control: Industrial Applications. CRC Press, Inc., Boca Raton, FL, USA, 1998.

[8] Puttige, Vishwas Ramadas. "NEURAL NETWORK BASED ADAPTIVE CONTROL FOR AUTONOMOUS FLIGHT OF FIXED WING UNMANNED AERIAL VEHICLES." School of Aerospace, Civil and Mechanical Engineering Australian Defence Force Academy University of New South Wales, Aug. 2008.

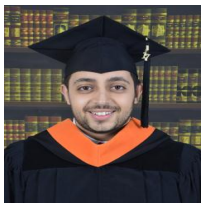
[9] Simon Haykin. Feedforward Neural Networks: An Introduction. Wiley, 2000

[10]J. J. Hop_eld, Neural networks and physical systems with emergent collective computational abilities, Proceedings of the National Academy of Sciences 79 (1982)

[11]Han, Su-Hyun et al. "Artificial Neural Network: Understanding the Basic Concepts without Mathematics." Dementia and neurocognitive disorders vol. 17,3 (2018): 83-89

[12]Shaikh, Faizan. "Introduction to Gradient Descent Algorithm along Its Variants." Analytics Vidhya, 24 June 2019.

BIOGRAPHIES



Andrew Hany Fahim is a graduate of Electrical Engineering from the American University in Dubai. Currently pursuing his Master of Science degree in Electrical Engineering, specializing in Control Systems, at Rochester Institute of Technology (RIT).

Dr Abdulla Ismail obtained his B.Sc. ('80), M.Sc. ('83), and Ph.D. ('86) degrees, all in electrical engineering, from the University of Arizona, U.S.A. Currently, a professor of Electrical Engineering and assistant to the President at the Rochester Institute of Technology, Dubai, UAE.