Perspectives in Communication, Embedded-Systems and Signal-Processing (PiCES)
ISSN: 2566-932X, Volume 2, Issue 8, November 2018
Proceedings of National Conference on Emerging Trends in VLSI, Embedded and Networking (NC-EVEN 18), April 2018

# Adaptive Controller, PID Controller and AI in Small UAV Autopilot System Design

Devmallya Karar[1], Natya S[2]

1. Student, B.E, Dept. of ECE, M.S.Engineering College, Bengaluru, Karnataka, India.
2. Assistant Professor, Dept. of ECE, M.S.Engineering College, Bengaluru, Karnataka, India

***Abstract:*** ***In recent years the Unmanned Aerial Vehicles (UAVs) are use for different applications such as in transportation, surveillance, agriculture, and search and rescue, and also their possible enormous economic impact; UAVs are still it is not used in fully autonomous commercial flights. The main reason is for the safety of the flight. Generally, pilots have to control the aircraft when a complex situation happens where even an advanced autopilots systems are not able to manage. Artificial Intelligence based methods and Adaptive Controllers has proved themselves to be an efficient in these scenarios with uncertainties. However, they also introduce another concern: non-determinism. This paper tries to find a solution on how such algorithms can be utilized. The methods which we used are based on an adaptive model to check the performance of a control parameter—proposed by a nondeterministic adaptive controller or AI-based optimizer—before it is deployed on the physical platform. On the other hand we also used a backup mechanism to recover the drone in case of failure. A Neural Network is used to model the aircraft, and a Genetic Algorithm is applied to optimize the PID controller of a quadcopter. In this embedded system and networking is used for signal transfer.***

***Keywords: Artificial Intelligence; Neural Network; Adaptive Controller; Genetic Algorithm; PID Controller; Embedded System; Quadcopter.***

## I. INTRODUCTION

UAVs are the future of defence system, observation and delivery but there are many issues due continuous advancement of UAVs technology. The major issues with UAVs are to control which can handle power fluctuations, weather-related exigencies and shifting loads. Any of these situations can destabilize the platform in flight.

Some of the methods for tuning control systems, like Ziegler-Nichols classical tuning method for Proportional Integral Derivative (PID) Controllers, are used in many applications but also have drawbacks when applied to dynamic systems. Classical methods and traditional hand tuning assume the parameters which are being set before starting the mission. This type of offline parameter tuning is not able to adapt during the flight, because of that it is difficult to find the issues UAVs have to face today, such as sudden balance changes in package delivery missions, changing weather patterns, minor structural damage, and other issues which are not yet discovered. These kind of issues is required to check on-the-fly by the onboard computer system by adjusting the control parameters in real time, because of this a controller is required which can adapt to this situation and optimize itself based on current circumstances.

Many adaptive approaches and optimization methods like Artificial Intelligence (AI) based algorithms which has been successfully implemented in a variety of nonlinear complex control problems and have shown their efficiency and superiority in reaching suboptimal solutions (e.g., Evolutionary Algorithms like Genetic Algorithm (GA), Neuro Fuzzy methods, and other heuristic approaches) are lacking determinism and the guarantee that the system being controlled will remain in the safe state during the parameter search process or the update transition. The absence of this kind of guarantee is not acceptable in aviation until and unless if it does not meet the high standards of safety assurance where evidence of correct, reliable behaviour of the control system should be shown before the flight.

This paper try to develop a framework for UAVs that enables them to benefit from both the adaptability and performance of AI based algorithms while retaining the stability, robustness, and predictability of classical methods. This approach is based on two-step checking of the modified control parameters. In the first step, the parameters will be checked in the adaptive computational model of the drone to assess whether the proposed parameters are qualified to be updated on the real platform; this approach has the benefit that testing on the virtual model boosts the optimization process significantly by allowing for several hundred checks per minute rather than waiting for the mechanical system to respond, which takes several seconds at least. In the second step, if the parameter update caused too much instability in the platform, the system will roll back to the latest best parameter set immediately to prevent further damage to the UAV.

## II. BACKGROUND AND METHODOLOGY

Determinism and predictability are major concerns when it comes to safety assurance and certification for an airborne system. However, there are many elements in real world flight which are not deterministic, such as environmental situations, equipment failures, and sudden mission changes, which may destabilize the whole



Conference held at Brindavan College of Engineering, Yelahanka, Bagalur Main Rd, Dwarka Nagar, Bengaluru - 560063

system. Today, human pilots are responsible for handling such situations, and the safety of the flight is dependent on the correct decisions of the pilot. On the other hand, with the advent of incredibly inexpensive drones and the hundreds of different applications they can perform, there is a huge interest in automating the entire flight mission. Removing the human pilots from the control loop has two main advantages: first, it cuts the UAV-based service costs significantly, and second, it prevents human error. Now the key question is: How can autopilots safely control drones in the face of all these nondeterministic factors?

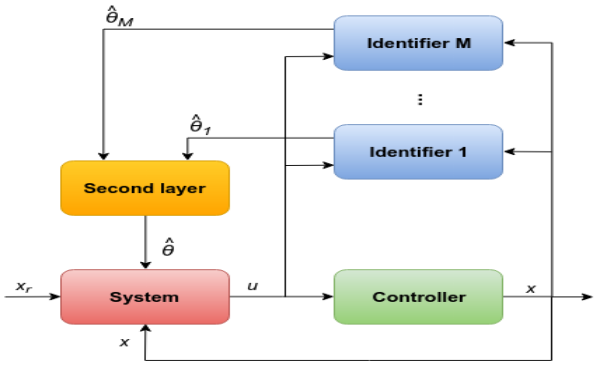



Fig 1. Block diagram of Adaptive Controller

Adaptive Controllers (AC) are capable of changing control parameters online based on the measured performance of the system. Artificial Intelligence (AI) methods are designed to learn from experience and make decisions according to extrapolations from learned information. An AI-based adaptive controller must be able to propose sub-optimal solutions in case of uncertainty, and this method has been successfully tested in several aerospace projects. Such success suggests that this kind of controller can be a good candidate for replacing human pilots. However, it is not always feasible to guarantee the output of such a system because of its inherent non-determinism.

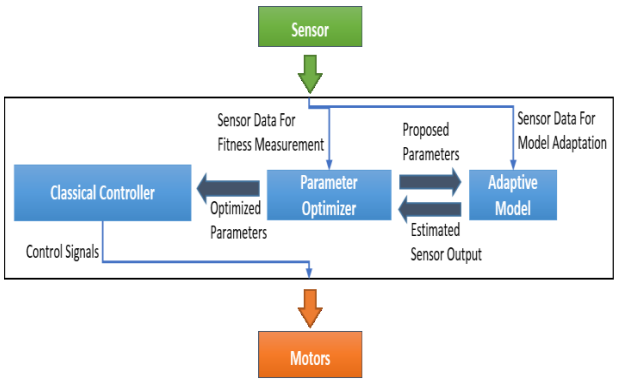



Fig 2. Framework for the adaptive controller

As illustrated in Fig. 2, we have developed a framework in which the adaptive controller consists of three subsystems: classical controller, adaptive model, and parameter optimizer. The classic controller can be any deterministic control algorithm which has been tuned, tested and verified to behave desirably in "normal" situations. The adaptive model can be any numerical model which can simulate the system and use system inputs and outputs to update itself within a certain time constraint. Finally, the parameter optimizer can be any optimization algorithm which alters control parameters to get a higher "fitness" value. The optimizer's offered parameters are nondeterministic, however, and can cause instability in the control system. For this reason, the new parameters will be tested on the model before changing them on the flight controller. If the calculated fitness of the proposed parameters is higher than the current fitness, they will be applied. A further benefit of this method is that it can boost the optimization process in Evolutionary Algorithm (EA) or other Machine Learning (ML) based optimizers due to fast fitness estimation; it is expected that fitness evaluation on a numerical model utilizing today's fast onboard processors could occur hundreds of times faster than testing on the physical platform.

To assess the viability of this approach, an actual quadcopter was built and tested in flight with the following control subsystems. A double PID algorithm was employed as the classic controller to manage the attitude and the location of the drone; a Multi-Layer Perceptron Neural Network represents the adaptive model of the drone; lastly, a Genetic Algorithm optimizes the parameters for the PID controller. The following sections will elaborate on the details of these structures.

## III. MODELLING

This study includes considerations for two drone modelling methods: the classical method and an online learning method.

### *A. Classical Method*

The classical method uses equations that describe the dynamics and kinematics of the quadcopter to determine the relationships between the motors' thrust values and the pose of the drone. Given the thrust of each motor as and the torque applied by the motors in each of the three rotational directions, a linear model for the quadcopter in all six degrees of freedom.

$$\begin{bmatrix} \ddot{\theta} & \ddot{x} \\ \ddot{\phi} & \ddot{y} \\ \ddot{\psi} & \ddot{z} \end{bmatrix} = \begin{bmatrix} \frac{l}{I_{xx}}\tau_\theta & -g\theta \\ \frac{l}{I_{yy}}\tau_\phi & g\phi \\ \frac{l}{I_{zz}}\tau_\psi & -\frac{\sum_{i=1}^{4} T_i}{m} \end{bmatrix}$$

This model relies on many assumptions and the quantity ι, for instance, is the length of the arm from the geometric centre of the quadcopter "cross" to a propeller this is assumed to be the same for all four arms. The assumed symmetry about the centre of mass drives the concept of an inertia matrix I as a purely diagonal matrix. The thrust and torque values in the above equation depend on many factors that must stay fixed over time

and remain identical for all four motors in order for the model to be valid ; these include (but are not limited to) the ratio of motor back-EMF to RPM, the density of the surrounding fluid, and the size, stiffness, and configuration of the propeller. In order to model the motors properly, many component parameters must be taken into account that are either not specified or not precisely defined by the manufacturer(s). This model also overlooks effects due to wind and the drag induced by lateral motion of the vehicle.

It is possible to account for these factors in the model at the cost of greatly complicating it. Another option is to treat many of the abovementioned factors as though they have negligible effects if great care is taken to ensure the symmetric nature of the airframe, the quiescence of the fluid medium, and the selection of closely matched drive components.

*B. Learning Method*

The learning methods use data from actual flight (inputs to the motors and outputs from sensors) to estimate the characteristics of the UAV; accordingly, these methods are inherently adaptive. An online learning approach allows the model to set its own parameters based on feedback from the platform; unlike the classical method, this allows a relatively simple model to handle changes like motor thrust fluctuations due to decreasing battery voltage or temperature, relocation of the centre of mass due to shifting payload, or component wear over time. Forming a non-linear model in the learning process requires a sufficient amount of data. The structure of such a mechanism should be adequate to mimic the dynamics of the system. Sensor readings and motor control signals must be logged and correlated, and signal conditioning (filtering) may need to be applied to the sensor data before it is fit for presentation to the self-learning "brain." Once this problem is solved, the data may need to be processed.

Multi-Layer Perceptron (MLP) Artificial Neural Networks (ANN) have a successful history in modelling and nonlinear robotic control. Calise and Rysdyk applied neural networks to aircraft control almost two decades ago. More recently, Isa and Arshad used a perceptron for an underwater glider, and Farzanegan, Banadaki, and Menhajused this method to control a single-link flexible joint. Zairi and Hazry used a similar system with two hidden layers for stabilizing a quadrotor in flight.

In this paper, six MLP Neural Networks have been designated for modelling the aircraft and a Levenberg Marquardt back propagation algorithm performs the training for the networks. These networks consist of one neuron in the input layer, 10 neurons in a single hidden layer, and one neuron in the output layer (Fig. 2). Each network models the behaviour of one degree of freedom of the robot. Note that a quadrotor has 6 degrees of freedom (pitch, roll, yaw, x, y, z) and the reason we used six networks instead of one network with six inputs and outputs is the point that this network must be trained, tested, and utilized onboard and in real time; consequently, the update mechanism is very time critical and needs to be implemented very efficiently. The fact is, not every change on the UAV necessarily affects all the parameters; in other words, there are incidents which only affect one or two parameters. As a result, there is no need to update the network as a whole. This discrete implementation allows more parallelism and faster convergence especially in multicore hardware. Figure 3 demonstrates the convergence of the neural network and the optimal result is obtained at 130th epoch in 3 seconds. It is noteworthy to mention that the training process has been executed off-board on an Intel Core i7 processor; however, our plan is to utilize a NVidia CUDA enabled Google Tango tablet to perform GPU-based parallel NN modelling on board and we expect execution in that platform takes reasonable amount of time.

Since measured signals are often noisy, and the chosen learning method has a neural network at heart, smoothing greatly aids convergence of the model. In this case, we employed Savitzky-Golay filtering to clean the signal without loss of amplitude. As depicted in figure 4, the filter smoothed away the high frequency noise in exemplary fashion and these filtered inputs and corresponding outputs fed the network. Note that in this graph "raw data" represents the autopilot output which in fact is processed by Extended Kalman Filter (EKF); however, for the next phase it will be treated as a raw signal.

Our network's inputs are based on a combination of the four PWM signals controlling the four motors. These combinations were selected to ensure interactive effects among the motors would affect the output of the neural net.

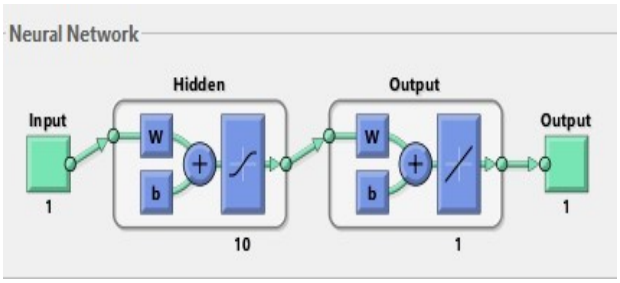



Fig 3. Neural Network configuration

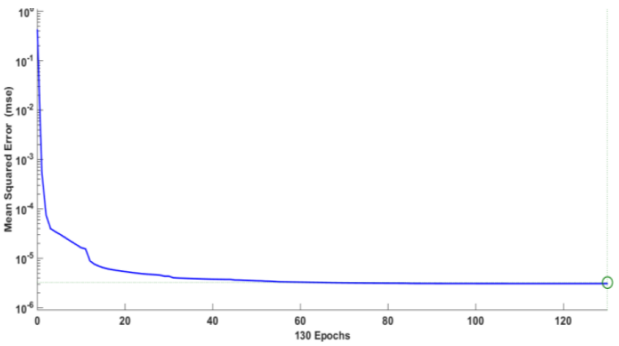



Fig 4. Neural Network Training Error Convergence

Let $PWM_i$ represent the control signal for motor i $\in$ {1,2,3,4}. The inputs and outputs of the model are describes in table I.

Table 1. Inputs and outputs of the Network.

| Network# | Input | Output |
|---|---|---|
| 1 | $PWM_3 + PWM_4 - PWM_1 - PWM_2$ | $\ddot{\psi}$ |
| 2 | $PWM_1 + PWM_4 - PWM_2 - PWM_3$ | $\ddot{\Phi}$ |
| 3 | $PWM_1 + PWM_3 - PWM_2 - PWM_4$ | $\ddot{\theta}$ |
| 4 | $PWM_1 + PWM_2 + PWM_3 + PWM_4$ | $\ddot{Z}$ |
| 5 | $\Phi$ | $\ddot{Y}$ |
| 6 | $\theta$ | $\ddot{X}$ |

## IV. OPTIMIZATION METHOD

Genetic algorithms (GA) have been widely used to help optimize PID controllers and have proven their efficiency and performance in many applications. Paper shows that this method has also been tested on simplified models of quadcopter and is confirmed to be successful in finding optimized coefficients for PID controllers. Genetic Algorithms are built on the ideas and mechanics of natural selection. Starting with no knowledge of an optimum solution, responses from the environment cause selection operator to create generations of candidate solutions that are refined to arrive at an optimum or near-optimum solution.

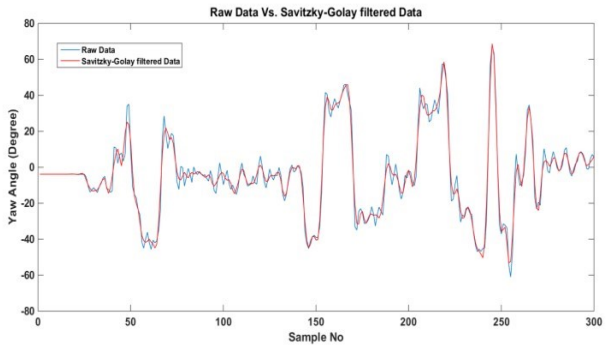



Fig 5. Savitzky-Golay filter used for data smoothing

Some terminology is needed to understand how GA is implemented here. In this case, a gene is a number which represents one of the PID coefficients. A chromosome is an array of all the PID coefficients that the autopilot uses to control the drone in one axis, and a generation is a set of 10 chromosomes.

There are some issues in using GA as an online control optimizer for UAVs which must be addressed: first, the time of convergence is not guaranteed, which may lead to relatively long training times to find an optimal solution. In this situation the battery life of the drone (in other words, the 'fly time') is smaller than the time needed to gain an optimum solution. Offline optimizers overcome this challenge by making multiple flights to gain the solution, i.e. using previous values to seed new values; however, this solution is not practical for online optimizers. The second problem is that sometimes the generated solution is not a proper control parameter and may cause enormous instability in the drone or even cause the drone to crash. To overcome this problem, we utilize our adaptive model to estimate the fitness value of the system. Only if the estimated fitness is higher than the current fitness will it be evaluated on the real platform. This procedure both speeds up the process and prevents inappropriate parameters that can cause failed off-springs from being sent to the actual platform. Moreover, to maintain optimum performance, the parameters that perform the best and have been tested on the platform (including initial manual parameters) will be kept as the backup parameter set. Thus, even if the adaptive model failed to recognize an incorrect control parameter and it is applied to the physical system, if the resulting behaviour causes state measurements to exceed their limits, the system will automatically rollback the parameters to the last best values, and a fitness value of 0 will be assigned to that chromosome. Since quadcopters are inherently unstable and agile, the transition takes place very quickly and the drone is able to recover in most cases.

As shown in figure 6, the optimization process starts with initialization of the generation which will be created randomly. This step is followed by computation and evaluation of the fitness functions. After this, the algorithm uses crossover, mutation, and random chromosome generation operators to shape the next generation, with repetition of steps 2 and 3 until an optimum solution is found. Each fitness value is a number between 0 and 100, and higher fitness means better performance. For measuring fitness, the following parameters are considered: Overshoot / undershoot, settling time, mean error, and error variance.

When selecting the finalists of the previous generation there are four qualifications in order to pass to the next generation.

- The top two (highest fitness values in the generation) chromosomes are transferred directly to next generation
- Four pairs of chromosomes are selected randomly for crossover, with the selection probability proportional to their fitness values.
- Two random chromosomes are selected randomly for mutation.

One new random chromosome is generated in each generation.

As depicted in Fig. 7, after 15 generations the fitness of yaw controller reaches 44 while the best manually tuned PID controller in our experiments could not get fitness above 35.

### *A. Uses of PID Controllers*

The quadcopter PID tuning is required to train the copter. The balancing the different aspects of the flight characteristics to make the craft respond perfectly for a particular flying style is required. We want a quad to feel snappy, but without oscillations, it is not at all possible to fly. The key is finding where the quadcopter is balancing or not. A working knowledge of PID tuning will help us to achieve this, and the more we work with PID settings, it will be easier to tune our quads to fly exactly the way we want them to. Back in the early days the flight controller firmware was not optimized. A quadcopter would always fly badly with default PID values, which made PID tuning absolutely essential. But that's no longer necessarily the case (at least for mini quads), the sophisticated noise filtering and optimized algorithms in modern FC software, improvements have enabled quadcopters to fly great out of the box.


Fig 6. Genetic Algorithm flowchart

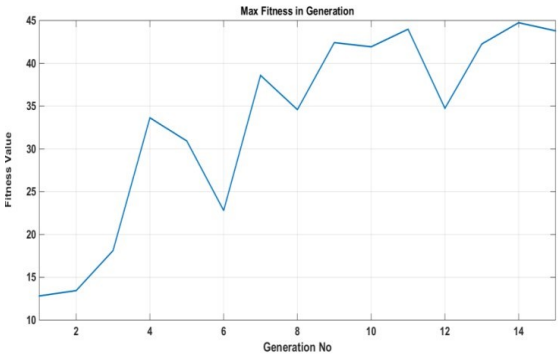
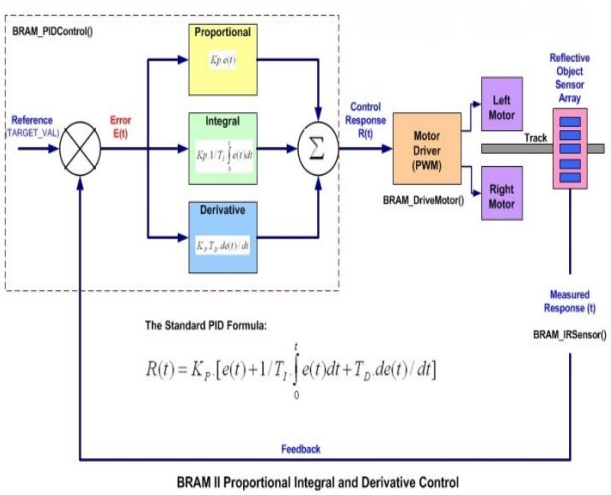



Fig 7. Best fitness per generation

That's not to say we can forget about PID tuning, there is always different way to the improvement in a quad's performance. Knowing how to tune PID provides the capability to change a quad that "flies well", into one that "flies perfectly" for your individual style.

- "P" is the present error, the further it is from the set-point, the harder it pushes.
- "D" is a prediction of future errors, it is that how fast you are approaching a set-point and counteracts P when it is getting close to minimize overshoot.
- "I" is the accumulation of past errors, it forces that happen over time; for example if a quad constantly drifts away from a set-point due to wind, it will speed up the motors to counteract it.




BRAM II Proportional Integral and Derivative Control

Fig 8. Block diagram of PID Controller.

From the PID controller reading sensor data to calculating the output required a process which is called a "loop". Modern flight controllers in racing drones are capable of doing thousands of "loops" per second.

Fig 9. Quadcopter.

The time takes for the flight control to complete a loop, is called "loop time". Loop time can be measured in millisecond, but it's measured in Hz. For example:

- A loop that takes 1 second = 1 cycle per second = 1Hz
- A loop that takes 1ms (0.001 second) = 1KHz

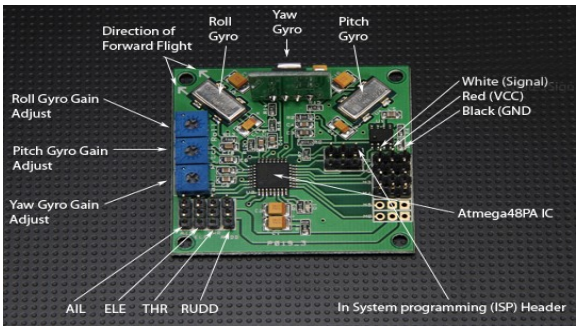



Fig 10. PID Controller Embedded board.

It is now quite common to see flight controllers that are capable of doing 8 KHz loop time, some can even do up to 32 KHz. But whether faster is better or not, that's another long topic. There are pro's and con's doing 32KHz, so many people prefer to stick with 8KHz or even lower loop time. There are many types of embedded circuit board which are being use to control the Quadcopter. Below figure 10it is one kind of board which is being used.

## V. CONCLUSION AND FUTURE WORKS

This paper proposed a method to implement nondeterministic algorithms for controlling UAVs with higher reliability. This method is based on 1) predicting the behaviour of the controller by testing it online on an adaptive model before the control signals are applied to the UAV and 2) constantly monitoring the performance of the controller after applying the changes proposed by the optimizer and recovering the platform in case of unfavourable behaviour. To test the practicality and performance of this method an extremely unpredictable optimizer (Genetic Algorithm) was implemented to optimize the PID coefficients of a quadcopter and a MLP neural network was designed to model the physical platform based on the inputs and outputs of the system. The experimental results suggest that this method is feasible; however, we need to perform more experiments with different methods of modelling, various AI-based and classic optimizers, and inherently adaptive controllers to be able to claim the complete success of this approach confidently.

## REFERENCES

[1] S. Bhattacharyya, D. Cofer, D.J. Muliner, J Mueller, E. Engstorm, "Certification considerations for adaptive systems," NASA STI Program/Mail Stop 148 Report, NASA Langlet Research Center, Hampton, Virginia 23681-2199, 2015.

[2] M. Sharma and A. J. Calise, "Neural-network augmentation of existing linear controllers," Journal of Guidance, Control, and Dynamics, vol. 28, no. 1, pp.12–19, 2005.

[3] K. Wise, E. Lavretsky, and N. Hovakimyan, "Adaptive control of flight: theory, applications, and open problems," in American Control Conference, 2006, pp. 5966–5971. IEEE, 2006.

[4] K. A.Wise, J. S. Brinker, A. J. Calise, D. F. Enns, M. R. Elgersma, and P. Voulgaris, "Direct adaptive reconfigurable flight control for a tailless advanced fighter aircraft," International Journal of Robust and nonlinear Control, vol.9, no. 14, pp. 999–1012, 1999.

[5] K. N. Maleki, "A platform independent evolutionary approach to UAV optimized control," University of Tulsa, Tulsa, 2015.

[6] A. Calise and R. Rysdyk, "Nonlinear adaptive flight control using neural networks," IEEE Control Systems, vol. 18, no. 6, pp. 14-25, 1998.

[7] K. Isa and M. R. Arshad, "Neural networks control of hybrid-driven underwater glider," in Oceans - Yeosu, Yeosu, 2012.

[8] B. Farzanegan, S. D. Banadaki and M. B. Menhaj, "Direct artificial neural network control of single link flexible joint," in 4th International Conference on Control, Instrumentation, and Automation (ICCIA), Qazvin, Iran, 2016.

[9] S. Zairi and D. Hazry, "Adaptive neural controller implementation in autonomous mini aircraft quadrotor (AMAC-Q) for attitude control stabilization," in Signal Processing and its Applications (CSPA), 2011 IEEE 7th International Colloquium on, Penang, 2011.

[10] A. Gibiansky, "Quadcopter dynamics and simulation," 23 November 2012. [Online]. Available: http://andrew.gibiansky.com/blog/physics/quadcopter-dynamics/.

[11] F.A.T Al-Saedi, R.A. Sabar, "Design and implementation of autopilot system for quadcopter," IJCSET, Vol 5, Issue 6,190-199, June 2015.

[12] A. Jayachitra and R. Vindoha, "Genetic algorithm based PID controller tuning approach for continuous stirred tank reactor," Journal of Advances in Artificial Intelligence, Volume 2014, Article No.9, 2014.

[13] H. Noshahri, H. Kharrati, "PID controller design for unmanned aerial vehicle using genetic algorithm," IEEE/ISIE 23rd International Symposium on Industrial Electronics, pp. 213-217, 2014.

[14] A. Löfwenmark and S. Nadjm-Tehrani, "Challenges in future avionic systems on multi-core platforms," Software Reliability Engineering Workshops (ISSREW), 2014 IEEE International Symposium on, Naples, 2014, pp. 115-119.

[15] J. Littlefield-Lawwill and L. Kinnan, "System considerations for robust time and space partitioning in integrated modular avionics," 2008 IEEE/AIAA 27th Digital Avionics Systems Conference, St. Paul, MN, 2008, pp. 1.B.1-1-1.B.1-11

[16] S. H. VanderLeest, "Taming interrupts: deterministic asynchronicity in an ARINC 653 environment," 2014 IEEE/AIAA 33rd Digital Avionics Systems Conference (DASC), Colorado Springs, CO, 2014, pp. 8A3-18A3-11