Tuning of PID Controllers for Quadcopter System using Hybrid Memory based Gravitational Search Algorithm – Particle Swarm Optimization

Nizar Hadi Abbas, PhD
Assistant Professor
College of Engineering,
University of Baghdad.

Ahmed Ramz Sami
Research Scholar
College of Engineering,
University of Baghdad.

ABSTRACT

Quadrotors are coming up as an attractive platform for unmanned aerial vehicle (UAV) research, due to the simplicity of their structure and maintenance, their ability to hover, and their vertical take-off and landing (VTOL) capability. With the vast advancements in small-size sensors, actuators and processors, researches are now focusing on developing mini UAV's to be used in both research and commercial applications. This work presents a detailed mathematical nonlinear dynamic model of the quadrotor which is formulated using the Newton-Euler method. Although the quadrotor is a 6 DOF under-actuated system, the derived rotational subsystem is fully actuated, while the translational subsystem is under-actuated. The derivation of the mathematical model was followed by the development of the controller to control the altitude, attitude, heading and position of the quadrotor in space, which is, based on the linear Proportional-Derivative- Integral (PID) controller; thus, a simplified version of the model is obtained. The gains of the controllers will be tuned using optimization techniques to improve the system's dynamic response. The standard Gravitational Search Algorithm (GSA) was applied to tune the PID parameters and then it was compared to Hybrid Memory Based Gravitational Search Algorithm – Particle Swarm Optimization tuning, and the results shows improvement in the new algorithm, which produced enhancements by (40.126%) compared to the standard algorithm.

General Terms

Optimization Algorithm, gravitational search algorithm, Hybrid memory based gravitation search algorithm, Quadcopter control, PID tuning, heuristic optimization algorithm.

Keywords

Quadcopter, PID, UAV, Flying robot, GSA, MBGSA-PSO.

1. INTRODUCTION

The quadrotor unmanned aerial vehicle (UAV) have been an increasingly popular research topic in recent years due to their low cost, maneuverability, and ability to perform a variety of tasks. They are equipped with four motors typically designed in a "X" configuration, each two pairs of opposite motors rotating clockwise and the other motor pair rotating counter-clockwise to balance the torque. Compared with the conventional rotor-type aircraft, has no tail, quadrotors advantage compared with normal airplanes is that it has more compact structure and it can hover. Quadcopter are used in many fields like search and rescue, area mapping, aerial photography, inspection of power lines, traffic monitoring, crop spraying, border patrol, surveillance for illegal imports and exports and fire detection and control.

The quadrotor does not have complex mechanical control system because it relies on fixed pitch rotors and uses the variation in motor speed for vehicle control. However, these advantages comes with disadvantage. Controlling a quadrotor is not easy because of the coupled dynamics and its commonly under-actuated design configuration and uses more energy of battery to keep it position in the air. In addition, the dynamics of the quadrotor are highly non-linear and several effects are encountered during its flights, thus making its control a good research field to explore. This opened the way to several control algorithms and different optimization technique for tuning that are proposed in the literature.

Going through the literature, one can see that most of the papers use two approach for modelling the mechanical model. They are Newton-Euler and Newton-Lagrange but the most used one and it is most familiar and better in modelling is Newton-Euler [2, 3, 6]. Also regarding the control of quadcopter, there is a good amount of research done both in linear and nonlinear control methods, [8]. However, for practicality the linear controller especially PID perform better in practical implementation and that is the reason PID was used as the controller for the quadcopter in this work [5, 9, 10]. In addition, some researchers work on the stabilization of quadcopter that means inner loop [11, 13]. And other researchers include the position control of quadcopter to cover all topics in quadcopter control [12]. Tuning the parameters of the PID controllers is very important to get the best performance of quadcopter. Thus in order to get the best (optimal) setting to achieve the objective an optimization algorithm must be used. In this work the Gravitational Search Algorithm was applied which is inspired by [1], then it was improved to generate new Hybrid algorithm called Memory Based Gravitational Search Algorithm – Particle Swarm Optimization (MBGSA) which achieved better results it combine the work in [14, 15]. As much of research done in this field still not all tuning algorism was tested, Gravitational Search Algorithm is one of them. Also this work explore new optimization algorithm which performed better. Both algorithms were measured statically and it turns out that the new algorithm is performing better than the normal algorithm always in getting lower objective value (47.04%) lower.

This paper discuss the formulation of mathematical model of quadcopter in section 2. After it the control system of the quadcopter which is PID and how to apply it to stabilize the quadcopter and control its position in section 3. In section 4 will be about the optimization techniques used tune the PID parameter Kp, Ki and Kd and how to enhanced the algorithm for tuning the parameters more effectively and better than the normal one. Section 5 will contain the simulation results and the discussion in section 6.

2. MATHEMATICAL MODEL

Designing a control system for physical systems is commonly started by building a mathematical model. The model is very important because it gives an explanation of how the system acts to the inputs given to it. In this research, the mathematical model equations of motion are derived using a full quadcopter with body axes as shown in **Fig. 1**.

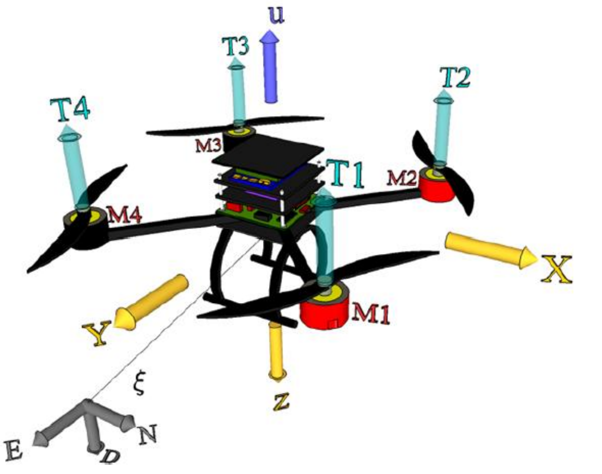

Fig 1: Quadcopter at hovering state with the main acting forces and torques.

2.1 Quadcopter Fundamentals

The quad-rotor is a helicopter equipped with four motors and propellers mounted on them. It is very well modelled with a cross (X) configuration style. Each opposite motors rotate in the same direction counter-clockwise and clockwise. This configuration of opposite pairs' directions completely eliminates the need for a tail rotor, which is needed for stability in the conventional helicopter structure. **Fig. 1** shows the model in stable hover, where all the motors rotate at the same speed, so that all the propellers generate equal lift and all the tilt angles are zero [6].

Four basic movements govern the quadcopter movement, which allow the quadcopter to reach a certain altitude and attitude and they are;

a) Throttle

It is provided by concurrently increasing or decreasing all propeller speeds with the same amount and rate. This generates a cumulative vertical force from the four propellers, with respect to the body-fixed frame. As a result, the quadcopter is raised or lowered by a certain value.

b) Roll

It is provided by concurrently increasing or decreasing the left propellers speed and decreasing or increasing the right propellers speed at the same rate. It generates a torque with respect to the x axis which makes the quadcopter to tilt about the axis, thus creating a roll angle. The total vertical thrust is maintained as in hovering; thus this command leads only to a roll angular acceleration.

c) Pitch

The pitch and roll are very similar. It is provided by concurrently increasing or decreasing the speed of the rear propellers and by decreasing or increasing the speed of the front propellers at the same rate. This generates a torque with respect to the y axis which makes the quad-rotor to tilt about the same axis, thereby creating a pitch.

d) Yaw

This command is provided by increasing (or decreasing) the opposite propellers' speed and by decreasing (or increasing) that of the other two propellers. It leads to a torque with respect to the z axis which makes the quadrotor turn clock wise or anti clock wise about the z-axis.

2.2 Quadcopter Dynamics

There are two coordinate systems to be considered in quadcopter dynamics, shown in **Fig. 2**:

- The earth inertial frame (E-frame)
- The body-fixed frame of the vehicle (B-frame)

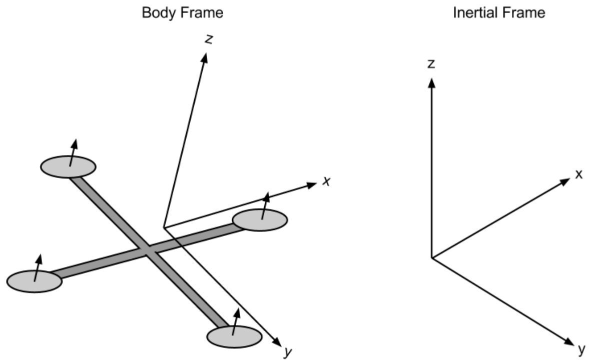

Fig 2: Inertial frame and body frame of quadcopter.

They are related through three successive rotations:

- Roll: Rotation of φ around the x-axis $\mathbf{R_x}$;
- Pitch: Rotation of θ around the y-axis $\mathbf{R_y}$;
- Yaw: Rotation of ψ around the z-axis $\mathbf{R_z}$.

Switch between the coordinates system using rotation matrix **R** which is a combination of rotation about 3 mentioned rotations also **Fig. 3** shows how rotation is applied [12]:

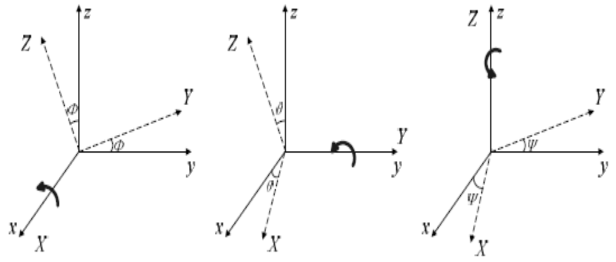

Fig 3: Rotations about X, Y, and Z axes.

$$\boldsymbol{Rx} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\,\phi & -\,sin\,\phi \\ 0 & sin\,\phi & cos\,\phi \end{pmatrix} \quad (1)$$

$$\boldsymbol{Ry} = \begin{pmatrix} cos\,\theta & 0 & sin\,\theta \\ 0 & 1 & 0 \\ -sin\,\theta & 0 & cos\,\theta \end{pmatrix} \quad (2)$$

$$\boldsymbol{Rz} = \begin{pmatrix} cos\,\psi & -sin\,\psi & 0 \\ sin\,\psi & cos\,\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (3)$$

$$\boldsymbol{R} = Rx\,.Ry\,.Rz = \begin{pmatrix} c\,\psi\,c\,\phi & c\,\psi\,s\,\theta\,s\,\phi & c\,\psi\,s\,\theta\,c\,\phi + s\,\psi\,s\,\phi \\ s\,\psi\,c\,\theta & s\,\psi\,s\,\theta\,s\,\phi & s\,\psi\,s\,\theta\,c\,\phi - s\,\phi\,c\,\psi \\ -s\,\theta & c\,\theta\,s\,\phi & c\,\theta\,c\,\phi \end{pmatrix} \quad (4)$$

Where c refers to cosine and s refers to sine.

The developed model in this work assumes the following:

- The structure is supposed rigid.
- The structure is supposed symmetrical.
- The center of gravity and the body fixed frame origin are assumed to coincide.
- The propellers are supposed rigid.
- Thrust and drag are proportional to the square of propeller's speed.

Using the Newton-Euler formalism to create a nonlinear dynamic model for the quadrotor, in this research, the Newton-Euler method is employed for both the main body and rotors. The general form of Newton-Euler equation is expressed as:

$$\begin{bmatrix} mI_{3*3} & 0 \\ 0 & I \end{bmatrix} \begin{Bmatrix} \dot{V} \\ \dot{\omega} \end{Bmatrix} + \begin{Bmatrix} \omega \times mV \\ w \times I\omega \end{Bmatrix} = \begin{Bmatrix} F \\ \tau \end{Bmatrix} \tag{5}$$

Eq. (5) is a generic form of equations of motion. It can be applied in any position in the coordinate system. In this case, the main point is the center of mass of the quadcopter. Referring to the body frame (B) in **Fig. 3**, Eq.(5) cut-down to:

$$\begin{bmatrix} mI_{3*3} & 0 \\ 0 & I \end{bmatrix} \begin{Bmatrix} \dot{V} \\ \dot{\omega} \end{Bmatrix} + \begin{Bmatrix} 0 \\ w \times I\omega \end{Bmatrix} = \begin{Bmatrix} F \\ \tau \end{Bmatrix} \tag{6}$$

Regarding the main body of the quadcopter and Eq. (6), the translational dynamics of the quadcopter in the body frame (B) is defined as:

$$m \begin{Bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{Bmatrix}_B = \begin{Bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} b\Omega_i^2 \end{Bmatrix} - R_{xyz}^{-1} \begin{Bmatrix} 0 \\ 0 \\ mg \end{Bmatrix} \tag{7}$$

Which will be described in the earth frame (E) **Fig. 3** through Eq. (7) as:

$$m \begin{Bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{Bmatrix}_E = R_{xyz}^{a} m \begin{Bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{Bmatrix}_B \tag{8}$$

$$= R_{xyz}^{a} \begin{Bmatrix} 0 \\ 0 \\ \sum_{i=1}^{4} b\Omega_i^2 - R_{xyz}^{-1} mg \end{Bmatrix} \tag{9}$$

$$m \begin{Bmatrix} \ddot{X} \\ \ddot{Y} \\ \ddot{Z} \end{Bmatrix}_E = R_{xyz}^{a} \begin{Bmatrix} -sin\theta\cos\phi \sum_{i=1}^{4} b\Omega_i^2 \\ sin\varphi \sum_{i=1}^{4} b\Omega_i^2 \\ -mg + cos\theta\cos\phi \sum_{i=1}^{4} b\Omega_i^2 \end{Bmatrix} \tag{10}$$

From Eq. (6), the main body rotational dynamics can be described in the body frame (B) as

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{Bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{Bmatrix} + \begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix} = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix} \tag{11}$$

$$\begin{Bmatrix} I_{xx}\ddot{\phi} \\ I_{yy}\ddot{\theta} \\ I_{zz}\ddot{\Psi} \end{Bmatrix} + \begin{Bmatrix} \dot{\theta}\dot{\Psi}\left(I_{zz} - I_{yy}\right) \\ \dot{\phi}\dot{\Psi}(I_{xx} - I_{zz}) \\ \dot{\phi}\dot{\theta}\left(I_{yy} - I_{xx}\right) \end{Bmatrix} = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_e + \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_r \tag{12}$$

Where subscripts e and r refer to moments due to external forces, which ultimately caused by thrust and drag from rotors, and moments due to rotor gyro effect, respectively.

2.3 Rotor Dynamics

The rotor dynamics can be described using Eq. (5) by considering coordinate system of each rotor, which is the same plane as the body frame for X and Y axes while Z axis coincides with rotation of the rotor. Considering rotational dynamics of each rotor in the form of Newton-Euler:

$$\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}_{r,i} \begin{Bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{Bmatrix}_{r,i} + \begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix}_{r,i} \times \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}_{r,i} \begin{Bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{Bmatrix}_{r,i} = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_{r,i} \tag{13}$$

$$\begin{Bmatrix} I_{xx}\ddot{\phi} \\ I_{yy}\ddot{\theta} \\ I_{zz}\ddot{\Psi} \end{Bmatrix}_{r,i} + \begin{Bmatrix} \dot{\theta}\dot{\Psi}\left(I_{zz} - I_{yy}\right) \\ \dot{\phi}\dot{\Psi}(I_{xx} - I_{zz}) \\ \dot{\phi}\dot{\theta}\left(I_{yy} - I_{xx}\right) \end{Bmatrix}_{r,i} = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_{r,i} \tag{14}$$

for i= 1,2,3,4 and denotes i^{th} rotor. Since the rotors always rotate about their Z-axes at the rate of Ω with the moment of inertia about Z-axis of J_r and have very low masses, $I_{xx}$and I_{yy} can then be omitted and the dynamics of each rotor reduces to:

$$\begin{Bmatrix} 0 \\ 0 \\ J_r\dot{\Omega} \end{Bmatrix}_i + \begin{Bmatrix} \dot{\theta}\Omega J_r \\ -\dot{\phi}\Omega J_r \\ 0 \end{Bmatrix}_i = \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_{r,i} \tag{15}$$

Note that Eq. (15) is a function of rotor speed Ω. Since rotor 1 and 3 rotate in the opposite direction of rotor 2 and 4, one can define the total rotor speed as:

$$\Omega = \Omega_1 - \Omega_2 + \Omega_3 - \Omega_4 \tag{16}$$

From Eq. (14) and (15), the total moment due to gyro effect from all rotors can be expressed as:

$$\begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_r = \sum_{i=1}^{4} \begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_{r_a}$$

$$= \sum_{i=1}^{4} \left[\begin{Bmatrix} 0 \\ 0 \\ J_r\dot{\Omega} \end{Bmatrix}_i + \begin{Bmatrix} \dot{\theta}\Omega J_r \\ -\dot{\phi}\Omega J_r \\ 0 \end{Bmatrix}_i \right]_a \tag{17}$$

$$\begin{Bmatrix} \tau_x \\ \tau_y \\ \tau_z \end{Bmatrix}_r = \begin{Bmatrix} \dot{\theta}\Omega_r J_r \\ -\dot{\phi}\Omega_r J_r \\ J_r\dot{\Omega}_r \end{Bmatrix}_a \tag{18}$$

2.4 Equation of Motion (EoM)

No Now that all necessary dynamics of the entire model has been established, one can write the complete equations of motion of the quadrotor. Combining Eq. (4), (10), (12), (14), (15) and (18) yields:

$$\begin{cases} translational \begin{cases} m\ddot{X} = -sin\theta cos\phi \sum_{i=1}^{4}[b\Omega_i^2] \\ m\ddot{Y} = sin\phi \sum_{i=1}^{4}[b\Omega_i^2] \\ m\ddot{Z} = -mg + cos\theta cos\phi \sum_{i=1}^{4}[b\Omega_i^2] \end{cases} \\ Rotational \begin{cases} I_{xx}\ddot{\phi} = \dot{\theta}\dot{\Psi}\left(I_{zz} - I_{yy}\right) - \dot{\theta}\Omega_r J_r + lb(-\Omega_2^2 + \Omega_4^2) \\ I_{yy}\ddot{\theta} = \dot{\phi}\dot{\Psi}(I_{xx} - I_{zz}) + \dot{\phi}\Omega_r J_r + lb(\Omega_1^2 - \Omega_3^2) \\ I_{zz}\ddot{\Psi} = \dot{\phi}\dot{\theta}\left(I_{yy} - I_{xx}\right) + J_r\dot{\Omega}_r + \sum_{i=1}^{4}[(-1)^i d\Omega_i^2] \end{cases} \end{cases} \tag{19}$$

2.5 Model Parameters

The quadcopter parameters was calculated practically by taking the measurements from the DJI F-450 frame and the applying two tests on a2212 13t 1000kv brushless motor to calculated thrust factor and drag factor, also the total weigh was calculated all the results can be seen in **Table 1**.

Table 1. Model parameter which is taken from real quadcopter system datasheets.

Symbol	Value	Description
m	0.964 Kg	Total mass of quadcopter
l	0.22 m	Distance form center of quadcopter to the motor
I_{xx}	8.5532×10^{-3}	Quadcopter moment of inertia around X axes
I_{yy}	8.5532×10^{-3}	Quadcopter moment of inertia around Y axes
I_{zz}	1.476×10^{-2}	Quadcopter moment of inertia around Z axes
J_r	5.225×10^{-5}	Rotational moment of inertia around the propeller axis
b	7.66×10^{-5}	Thrust coefficient
d	5.63×10^{-6}	Drag coefficient

3. PID CONTROL

PID controllers have been used in a broad range of controller applications. It is for-sure the most applied controller in industry. The PID controller shown in Fig. 4 has the advantage that parameters (Kp, Ki, Kd) are easy to tune, which is simple to design and has good robustness. However, the quadcopter includes non-linearity in the mathematical model and may include some inaccurate modeling of some of the dynamics, which will case bad performance of the control system, thus it is needed form the designer to be careful when neglecting some effects in the model or simplifying the model [9].

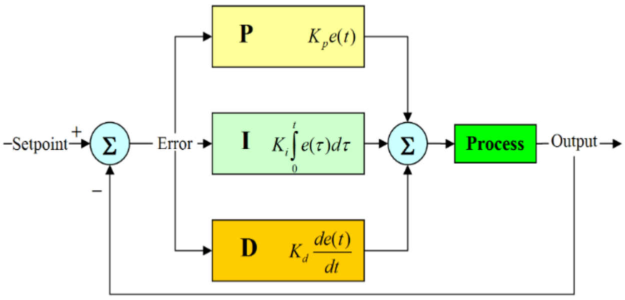

Fig 4: General structure of PID controller.

Stabilization is very important for an under-actuated system like a quadrotor, as it is inherently unstable due to its six degrees of freedom and four actuators. A control system is modeled for the quadcopter using four PID controllers to control the Attitude (roll, pitch and yaw) and the Altitude (Z height) to introduce stability these four controller form the inner loop of control for the quadcopter system. And then two more PID controllers are used to control the position of the quadcopter (X and Y axes) and the output of these two controllers will be input to the roll and pitch controllers these two PID's will form the outer control loop**.**

3.1 Inner Loop Control

Inner control loop controls quadcopter altitude and attitude. Input variables for inner loop can be divided in two parts, desired and sensor signals. Desired signals are obtained from the control signals coming directly form the pilot or the autopilot program. These signals are the Height (altitude) and pointing (Yaw) of the quadcopter the other two signals desired roll and pitch comes for the output of the outer loop control since they are translated form the desired x and y position in the outer PID's. **Fig .5** show the complete control system for the quadcopter including the inner loop and the outer loop.

Altitude control

Equation for the thrust force control variable $U1$ is:

$$U_1 = K_{PZ}e_Z + K_{IZ}\int e_Z - K_{DZ}\frac{d}{dt}e_Z \qquad (20)$$

Where K_{PZ}, K_{IZ} and K_{DZ} are three altitude PI-D controller parameters. e_Z is the altitude error, where $e_Z = Z_{des} - Z_{mes}$. Z_{des} is the desired altitude and Z_{mes} is the measured altitude.

Roll control

Equation for the roll moment control variable $U2$ is:

$$U_2 = K_{P\phi}e_\phi + K_{I\phi}\int e_\phi - K_{D\phi}\frac{d}{dt}e_\phi \qquad (21)$$

Where $K_{P\phi}$, $K_{I\phi}$ and $K_{D\phi}$ are three roll angle PI-D controller parameters. e_ϕ is the roll angle error, where $e_\phi = \phi_{des} - \phi_{mes}$. ϕ_{des} is the desired roll angle and ϕ_{mes} is the measured roll angle.

Pitch control

Equation for the pitch moment control variable $U3$ is:

$$U_3 = K_{P\theta}e_\theta + K_{I\theta}\int e_\theta - K_{D\theta}\frac{d}{dt}e_\theta \qquad (22)$$

Similar to the roll control, $K_{P\theta}$, $K_{I\theta}$ and $K_{D\theta}$ are three pitch angle PI-D controller parameters. e_θ is the pitch angle error, where $e_\theta = \theta_{des} - \theta_{mes}$. θ_{des} is the desired pitch angle and θ_{mes} is the measured pitch angle.

Yaw control

Equation for the yaw moment control variable $U4$ is:

$$U_4 = K_{P\Psi}e_\Psi + K_{I\Psi}\int e_\Psi - K_{D\Psi}\frac{d}{dt}e_\Psi \qquad (23)$$

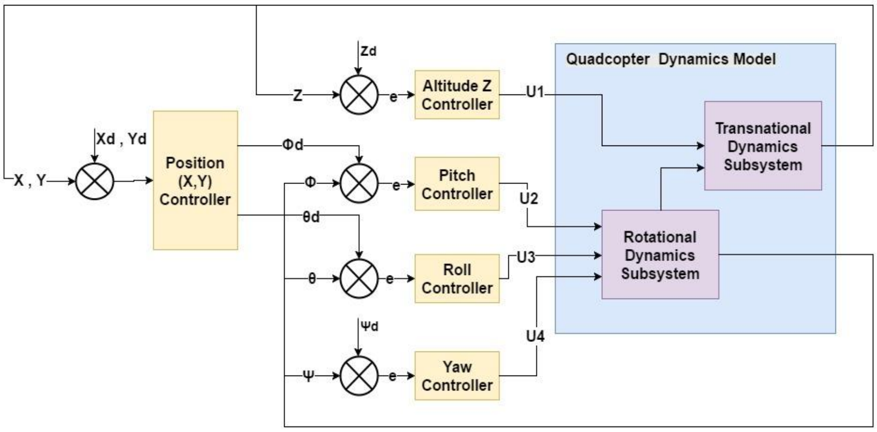
Zd
Z
e
Altitude Z Controller
U1
Quadcopter Dynamics Model
Transnational Dynamics Subsystem
Xd , Yd
X , Y
Position (X,Y) Controller
Φd
Φ
e
Pitch Controller
U2
θd
θ
e
Roll Controller
U3
Rotational Dynamics Subsystem
Ψd
U4
Ψ
e
Yaw Controller

Fig 5: Block diagram of the complete quadcopter control

Where $K_{P\Psi}$, $K_{I\Psi}$ and $K_{D\Psi}$ are three yaw angle PI-D controller parameters. e_{Ψ} is the yaw angle error, where $e_{\Psi} = \Psi_{des} - \Psi_{mes}$. Ψ_{des} is the desired yaw angle and Ψ_{mes} is the measured yaw angle.

3.2 Outer Control Loop

Outer control loop is applied since the quadcopter is under-actuated system and it is not applicable to control all of the quadcopter 6-DOF straightly. As mentioned earlier, the inner loop directly controls 4-DOF (three angles and altitude). To be able to control X and Y position, outer loop is implemented. The outer control loop outputs are desired roll and pitch angles, which they are the inputs to the inner loop for the desired X and Y position.

Eq. (24) and (25) are the equations for the quadcopter X and Y linear accelerations:

$$\ddot{X} = (cos\Psi sin\theta cos\phi + sin\Psi sin\phi)\frac{U_1}{m} \qquad (24)$$

$$\ddot{Y} = (sin\Psi sin\theta cos\phi - cos\Psi sin\phi)\frac{U_1}{m} \qquad (25)$$

Quadcopter dynamics of the X and Y linear accelerations can be simplified in to:

$$\ddot{X} = (\theta cos\Psi + \phi sin\Psi)\frac{U_1}{m} \qquad (27)$$

$$\ddot{Y} = (\theta sin\Psi - \phi cos\Psi)\frac{U_1}{m} \qquad (28)$$

Now Eq. (27) and (28) are put in matrix notation:

$$\begin{bmatrix}\ddot{X}\\ \ddot{Y}\end{bmatrix} = \frac{\mathrm{U}_1}{\mathrm{m}}\begin{bmatrix} sin\Psi & cos\Psi\\ -cos\Psi & sin\Psi\end{bmatrix}\begin{bmatrix}\phi\\ \theta\end{bmatrix} \qquad (29)$$

The desired pitch and roll:

$$\begin{bmatrix}\phi_d\\ \theta_d\end{bmatrix} = \frac{\mathrm{m}}{\mathrm{U}_1}\begin{bmatrix} -sin\Psi & -cos\Psi\\ cos\Psi & -sin\Psi\end{bmatrix}^{-1} \qquad (30)$$

The complete control system with the dynamic system is applied in Matlab-Simulink can be shown in **Fig. 6**.

4. OPTIMIZATION

Optimization is the process of making something better. Optimization consists in trying variations on an initial solution and using the information gained to reach global optimum. A computer is the perfect tool for optimization since it can perform a lot iteration.

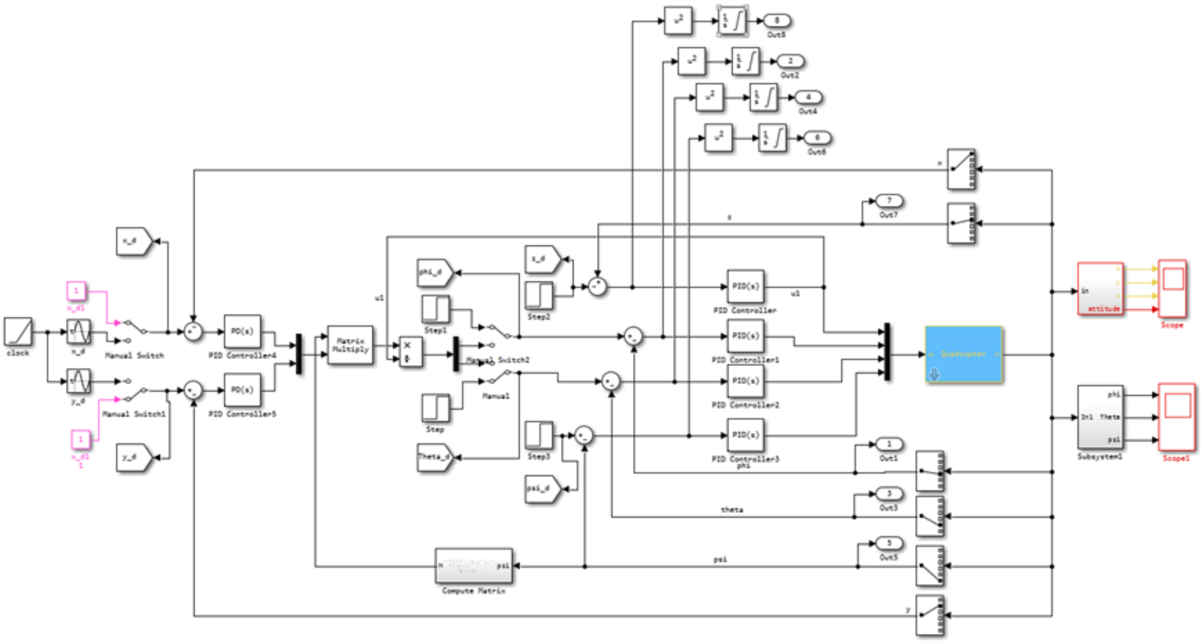

Fig 6: Complete Simulink model and control system for quadcopter in Matlab.

4.1 Cost Function

Cost function or objective function formulation is very important in optimization since it is the main parameter to measure the performance of the optimization technique and decide whether the solution will suit the problem or not. In this work, the objective function was built to achieve two goals optimize the consumption of battery and speed up the response time. After researching about the battery consumption, it was found that reducing the oscillation and the overshoot of the response of the quadcopter as much possible will reduce unnecessary power consumption [11]. An error objective function was used to make sure that the response is fast to follow the control signal. Integral square error (ISE) was used in the objective function the total objective function is shown in Eq. (31):

$$CostFunction = \alpha_\phi Mp_\phi + \beta_\phi \int |e_\phi|^2\, dt + \alpha_\theta Mp_\theta + \beta_\theta \int |e_\theta|^2\, dt + \alpha_\Psi Mp_\Psi + \beta_\Psi \int |e_\Psi|^2\, dt + \alpha_z Mp_z + \beta_z \int |e_z|^2\, dt \qquad (31)$$

Where Maximum peak over shoot is calculated by: $Mp = \frac{u_d - u_a}{u_a} \times 100$, u_d: desired input (Step input $u_d = 1$) , u_a: actual input.

4.2 Review of GSA Algorithm

Gravitational Search Algorithm (GSA) is an optimization algorithm based on the law of gravity. The algorithm have agents is considered as object and its performance is measured by its mass, All these agents attract each other by the force of gravity, this gravity force causes a global movement of all agents towards the agents with heavier masses. The mass which correspond cost function value of the agent. The position of the agent corresponds to a solution of the problem. The position of the *i*th agent of system with N agents:

$$X_i = \left(x_i^1, \dots, x_i^N\right) \qquad (32)$$

The force acting on mass ‘i’ from mass ‘j’ as following:

$$F_{ij}^d(t) = G(t)\frac{M_{pi} \times M_{aj}}{R_{ij}(t) + \varepsilon}\left(x_j^d(t) - x_i^d(t)\right) \qquad (33)$$

where M_{aj} is the active gravitational mass related to agent j, M_{pi} is the passive gravitational mass related to agent i, G(t) is gravitational constant at time t, e is a small constant, and $R_{ij}(t)$ is the Euclidian distance between two agents i and j. The total force that acts on agent i in a dimension d be a randomly weighted sum of dth components of the forces exerted from other agents:

$$F_i^d(t) = \sum_{j=1, j\neq i}^{N} rand_j F_{ij}^d(t) \qquad (34)$$

Where $rand_j$ is a random number in the interval [0, 1]. by the law of motion, the acceleration of the agent i at time t, and in direction dth, is given as follows:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)} \qquad (35)$$

Where M_{ii} is the inertial mass of ith agent. The velocity of an agent is considered as a fraction of its current velocity added to its acceleration. Therefore, its position and its velocity could be calculated as follows:

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \qquad (36)$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \qquad (37)$$

A heavier mass means a more efficient agent. This means that better agents have higher attractions and walk more slowly. To update the gravitational and inertial masses this following equations are used:

$$M_{ai} = M_{pi} = M_{ii} = M_i \qquad (38)$$

$$m_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \qquad (39)$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \qquad (40)$$

The different steps of the proposed algorithm are the followings:

a) Search space identification.
b) Randomized initialization.
c) Fitness evaluation of agents.
d) Update G(t), best(t), worst(t) and M_i(t) for i = 1,2,. . .,N.
e) Calculation of the total force in different directions.
f) Calculation of acceleration and velocity.
g) Updating agents’ position.
h) Repeat steps c to g until the stop criteria is reached.
i) End.

The algorithm is more explained in the flowchart below **Fig. 7**

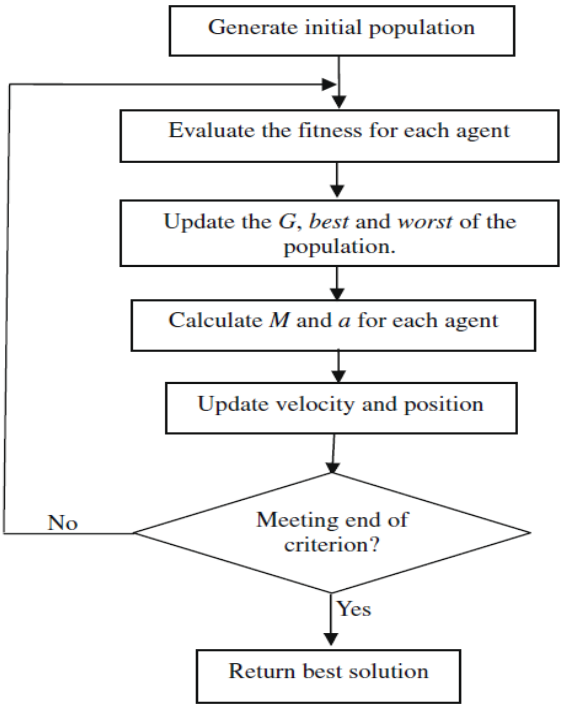
Generate initial population
Evaluate the fitness for each agent
Update the *G*, *best* and *worst* of the population.
Calculate *M* and *a* for each agent
Update velocity and position
Meeting end of criterion?
No
Yes
Return best solution

Fig 7: Flowchart of Gravitational search algorithm.

4.3 MBGSA Algorithm

The standard gravitational search algorithm (GSA) is a memory less heuristic optimization algorithm. Thus, the positions of agents depend only on the previous iteration. There is always a chance to lose the optimal solution because of not considering the best solution from previous iterations. This disadvantage reduces the performance of GSA when dealing with complicated optimization problems such as tuning multiple PID’s of quadcopter control system. Yet, the MBGSA uses the overall best solution of the agents from previous iterations in the calculation of agents’ positions. Consequently, the agents try to improve their positions by always searching around overall best solutions [15]. The following equation will be modified:

$$F_{ij}^{d}(t) = G(t)\frac{M_{pi} \times M_{aj}}{R_{ij}(t)+\varepsilon}\left(pbest_{j}^{d}(t) - x_{i}^{d}(t)\right) \quad (41)$$

$$m_i(t) = \frac{fit_i(t) - worst_{pbest}(t)}{best_{pbest}(t) - worst_{pbest}(t)} \quad (42)$$

The best position of any agent is stored as the agent's personal best position (pbest).

4.4 PSO Hybridization

The hybridization of PSO with MBGSA in this work will be using low-level co-evolutionary heterogeneous hybrid [14]. The hybrid is low-level because both algorithms functionality will be combined. It is co-evolutionary because the algorithms will be used in parallel. It is heterogeneous because there are two different algorithms that are involved to produce results [16].

The basic idea of PSO hybridization is to combine the ability of social thinking (gbest) in PSO with the local search capability of MBGSA. The combination of these algorithms will modify the speed equation:

$$v_i(t+1) = w \times v_i(t) + c_1' \times \text{rand} \times a_i(t) + c_2' \times \text{rand} \times (\text{gbest} - \text{x}_\text{i}(t)) \quad (43)$$

5. SIMULATION RESULTS AND DISCUSSION

5.1 Simulation Parameters

The Algorithm setting are shown in **Table 2.**

Table 2. GSA parameters.

Symbol	Value	Description
N	40	Size of the swarm " no of objects "
Max_Iteration	60	Maximum number of "iterations
G0	1	Gravitational constant
α	4	Increasing rate of gravitational constant
C1	0.5	Pso speed constant
C2	1.5	Pso speed constant

5.2 Simulation Results

The Gravitational Search algorithm was applied to the simulation model in Simulink in order to tune the PID's gains of the inner loop and then the outer loop. The inner loop which is 4 PID's (12 parameter) then the algorithm is applied second time to tune the outer control loop 2 PID (6 parameters). After simulation the following results was achieved considering that number of iteration is 60 and number of population is 40, the step response for the Roll, pitch and yaw which is essential to check for the stability of the quadcopter and the objective function. The step response is shown in **Fig. 8**. In addition, **Table 3** is showing the values of PID and the corresponding overshoot and Integral square error ISE.

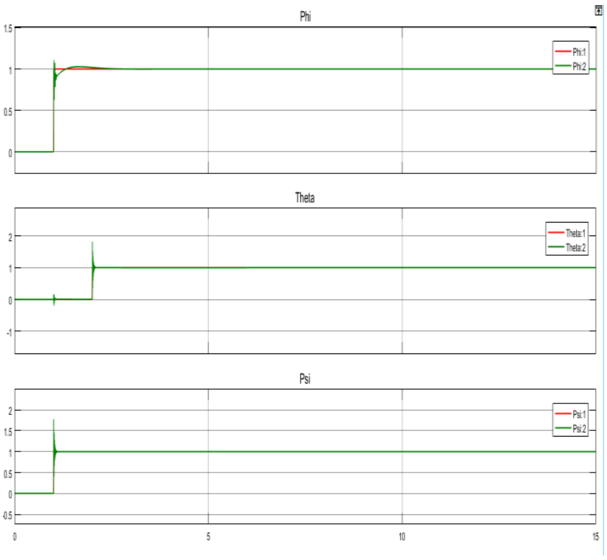

Fig 7: Step response of roll pitch and yaw after tuning the PID's with GSA (1: refers to desired signal, 2: refers to actual signal.

Table 3. PID values for roll, pitch, yaw and attitude and their corresponding overshot and ISE tuned by GSA.

Roll				
Kp	Ki	Kd	Overshoot%	ISE
32.6064	66.5007	6.2602	10.9933	0.00517
Pitch				
Kp	Ki	Kd	Overshoot%	ISE
21.6032	35.9199	48.1552	41.4366	0.00504
Yaw				
Kp	Ki	Kd	Overshoot%	ISE
47.3974	21.4835	54.9827	37.2717	0.005013
Z-Attuide				
Kp	Ki	Kd	Overshoot%	ISE
53.3939	36.2772	63.7173	20.0752	63.7173
X-Postion				
Kp	Ki	Kd	Overshoot%	ISE
1.6152	0	0.64219	40.241	0.2528
Y-Position				
Kp	Ki	Kd	Overshoot%	ISE
2.7621	0	0.792	21.231	0.1285

The tracking of the desired trajectory in the space is shown in **Fig. 8**, It is noticed that the normal GSA optimization is not following the exact path of the trajectory and it is because that the algorithm is fully random and heuristic thus it converges to local optimum. Figure **Fig. 9** shows the tracking in each single axes to check each axes independently. Testing the outer PID controller with step response it shows that it needs more iteration to get better results or improve the algorithm to achieve better result for the same setting.

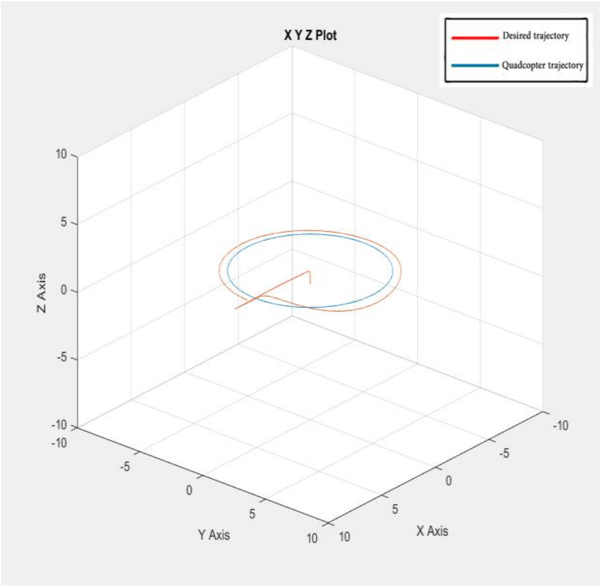

Fig 8: Trajectory tracking of quadcopter in 3D space after tuning the PID's with GSA.

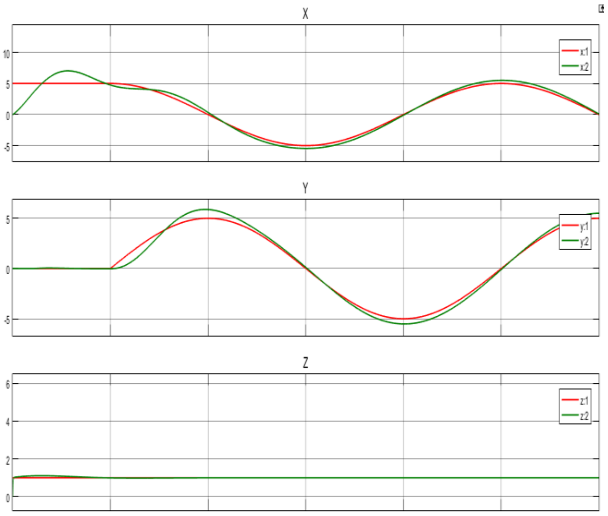

Fig 9: X, Y and Z axis response on single axes after tuning the PID's with GSA (1: refers to desired signal, 2: refers to actual signal).

The new Algorithm Hybrid Memory Based Gravitational Search algorithm – Practical Swarm Optimization algorithm was test also to tune the quadcopter control system. The MGSA-PSO was applied to the complete model in Simulink to tune the PID's gains of the inner (loop 4 PID's, 12parameters), Then the algorithm was applied second time to tune the outer control loop (2 PID's, 6 parameters). The following results was achieved considering that number of iteration is 60 and number of population is 40, the step response for the Roll pitch and yaw which is essential to check for the stability of the quadcopter the step response is shown in **Fig. 10** was better than the previous. The tracking of the desired trajectory in the space is shown **Fig. 11** was very good the desired track and actual track were almost the same. It is because that the hybrid algorithm has gained good search abilities form the PSO and also gained Memory to keep the best solution and search around it thus enables the algorithm to converge best optimum solution.

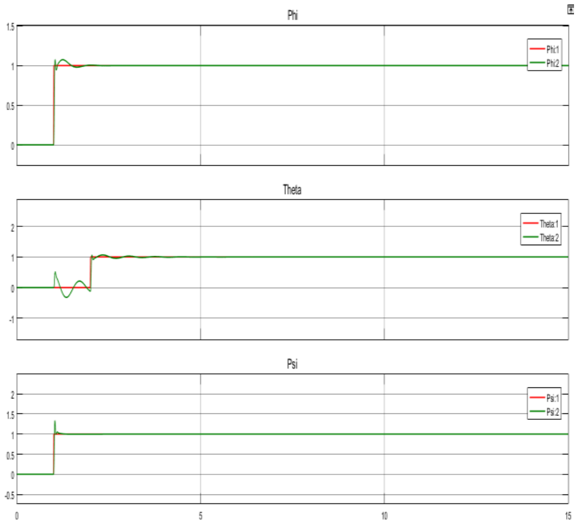

Fig 10: Step response of roll pitch and yaw after tuning the PID's with MBGSA-PSO (1: refers to desired signal, 2: refers to actual signal).

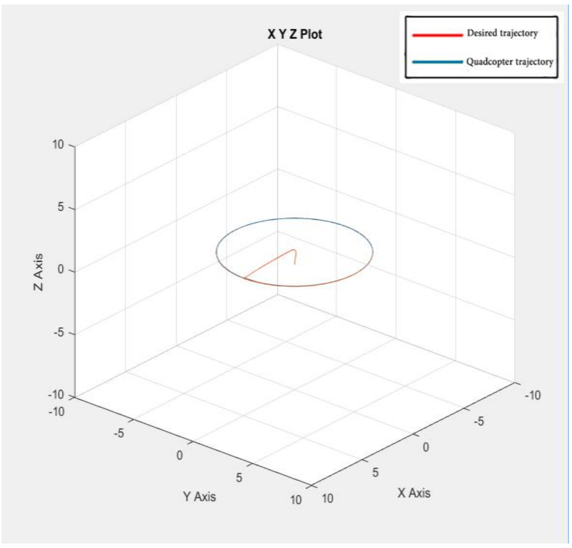

Fig 10: Trajectory tracking of quadcopter in 3D space after tuning the PID's with MBGSA-PSO.

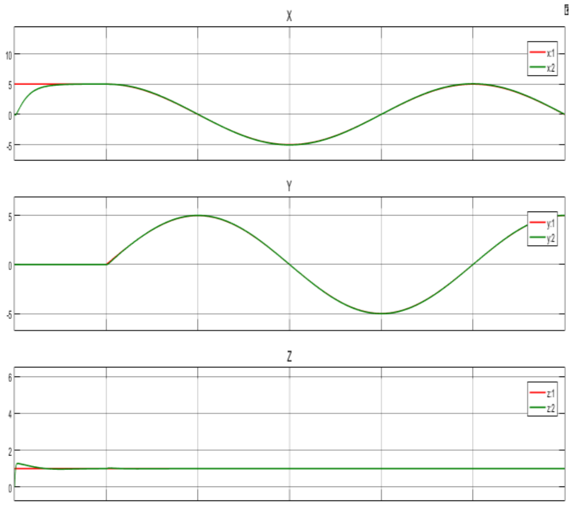

Fig 10: Position response for step input signal after tuning the PID's with MBGSA-PSO (1: refers to desired signal, 2: refers to actual signal).

Ten experiment were carried out for the previous GSA algorithm and the MBGSA-PSO algorithm because judgment cannot be based on one test since the algorithms are random procedures so it must be verified statistically and prove by statistics that the enhancement is better than the standard algorithm. As shown in **Table 5**, the normal algorithm (GSA) obtain (191.7409) as an average of 7 samples for the objective function where the improved algorithm (MBGSA-PSO) has achieved (101.5396) as an average of 7 samples for the objective function. That means (40.126%) improvement in MBGSA-PSO algorithm against the ordinary algorithm.

Table 4. PID values for roll, pitch, yaw and attitude and their corresponding overshot and ISE tuned by MBGSA-PSO.

Roll				
Kp	Ki	Kd	Overshoot%	ISE
5.1889	60.1512	0.77936	7.1889	1.4457
Pitch				
Kp	Ki	Kd	Overshoot%	ISE
1.7977	60	0.77176	68.5167	0.0731
Yaw				
Kp	Ki	Kd	Overshoot%	ISE
20.2211	46.3993	1.6346	34.1264	0.00992
Z-Attuide				
Kp	Ki	Kd	Overshoot%	ISE
59.97	61.2318	30.6211	29.357	0.05706
X-Postion				
Kp	Ki	Kd	Overshoot%	ISE
18.6152	0	6.4759	0	0.0123
Y-Position				
Kp	Ki	Kd	Overshoot%	ISE
27.7621	0	29.6963	0	0.0213

Table 5: Comparison between GSA and MBGSA-PSO with 10 sample to prove the enactment statistically

	GSA	**MBGSA-PSO**
	199.4264	92.7411
	195.1306	120.0692
	181.6209	167.942
	203.7294	105.7158
	187.992	81.5016
	191.1495	122.2561
	183.1372	81.7381
	186.7428	120.1246
	192.222	98.2723
	200.5676	131.2133
Statistical average	192.718	115.387

The enhancement percentage can be calculated according to the following equation.

$$EP]_{Algorithim\ A\ over\ Algorithm\ B} = \left(1 - \frac{OB]_A}{OB]_B}\right) \times 100\% \qquad (44)$$

Where EP refers to enhancement percentage and OB is for Objective function.

The enhancement of HMBGSA-PSO which is the new version of GSA over its original algorithm is:

$$EP]_{HMBGSA\ over\ GSA} = \left(1 - \frac{115.387}{192.718}\right) \times 100\% = 40.126\% \qquad (45)$$

6. CONCLUSION AND FUTURE WORK

Form the work done on the quadcopter control system, the PID controller is a suitable controller because it is simple and at the same time it can control the quadcopter and produce a control signals that is fast to follow the refresh speed of the sensors. The Optimization algorithms are essential for tuning the PID's and get the best optimal value, GSA algorithm in the literature was not used to tune multiple PID's simultaneously but in this work it was tested and showed a weak performance with the setting obtained from it. So the GSA was improved in this work to generate new algorithm which handle multi PID tuning in a very efficient way which is called Hybrid Memory Based Gravitational Search Algorithm – Particle Swarm Optimization (MBGSA-PSO). It was the combination form improving the GSA memory and hybridization it with Particle Swarm Optimization PSO. The HMBGSA-PSO average objective function was (115.387) and compared to GSA (192.718) thus achieved enhancement by (40.126%).

This results obtained in this work will be implemented in the future work in a swarm of quadcopters that are controller by a central algorithm and each quadcopter in the swarm will have this tuning but the X , Y and Z control signals will be controller by swarm algorithm, to study the behavior of the swarm.

7. ACKNOWLEDGMENTS

Our thanks to everyone who helped us in completing this work.

8. REFERENCES

[1] Rashedi, E., Nezamabadi-pour, H., Saryazdi, S., 2009, GSA: A Gravitational Search Algorithm, Information Sciences Vol. 179, pp. 2232–2248, Kerman, Iran.

[2] Beard, R.W., 2008, Quadrotor Dynamics and Control, Report, BYU ScholarsArchive, Brigham Young University, Provo.

[3] Benić, Z., Piljek, P., and Kotarski, D., 2016, Mathematical modelling of unmanned aerial vehicles with four rotors. Interdisciplinary Description of Complex Systems Vol.14, No.1, pp. 88-100.

[4] Bresciano, T., 2008, Modelling, Identification and Control of a Quadrotor Helicopter, M.Sc. Thesis. Lund University – Department of Automatic Control, Lund, Sweden.

[5] Bouabdallah, S., 2004, PID vs LQ Control Techniques Applied to an Indoor Micro Quadrotor, IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendal, Japan, pp. 2451–2456.

[6] Bouabdallah, S., 2006, Design and Control of Quadrotors with Application to Autonomous Flying,

[7] Ph.D. Thesis, Ecole Polytechnique Federale de Lausanne, Lausanne, Switzerland.

[8] ElKholy, M., N., 2014, Dynamic Modeling and Control of a Quadrotor Using Linear and Nonlinear Approaches, Master Thesis, School of Sciences and Engineering, American University in Cairo, Egypt.

[9] Kotarski, D., Benić, Z., and Krznar, M., 2016, Control Design For Unmanned Aerial Vehicles With Four Rotors, Interdisciplinary Description of Complex Systems Vol. 14, No. 2, pp. 236-245.

[10] Li, J., Li, Y., 2011, Dynamic Analysis and PID Control for a Quadrotor, IEEE International Conference on Mechatronics and Automation, Beijing, China.

[11] Oscar A., 2015, Get Longer Flight Time On Multicopter, website article https://oscarliang.com/maximize-flight-time-quadcopter/

[12] Pipatpaibul, P., and Ouyang P. R., 2011, Quadrotor UAV control : online learning approach, Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2011 August 28-31, 2011, Washington, DC, USA.

[13] Salih A. L., Moghavvemi M., Haider, Mohamed A. F., and Gaeid K. S., 2010, Flight PID controller design for a UAV quadrotor, Scientific Research and Essays Vol. 5, No. 23, pp. 3660-3667.

[14] Mirjalili, S., Hashim, S. Z. M., 2010, A New Hybrid PSOGSA Algorithm for Function Optimization, International Conference on Computer and Information Application, Tianjin, China.

[15] Darzia, S., Kiongb T. S., Islamc M. T., Soleymanpourd, H. R., Kibriaca S., 2016, A memory-based gravitational search algorithm for enhancing minimum variance distortionless response beamforming, Applied Soft Computing Vol. 47, pp.103–118.

[16] Talbi E. G, 2002, A Taxonomy of Hybrid Metaheuristic, Journal of Heuristics, vol. 8, no. 5, pp. 541-546.