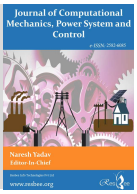
Vol.7 No.3 July 2024

Journal of Computational Mechanics, Power System and Control

Received 6 July, Revised 12 July, Accepted 31 July



Optimization of Quadcopter PID Controller Gains using Ant Colony Optimization and Genetic Algorithm

Adeleke Olorunnisola
Federal University of Technology, Akure
adelekeoo@futa.edu.ng

Olurotimi Dahunsi
Federal University of Technology, Akure
oadahunsi@futa.edu.ng

Abstract: Proportional Integral Derivative (PID) controllers stand as the cornerstone in most robotic applications due to their inherent simplicity and practicality. However, the manual tuning of these controllers often proves to be an ineffective approach to obtaining optimal gains for specific operational requirements. Consequently, there arises a demand for computational intelligence, leveraging meta-heuristic algorithms to systematically determine the most suitable combination of gains. Among the plethora of meta-heuristic algorithms, Genetic Algorithm has gained prominence for its efficacy in intelligent problem-solving. Similarly, Ant Colony Optimization Algorithm is recognized for its effectiveness. This paper conducts a comparative analysis of the performance of PID controllers when tuned using these algorithms. The investigation involves a simulation using MATLAB 2023a, with documentation of results presented. Through this analysis, the paper aims to provide insight into how ACO and GA can be used to tune the PID controller for quadcopters.

Keywords: PID, Genetic Algorithm, Ant Colony Optimization, Quadcopter, UAV

Nomenclature

Abbreviation	Expansion
PID	Proportional Integral Derivative
UAVs	Unmanned Aerial Vehicles
RFID	Radio Frequency Identification
MPC	Model Predictive Control
IMC	Internal Model Control
SMC	Sliding Mode Control
ACO	Ant Colony Optimization
GA	Genetic Algorithm

1. Introduction

UAVs have become indispensable assets across various industries, notably enhancing the manufacturing sector. By integrating cutting-edge technologies like machine vision [1] and RFID [2]. These vehicles can perform aerial surveys to manage warehouse inventory [3], track the movement of raw materials and finished products within the supply chain, and facilitate material handling within manufacturing facilities [4], among other essential functions. The critical need for UAV stability and safety during operation revolves around a control system, primarily managing rotor velocity. PID controllers reign supreme across 90% of industries due to their unmatched simplicity, clear functionality, applicability, and user-friendly nature, which surpasses other advanced control schemes like MPC, IMC, and SMC [5].

However, the challenge in implementing a PID controller lies in the intricate process of tuning system gains to attain requisite stability, prompting the application of soft computing methodologies utilizing algorithms for assistance [6]. An attempt has been made to use a modified PID which is tuned by GA [7]. Differential Evolution was also used to tune the PID gains to control the attitude of a quadrotor [8]. Similarly, there has been a comparison between the traditional tuning method by the Ziegler-Nichols model and the Genetic algorithm to tune PD, PI, and PID controllers [9] [10] [11]. Lastly, [12] used an artificial bee colony to tune the PID parameters of a flight control system. Although the Genetic Algorithm stands out for its broad applicability and straightforward design, several alternative algorithms such as the ACO algorithm which is a swarm intelligence-based algorithm could work too.

This study endeavors to investigate and compare the efficacy of these two algorithms concerning their suitability in enhancing stability for a designed quadcopter over a specified duration. Recognizing

https://doi.org/10.46253/jcmps.v7i3.a3

the limitations of manual tuning in obtaining optimal gains for specific operational requirements, the research aimed to explore the use of computational intelligence, specifically the Genetic Algorithm and Ant Colony Optimization Algorithm, for tuning PID controllers effectively. The results provided insights into the comparative efficacy of these two algorithms in enhancing the performance of robotic systems.

The rest of the paper is organized as Section 2 develops the system model, section 3 covers Control Optimization techniques, section 4mention the result, section 5 discusses the result and Section 6 concludes the paper.

2. System Modelling

Developing a mathematical model for a quadcopter in a three-dimensional space offers a systematic method to establish connections between inputs and outputs, facilitating the design of a robust controller to ensure system stability [13]. The quadcopter classified as a UAV, features four rotors affixed to a set of intersecting axles. Notably, rotors 1 and 3 operate in a counter-clockwise direction, while rotors 2 and 4 rotate clockwise, establishing a torque-balanced configuration upon the body frame, as depicted in Fig. 1.

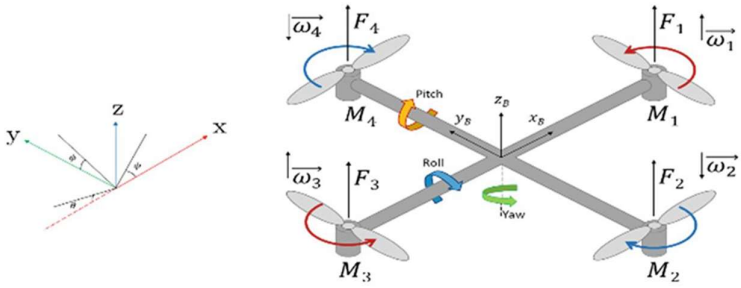


Fig. 1. *Schematic diagram of a quadcopter motions*

Quadcopters exhibit movement along six axes through a combination of rotational and translational maneuvers. The control inputs encompass:

1. **Thrust (z):** This force, perpendicular to the quadcopter's orientation, governs its directional movement in the airspace. Its magnitude corresponds to uniform changes in the speed of all four rotors.
2. **Roll angle (ϕ):** Generated by the quadcopter's rotation around the x-axis, maintaining a consistent aerial level while executing horizontal movements through adjustments in the speeds of rotors 2 and 4.
3. **Pitch angle (θ):** Produced by the quadcopter's rotation around the y-axis, influenced by adjustments in rotor 1 and rotor 3 speeds.
4. **Yaw angle (ψ):** Elicited by the quadcopter's rotation around the z-axis, achieved by increasing the angular velocity of one pair of rotors (spinning in the same direction) while decreasing the angular velocity of the other pair. Table 1 represents the summary of the movements.

Table 1. *Summary of quadcopter rotational and translational movements.*

Expression	Description
Theta	Pitch (Euler) angle – y-axis rotation
Phi	Roll (Euler) angle – x-axis rotation
Psi	Yaw (Euler) angle – z-axis rotation
z	Thrust angle - altitude

Euler angles which define the angular position of the quadcopter are denoted by η, as follows:

$$\vec{\eta} = \begin{Bmatrix} \phi \\ \theta \\ \Psi \end{Bmatrix} \tag{1}$$

The rotation matrix R is a 3x3 matrix that transforms coordinates from one reference frame to another. For a quadcopter, this matrix helps in converting body-fixed coordinates to the inertial frame (or vice versa). The rotation matrix R is represented as.

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{2}$$

Roll (ϕ) is calculated directly from the elementr_{23}using the arcsine function. Yaw (ψ) is calculated using the atan2 function with elementsr_{21} and r_{22}, adjusted by the roll angle. Pitch (θ) is calculated using the atan2 function with elements r_{13} and r_{33}, also adjusted by the roll angle. These formulas allow the extraction of the Euler angles from the rotation matrix, providing a way to determine the orientation of the quadcopter in terms of roll, pitch, and yaw. The Roll, Pitch, and Yaw angles ϕ, θ, Ψ are calculated as follows:

$$\phi = \sin^{-1}\left(r_{23}\right) \tag{3}$$

$$\Psi = a\tan 2\left(-\frac{r_{21}}{\cos(\phi)}, \frac{r_{22}}{\cos(\phi)}\right) \tag{4}$$

$$\theta = a\tan 2\left(-\frac{r_{13}}{\cos(\phi)}, \frac{r_{33}}{\cos(\phi)}\right) \tag{5}$$

The linear position of the quadcopter with respect to Earth's inertial reference frame (x, y, z)is denoted by $\vec{\zeta}$.

$$\vec{\zeta} = \begin{Bmatrix} x \\ y \\ z \end{Bmatrix} \tag{6}$$

A combination of the equation (1) and (6), generates the array q , that is:

$$q = \begin{bmatrix} \vec{\zeta} \\ \vec{\eta} \end{bmatrix} \tag{7}$$

However, the linear and angular velocities can be modeled as the matrices below which are denoted by $\vec{\Upsilon}$ and $\vec{\Omega}$:

$$\vec{\Upsilon} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{8}$$

$$\vec{\Omega} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -s(\theta) \\ 0 & c(\phi) & c(\theta)s(\phi) \\ 0 & -s(\phi) & c(\theta)c(\phi) \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\Psi} \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\Psi} \end{bmatrix} = \begin{bmatrix} 1 & s(\phi)t(\theta) & c(\phi)t(\theta) \\ 0 & c(\phi) & -s(\phi) \\ 0 & s(\phi)\sec(\theta) & c(\theta)\sec(\phi) \end{bmatrix}^{-1} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{10}$$

The rate of change of the roll rate $\dot{p}$ depends on the product of the pitch and yaw rates, scaled by the moments of inertia, and the external torque around the x-axis.

$$\dot{p} = \frac{I_{yy} + I_{zz}}{I_{xx}}.qr + \frac{1}{I_{xx}}.M_1 \tag{11}$$

Where

I_{xx}, I_{yy}, I_{zz} -Moments of inertia around the x, y, and z axes, respectively.

q- Pitch rate (angular velocity around the y-axis).

r- Yaw rate (angular velocity around the z-axis).

M_1- External moment or torque applied around the x-axis

The rate of change of the pitch rate $\dot{q}$ depends on the product of the yaw and roll rates, scaled by the moments of inertia, and the external torque around the y-axis. M_2 is the external moment or torque applied around the y-axis.

$$\dot{q} = \frac{I_{zz} + I_{xx}}{I_{yy}}.rp + \frac{1}{I_{yy}}.M_2 \tag{12}$$

The rate of change of the yaw rate $\dot{r}$ depends on the product of the roll and pitch rates, scaled by the moments of inertia, and the external torque around the z-axis. M_3 is the external moment or torque applied around the z-axis.

$$\dot{r} = \frac{I_{xx} + I_{yy}}{I_{zz}}.pq + \frac{1}{I_{zz}}.M_3 \tag{13}$$

2.1 Rotation Matrices at a given Attitude

The rotation matrix taking Yaw, Pitch, and Roll sequence can be expressed as follows:

$$R(\Psi) = \begin{bmatrix} c(\Psi) & s(\Psi) & 0 \\ -s(\Psi) & c(\Psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{14}$$

$$R(\theta) = \begin{bmatrix} c(\theta) & 0 & -s(\theta) \\ 0 & 1 & 0 \\ s & 0 & c(\theta) \end{bmatrix} \tag{15}$$

$$R(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi) & s(\phi) \\ 0 & s(\phi) & c(\phi) \end{bmatrix} \tag{16}$$

To obtain the combined rotation matrix R, we multiply the individual rotation matrices in the order of the rotations.

$$R = R(\Psi) * R(\theta) * R(\phi) \tag{17}$$

Hence the rotation matrices can be written as

$$R = \begin{bmatrix} c(\Psi)c(\theta) & -s(\Psi)c(\phi) + c(\Psi)s(\theta)s(\phi) & s(\Psi)s(\phi) + c(\Psi)s(\theta)c(\phi) \\ s(\Psi)c(\theta) & c(\Psi)c(\phi) + s(\Psi)s(\theta)s(\phi) & -c(\Psi)s(\phi) + s(\Psi)s(\theta)c(\phi) \\ -s(\theta) & c(\theta)s(\phi) & c(\theta)c(\phi) \end{bmatrix} \tag{18}$$

This matrix can be used to transform coordinates from one frame to another, considering all three rotations in the specified order.

2.2 State-Space Modelling

Using the state space model, one can formulate the state variables of the rotational and translational dynamics. This comprehensive model is essential for designing control systems that can stabilize and maneuver the quadcopter effectively as follows:

$$\dot{x} = f(\vec{x}, \vec{u}) \tag{18}$$

The state vector is represented as

$$\vec{x} = [X, Y, Z, \dot{X}, \dot{Y}, \dot{Z}, \phi, \Psi, \theta, p, q, r]^T \tag{19}$$

Where X, Y, and Z represent the position coordinates in the inertial frame. The input vector $\vec{u}$ consists of the total thrust and the moments around the three axes. T_Σ is the total thrust produced by the quadcopter's motors.

$$\vec{u} = [u_1, u_2, u_3, u_4]^T = [T_\Sigma, M_1, M_2, M_3, M_4]^T \tag{20}$$

The translational dynamics of the equation of motion are

$$\dot{X} = \dot{X} \tag{21}$$

$$\dot{Y} = \dot{Y} \tag{22}$$

$$\dot{Z} = \dot{Z} \tag{23}$$

Then

$$\ddot{X} = -\frac{1}{m}(\sin\Psi.\sin\phi + \cos\Psi.\sin\theta.\cos\phi).T_\Sigma \tag{24}$$

$$\ddot{Y} = -\frac{1}{m}(-\cos\Psi.\sin\phi + \sin\Psi.\sin\theta.\cos\phi).T_\Sigma \tag{25}$$

$$\ddot{Z} = -\frac{1}{m}\cos\theta.\cos\phi.T_\Sigma + g \tag{26}$$

Here, m is the mass of the quadcopter, and g is the gravitational constant.

During state equations linearization we can obtain

$$\dot{\phi} = p + s(\phi)t(\theta).q + c(\phi)t(\theta).r \tag{27}$$

$$\dot{\theta} = c(\phi).q - s(\phi).r \tag{28}$$

$$\dot{\Psi} = s(\phi)\sec(\theta).q + c(\phi).r \tag{29}$$

$$\dot{p} = \frac{I_{yy} - I_{zz}}{I_{xx}}.qr + \frac{1}{I_{xx}}.M_1 \tag{30}$$

$$\dot{q} = \frac{I_{zz} - I_{xx}}{I_{yy}}.rp + \frac{1}{I_{yy}}.M_2 \tag{31}$$

$$\dot{r} = \frac{I_{xx} - I_{yy}}{I_{zz}}.pq + \frac{1}{I_{zz}}.M_3 \tag{32}$$

Linearization around the (hovering) equilibrium point, $\vec{e} = [0,0,0,0,0,0,0,0,\Psi_e,0,0,0]^T$, $\vec{u}_e = [mg,0,0,0]^T$ yields a linear quadcopter state-space equation as follows:

$$\vec{x}' = A\vec{x}' + B\vec{u}' \tag{33}$$

Where the matrix A is a 12x12 matrix with specific elements as given, where g represents the gravitational constant and Ψ_e is a specific angle or parameter involved in the function f.

$$A = \frac{\partial f(\vec{x},\vec{u})}{\partial \vec{x}}|_{\vec{u}=\vec{u}_e,\vec{x}=\vec{x}_e} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -g\sin(\Psi_e) & -g\cos(\Psi_e) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & g\cos(\Psi_e) & -g\sin(\Psi_e) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B = \frac{\partial f(\vec{x},\vec{u})}{\partial \vec{x}}_{\vec{x}=\vec{x}_e,\vec{u}=\vec{u}_e} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{m} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} \tag{34}$$

And $\vec{x}' = \vec{x} - \vec{x}_e, \vec{u}' = \vec{u} - \vec{u}_e$ stands for deviation of states and control inputs from steady state.

3. Control Optimization Techniques

3.1. Genetic Algorithm

The GA is a stochastic optimization technique used for solving nonlinear equations and complex optimization problems. Unlike deterministic methods, GA uses probabilistic transition rules to manage a population of potential solutions, called individuals or chromosomes, which evolve through generations.
GA starts with a random population, represented by real-valued numbers or binary strings (chromosomes). It simulates solution evolution using a fitness function and genetic operators: reproduction, crossover, and mutation. Each chromosome's performance is evaluated by an objective function that assigns a fitness score, guiding the selection of superior solutions.
The key operations in GA are:

i) Reproduction: Selecting individuals based on fitness to produce offspring.
ii) Crossover: Combining pairs of chromosomes to create new ones.
iii) Mutation: Introducing random variations.

The process continues iteratively, with the population evolving towards optimal solutions. The algorithm employs a "survival of the fittest" strategy, relying on error value computation for fitness assessment, ensuring the best solutions progress through generations.

PSEUDO CODE FOR GENETIC ALGORITHM (GA)

```
1.  Initialization → Initialize the population with random chromosomes
2.  Fitness Evaluation → Evaluate the fitness of each chromosome in the population
3.  while termination condition not met do
4.      Select a mating pool from the current population based on fitness
5.      Apply reproduction to create offspring from the mating pool
6.      Apply crossover on pairs of offspring to combine their genes
7.      Apply mutation on offspring to introduce random variations
8.      Evaluate the fitness of each offspring
9.      Select the next generation population based on fitness (survival of the fittest)
10.     Iterative Refinement → Replace the current population with the new generation
11. end while
12. Output the best solution found
```

3.2. Ant Colony Optimization Algorithm

ACO is also a metaheuristic algorithm inspired by ant foraging behavior, used to solve complex optimization problems through probabilistic decision-making and collective intelligence. ACO operates iteratively with artificial ants representing PID gains, which are the proportional, integral, and derivative components of a controller.

The process begins with initializing a group of ants to explore potential PID gain values. Ants iteratively construct gain sets using heuristic information and existing knowledge about the system's dynamics. An objective function evaluates each set of gains based on the quadcopter's performance, assigning fitness scores.

ACO uses these fitness scores to reinforce and select better-performing gain sets. Pheromone updates adjust the likelihood of selecting specific gain values, favoring those that improve flight characteristics. Iterative updates of pheromone levels guide the algorithm towards optimal PID controller values, with evaporation mechanisms ensuring exploration to avoid local optima.

PSEUDO CODE FOR ANT COLONY OPTIMIZATION ALGORITHM (ACO)

```
1.  Initialization: Initialize parameters such as number of ants and pheromone levels
2.  Problem definition: specify the solution representation
3.  while termination condition not met do
4.      for each ant in the colony do
5.          Construct a solution based on pheromone levels and heuristic information
6.      end for
7.      Evaluate the quality of each constructed solution using the objective function
8.      Update pheromone levels based on the fitness scores of the solutions
9.      if convergence criterion is met then
10.         break
11.     end if
12. end while
13. Output the best solution found
```

4. Result

The research objective was to develop a PID controller, applying GA and ACO to optimize the PID gains. The fitness function was our cost function and the minimizing of the cost function was accomplished to get a more stable controller, to improve the transient and steady-state response like settling time, rise time, and peak time. The tables below highlight the parameters and the results achieved.

Table 2. *Parameters Used in Genetic Algorithm*

Parameters	Value/Type
Maximum generations	60
Population size	50
Encoding	Binary
Selection	Uniform
Crossover	Single point crossover
Mutation	Uniform

Table 3 provides the ranges of parameters for the PID controller gains for each motor in the quadcopter. Each motor has three PID controller gains: K_p (Proportional gain), K_i (Integral gain), and K_d (Derivative gain). These ranges indicate the values within which the PID controller gains can be adjusted for each motor to optimize the performance of the quadcopter.

Table 3. *Genetic Algorithm PID Parameters range*

Parameters	K_p(Lower -Upper)	K_i(Lower -Upper)	K_d(Lower -Upper)
Motor 1	0.1 – 2.0	0 - 0	0.1 – 1.0
Motor 2	0.1 – 2.0	0 - 0	0.1 – 1.0
Motor 3	0.5 – 1.0	0 - 0	0.1 2.0
Motor 4	5.0 – 6.0	0.1 – 2.0	0.0 – 0.0

The PID gain values were obtained through the Genetic Algorithm for each of the four motors in the quadcopter. These values represent the optimized parameters for the PID controller of each motor, determined through the Genetic Algorithm to enhance the quadcopter's performance. The Genetic Algorithm iteratively refines these gains based on the quadcopter's response to achieve better stability and control during flight. By fine-tuning the PID gains using the Genetic Algorithm, the quadcopter can effectively respond to external disturbances and maintain its desired trajectory with improved accuracy and efficiency. The PID gain values of the four motors are represented in Table 4.

Table 4. *Genetic Algorithm PID gain values*

Parameters	K_p	K_i	K_d
Motor 1	0.3212	0	0.1
Motor 2	0.1163	0	0.1027
Motor 3	0.5	0	0.1277
Motor 4	5.0	0.1	0

In the PID controllers, settling time is the time taken for the system's response to reach and remain within a certain percentage (usually 5% or 2%) of the desired value after a step input. The parameter or variable related to the settling time of a PID controller is tuned using a Genetic Algorithm. Fig. 2 illustrates the settling time performance of the PID controller when tuned using a Genetic Algorithm. The settling time is a crucial performance metric in control systems as it indicates how quickly the system stabilizes after a disturbance or set point change.

By analyzing the settling time results depicted in Fig. 2, researchers can evaluate the effectiveness of using a Genetic Algorithm to tune the PID controller for improved system performance. A shorter settling time depicted in the figure indicates that the algorithm efficiently converges to the optimal PID controller gains. Conversely, a longer settling time suggests that the algorithm takes more iterations or time to fine-tune the gains to achieve the desired performance.

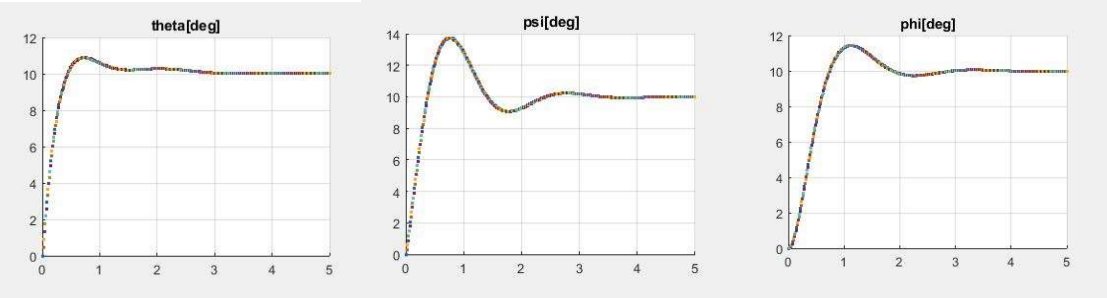


Fig. 2 . *Genetic Algorithm PID settling time*

In the ACO, the typical parameters of a PID controller used in control systems are K_p, K_i, and K_d. When the ACO algorithm runs, you might find that the best-performing Kp values are between 2 and 4. You can then adjust the Kp range to [2, 4] for finer tuning in subsequent iterations. By iteratively adjusting the ranges based on the performance of the solutions, you can effectively optimize the PID controller parameters using ACO.

Table 5. *Ant Colony Optimization Algorithm PID Parameters range*

Parameters	K_p(Lower -Upper)	K_i(Lower -Upper)	K_d(Lower -Upper)
Motor 1	0.1 – 2.0	0 - 0	0.1 – 1.0
Motor 2	0.1 – 2.0	0 - 0	0.1 – 1.0
Motor 3	0.5 – 1.0	0 - 0	0.1 - 2.0
Motor 4	5.0 – 6.0	0.1 – 2.0	0.0 – 0.0

The parameters and values of the ACO algorithm are mentioned in Table 6.

Table 6. *Parameters Used in Ant Colony Optimization Algorithm*

Parameters	**Value/Type**
Maximum generations	5
Colony size	10
Alpha	1.0
Beta	2.0
Ant Movement Strategy	Stochastic
Maximum Iterations without change	10

Each ant in the colony constructs a solution by selecting K_p, K_i, and K_d values within the specified ranges. The selection is influenced by the pheromone trails and possibly heuristic information. By carefully defining the initial ranges, performance criteria, and pheromone update rules, ACO can effectively optimize the PID gain values for a variety of control systems. It is explained in Table 7.

Table 7. *Ant Colony Optimization Algorithm PID gain values*

Parameters	K_p	K_i	K_d
Motor 1	0.2056	0	0.7990
Motor 2	1.6088	0	0.3948
Motor 3	0.56	0	0.3283
Motor 4	5.3612	1.2581	0

In the case of the ACO Algorithm applied to tuning PID controllers for a quadcopter, the settling time represents how fast the algorithm can find the best combination of gains for the controller. Fig. 3 illustrates the settling time of the ACO Algorithm during the tuning process of the PID controller for the quadcopter.

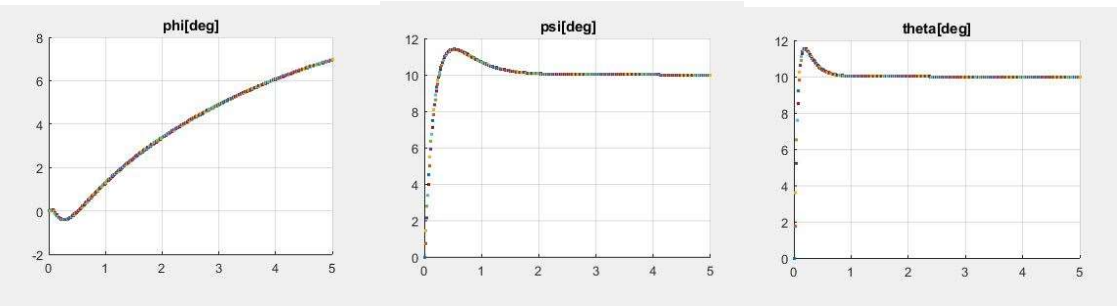


Fig. 3. *Ant Colony Optimization Algorithm Settling Time*

5. Discussion

When comparing the GA and ACO algorithms, GA uses a fixed number of generations and ACO runs on a specific number of iterations to optimize the pheromone trails. Comparing the results from the graph, it is obvious that GA performed better and was able to get the phi, psi, and theta angles to a steady state thereby leading to a stable motion as compared to that of ACO which was unable to get the phi angle to a steady state even within the simulation time (5secs). Although it appears the settling time for ACO is faster, however, generally GA can be said to perform better. The exact performance can vary based on specific implementation details and the tuning of both algorithms. Therefore, it's recommended to empirically test both methods on the actual system to make a final determination.

6. Conclusion

In this article, the dynamic modeling of a quadcopter is presented. Based on this model, to control the altitude of the quadcopter, a control strategy was developed using GA and ACO. The study aimed to showcase the effectiveness of the Genetic Algorithm and Ant Colony Optimization Algorithm in optimizing PID controller gains, highlighting their potential for enhancing the performance of robotic systems, The fitness function was our cost function, and the minimizing of the cost function was accomplished to get a more stable controller, to improve the transient and steady-state response like settling time, rise time and peak time. The findings of the study suggest that both the Genetic Algorithm and Ant Colony Optimization Algorithm can be valuable tools in fine-tuning PID controller gains to improve the overall performance and efficiency of robotic systems.

Compliance with Ethical Standards

Conflicts of interest: Authors declared that they have no conflict of interest.

Human participants: The conducted research follows the ethical standards and the authors ensured that they have not conducted any studies with human participants or animals.

References

[1] M. Achtelik, T. Zhang, K. Kuhnlenz, and M. Buss, “Visual tracking and control of a quadcopter using a stereo camera system and inertial sensors”, in *2009 International Conference on Mechatronics and Automation*, Aug. 2009, pp. 2863–2869. doi: 10.1109/ICMA.2009.5246421.

[2] A. Milella, P. Vanadia, G. Cicirelli, and A. Distante, "RFID-based environment mapping for autonomous mobile robot applications", in *2007 IEEE/ASME international conference on advanced intelligent mechatronics*, Sep. 2007, pp. 1–6. doi: 10.1109/AIM.2007.4412601.

[3] Y.-H. Li and H.-Y. Lin, "Development of UAV Localization and Navigation Techniques for Warehouse Inventory Inspection", in *2023 IEEE International Conference on Industrial Technology (ICIT)*, Apr. 2023, pp. 1–6. doi: 10.1109/ICIT58465.2023.10143122.

[4] M. Lieret, V. Kogan, S. Döll, and J. Franke, "Automated in-house transportation of small load carriers with autonomous unmanned aerial vehicles", in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*, Aug. 2019, pp. 1010–1015. doi: 10.1109/COASE.2019.8843183.

[5] A. Jayachitra and R. Vinodha, "Genetic Algorithm Based PID Controller Tuning Approach for Continuous Stirred Tank Reactor", *Adv. Artif. Intell.*, vol. 2014, p. e791230, Dec. 2014, doi: 10.1155/2014/791230.

[6] Rahul .B, "Genetic Algorithm tuned PID Controller for Aircraft Pitch Control", *Int. J. Res. Eng. Appl. Amp Manag. IJREAM*, Jan. 2019, Accessed: Nov. 07, 2023. [Online]. Available: https://www.academia.edu/44517591/Genetic_Algorithm_tuned_PID_Controller_for_Aircraft_Pitch_Control

[7] K. Khuwaja, N.-Z. Lighari, I. C. Tarca, and R. C. Tarca, "PID Controller Tuning Optimization with Genetic Algorithms for a Quadcopter", *Recent Innov. Mechatron.*, vol. 5, no. 1, Art. no. 1, Apr. 2018, doi: 10.17667/riim.2018.1/11.

[8] A. Gün, "Attitude control of a quadrotor using PID controller based on differential evolution algorithm", *Expert Syst. Appl.*, vol. 229, pp. 120518, Nov. 2023. doi: 10.1016/j.eswa.2023.120518.

[9] A. A. M. Yusuf, N. J. Agung, and S. Unang, "Implementation real value genetic algorithm to determine three PID parameter", in *2015 International Conference on Control, Electronics, Renewable Energy and Communications (ICCEREC)*, Aug. 2015, pp. 198–202. doi: 10.1109/ICCEREC.2015.7337044.

[10] T. Ahmmed, I. Akhter, S. M. R. Karim, and F. A. S. Ahamed, "Genetic Algorithm Based PID Parameter Optimization", *Am. J. Intell. Syst.*, vol. 10, no. 1, pp. 8–13, 2020.

[11] M. F. Q. Say, E. Sybingco, A. A. Bandala, R. R. P. Vicerra, and A. Y. Chua, "A Genetic Algorithm Approach to PID Tuning of a Quadcopter UAV Model", in *2021 IEEE/SICE International Symposium on System Integration (SII)*, Jan. 2021, pp. 675–678. doi: 10.1109/IEEECONF49454.2021.9382697.

[12] F. Lin and X. Wang, "(PDF) PID parameters tuning of UAV flight control system based on artificial bee colony algorithm", Accessed: Nov. 17, 2023. [Online]. Available: https://www.researchgate.net/publication/300484014_PID_parameters_tuning_of_UAV_flight_control_system_based_on_artificial_bee_colony_algorithm

[13] A. Sheta, M. Braik, D. R. Maddi, A. Mahdy, S. Aljahdali, and H. Turabieh, "Optimization of PID Controller to Stabilize Quadcopter Movements Using Meta-Heuristic Search Algorithms", *Appl. Sci.*, vol. 11, no. 14, Art. no. 14, Jan. 2021. doi: 10.3390/app11146492.