Journal of Computing and Communication Vol.4 , No.1 , PP. 31-42 , 2025

Evolutionary Design of a PID Controller Using Metaheuristics Search Algorithms

Vennapusa M. Kumar Reddy[a], Mudimela S. Kumar Reddy [a], Karnati Suresh [a], Alaa Sheta [b], Walaa H. Elashmawi [c], Adel Djellal [d], Abdel karim M. Baareh [e]

[a]*Computer Science Department, Missouri State University, Springfield, MO 65807, USA*

[b]*Department of Computer Science, Southern Connecticut State University, New Haven, CT, USA*

[c]*Department of Computer Science, Misr International University, Cairo, Egypt*

[d]*Department of E.E.A., National Higher School of Technology and Engineering, 23005, Annaba, Algeria*

[e]*Applied Science Department, Ajloun University College, Al-Balqa Applied University, Ajloun, Jordan*

*Corresponding Author: Walaa H. Elashmawi [***walaa.hassan@miuegypt.edu.eg***]

ARTICLE DATA

Article history:
Received 13 December 2024
Revised 27 Jan 2025
Accepted 28 Jan 2025
Available online

Keywords:
PID Controller, Swarm Intelligence Algorithm, Evolutionary Algorithm, Metaheuristic Search Algorithms

ABSTRACT

Proportional-Integral-Derivative (PID) controllers are prominent due to their superior functionality and ease of use. However, optimizing their parameters presents a significant challenge. Adjusting parameters must be done carefully and cautiously because improper calibration can compromise the system's stability. Although classic tuning techniques, such as the Ziegler-Nichols (ZN), are frequently employed, their efficiency is restricted due to the intricate and ever- changing nature of the systems, often leading to parameter settings that could be more optimal. Therefore, the need for a more accurate parameter-tuning technique is urgent. Various optimization strategies are used to fine-tune parameters with more precision. These methods include Gray Wolf Optimization (GWO), Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Ant Colony Optimization (ACO). These methods are applied to fine-tune the PID parameters for a Direct Current (DC) motor to achieve optimal performance, and a comparative analysis of the results is conducted. Various fitness functions encompass performance metrics such as rise time, overshoot, peak time, settling time, and mean square error (MSE). These metrics are incorporated into the corresponding optimization approaches to quantitatively assess the controller's performance. Various test cases have been utilized and the GA outperforms other algorithms ranging from 17% to 28% where rise time, settling time, and MSE are significant in the fitness function.

1. Introduction

The PID (Proportional–Integral–Derivative) Controller stands as a closed-loop control system widely applied within industrial control systems [1]. Growing research has illustrated the advantages of meta-heuristic algorithms in problem-solving contexts. Evolutionary algorithms (EAs) and Swarm Intelligence (SI) techniques have established notable search and optimization abilities in solving many exciting optimization problems [2]. Particularly, Firefly Algorithm (FA) and Particle Swarm Optimization (PSO) are leading methods within EAs and SI, respectively [3-5]. GA's ability is shown in solving real-world problems, particularly those with constraints or discrete parameter sets. This research explores the advantages of utilizing several metaheuristic search algorithms to fine-tune the parameters of a PID used to control a Direct Current (DC) motor system. By optimizing the model parameters, we aim to enhance the controller response. This inquiry delves into a novel fitness function prioritization, amalgamating all the performance above control aspects of the PID. The velocity regulation of a DC motor is realized via PID tuning algorithms [1],[6]. The ensuing outcomes of various scenarios are meticulously examined and juxtaposed to discern the most optimal configuration for the PID.

The paper is organized as follows: In the next Section, a synopsis of the background and related studies is provided. Followed by an illustration of the problem statement. Then, delves into the mathematics of the PID controller. A comprehensive discussion on evolutionary algorithms is presented, followed by an investigation of swarm intelligence algorithms. Afterward, the conceptual model of the DC motor is offered, and diverse fitness functions are introduced. Subsequently, the experimental results were derived from the algorithms. Finally, the study's conclusion and insights into forthcoming research directions are presented..

2. Background and Related Work

Several heuristic approaches were utilized to fine-tune the parameters of PIDs [1]. The classical tuning criteria known as Ziegler-Nichols [7] were initially employed. However, attaining optimal or nearly optimal PID parameters through this formula in industrial settings often proves challenging. Nevertheless, recent strides in computational techniques have paved the way for developing optimization algorithms geared toward adjusting control parameters to enhance performance.

In earlier research, scholars have delved into the application of Evolutionary algorithms and swarm intelligence for parameter tuning in DC motors [8-10]. GA, derived from the concept of natural selection and belonging to the broader spectrum of evolutionary algorithms, is one such method that has been harnessed to fine-tune PID parameters in DC motors by employing diverse fitness functions [11] [12]. The results indicate that the GA approach surpasses manual tuning techniques like the Ziegler-Nichols method. GA initiates with a set of potential solutions, and through multiple iterations involving processes like selection, crossover, and mutation, the most robust individuals endure, constituting the ultimate solution. PSO, a more recent heuristic strategy introduced by Kennedy and Eberhart [4], emulates a streamlined social system and proficiently addresses continuous nonlinear optimization challenges. Compared to alternative stochastic methodologies, PSO demonstrates swift production of high-quality solutions with consistent convergence characteristics. Consequently, the field demands sophisticated meta-heuristic algorithms like ACO and GWO, belonging to the Swarm Intelligence domain. ACO emulates how ants discover the optimal route from their nest to a food source by depositing pheromone traces along routes they consider optimal [13]. GWO is a metaheuristic search algorithm inspired by the *Canis lupus* behavior [14]. GWO maintains three types of wolves: alpha α, beta β, and delta δ. The function of these types is to lead the search for the prey. α represents the leading best solution in our search. The other two solutions come next in line. The remaining wolves are designated as omega ω solutions. Gray wolves can locate the position of prey and create a circle around it. The α wolf always leads the hunting process. This emulation captures the intricate hierarchy of leadership within a wolf pack. Importantly, solutions rooted in heuristic principles consistently demonstrate superior performance in real-world scenarios, providing advantages over conventional methods. Not only do they optimize memory usage, but they also deliver enhanced results. Moreover, the authors [15] provided a comprehensive evaluation of contemporary and traditional approaches to tweaking the parameters of PID controllers by utilizing metaheuristic algorithms.

3. Problem Statement

A DC motor operates by converting DC electrical power into mechanical power [2]. Its speed and torque are regulated through the manipulation of voltage and current. Any alterations in the input parameters of a DC motor can directly impact its system behavior. PID controllers come into play to maintain precise control over the DC motor. However, for optimal performance, it becomes imperative to fine-tune the parameters of these PID controllers [11][12][16].

Evolutionary Algorithms are renowned for their superior performance, attributed to their capacity to effectively and swiftly tackle diverse problems. Notably, these algorithms are able to overcome the challenge of local minimum entrapment, a pitfall encountered by other methodologies in path planning scenarios. The enhanced performance of evolutionary algorithms is underpinned by their non-greedy nature, enabling a harmonious balance between locally and globally optimal solutions. All evolutionary algorithms share a common trait of emulating nature's intelligent bionic evolution. For instance, the foraging behavior of ants inspired the "ant colony optimization" approach, while the Crow Search Algorithm replicates a crow's search and hiding strategies for food. Genetic algorithms are rooted in Darwin's theory of biological evolution and the natural selection processes governing genetic evolution.

4. PID Controller Design

The PID controller assesses the error as the variance between the setpoint, representing the desired or target value, and the process variable, denoting the current measured value of the process. In response, the controller adjusts the process control inputs to diminish this error. The three basic concepts of PID are described below and illustrated in Figure 1.

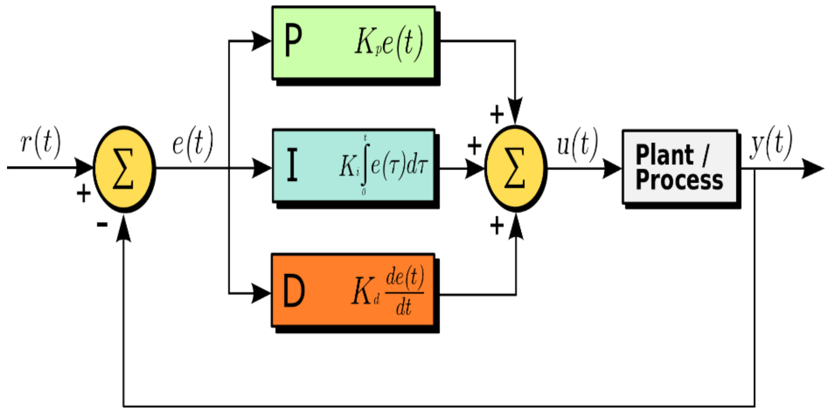


FIGURE 1. PID Controller System

- P -controller multiplies the proportional constant with the error to get the output. It reduces the fluctuation in the output. P depends on the present error.
- I-controller is used to minimize steady-state error, yet there is always a persistent offset between the set value and the output. I-controller integrates steady-state error until it reaches the value zero. It depends on past errors.
- D-controller depends on the change in error. It predicts the future behavior of the error. When the set point is modified By predicting the error's potential future behavior, D-controller makes the output usually responds. D-controller depends on future errors.

One form of PID Controller is expressed as in Equation 1 with a list of definions in Table 1. It can be expressed as a transfer function in Equation 2.

$$u(t) = G \times \left(K_p e(t) + K_i \int e(t)\ dt + K_d \frac{de(t)}{dt}\right) \cdot e^{-K_N t} \tag{1}$$

$$U(S) = K_p + K_i \times \frac{1}{s} + K_d \times s \tag{2}$$

where the output of the PID control circuit is denoted as $u(t)$, K_p is the proportional gain, K_i is the integral gain, K_d is the derivation gain and $e(t)$ is the error signal (i.e., $y(t) - r(t)$).

TABLE 1: Definitions of variables used in the PID controller

Variable	Definition
$u(t)$	Control output at time t
G	System gain
K_p	Proportional gain
K_i	Integral gain
K_d	Derivative gain
$e(t)$	Error signal at time t
$\int e(t)\ dt$	Integral of the error signal over time
$\frac{de(t)}{dt}$	Derivative of the error signal with respect to time

K_N	Exponential decay constant
t	Time variable
$e^{-K_N t}$	Exponential decay factor

The classical PID controllers are extensively employed in industrial control systems owing to their simplicity and efficacy [17].

5. Evolutionary Algorithms

Evolutionary Algorithms (EA) derive insights from Darwin's theory of natural selection, which centers on the principle of "survival of the fittest." These algorithms employ biological operations such as mutation, crossover, and selection. Their application extends to optimization techniques and real-world problem-solving.

5.1. Genetic Algorithm (GA)

Genetic Algorithms (GAs), a randomized, global search technique, emulate the principles of natural evolution. Over time, computational systems have progressively enhanced their efficacy, rendering them valuable for specific optimization tasks. The most favorable solution is ascertained based on environmental feedback, in conjunction with evolution operators like reproduction, crossover, and mutation. Commencing without prior knowledge of the optimal solution, the genetic algorithm averts the pitfalls of local minima and convergence to suboptimal outcomes by initiating multiple independent search points, thus conducting concurrent explorations. The step-by-step process of the Genetic Algorithm, inspired from [2] is delineated in Algorithm 1.

Algorithm 1: GA Algorithm

```
1. Procedure:
    (a) Set count for the number of generations; $t = 0$;
    (b) Create initial population of solutions $P(0)$ randomly
    (c) Evaluate the fitness function for each individual $P(t)$, where $t$ is the population index $t$
2. While Termination criteria is not satisfied do:
    (a) $t$++
    (b) Create the next generation
    (c) Introduce crossover among a subset of the population $t$
    (d) Introduce mutation within a portion of the population $t$
    (e) Evaluate the fitness for the next generation.
3. Return best solution
```

6. Swarm Intelligence Algorithms

Swarm Intelligence (SI) is a computational approach to addressing intricate challenges. Within this paradigm, a swarm constitutes an assembly of discrete entities, which could encompass individuals or animals. Each entity's accumulated information collectively contributes to the problem-solving process, ultimately deriving a solution.

6.1. Particle Swarm Optimization (PSO)

Kennedy and Eberhart [4] originally proposed Particle Swarm Optimization (PSO); they drew inspiration from sociobiological observations. They observed that a collective of birds or fish moving in unison could harness the collective experiences of all members, leading to enhanced outcomes. While one bird searches for food, the entire flock benefits from shared discoveries, augmenting their collective hunting success.

In the PSO framework, the simulation mimics the behavior of a bird flock, with each "bird" aiding in the quest for the optimal solution within a multi-dimensional problem space. The best position within the group encapsulates this individual contribution. Additionally, the global best position within the flock signifies the best solution as represented in Equation 3 and 4 [18]. Through iterative particle velocity and position updates, the algorithm navigates toward improved solutions, continuing until predefined stopping criteria are fulfilled. While PSO solutions often approach the ideal outcome, the algorithm's superiority as the ultimate solution is not guaranteed in all cases.

$$V_i = \omega V_{i-1} + c_1 r_1 (P_{best} - X_{i-1}) + c_2 r_2 (G_{best} - X_{i-1}) \quad (3)$$
$$X_i = X_{i-1} + V_i \quad (4)$$

Where V_i to denote the particle's speed and X_i to represent the particle's position. P_{best} stands for the best individual value of a particle, and G_{best} stands for the best global value of the entire population. c_1, c_2 represents the acceleration coefficients, ω is the intertia weight, and r_1, r_2 represents the random numbers. Inspired from [2], the sequential PSO algorithm is given in Algorithm 2.

Algorithm 2: PSO Algorithm

```
1. Initialize position $X_i(0)$ and velocity $V_i(0)$ for each particle randomly as in Equation 3
2. While The termination condition has not been met do:
   (a) For $i = 1 \ldots it_{max}$ Do:
       • Compute the Fitness Function
       • Revise the current and global best positions
       • Calculate each particle's speed $V_i$
       • Revise each particle's position $X_i$
```

6.2. Ant Colony Optimization (ACO)

The inception of ACO was initially put forth by Marco Dorigo [13]. Ants naturally gravitate towards living in colonies and engaging with one another through various communication modes, including touch, sound, and pheromones. For ants, the paramount goal revolves around locating food, a factor that significantly influences their behavior. As ants traverse their environment, they deposit pheromone-like organic substances onto the ground to establish navigational cues. These pheromones are chemical messengers, facilitating communication among ants. Upon stumbling upon a food source, an ant accumulates provisions and leaves pheromones along its path back to the colony. The amount and caliber of the food influence the amount of pheromones released by the ant. Other ants detect this olfactory trail and follow it to reach the food source. The intensity of the pheromone concentration directly correlates with the likelihood of other ants opting for the same route.

Considering a network where ants can traverse between nodes, the probability of ant k stationed at node I transitioning to another node in the network hinges upon the extent of pheromone presence along the path. This probabilistic aspect is depicted in Equation 5.

$$p_k^{ij} = \begin{cases} \dfrac{\left(\tau_{ij}^k\right)^\alpha \left(\eta_{ij}^k\right)^\beta}{\sum_{l \in N_i^k} \left(\tau_{il}^k\right)^\alpha \left(\eta_{il}^k\right)^\beta} & if \;\; j \in N_i^k \\ 0 & if \;\; j \notin N_i^k \end{cases} \quad (5)$$

where τ_{ij} represents the quantity of pheromone deposited for the transition from state i to j, $\alpha \geq 0$ is a parameter regulating the impact of τ_{ij}, η_{ij} signifies the desirability of the state transition ij, and $\beta \geq 1$ is a parameter that regulates the impact of η_{ij}. Additionally, τ_{il} and η_{il} represent the trial level and attractiveness for other potential state transitions. The evaporation rate, which reduces the value of deposited pheromone over time, is determined by Equation 6.

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \sum_{k}^{m} \Delta\tau_{ij}^{k} \tag{6}$$

Here, ρ stands for the coefficient of pheromone evaporation, m represents the number of ants, and $\Delta\tau_{xy}^{k}$ denotes the amount of pheromone laid down by k^{th} ant. Once the pheromones evaporate and the ants that just crossed the trail have laid fresh pheromones, the new pheromone levels are updated according to Equation (7).

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & if\ \ k^{th}\ ant's\ turn \\ 0 & Otherwise \end{cases} \tag{7}$$

where L_k represents the cost of the tour length undertaken by the k^{th} ant and Q is a constant. The detailed Algorithm of ACO is given in Algorithm 3.

Algorithm 3: ACO Algorithm

```
1. Procedure:
    (a) Initialize the system parameters
2. While Termination state is not attained do:
    (a) Generate Solutions
    (b) Calculate fitness
    (c) Update Pheromone in Equation 5
    (d) Repeat
```

6.3. Grey Wolf Optimization (GWO)

The GWO was first presented by Mirjalili and his collaborators [14]. This algorithm derives its inspiration from the intricate behaviors exhibited by wolves in their living and hunting dynamics. Wolves tend to operate in cohesive packs, which are organized into distinct categories: alpha (α), beta (β), delta (δ), and omega (ω). Among them, alpha wolves possess the highest rank, serving as leaders to the other wolves. Following them in precedence are β and δ wolves, while ω wolves assume the lowest priority. Each category of wolves has its designated leader.

When engaged in hunting, the wolves collectively adopt a strategy. First, they identify their target prey and subsequently update their individual positions accordingly. During this process, every wolf within the pack adheres to the guidance of their respective leaders, belonging to the alpha, beta, and delta categories according to Equation 8(a-f). This orchestration ensures that the pack operates cohesively to optimize their collective hunting endeavor.

$$\overrightarrow{D_\alpha} = \left|\overrightarrow{C_1} * \overrightarrow{X_\alpha}(t) - \mathcal{N}(t)\right| \tag{8-a}$$

$$\overrightarrow{X_1}(t) = \overrightarrow{X_\alpha}(t) - \overrightarrow{A_2} * \overrightarrow{D_\alpha} \tag{8-b}$$

$$\overrightarrow{D_\beta} = \left|\overrightarrow{C_2} * \overrightarrow{X_\beta}(t) - \vec{X}(t)\right| \tag{8-c}$$

$$\overrightarrow{X_2}(t) = \overrightarrow{X_\beta}(t) - \overrightarrow{A_2} * \overrightarrow{D_\beta} \tag{8-d}$$

$$\overrightarrow{D_\delta} = \left|\overrightarrow{C_3} * \overrightarrow{X_\delta}(t) - \vec{X}(t)\right| \tag{8-e}$$

$$\overrightarrow{X_3}(t) = \overrightarrow{X_\delta}(t) - \overrightarrow{A_2} * \overrightarrow{D_\delta} \tag{8-f}$$

Therefore, updating the position of grey can be computed according to Equation 9.

$$\vec{X}(t+1) = AVERAGE\left(\overrightarrow{X_1}, \overrightarrow{X_2}, \overrightarrow{X_3}\right) \tag{9}$$

where \$$t$\$ represents the actual iteration, \$$A$\$ and \$$C$\$ are coefficients associated with vectors, and X_α, X_β, and X_δ denote the positions of the α, β, and δ wolves, respectively, with X signifying any wolf's location. The coefficients A and C are determined as follows:

$$\vec{A} = 2\vec{a} \times \overrightarrow{r_1} - \vec{a} \tag{10}$$

$$\vec{C} = 2\overrightarrow{r_2} \tag{11}$$

As the iterations go on, the value decreases from 2 to 0, r_1 and r_2 are random values, $\{r_1, r_2\} \in [0,1]$. The step-by-step procedure of the GWO Algorithm is given in Algorithm 4.

Algorithm 4: GWO Algorithm

```
1. Generate initial population $X_i, i = 1,..., n$
2. Calculate $a, A,$ and $C$ using Equations 10 and 11.
3. Determine the fitness of each search wolf.
    (a) $X_\alpha$ = best search wolf
    (b) $X_\beta$ = second best search wolf
    (c) $X_\delta$ = third best search wolf
4. **While** $t < it_{max}$ **Do**:
    (a) **For each** wolf **Do**:
        • Randomly initialize $r_1$ and $r_2$
        • Update the position of current wolfusing Equation 8
    (b) Compute $a, A,$ and $C$
    (c) Compute the fitness of all the wolves
    (d) Update $X_\alpha$, $X_\beta$, and $X_\delta$
    (e) $t$++
5. **Return** $X_\alpha$
```

7. Problem Formulation

The PID Controller governs the attributes of the DC motor, ensuring their regulation. We can attain the targeted output through parameter adjustment within the PID while minimizing errors. The DC motor's traits are consolidated into a singular function for study purposes. To analyze its dynamics, the DC motor is encapsulated within a Transfer function:

$$G(s) = \frac{1}{s^3 + 9 \times s^2 + 23 \times s + 15} \tag{12}$$

A closed-loop feedback controller (CLFC) is used to regulate the dynamics of the DC motor. PID controller is represented as in Equation 13.

$$U(S) = K_p + K_i \times \frac{1}{s} + K_d \times s \tag{13}$$

7.1. Fitness Function

Within optimization problems, the fitness function is pivotal, guiding the algorithm toward identifying the optimal solution. Essentially, this function assesses the system's performance output and measures the extent to which a proposed solution deviates from the desired setpoint. Consequently, the fitness function considers essential metrics such as settling time, rise time, etc. In the context of this research, our selected fitness functions encompass rise time, peak time, settling time, overshoot, and Mean Square Error (MSE). Specifically, MSE computes the average squared difference between the setpoint and the process variable according to Equation 14.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \big(y(i) - w(i)\big)^2 \tag{14}$$

Where $y(i)$ and $w(i)$ are DC motor output and desired output, respectively.

It is important to first establish certain key definitions before outlining the fitness function.

1. **Rise-time** (t_r) is the time taken to transition from 10% to 90% of the steady-state value.
2. **Settling-time** (t_s) is the time needed for the output to reach and consistently stay within a specified error margin.
3. **Peak-time** (t_p) is the time at which the first response peak value occurs.
4. **Overshoot**(t_o) is the fact that the output value exceeds the desired steady-state value.

In this research, we propose the following fitness function:

$$F = \gamma_1 \times t_r + \gamma_2 \times t_s + \gamma_3 \times t_p + \gamma_4 \times t_o + \gamma_5 \times MSE \tag{15}$$

where: $\gamma_{1\ldots5}$ represent the weights changed to alter how well each function contributes to fitness. The above fitness function gives importance to all the parameters, and by changing the weights, we can get the desired fitness function.

8. Experiemental Results

To refine the PID controller's parameters, a trial-and-error method was utilized to identify the most appropriate parameters for Genetic Algorithms (GAs), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Grey Wolf Optimization (GWO). The setting parameters are presented in tables 2-5, respectively.

TABLE 2: GA Parameter Settings.

Parameter	Value
Population Size	100
Mutation Fraction	0.1
Crossover Fraction	0.6
Lower Bound	100
Upper Bound	500
Iterations	100

TABLE 3: PSO Parameter Settings.

Parameter	Value
Swarm Size	100
Inertia Weight (ω)	0.2
Cognitive Parameter (c_1)	0.8
Social Parameter (c_2)	0.8
Lower Bound	100
Upper Bound	500
Iterations	100

TABLE 4: ACO Parameter Settings.

Parameter	Value
Number of Ants	30
Initialisation of pheromone	100
Pheromone Weight (ω)	0.2

Heuristic data Weight (α)	0.8
Evaporation rate (ρ)	0.7
Lower Bound	100
Upper Bound	500
Iterations	100

TABLE 5: GWO Parameter Settings.

Parameter	**Value**
Number of Search Agents	30
Lower Bound	100
Upper Bound	500
Iterations	100

8.1. Test Case 1

The fitness function has been adjusted with the following parameters: $\gamma_1 = 10$, γ_2= 20, γ_3= 10, γ_4= 10, and γ_5= 40. The relevant findings are displayed in Table 6. Figure 2 depicts the convergence curve and the step response.

TABLE 6: Calculated time for each algorithm (case 1)

Algorithm	t_r	t_s	t_p	t_o
GA	0.3203	1.5436	0.5703	0.5172
PSO	0.2643	1.4926	0.4894	1.5297
ACO	0.2309	1.9409	3.6494	1.9066
GWO	0.2643	1.4898	0.4894	1.5394

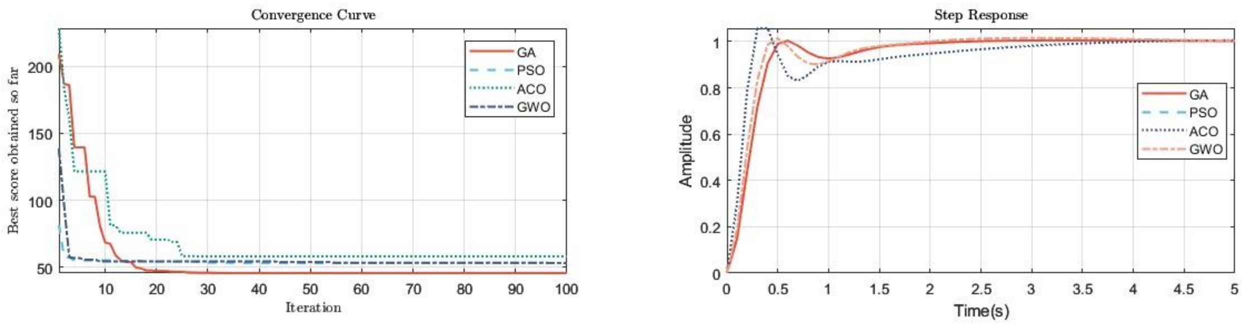


FIGURE 2. Curves for $\gamma_1 = 10$, γ_2= 20, γ_3= 10, γ_4= 10, and γ_5= 40

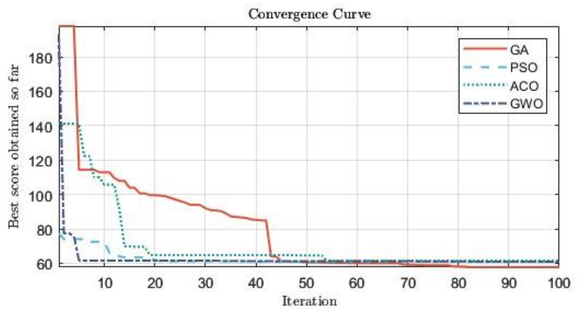
From Table 6, ACO is the fastest in terms of rise-time (t_r) but has significant drawbacks in terms of settling time, peak time, and overshoot, which suggests that it is quick to react but prone to instability and slower convergence to the steady-state value. PSO and GWO are more balanced, with good performance in rise-time, settling-time, and peak-time, though PSO suffers from a high overshoot. GWO appears to be more stable, with a lower overshoot compared to PSO. However, GA shows the most controlled behavior, with the lowest overshoot and reasonable times in other parameters, though it is slower in rise-time and peak time. It may be a better choice for systems requiring precision and minimal deviation from the steady-state value. Therefore, giving significance to rising time, settling time, and MSE in the fitness function, GA outperforms PSO by 17%, GWO, and ACO by 28%. This is evident in the performance indices, where rising time, peak time, and settling time exhibit lower values, signifying better performance. However, it is noteworthy that the overshoot experiences a considerable increase in this scenario.

8.2. Test Case 2

For the second test case, we have adjusted the values of the fitness function to: $\gamma_1 = 1$, γ_2= 30, γ_3= 1, γ_4= 10, and γ_5= 40. Table 7 contains the relevant findings, while Figure 3 shows the convergence curve and step response.

TABLE 7: Calculated time for each algorithm (case 2)

Algorithm	t_r	t_s	t_p	t_o
GA	0.281	1.4236	0.524	1.3588
PSO	0.2637	1.3941	0.4891	1.771
ACO	0.251	2.7848	0.5004	5.8543
GWO	0.2637	1.402	0.4892	1.7482



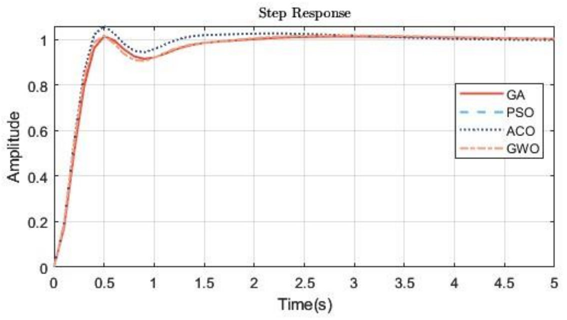


FIGURE 3. Curves for $\gamma_1 = 1$, γ_2= 30, γ_3= 1, γ_4= 10, and γ_5= 40

Based on the results of rise time (t_r) listed in Table 7, ACO is the fastest, followed closely by GA, which shows a 10.7% improvement compared to ACO. PSO and GWO show a smaller improvement of 4.9%. For Settling time (t_s), PSO is the best performer here, with a small 2.1% improvement over GA, while ACO shows a substantial improvement of 50% over PSO, making it the most improved in this parameter. According to peak-time (t_p), both PSO and GWO perform equally well in this parameter, achieving the lowest peak time. GA and ACO show moderate improvements over the best values. However, GA is the best performer in terms of Overshoot (t_o=1.3588). All methods converge almost simultaneously in the fitness function when the MSE is given more weight. The results achieved surpass those of the prior function; nonetheless, the overshoot is very high.

8.3. Test Case 3

In the third test case, the fitness function is modified with the following values: $\gamma_1 = 20$, γ_2= 50, γ_3= 20, γ_4= 10, and γ_5= 50. Table 8 displays the outcomes that were achieved. Figure 4 displays the convergence curve and step response.

TABLE 8: Calculated time for each algorithm (case 3)

Algorithm	t_r	t_s	t_p	t_o
GA	0.2757	1.31	0.4962	1.865
PSO	0.2631	1.3322	0.489	1.9999
ACO	0.2336	2.5135	0.4414	3.7458
GWO	0.2632	1.3371	0.489	1.9822

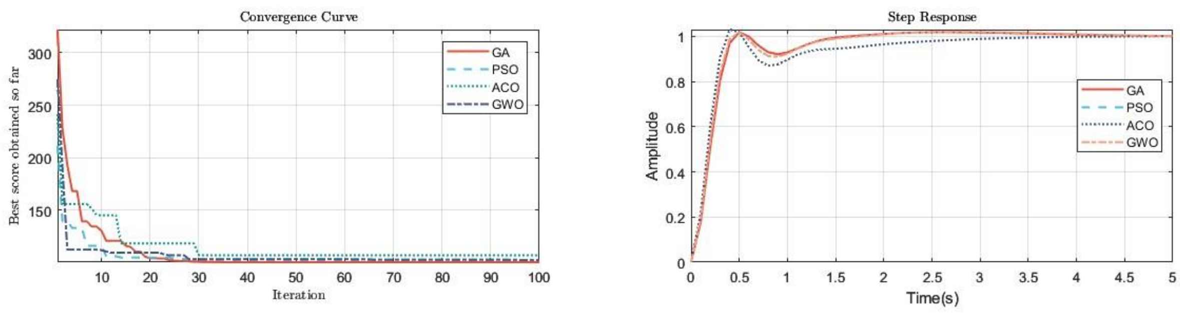


FIGURE 3. Curves for $\gamma_1 = 20$, γ_2= 50, γ_3= 20, γ_4= 10, and γ_5= 50

While the ACO performs best with the shortest rise time of ($\boldsymbol{t_r}$=0.2336) , the GA shows the most improvement in rise time, with a 15.3% reduction compared to ACO. Also, GA performs best in $\boldsymbol{t_s}$=1.3100, $\boldsymbol{t_o}$=1.8650 which makes it a good choice for stability and control.

Despite assigning significant weight to Overshoot in the fitness function, there is no observed reduction in overshoot when compared to the other performance indices.

9. Conclusion and Future Work

This paper aimed to enhance the parameters of a Proportional-Integral-Derivative (PID) to control a Direct Current (DC) motor system through the utilization of Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), and Grey Wolf Optimization (GWO) techniques. The fitness function considered in this study encompassed various performance metrics, including rise-time, settling-time, peak-time, overshoot, and Mean Square Error. The study's findings demonstrated that the proposed methodology yielded significant improvements in terms of rise time, peak time, response precision, and error reduction compared to conventional approaches. Moreover, the step response exhibited enhanced performance, establishing the superiority of the proposed PID controller over existing designs. Future research endeavors could explore further adjustments to the weighting of the fitness function and the investigation of novel algorithms to enhance the PID controller's parameters further using various evolutionary and swarm-based optimization techniques.

References

[1] Stephen Bassi Joseph, Emmanuel Gbenga Dada, Afeez Abidemi, David Opeoluwa Oyewola, and Ban Mohammed Khammas. Metaheuristic algorithms for PID controller parameters tuning: Review, approaches and open problems. Heliyon, page e09399, 2022.

[2] DheerajReddy Maddi, Alaa Sheta, Dharani Davineni, and Heba Al-Hiary. Optimization of PID controller gain using evolutionary algorithm and swarm intelligence. In 2019 10th International Conference on Information and Communication Systems (ICICS), pages 199–204. IEEE, 2019.

[3] Alaa Sheta, Malik S Braik, and Sultan Aljahdali. Genetic algorithms: a tool for image segmentation. In 2012 international conference on multimedia computing and systems, pages 84–90. IEEE, 2012.

[4] James Kennedy. Particle swarm optimization. encyclopedia of machine learning. Springer, 760:766, 2011.

[5] Malik Braik and Alaa Sheta. Exploration of genetic algorithms and particle swarm optimization in improving the quality of medical images. Computational intelligence techniques in handling image processing and pattern recognition. Lambert Academic Publishing (LAP), Germany, pages 329–360, 2011.

[6] Mirza Muhammad Sabir and Junaid Ali Khan. Optimal design of pid controller for the speed control of dc motor by using metaheuristic techniques. Advances in artificial neural systems, 2014, 2014.

[7] J.G. Ziegler and N.B. Nichols. Optimum settings for automatic controllers. Transactions of the American Society of Mechanical Engineers, 64(7):759–768, 1942.

[8] Mahmud Iwan Solihin, Lee Fook Tack, and Moey Leap Kean. Tuning of PID controller using particle swarm optimization (PSO). In Proceeding of the international conference on advanced science, engineering and information technology, volume 1, pages 458–461, 2011.

[9] N Kundariya and J Ohri. Design of intelligent PID controller using particle swarm optimization with different performance indices. International Journal of Scientific & Engineering Research, 4(7):1191–1194, 2013.

[10] K Latha, V Rajinikanth, and PM Surekha. PSO- based PID controller design for a class of stable and unstable systems. International Scholarly Research Notices, 2013, 2013.

[11] PM Meshram and Rohit G Kanojiya. Tuning of PID controller using ziegler-nichols method for speed control of DC motor. In IEEE-international conference on advances in engineering, science and management (ICAESM-2012), pages 117–122. IEEE, 2012.

[12] Saeed Abedini and Hassan Zarabadipour. Tuning of an optimal PID controller with iterative feedback tuning method for DC motor. In The 2nd

International Conference on Control, Instrumentation and Automation, pages 611–615. IEEE, 2011.
[13] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. Ant colony optimization. IEEE computational intelligence magazine, 1(4):28–39, 2006.
[14] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. Grey wolf optimizer. Advances in engineering software, 69:46–61, 2014
[15] Stephen Bassi Joseph, Emmanuel Gbenga Dada, Afeez Abidemi, David Opeoluwa Oyewola, and Ban Mohammed Khammas. Metaheuristic algorithms for pid controller parameters tuning: review, approaches and open problems. Heliyon, 8(5):e09399, 2022.
[16] Hang Wu, Weihua Su, and Zhiguo Liu. PID controllers: Design and tuning methods. In 2014 9th IEEE Conference on industrial electronics and applications, pages 808–813. IEEE, 2014.
[17] Ahmed M. Nassef, Mohammad Ali Abdelkareem, Hussein M. Maghrabie, and Ahmad Baroutaji. Metaheuristic-based algorithms for optimizing fractional-order controllers—a recent, systematic, and comprehensive review. Fractal and Fractional, 7(7), 2023.
[18] Liu Nanping, Xia Kewen, Zhou Juan, and Ge Chun. Numerical simulation on transistor with cqpso algorithm. In 2009 4th IEEE Conference on Industrial Electronics and Applications, pages 3732–3736. IEEE, 2009