https://doi.org/10.64142/jeai.1.3.37

Journal of Engineering and Artificial Intelligence

ISSN: 3117-3357

Research Article

JEAI-25-37

Intelligent Tuning of PID Parameters Using Nature-Inspired Algorithms

Ibrahim Mammadov[*], **Samrad Maharramov and Kanan Aghakishiyev**

Department of Electrical and Electronics Engineering, Sakarya University, Sakarya, Turkey

[*]**Corresponding author:** Ibrahim Mammadov, Department of Electrical and Electronics Engineering, Sakarya University, Sakarya, Turkey, E-mail: ibrahim.mammadov@ogr.sakarya.edu.tr

Received date: October 10, 2025; **Accepted date:** November 8, 2025; **Published date:** November 11, 2025

Citation: Mammadov I, Maharramov S, Aghakishiyev K (2025) Intelligent Tuning of PID Parameters Using Nature-Inspired Algorithms. J Eng Artif Intell Vol.1 No.3: 37.

Abstract

This paper proposes the application of both the Firefly Algorithm (FA) and Particle Swarm Optimization (PSO) for tuning the PID controller parameters of first-, second-, and third-order dynamic systems. FA is distinguished by its simplicity, stable convergence behavior, and superior computational efficiency. Inspired by the bioluminescent behavior of fireflies, the Firefly Algorithm is a well-established meta-heuristic optimization technique. In this study, FA's performance is benchmarked against the widely used PSO method. Simulation results show that FA yields slightly better improvements in step-response metrics namely rise time, settling time, and overshoot while PSO can attain lower fitness values in some scenarios. FA demonstrates the ability to quickly and reliably optimize controller parameters, even for complex system models. Detailed analyses across multiple test cases confirm that FA not only excels at parameter optimization but also provides enhanced stability and robustness in controller design. PSO, on the other hand, sometimes achieves marginally lower fitness values, indicating its potential for fine-tuning in specific cases. This work highlights the applicability of both FA and PSO in academic and industrial control-system design, emphasizing each method's strengths and trade.

Keywords: Firefly algorithm; PID controller; Optimization; Particle Swarm Optimization (PSO); Control systems; Meta-heuristic algorithms; Dynamic systems; Performance analysis

Introduction

Traditional PID controllers are widely used in various industrial processes. Their popularity stems from their simple functionality, high reliability, and broad applicability. PID control algorithm is used to control almost all control levers in process industries and is the basis of advanced control algorithms and strategies. However, industrial applications of PID controllers show that parameter tuning is still a challenging problem in many applications. Traditionally, this problem has been addressed by trial-and-error method. In recent years, more systematic methods have been introduced [1-4]. Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) are frequently used approaches to find the optimal PID parameters [5]. Despite the number of methods proposed, these methods are not fully systematic and may sometimes exhibit poor performance in practice. In recent years, metaheuristic approaches have attracted the attention of researchers interested in engineering control problems [6]. In 1997, Bagis introduced an improved Genetic Algorithm (GA) method to determine the optimal parameters for PID controllers [7]. Puangdownreon and Sujitjorn presented an adaptive tabu search procedure based on backtracking and additional use of adaptive search trajectory mechanism for the optimal PID controller parameters [8]. In order to achieve the desired closed-loop system response, DEA based PID tuning studies were discussed by Bagis and Savascihabes [9]. In this paper, an efficient tuning method is required to find the optimal PID parameters. This method is based on the Firefly Algorithm (FA). In fact, the Firefly Algorithm (FA) is a new meta-heuristic optimization method inspired by the behavior of real fireflies. Xin-She Yang from Cambridge University first proposed the concept of utilizing cooperative firefly agents for optimization problems [10], which was later expanded upon in subsequent studies [11,12]. Recently, it has been adapted to solve various design problems. FA has been successfully used in permutation flow priority scheduling problems [13], clustering problems [14] and mixed variable structure optimization [15]. The Ant Algorithm demonstrates excellent performance across different application domains. In this study, a novel tuning approach is needed to adjust the PID controller parameters for various processes.

The obtained results are compared with the existing method Particle Swarm Optimization (PSO) by simulation. Simulation experiments show that FA based approach exhibits different performance compared to PSO method. In addition, there are similar studies comparing PSO and other optimization methods, for example, in the study of A. Köybaşı and İ. Yazıcı, PSO is compared with grey wolf optimization [16]. Firefly Algorithm (FA) is an effective tool in PID parameter optimization as a powerful meta-heuristic method inspired by nature. There are studies comparing FA with traditional methods such as Ziegler-Nichols [17]. In this study, the performance of FA is compared with PSO, a modern optimization method.

The paper is composed of sections. Section I, Introduction. Section II, presents the formulation of the PID parameter tuning problem. Section III, explains the basic features of FA. Section IV, explains the implementation of the proposed approach. Section V, presents the numerical simulations and comparisons. Finally, Section VI, combines the results.

Mathematical Representation of the PID Parameter Optimization Problem

The block diagram presented in **Figure 1** is taken into consideration. In this figure, $G_{PID}(s)$ is the controller and $G_P(s)$ this represents the process to be controlled. In practical applications, the output of the PID controller and the transfer function (in a parallel structure) are expressed by the following equations, respectively [18]:

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de}{dt} \tag{1}$$

$$u(t) = k_p\ (t) \left[e(t) + \frac{1}{T_i}\int e(t)dt + T_d \frac{de(t)}{dt}\right] \tag{2}$$

$$G_{PID}(s) = k_p \left[1 + \frac{1}{T_i s} + T_d s\right] \tag{3}$$

The first equation (1) shows the general formula of the PID controller in the time domain. It is the equation used to calculate the output (control signal) of the system.

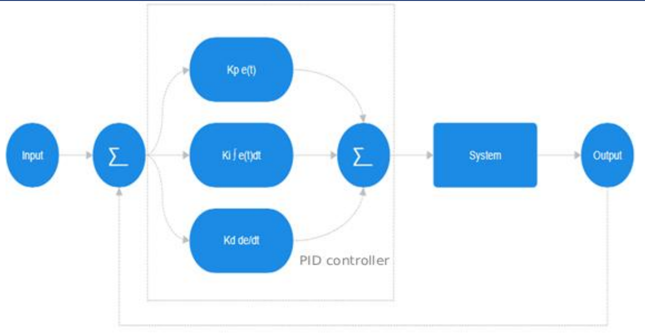


Figure 1: Block diagram of basic PID control system.

$u(t)$, the control signal, is the signal applied to the input of the system. It is the output of the PID controller. K_p, proportional gain. This term is used to adjust the control signal proportionally to the error signal in the system. The larger the error signal, the more $u(t)$. K_i, integral gain. This term is used to adjust the control signal for error signals that accumulate over time. It is used to correct long-term error conditions. K_d, derivative gain. This term is used to adjust the control signal based on the rate of change (derivative) of the error signal. It tries to predict the future behavior of the system and reacts to rapid changes. $e(t)$, the error signal, is the difference between the desired (reference) value and the current system output. $\int e\,(t)dt$, is the integral of the error signal. The integral term takes into account the accumulation of error over time. $\frac{de(t)}{dt}$, is the derivative of the error signal. The derivative component considers the error's rate of change [19].

The third equation (3) shows the transfer function of the PID controller expressed by the Laplace transform. This formula is used to understand how the PID controller behaves in the frequency domain and the overall response of the system [20].

$G_{PID}(s)$, the transfer function of the PID controller. It expresses the system behavior in terms of the Laplace transform. K_p, proportional gain term. $\frac{K_i}{s}$, integral term, the s term represents the integration operation in the Laplace transform. $K_d s$, derivative term. The s term represents the differentiation operation in the Laplace transform [21].

For a given system or process, the primary objective of the PID tuning problem is to efficiently adjust the PID controller parameters to achieve the desired performance with an optimal closed-loop time response. To accomplish this, the proposed FA-based PID tuning approach is illustrated schematically in **Figure 2** [21].

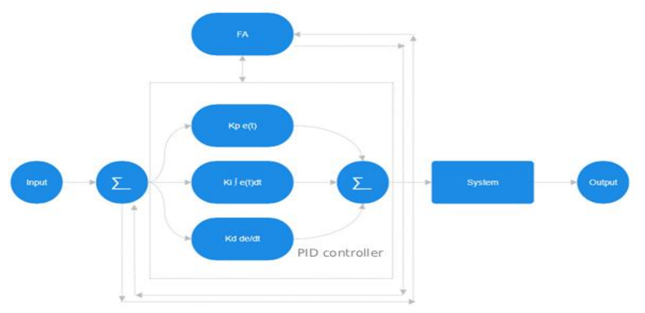


Figure 2: PID tuning scheme with FA.

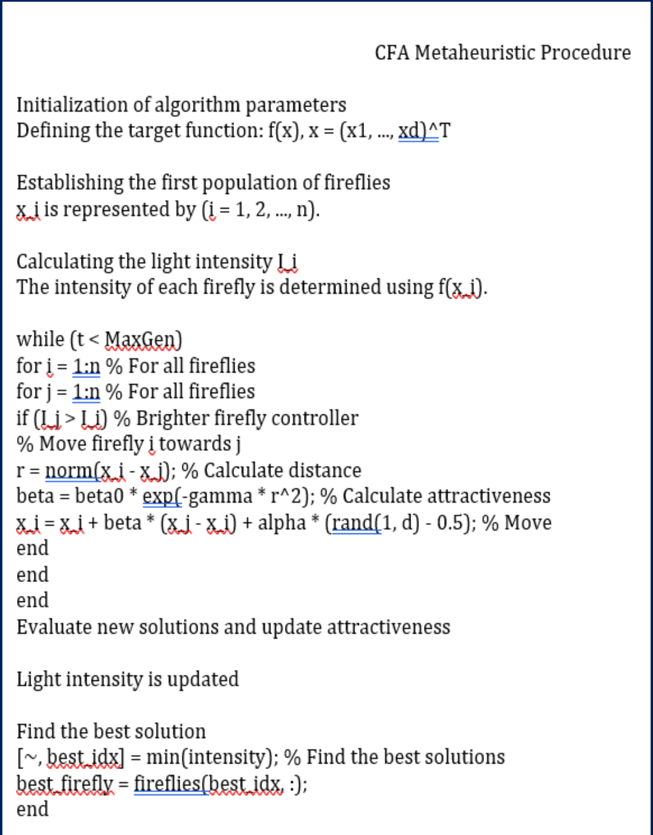
The FA aims to identify these optimal parameter values by minimizing a predefined fitness function. In the time domain, this fitness function can be formulated using various performance metrics, including the absolute error over time, rise time, settling time, overshoot, and steady-state error. In this study, the following performance function is utilized as the fitness function for PID controller design:

$$F = \left((1 - e^{-\rho})(OS + E_{ss}) + e^{-\rho}(t_s + t_r)\right) \quad (4)$$

F here represents the fitness function. Fitness function is a criterion used to evaluate the performance of the system. Here, it forms a total value that measures the performance of the PID control system. OS (Overshoot) indicates the amount by which the system response exceeds the desired value. That is, it is the peak point at which the system responds and returns after a certain period of time. E_{ss}(Steady-State Error), It is the amount of error that shows how close the system is to the target value in the long run. It is observed when the system is in a stable state. t_s(Settling Time), The time it takes for the system response to remain within the desired tolerance limits. In other words, it refers to the duration required for the system response to attain a steady state. t_r(Rise Time), It represents the duration required for the system to reach the target value for the first time from its initial state. That is, it represents the rise time of the system's response. ρ, is a parameter used in this formula and is usually used as a weighting factor. It is set to determine the degree of importance of the terms used in the fitness function. That is, it controls which metrics will be evaluated with more importance in terms of system performance.

Fundamentals of the Firefly Algorithm

The Firefly Algorithm is a nature-inspired optimization technique that mimics the flashing behavior of fireflies [22]. It is built upon three idealized principles derived from the natural characteristics of fireflies: (1) All fireflies are identical in terms of gender and move toward brighter and more attractive fireflies. (2) The attractiveness of a firefly is directly proportional to its brightness, which diminishes as the distance between fireflies increases. (3) If a firefly does not find another one that is brighter or more attractive, it moves randomly. In the context of optimization problems, the flashing light represents the fitness function, helping to achieve optimal solutions efficiently. The primary steps of the standard Firefly Algorithm are outlined in the pseudo-code shown in **Figure 3** [23].



```
CFA Metaheuristic Procedure

Initialization of algorithm parameters
Defining the target function: f(x), x = (x1, ..., xd)^T

Establishing the first population of fireflies
x_i is represented by (i = 1, 2, ..., n).

Calculating the light intensity I_i
The intensity of each firefly is determined using f(x_i).

while (t < MaxGen)
for i = 1:n % For all fireflies
for j = 1:n % For all fireflies
if (I_j > I_i) % Brighter firefly controller
% Move firefly i towards j
r = norm(x_i - x_j); % Calculate distance
beta = beta0 * exp(-gamma * r^2); % Calculate attractiveness
x_i = x_i + beta * (x_j - x_i) + alpha * (rand(1, d) - 0.5); % Move
end
end
end
Evaluate new solutions and update attractiveness

Light intensity is updated

Find the best solution
[~, best_idx] = min(intensity); % Find the best solutions
best_firefly = fireflies(best_idx, :);
end
```

Figure 3: CFA metaheuristic pseudocode.

This algorithm employs two main mechanisms for solution search: attractiveness and movement, which are defined as follows:

Attractiveness

The attractiveness function of a firefly is represented by the following monotonically decreasing function:

$$\beta(r) = \beta_0 \, exp(-\gamma r^m), \qquad m \geq 1 \quad (5)$$

$\beta(r)$ Attractiveness at distance r. Indicates how attractive the firefly is to other fireflies. Attractiveness decreases as distance increases. β_0 Initial attractiveness coefficient. It is the attractiveness value when r=0 (*i.e.* two fireflies are very close to each other). γ Absorption coefficient. A parameter that determines how quickly the attraction decreases with distance. r The distance between two fireflies. As this distance increases, the attraction decreases. m It is a parameter that controls the effect of distance on attractiveness. $m \geq 1$ Under the condition that the relationship between distance and attractiveness is strengthened or weakened. Usually m=1, but larger values cause the attractiveness to decrease more rapidly with distance.

This formula shows how the attractiveness level of fireflies changes with distance. β_0 initial attraction, γ ve m parameters control how quickly the attraction decreases and how it changes with distance. $m \geq 1$ This restriction implies that attractiveness is negatively correlated with distance, and that attractiveness decreases more rapidly as distance increases. $exp(-\gamma r^m)$ This is an expression that shows that attraction decreases exponentially as distance increases. This causes the fireflies' attraction to fireflies that are farther away to decrease rapidly.

The distance r between any two fireflies i and j at locations X_i and X_j respectively can be defined as cartesian or Euclidean as follows [24]:

$$r_{ij} = \sqrt{\sum_{k=1}^{d} \left(x_{i,k} - x_{j,k}\right)^2} \quad (6)$$

Here , r_{ij} represents the Euclidean distance between firefly i and firefly j, $x_{i,k}$ i is the position of firefly i on the k-coordinate axis (*e.g.,* x y or z), $x_{j,k}$ is the position of firefly i on the k-coordinate axis (*e.g.,* x y or z), d is the number of dimensions of the problem (*e.g.,* d=2 in a two-dimensional problem), k is the index of the coordinate axes (*e.g.,* k=1,2,3 for x,y,z).

Motion

The movement of firefly i toward a brighter firefly j is described by the following equation [24]:

$$x_i^{(t+1)} = x_i^{(t)} + \beta_0 \exp\left(-\gamma r_{ij}^2\right)\left(x_j - x_i\right) + \alpha(rand - 0.5) \quad (7)$$

Here, $x_i^{(t)}$ is the current position of firefly i (its current location), $\beta_0 \exp\left(-\gamma r_{ij}^2\right)$ is the attraction function. It depends on the distance between two fireflies. β_0 is the maximum attraction coefficient,γ The coefficient that determines how quickly light decreases with distance, r_{ij} The Euclidean distance between fireflies i and j,$\left(x_j - x_i\right)$ The vector that directs firefly i towards firefly j.

This expression represents the motion towards the brighter firefly. $\alpha(rand - 0.5)$ randomness term. It adds the random component of the motion if there is no brighter firefly or if there is uncertainty.α is the randomness coefficient. It is adjusted according to the nature of the problem,rand: Generates a random number in the range (0,1).

First Term $(x_i^{(t)})$ - This term represents the current position of the firefly. The firefly starts moving from this position.

Second Term ($\beta_0 \exp\left(-\gamma r_{ij}^2\right)\left(x_j - x_i\right)$) - Represents the attraction force towards the brighter firefly. The attraction decreases exponentially with distance (r_{ij}).

Third Term ($\alpha(rand - 0.5)$) - A random movement component is included to prevent the algorithm from being trapped in local minima and to maintain diversity within the search space.

If there is a firefly (j) brighter than firefly i, then firefly i moves towards firefly j. If there is no brighter firefly $\left(I_j \leq I_i\right)$, it makes a random move. This mechanism allows the algorithm to both explore the coordinate axis for the solution and converge to the best solution.

Application of Firefly Algorithm to Tuning Pid Parameters

Optimal tuning of PID (Proportional-Integral-Derivative) controller parameters is a critical step to improve system performance in many engineering applications. Firefly Algorithm (FA) is an effective tool for such optimization problems as a powerful meta-heuristic method inspired by nature. The Firefly Algorithm (FA) is derived from the natural behavior of fireflies and incorporates this behavior into the solution-search process. This process provides dynamic convergence towards the best solution starting from a random initial population [25].

The implementation of Firefly Algorithm in PID controller optimization is accomplished through the following steps:

- Creating the initial population: FA starts with an initial population of feasible solution candidates. This population consists of fireflies representing the parameters of the control system. Each firefly represents a parameter set consisting of the values of the PID controller (K_p , K_i , K_d) .
- Simulation of closed loop system: For each firefly, the determined PID parameters are evaluated through a closed-loop simulation of the system. Throughout the simulation, both transient and steady-state responses are observed. This step is essential for assessing the effectiveness of the selected PID parameter set.
- Calculation of performance criteria: Simulation results are evaluated with the following performance criteria:
- ISE (Integral of Squared Error): Integral of squared error measures the overall error performance of the system

- ST (Settling Time): It is the time for the system response to reach steady state.
- OS (Overshoot): It refers to the maximum amount of overshoot in the system response.

These criteria are essential in determining the fitness function.

- Assessment of fitness function: The fitness value of each firefly is calculated by a function defined to optimize performance criteria [26].
- Firefly movement and attraction mechanism: In FA, each firefly moves towards brighter (*i.e.* better fitness) fireflies. The equation of motion consists of three components: the current position, the movement to other fireflies due to attraction, and the random search component. This allows the algorithm to both seek the global optimum and avoid getting stuck in local minima.
- Convergence and finding the best solution: All fireflies are ranked based on their fitness values, and the best solution within the population is identified. The algorithm proceeds either until the maximum number of iterations is reached or the fitness function satisfies a predefined threshold.
- Analysis of results and implementation: Once the algorithm is completed, the optimal PID parameter set is implemented in the system. The system's performance is then evaluated to confirm improvements in response time, overshoot, and error values of the controller:
- Fast convergence: Produces faster and more accurate results thanks to the dynamic attraction mechanism.
- Flexibility: Easily adaptable to different optimization problems.
- Avoiding local minimum: It avoids local traps thanks to the randomness factor.
- Flow Diagram

The following flow chart summarizes the application process for PID parameter tuning of FA:

- Firefly population is created.
- Closed-loop simulation program is run.
- ISE, ST and OS are calculated.
- Fitness function is evaluated.
- Fireflies are ranked and the best solutions are found.
- Repeat the process until the maximum number of iterations is reached.

This systematic process shows that FA provides an effective solution to PID parameter optimization problems. It has been observed that the algorithm provides significant improvements in controller design by optimizing the performance criteria **(Figure 4)**.

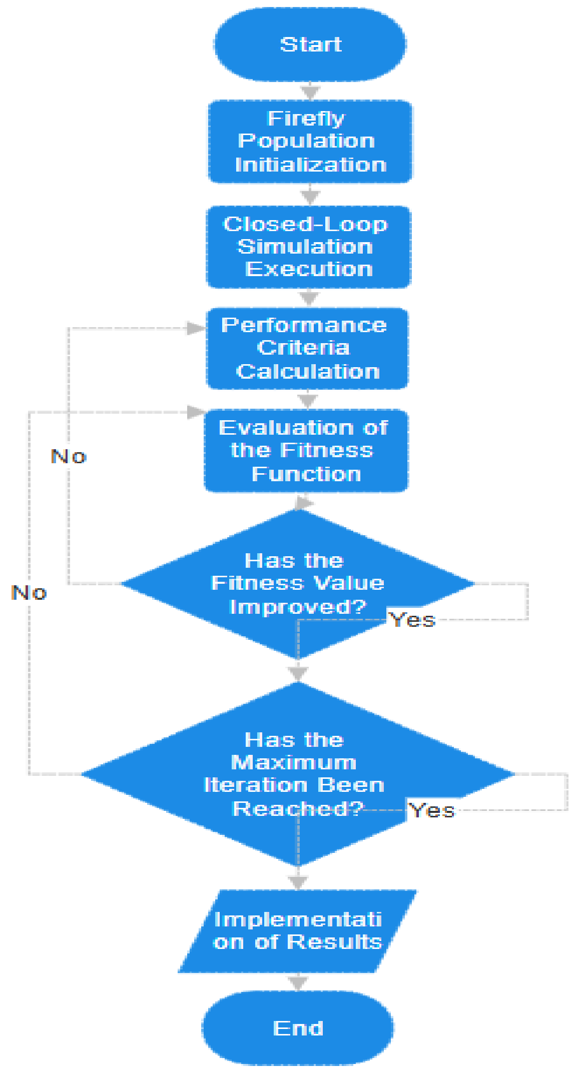

Start
Firefly Population Initialization
Closed-Loop Simulation Execution
Performance Criteria Calculation
Evaluation of the Fitness Function
Has the Fitness Value Improved?
No
Yes
Has the Maximum Iteration Been Reached?
No
Yes
Implementation of Results
End

Figure 4: Flowchart of the proposed FireFly algorithm for the PID controller.

Simulation Results

The proposed method was executed 12 times in MATLAB on an Intel Core i5-11400H 12-core 2.7 GHz PC [27,28]. Initial numerical tests were conducted to optimize the firefly algorithm parameters, and the best-obtained values are summarized in **Table 1**.

Table 1: FA parameter values.

Parameter	Explanation	Value
n	Number of fireflies	20
α	Randomness factor	0.2
β_0	Maximum attractiveness	1
γ	Light absorption coefficient	1
ng	Maximum number of iterations	100

Test

To verify the validity and effectiveness of the proposed approach, simulation experiments were performed on three different processes from the literature. The closed-loop response to a step change was compared with the results obtained using the PSO method. Transfer functions of these processes (G_1 G_2 G_3) were used [29].

For each process examined, the closed loop against step change was compared with the PSO method. For this purpose, the PSO-based PID controller parameters were obtained by optimizing the fitness function, which includes performance criteria such as overshoot, steady-state error, settling time and rise time. Simulation results revealed that the PSO-based approach outperformed traditional optimization methods in terms of response time and overall system stability [30].

For each test process, the PID controllers designed with the proposed approach and the PSO method are presented together with the obtained fitness functions and performance criteria of the processes. The comparison shows that the PID parameters optimized with the PSO method provide better results in control responses such as lower overshoot, shorter settling time and minimized steady-state error.

$$G_1(s) = \frac{K}{s+a} \tag{8}$$

$$G_2(s) = \frac{K}{s^2 + 2\zeta\omega_n s + \omega_n{}^2} \tag{9}$$

$$G_3(s) = \frac{K}{s^3 + as^2 + bs + c} \tag{10}$$

$$G_1(s) = \frac{\pi}{3s + 4} \tag{11}$$

$$G_2(s) = \frac{\pi}{3s^2 18s + 9} \tag{12}$$

$$G_3(s) = \frac{\pi}{3s^3+4s^2+3s+4} \tag{13}$$

Discussion and comparison

Closed loop responses to step change for each process were compared with the proposed method. For this purpose, PSO based PID controller parameters were optimized and obtained and the results were evaluated through simulations. For comparison, PID parameters were tested on different processes and performance criteria such as overshoot ($OS\%$), steady state error (E_{ss}), settling time (t_s) and rise time (t_r) were used.

The PID parameters optimized for the system are as shown in **Table 2** below:

Table 2: Optimized parameter values.

K_p	4.215
K_i	7.832
K_d	0.337

As a result, the PID controllers designed with the proposed approach and the PSO method, the obtained fitness functions and the performances of the processes are listed in **Table 3**.

The table includes the comparison of the PID parameters obtained with the PSO method with the PID parameters obtained with the Firefly Algorithm (FA) method. The obtained results show that in the comparison between the PSO and FA based approaches, the PSO based approach performs better in terms of control response and provides significant improvements in the stability and response time of the system. This comparison evaluates the results of both methods and reveals that PSO is a more effective optimization method.

Table 3: Best solutions using FA and PSO for different processes with ρ=0.9.

Function	Method	PID parameters			Performance				Fitness
		K_p	K_i	K_d	OS%	t_s	t_r	E_{ss}	F
$G_1(S)$	FA	7.445	10	0	0	0.49683	0.28092	0	0.31621
	PSO	7.3979	10	0	0	0.49581	0.28194	0	0.31621
$G_2(S)$	FA	10	5.2684	0	0	1.3311	0.82971	0	0.87582
	PSO	10	5.2694	0	0	1.3308	0.82965	0	0.87837
$G_3(S)$	FA	1.915	0.80528	6.8141	36.151	10.252	0.48609	0	46.88909
	PSO	0.58778	0.44441	2.5663	1.2432	11.409	0.95307	0	13.6053

The time responses for PSO and FA based PID controllers are also plotted in **Figures 5-7**.

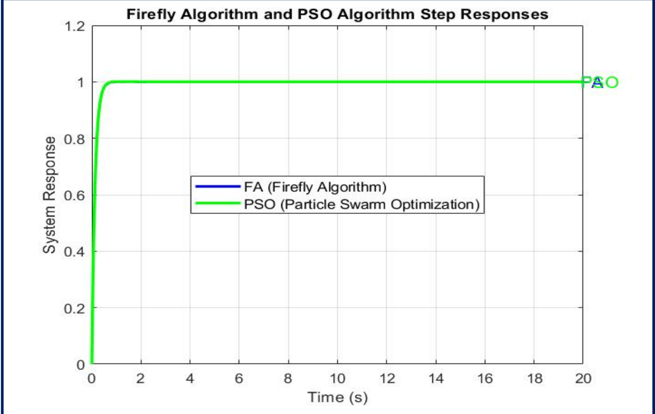


Figure 5: Step response of the G_1 (s)system using PID controllers optimized with the Firefly Algorithm (FA) and Particle Swarm Optimization (PSO).

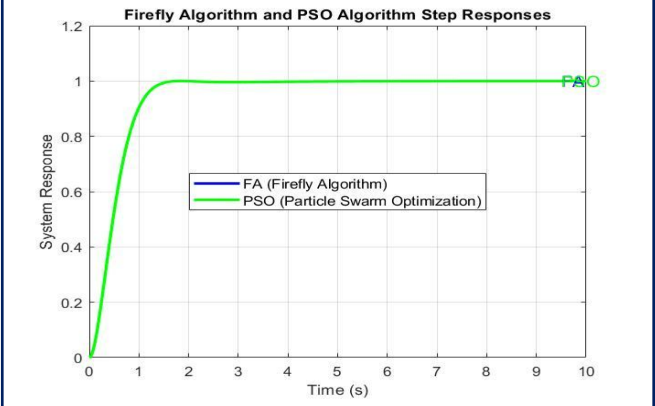


Figure 6: Step response of the G_2 (s)system using PID controllers optimized with the Firefly Algorithm (FA) and Particle Swarm Optimization (PSO).

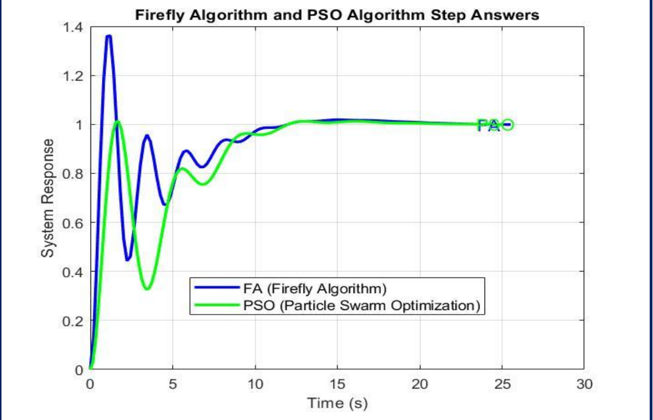


Figure 7: Step response of the G_3 (s) system using PID controllers optimized with the Firefly Algorithm (FA) and Particle Swarm Optimization (PSO).

Conclusion

In conclusion, This study has successfully optimized PID control parameters using both Firefly Algorithm (FA) and Particle Swarm Optimization (PSO). Each meta-heuristic minimizes a fitness function that captures key performance metrics: overshoot (OS), steady-state error (E_{ss}), rise time (t_r), and settling time (t_s). Results with FA: G_1 (s) system: FA yielded optimized PID parameters K_p = 7.445, K_i = 10, K_d= 0. With these values, the system exhibited OS = 0 %, E_{ss} = 0, rise time t_r = 0.28092 s, and settling time t_s = 0.49683 s, indicating a stable, high-performance response. The fitness value was 0.31621, demonstrating excellent optimization efficiency. $G_2(s)$system: FA produced K_p= 10, K_i= 5.2684, K_d= 0. The resulting performance had OS = 0 %, E_{ss} = 0, rise time t_r = 0.82971 s, and settling time t_s = 1.3311 s. Its fitness value was 0.87582, confirming high efficiency. $G_3(s)$ system: FA determined K_p= 1.915, K_i= 0.80528, K_d= 6.8141. This produced OS = 36.151 %, rise time t_r = 0.48609 s, and settling time t_s = 10.252 s, with a fitness of 46.88909. Although acceptable, the $G_3(s)$ system still requires further fine-tuning. Compared to PSO's fitness of 13.6053, FA shows a distinct advantage in complex cases. Overall, FA's dynamic attractiveness mechanism and stochastic exploration allow it to avoid local minima and converge rapidly across all test systems. Results with PSO: In the G_1 (s) and G_2 (s) systems, PSO attained fitness values very close to FA (0.31621 and 0.87837, respectively), occasionally outperforming FA by a narrow margin. This indicates that PSO can sometimes yield slightly lower fitness for simpler systems. For the G_3 (s) system, PSO's fitness was 13.6053 significantly worse than FA's 0.48609 revealing that PSO may require more delicate parameter tuning in complex, higher-order systems. In summary, FA generally ensures stable and rapid convergence, especially in complex scenarios, while PSO can offer marginal fitness improvements in simpler cases through additional fine-tuning. Application perspectives: FA and PSO meta-heuristics could be integrated into PLC-based greenhouse automation to enhance environmental control precision [31,32]. Applying FA- or PSO-tuned PID loops to asynchronous electric drives in pumping applications can improve real-time performance and energy efficiency under PLC environments. Extending this optimization framework to transformer monitoring and fault diagnosis would allow advanced PID control schemes to run on PLC networks for continuous condition assessment and preventive maintenance.

Shan et al. have demonstrated the adaptability of similar meta-heuristic tuning by developing a modified fuzzy-PID-Smith predictive compensation algorithm for a pH-controlled liquid fertilizer system. Furthermore, Mammadov et al. have shown that PSO-based optimization can effectively mitigate the Ferranti effect in long-distance transmission lines, further validating the robustness of meta-heuristic algorithms in complex power systems.Thus, both FA and PSO are effective tools for PID parameter optimization. FA particularly excels in complex scenarios by providing lower fitness values and stable convergence, whereas PSO may be preferable for additional fine-tuning in simpler systems. Future work could explore hybrid FA–PSO approaches and further fine-tuning strategies for highly dynamic control challenges.

Acknowledgments

The authors would like to acknowledge Sakarya University. Mammadov Ibrahim would like to express his special thanks to his girlfriend, Fidan Gafarova, for being by his side and supporting him during this stressful period.

Authors' Contributions

In this study, all authors studied and designed the article.

Competing Interests

The authors declare that they have no conflict of interest.

References

1. Bizuneh A, Mitiku H, Salau AO, Chandran K (2024) Performance analysis of an optimized pid-p controller for the position control of a magnetic levitation system using recent optimization algorithms. Measurement: Sensors 33: 101228. [Crossref], [Google Scholar]
2. Ray PK, Mohanty A (2019) A robust firefly-swarm hybrid optimization for frequency control in wind/pv/fc based microgrid. Appl Soft Comput 85: 105823. [Crossref], [Google Scholar]
3. Wang Y, Zhang Y, Wang W, Liu Z, Yu X, et al. (2023) A review of optimal design for large-scale micro-irrigation pipe network systems. Agronomy 13: 2966. [Crossref], [Google Scholar]
4. Zhang H, Liang Y, Zhang W, Xu N, Guo Z, et al. (2018) Improved pso-based method for leak detection and localization in liquid pipelines. IEEE Trans Ind Inform 14: 3143-3154. [Crossref], [Google Scholar]
5. Zhang X, Lin L (2023) Adaptive join query optimization in big data environments. IEEE Trans Knowl Data Eng 35: 1025-1036.
6. Bağış A (2007) Determination of the pid controllers parameters by using binary and real coded genetic algorithms. J Inf Sci Eng 23: 1469-1480. [Google Scholar]
7. Puangdownreong D, Sujitjorn S (2007) Obtaining an optimum pid controller via adaptive tabu search. Adapt Nat Comput Algo 747-755. [Crossref], [Google Scholar]
8. Bagis A, Savascihabes A (2010) PID tuning by using differential evolution algorithm for desired closed loop system response. Intell Syst Appl 170-174. [Google Scholar]
9. Sakai T, Yoshikawa T, Matsuo Y (2023) Development of firefly algorithm for multiobjective optimization. IEEE Access11: 125-137.
10. Yang XS (2008) Nature-inspired metaheuristic algorithm. Luniver Press. [Google Scholar]
11. Yang XS (2009) Firefly algorithms for multimodal optimization. Stochastic Algorithms: Found Appl 5792: 169-178. [Crossref], [Google Scholar]
12. Sayadia MK, Ramanziana R, Ghaffari-Nasaba N (2010) A discret firefly meta-heuristic with local search for makespan minimisation in permutation flow shop scheduling problems. Int J Ind Eng Comput 1: 1-10. [Crossref], [Google Scholar]
13. Senthilnath J, Omkar SN, Mani V (2011) Clustering using firefly algorithm: Performance study. Swarm Evol Comput1: 164-171. [Crossref], [Google Scholar]
14. Gandomi AH, Yang XS, Alavi AH (2011) Mixed variable structural optimization using firefly algorithm. Comput Struct 89: 2325-2336. [Crossref], [Google Scholar]
15. Köybaşı A, Yazıcı I (2020) Solution of test problems with grey wolf optimization algorithm and comparison with particle swarm optimization. Sakarya Uni J Sci 24: 1252- 1264. [Crossref], [Google Scholar]
16. Bendjeghaba O, Boushaki I, Zemmour N (2013) Firefly algorithm for optimal tuning of pid controller parameters. Int J Intell Eng Inform 5: 343-361, [Crossref], [Google Scholar]
17. Ogata K (2010) Modern control engineering. Prentice Hall. 213-230. [Google Scholar]
18. Nise NS (2011) Control systems engineering. John Wiley and Sons Inc 100-120. [Google Scholar]
19. Chen PC, Luo Y, Peng YB, Chen YQ (2021) Optimal robust fractional order (PID)-D-lambda controller synthesis for frst order plus time delay systems. ISA Trans 114: 136-149. [Crossref], [Google Scholar]
20. Dorf RC, Bishop RH (2011) Modern control systems. Prentice Hall. [Google Scholar]
21. Pradhan PC, Sahu RK, Panda S (2016) Firefly algorithm optimized fuzzy PID controller for AGC of multi-area multi-source power systems with UPFC and SMES. Eng Sci technol Int J 19: 338-354. [Crossref], [Google Scholar]
22. Kara T, Yildirim M (2015) PID controller design and tuning using firefly algorithm for an automated vehicle. J Electr Eng Tech 10: 843-851.
23. Xiao L (2021) Parameter tuning of PID controller for beer flling machine liquid level control based on improved genetic algorithm. Comput Intell Neurosci. [Crossref], [Google Scholar]
24. https://www.mathworks.com/matlabcentral/fileexchange/29693-firefly-algorithm.
25. https://www.mathworks.com/matlabcentral/fileexchange/52857-particle-swarm-optimization-pso

26. Fu C, Ma X, Zhang L (2021) Fuzzy-pid strategy based on pso optimization for ph control in water and fertilizer integration. IEEE Access 10: 4471-4482. [Crossref], [Google Scholar]

27. Maharramov S, Mammadov I, Damirova J (2024) plc əsasli istixana avtomatlaşdirilmasi. Pathei volume 43: 530-537.

28. Suleymanov N, Damirova J, Maharramov S (2024) Greenhouse automation: Using Programmable Logic Control (PLC) for smart plant irrigation system in greenhouses. Pathei 36: 252-264. [Google Scholar]

29. Mammadov I, Rahimli I (2024) Optimization and regulation of asynchronous electric drives for pumping machines. Вестник науки 783-787. [Crossref], [Google Scholar]

30. Aghakishiyev K, Mammadova G (2024) Fault analysis and monitoring of transformer condition. 4: 457-468. [Google Scholar]

31. Shan Y, Zhang L, Ma X, Hu X, Hu Z, et al. (2021) Application of the modifed fuzzy-pid-smith predictive compensation algorithm in a ph-controlled liquid fertilizer system. Process 9: 1506. [Crossref], [Google Scholar]

32. Mammadov I, Aghakishiyev K, Maharramov S (2025) An optimization-based approach for mitigating the ferranti effect through optimal selection and placement of shunt reactors in long-distance transmission lines. Вестник Ггнту Технические Науки 2: 664-680. [Crossref], [Google Scholar]