Int. J. Nonlinear Anal. Appl. In Press, 1–9
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2023.21415.4024

A review of adaptive tuning of PID-controller: Optimization techniques and applications

Israa Ahmed Abbas*, Muntadher Khamees Mustafa

Department of computer science, College of Science, University of Diyala, Baqubah, Iraq

(Communicated by Nallappan Gunasekaran)

Abstract

The PID controller's well-established advantages have led to its widespread application in practically all industrial operations. The effectiveness of the controller has a major impact on the overall performance of the system, making tuning an essential part of the system's operation, which made this a hot topic for academics to dig into for some time. This paper provides an overview of both contemporary and old PID tuning methods. Techniques can be categorized into two major groups, namely: Traditional and optimization tuning methods are two examples of tuning techniques. A comparison between some of the different methods has as well as being made available. The primary objective of this paper is to give thorough for PID professional controllers.

Keywords: DC Motor, PID Controller, Classical techniques, Optimization Techniques
2020 MSC: 37N40, 65K10

1 Introduction

As a general control loop feedback mechanism, the PID controller may be found in a variety of industrial control systems, especially those with exact mathematical models. The proportional, integral, and derivative values are used to calculate the PID controller [9]. The proportional word refers to adjusting the control output in proportion to the amount of mistake (the difference between the set point and the process variable) [6]. Integral term: This term reduces the offset by raising the system's type and order by one and boosting the system's reaction speed, albeit at the price of prolonged oscillations [3].

The reaction is defined by the derivative value, which is based on the pace at which the error is changing. The sum of these three acts issued to be imported into the regulated system [9]. Changes in the proportionality constants of these variables have an effect on the system's reaction. PID tuning or adjusting the PID proportionality constants is critical because of this [3]. Traditional parameter tuning methods and clever optimization algorithms are the two types of PID controller parameter tuning methods available. Traditional approaches to parameter tuning include the empirical, upwards curve, critical ratio, damping oscillatory, and relay feedback techniques. There has been tremendous influence on the real control system as a result of using or modifying the Z-N approach as an empirical parameter tuning method ; almost all PID controller suppliers and users do so [23]. Various strategies have been employed for PID tuning, with the Ziegler Nichols methodology being one of the first. There are two types of approaches: classical

*Corresponding author
Email addresses: scicompms2202@uodiyala.edu.iq (Israa Ahmed Abbas), alkarawis@uodiyaia.edu.iq (Muntadher Khamees Mustafa)

Received: January 2023 *Accepted:* March 2023

and computational or optimization techniques. Classical procedures establish assumptions about the plant and the desired output and seek to get some process characteristic analytically or visually. This knowledge is then utilized to select controller settings. These strategies are appropriate for initial iteration since they are computationally very quick and easy to apply. It's not always possible to get the expected effects from controller settings because of the assumptions made. Traditional methods were evaluated in this article. Techniques for Improving Results PID tuning for data modeling and cost function optimization has made use of these strategies. Genetic algorithms and differential evolution are two examples of computer methods for modeling complicated systems. There is a cost function that must be minimized in order for optimization techniques to work. Cost functions are classified into four types: integral absolute error (IAE), integral square error (ISE), integral time absolute error (ITAE), and integral time square error (ITSE).

$$IAE = \int_0^T |e(t)| \tag{1.1}$$

$$ISE = \int_0^T |e(t)|^2 \tag{1.2}$$

$$ITAE = \int_0^T t|e(t)| \tag{1.3}$$

$$ITSE = \int_0^T t|e(t)|^2. \tag{1.4}$$

Computer models are used to adjust PID controllers for optimal performance. If there are any process deviations detected, the PID settings are reset to provide the desired response. PID controller self-tuning effectively sets the PID parameters while simultaneously modeling the process using some computational model and compares the results. Adaptive approaches are categorized based on the idea that if the process dynamics change [16], the controller should alter its settings. There are two sorts of process dynamics variations: predictable and unexpected. If you have nonlinearities, you can use an auto-tuning strategy to deal with them. Then, a gain schedule may be employed to deal with the non-linearities [3].

2 Classical techniques

Assumptions concerning model assumptions and controller settings are commonly made in conventional methodologies. Step response data is used to determine the dynamics of these systems. Different traditional techniques have relied on various equations to specify this reaction [3].

2.1 Ziegler–Nichols

ZN rules, which are typically used to tune PID controllers for plants with unknown dynamics, may also be used to tune PID controllers for plants with known dynamics. Ziegler and Nichols presented guidelines for finding the values of proportional gain K_P, integral time Ti, and derivative time Td based on the transient response characteristic of a specific plant. It's best to use tuning rules when you have a linear, monotonic and slow system, a single-pole exponential "lag" or something like, and an analog controller. This approximation accurately depicts the frequency response rolloff in the great majority of situations, even though genuine plants are unlikely to have a flawless first-order lag characteristic in place. There will be an extra phase shift when using higher-order poles. For loop stability, phase shift is extremely significant, even though it has only a little impact on the gain rolloff form. The amplitude rolloff and the phase shift cannot be adequately matched by a single "lag" pole. Thus, the Ziegler-Nichols model uses an imaginary phase adjustment that does not alter the anticipated magnitude rolloff. As we get closer to the stability margin, we see a 180 degree phase change in the feedback loop. A phase shift of up to 90 degrees can be caused by a first order lag. The rest of the observed phase shift must be compensated for by an artificial phase adjustment. Between zero and critical frequency, the phase adjustment is considered to be a straight line with a 180-degree phase shift between two points. A time delay is the same as a "straight line" phase shift. If so, is this in line with the observed phase shifts? Let's wish for the best, as it is the least likely scenario [13]. Based on controller assumptions, the Ziegler and Nichols technique was the first to be created. Since the controller settings generated are very aggressive, they result in excessive overshoot and oscillatory response, which necessitate additional tuning. The first method's parameters are likewise difficult to estimate in a noisy environment.

2.2 Cohen-Coon

The Cohen-Coon method of controller tuning corrects the slow steady-state response provided by the Ziegler-Nichols method when there is a large dead time (process delay) relative to the open loop time constant; a large process delay is required for this method to be practical because otherwise unreasonable controller gains will be predicted. This approach can only be used with first-order models with time delay since the controller does not respond immediately to the disturbance (the step disturbance is progressive rather than instantaneous). The Cohen-Coon technique is a 'offline' tuning approach, which implies that after the input has achieved steady-state, a step change can be implemented. Once the output has been measured, the initial control settings may be evaluated based on the response. In Cohen-Coon approach, there are specified parameters to achieve a minimal offset and a standard decay ratio of 1:4. (QDR). A decay ratio of 1/4(QDR) means that the second oscillation has just a quarter of the amplitude of the first. Table 1 displays these options [19].

Table 1: Cohen Coon forecasts can be improved by using the standard suggested formulae.

	Kc	**Ti**	**Td**
P	$(P/Nl) * (1 + (R/3))$		
PI	$(P/Nl) * (0.9 + (R/12))$	$L * (30 + 3R)/(9 + 20R)$	
PID	$(P/Nl) * (1.33 + (R/4))$	$L * (30 + 3R)/(9 + 20R)$	$4L/(11 + 2R)$

where P is the percent change in input, N is the percent change in output/T, and L is T_{dead}, R is T_{dead}/T. This method is useful for systems with a time delay and a faster closed loop response time [22].

3 Optimization Techniques

Additional mathematical programming approaches that differ conceptually from the tar-based optimization methods have been developed in the last several years. Methods like this are considered to be contemporary or unconventional. Numerous techniques have been developed by studying biological, molecular, swarm-insect, and neural systems.

Genetic algorithms, simulated annealing, Particle Swarm Optimization, Ant Colony Optimization, Fuzzy Optimization, Neural-Network-Based Approaches, Crow Search Algorithm, Artificial Bee Colony and Grey Wolf Optimizer are some of the methods covered in this area.

3.1 Genetic Algorithms

Inherent in genetic algorithms are concepts from natural selection and evolutionary genetics. The genetic search strategy makes use of the natural processes of reproduction, genetic crossover, and mutation. Some of the ways in which GAs diverge from conventional optimization strategies are as follows: First, instead of starting with a single design point, a set of points (called trial design vectors) is used. Population size is often expected to be between two and four design variables n. To avoid being stuck at a single point, GAs employ many points as possible solutions. GAs only utilize the value of the goal function. No derivatives are used in the search algorithm. In natural genetics, chromosomes are represented as strings of binary values in GA design variables. Consequently, the search approach may be used to solve discrete and integer programming issues. The length of the string for continuous design variables can be changed to obtain any resolution required. In natural genetics, a design vector's objective function value is used as fitness. A new set of strings is generated with each generation by randomly selecting some of the existing strings and crossing them with new ones. However, GAs are not just a basic random search strategy since they are randomized. They effectively integrate fresh data with existing knowledge to find a new generation with improved fitness or an objective function value. The standard genetic algorithm is given below, and the algorithm flowchart is shown in Fig. 1.

GA is widely used in PID tuning and has many applications in control systems [12]. Girishraj et. al This study found that GA beat Ziegler Nichols, Skogestad modification and IMC rule [4] in terms of overshoot, disturbance rejection, gain margin, and phase marginality in a bioreactor PID controller. The scalability and applicability of GA in multivariable system tuning were examined. A DC motor's position and speed may be controlled using GA Reverse osmosis and cascade control systems' PIDs have been fine-tuned with the help of GA [11].

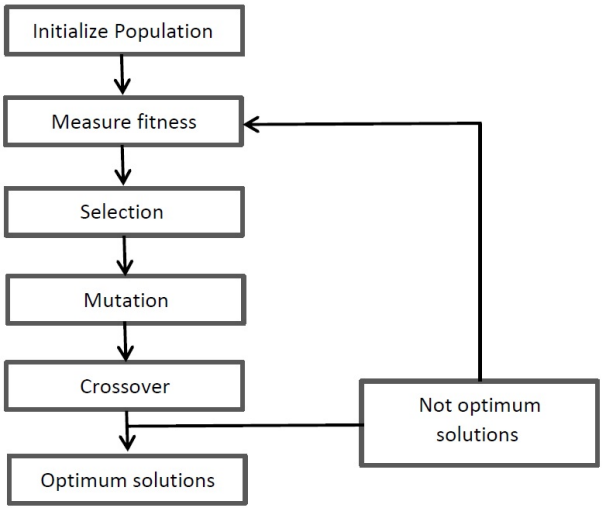

Figure 1: Flowchart of genetic algorithm based tuning

3.2 Simulated Annealing

This approach is based on a simulation of thermal annealing of critical hot solids. When the metal is heated to a high enough temperature, the atoms in the molten state are no longer bound to each other and may move freely with respect to one another. Atoms, on the other hand, are confined in their mobility when the temperature lowers. Crystals with the lowest possible internal energy are formed when the temperature decreases, since the atoms tend to organize themselves as a result of this tendency.

The PID tuning of time-delay controllers for high-performance drilling processes has made use of simulated annealing through the use of simulated heat. As a result of the multimodal search space's enormous size and nonlinearity, the typical Simulated annealing method is computationally costly The outcome is a modified form of SA called Orthogonal Simulated Annealing (OSA), which is used to optimize several fuzzy neural networks simultaneously while PID tuning different controllers [3].

3.3 Particle Swarm Optimization

When it comes to ants, termites, bees, and wasps, Particle Swarm Optimization (PSO) takes its cues from the behavior of these and other swarming insects, as well as from birds and fish. Particle swarm optimization is aimed to emulate the behavior of social creatures. A bee colony or a group of birds might be referred to as a particle. The swarm's collective or group intelligence is utilized by each person or particle in a swarm in a dispersed fashion. Since the swarm can communicate with each other instantaneously, even far away particles can follow an individual particle's discovery of an efficient route to food. Genetic algorithms, which are based on evolution, are known as behaviorally inspired algorithms. Swarm intelligence-based optimization approaches, on the other hand, are known as evolutionary algorithms. In 1995, Kennedy and Eberhart developed the PSO algorithm [17].

Due to its great computing efficiency, the Particle Swarm Optimization (PSO) approach established by Kennedy and Eberhart is frequently employed in a wide range of engineering challenges, such as the optimization of complex systems. In order to reduce the maximum overshoot, rising time, speed tracking error, steady state error, and settling time, this approach is applied. As a result of the optimization solution, the Pareto front or ideally surfaces may be found.

With the help of the movement of swarm intelligence, PSO is an effective stochastic optimization approach. PSO components include swarm size, velocity, position components, and the maximum number of iterations. PSO controller construction is depicted in Fig. 2.

Random candidate solutions are seeded into the Particle Swarm Optimization (PSO) model by a swarm of particles. They scan the d-dimension issue space again and over again for fresh approaches. Position vector X_k^i represents the location of every particle. I is the index of the particle and V_k^i is the speed of the particle represented by two vectors: a location and a velocity P_{best}^i denotes the particle's most recent and most preferred location. The P_{globel}^i vector is then used to hold the swarm's optimal location vector. At iteration k, the following formula determines the new velocity

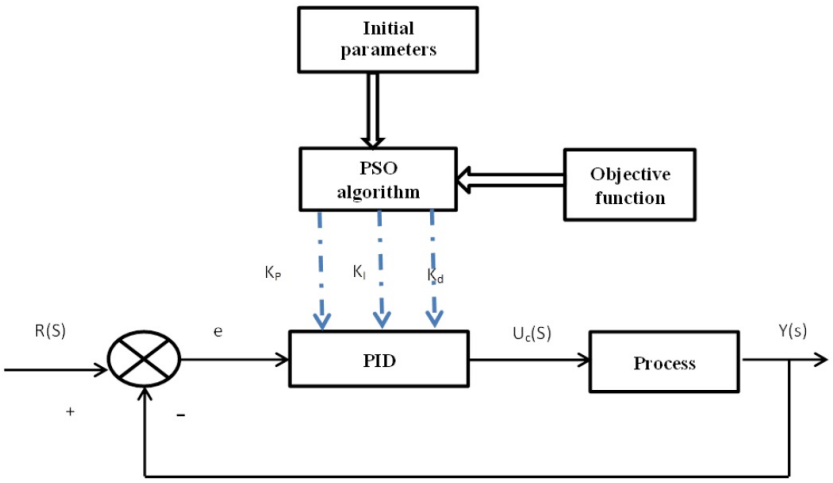

Figure 2: The structure of the PID controller with PSO algorithm

based on the old one:

$$V_k^i = wV_k^i + c_1R_1(P_{best}^i - X_k^i) + c_2R_2(P_{globel}^i - X_k^i) \tag{3.1}$$

The new location is then calculated by the sum of the previous and the new velocity:

$$X_{k+1}^i = X_k^i + V_k^i \tag{3.2}$$

There are two random numbers, referred to as R1 and R2. To choose its next move, a particle takes into account the memory of its best prior location and the swarm's most successful particle's experience [15].

3.4 Ant Colony Optimization

Authentic ant colonies work together to choose the quickest route from their homes to their sources of food. ACO, on the other hand, mimics this cooperative conduct. The approach was developed by Dorigo and his colleagues in the early 1990s. Each ant leaves a pheromone trail behind it as it searches for food. It becomes weaker as the number of ants going through it falls, and it becomes stronger as the number of ants travelling through it rises. An ant search algorithm that relies on the cooperation of several colonies is what we're talking about here [3].

Ants, according to Ibtissem Chiha and his coworkers, solve problems by applying a probability rule to their decisions. Two variables are used to derive the probability rule between nodes I and j.

$$p_{ij} = \frac{[T_{IJ}]^{\propto}[\eta_{ij}]^{\beta}}{\sum h \in s[\tau_{ih}]^{\alpha}[\eta_{ih}]^{\beta}} \tag{3.3}$$

Factor η_{ij} is the inverse of the cost function. This factor doesn't change while the algorithm is running; rather, the factor τ_{ih} (associated with pheromone and starting at a value of 0) is modified after each iteration. The heuristic or pheromone factor may be targeted with the help of the parameters α and β. For each path, the following is the pheromone amount change:

$$\nabla\tau_{ij}^A = \begin{cases} \frac{L^{\min}}{L^A}, & \text{if } i,j \in T^A; \\ 0, & \text{else.} \end{cases} \tag{3.4}$$

It's been determined that $L^{\min}$, proposed by ant A, will be the best answer so far. When it comes time for the following iteration, the pheromone chosen is,

$$\tau_{ij}^{(t)} = \rho\tau_{ij}^{(t-1)} + \sum_{A=1}^{NA} \Delta\tau_{IJ}^{A_{(t)}} \tag{3.5}$$

In order to eliminate poor decisions, the ant population and evaporation rate are both counted as NA and.

Ant colony optimization (ACO) was utilized to tune the PID. For minimizing a multi-objective function, it beat genetic algorithms and the Ziegler Nichols technique, although [3].

3.5 Fuzzy Based Optimization

Traditional designs formulate the optimization issue in exact language. Real-world issues, on the other hand, frequently include design data, goal function, and constraint formulations that are both ambiguous and linguistic in nature. PID controller settings may be dynamically modified in real-time in response to signal error and error using the Fuzzy logic controller (FLC). For each plant and each PID controller parameter range with which the FLC is to be utilized, the FLC's design characteristics change. There are no major changes to the controller's basic structure. In Fig. 3, we see the FLC, which is a frequent component in PID tuning [3].

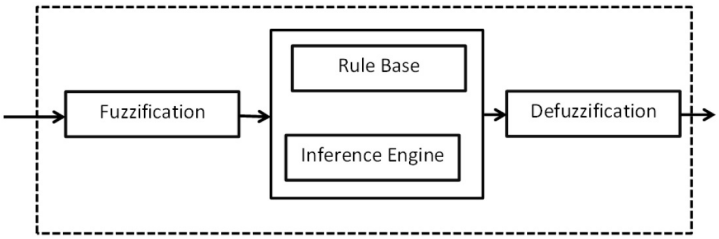

Figure 3: Basic block diagram for fuzzy "control"

The mistake and its derivatives are displayed in Fig. 3 as inputs to the fuzzy interface, as illustrated. Most often, in fuzzy interfaces, the Mamdani model, as described in [10], is utilized. process may be broken down into four sections:

1. Creating a fuzzy variable for each of the crisp inputs.
2. To determine the output for each rule, you must know its fuzzy predecessors.
3. Making an overall assessment of all ambiguous rules.
4. Mapping ambiguous output to clear ones (defuzzification).

It is the sort of controller and real-world experience that decide the fuzzy rules for a certain plant. According to Jantzen, it may be difficult to build integral rule bases. PD fuzzy systems with integrated error control, then, are the optimum choice. It's also possible to combine two fuzzy controllers in order to have the effect of a PID controller [3].

3.6 Neural-Network-Based Methods

The nervous system's enormous parallel processing capacity has been connected to its ability to handle perception difficulties in the midst of enormous volumes of sensory input. Optimization issues are increasingly being solved using neural computing techniques. In a massively parallel network of connected basic processors, the inputs are passed from one neuron to the next, and the computed output is sent on to the output nodes (neurons). There are several ways to describe a neural network, including describing each neuron as an individual, as well as the network as a whole.

PID controllers are not widely utilized because of the intrinsic limitations of neural network theory, such as the difficulty in determining the number of layers and the number of neurons per layer. ANN may, however, describe even extremely nonlinear systems. Using SVMs to create self-tuning controllers is more straightforward. Self-tuning PID controllers make up the majority of neural network work. Figure 4 depicts the flow chart utilized in [21] for self-tuning PID control with GA and ANN.

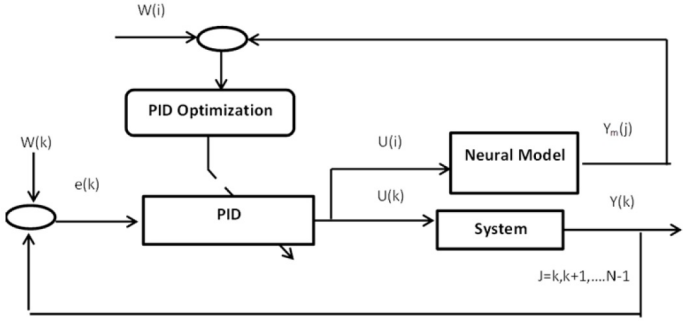

Figure 4: Flowchart of neural network based "controller"

3.7 Crow Search Algorithm

It's been recommended that CSA be used to improve swarm intelligence through simulation of crows' clever behavior when it comes to hiding and recovering food. There are minimal control parameters, and the method is easy to implement.

In the past, birds have been recorded monitoring each other and taking each other's food after the owner has gone away. If a crow has been robbed before, it will take additional care to avoid being a victim again, such as moving its hiding location. In reality, they can forecast a pilferer's conduct based on their own stealing experience and select the best line of action to secure their stockpiles [2].

Based on this hybrid technique, the PID controller and the Crow search algorithm (CSA) are combined to provide an internal function within the PID controller of the DC motor that may be utilized to develop good PID control settings. The selection of PID parameters is viewed as a problem of search space.

When searching for the high-correlated parameters vector, meta-heuristic techniques are acceptable since this results in disturbances that have no impact on DC motor performance [1].

Major sections of the framework may be broken down into two. A random set of parameters is used by the CSA algorithm to estimate the global optimum, after which the cost value for each individual is calculated and the best parameters are assigned to the DC motor's PID Controller system. Search space domain influences the design of each candidate vector. It is the number of PID parameters that determines the length of the vector. Each cell in the vector is determined by a parameter value. For the purpose of evaluating parameter subsets, the error performance indicators are used to produce the cost function, which is specified by the following equations, and is used to repeatedly adjust these parameters until a minimal cost function (desired response) has been achieved,

$$U(t) = K_p e(t) + \int e(t)dt + T_d \frac{de(t)}{dt} \tag{3.6}$$

The disparity between predicted and actual input values is known as "monitoring error" ($e(t)$). In order to calculate the derivative and integral of the error signal, a PID controller receives this signal and performs these calculations. Proportional gain (K_P) multiplied by error magnitude multiplied by the error's integral gain (K_i) and derivative gain (K_d) multiplied by error's derivative are now equal to controller's signal $U(t)$ The performance index is a metric used in the development of PID controllers, as can be seen in the example above (3.6).

The performance of a system controlled by a PID controller. To match a given specification, the set of PID parameters for the system may be tweaked using this approach to find a "optimal system". A final comparison is made between the empirical findings and the four suggested PIDCSA variants (ISE, IAE, ITAE, and ITSE) depending on the kind of error indicator functions and additional methods like the ZN method and PSO algorithm [1].

3.8 Artificial bee colony

For the artificial bee colony algorithm, Karaboga came up with the idea in 2005, and it is still in its infancy. In the same vein as previous approaches of swarm intelligence, this population-based evolutionary algorithm has been proved to be an effective and efficient solution for addressing the optimization issue [5].

The food source is a potential solution to the ABC algorithm's difficulty. The amount of nectar in a food source tells you a lot about the quality of that item. Employed bees, onlookers, and scout bees are all charged with discovering the greatest food sources, which are chosen at random [7].

Iteratively, the ABC computation connects each honey bee to a wide range of nutrient sources. Every cycle, a worker honey bee chooses a nearby food source and measures its nectar yield or overall health using the current environmental conditions.

$$V_{nj} = X_{nj} + \Psi^*_{nj}(X_{nj} + X_{mj}) \tag{3.7}$$

a creative sustenance asset, means a discretionary quantity between +1 and designate the inconsistently selected sustenance asset. Bees will migrate if the creative nutrition asset's health is superior to the previous one; else, they will stick with the old one. When honey bees finish their hunt, they pass on information about their food supplies to other honey bees. If all the utilised honey bee hives are providing nectar, a passing honey bee will weigh their chances of finding food in relation to the health of their food sources, which will be decided by their current state (3.8),

$$P_n = \frac{fit_n}{\sum_{n=1}^{k} fit_n} \tag{3.8}$$

Now, the estimation of the health of the clarify is based on the nectar quantity of the area's nourishment asset and the number of honey bees that are utilized. In the event that the honey bee colony is deprived of a source of nutrition, the used honey bees become scouts.

This is where scouts bring out the creative side of themselves through the use of conjecture (3.9),

$$X_n^j = X_{\min}^j + rand(0,1)(X_j^{\max} - X_j^{\min});\ j = 1, 2, ..., D \tag{3.9}$$

ABC was used to alter the PID for the DC engine's speed control. According to this source, a DC engine may be used to replace an unanticipated switch to a PID controller and an ABC calculation can be built up during the subsequent adjustment process [8].

3.9 Grey Wolf Optimizer Algorithm

Mirjalili et al. invented GWO, which is currently widely used across a wide range of scientific and industrial fields, including astronomy and physics. This algorithm depicts the social structure and hunting prowess of grey wolves. The GWO algorithm is always being tweaked in response to new applications and variants that are being developed. The typical GWO algorithm might be renamed as follows: (IGWO) [20].

A grey wolf's hunting strategy is broken down into three stages, each of which is addressed by the GWO algorithm. After following the prey at a distance, pursuing it for a brief period of time, then encircling it, the hunting process is complete The target is harassed and encircled until it halts and remains still. Finally, wolves assault the prey. Mathematically, Mirjalili et al [14] construct and characterize the hunting processes of grey wolves in accordance with their social order. Here is a basic introduction to the mathematical model: Grey wolf circling behavior may be described mathematically as follows:

$$\overrightarrow{D} = |\overrightarrow{C}\,\overrightarrow{X_P}(iter) - \overrightarrow{X}(iter)| \tag{3.10}$$

$$\overrightarrow{X}(iter+1) = \overrightarrow{X_P}(t) - \overrightarrow{A}\,\overrightarrow{D} \tag{3.11}$$

$\overrightarrow{X_P}$ and $\overrightarrow{X}$ are represents position vector of prey and grey wolf, respectively. Iter is the current iteration. The $\overrightarrow{A}$ and $\overrightarrow{C}$ are coefficient vectors, they are calculated as,

$$\overrightarrow{A} = 2\overrightarrow{a}\,\overrightarrow{r_1} - \overrightarrow{a} \tag{3.12}$$

$$\overrightarrow{C} = 2\overrightarrow{r_2} \tag{3.13}$$

$\overrightarrow{r_1}$ and $\overrightarrow{r_2}$ has the range [0,1] for random vectors. The number of components of a decreases linearly from 2 to 0. Mathematically, the hunt for grey wolves is modelled as:

$$\overrightarrow{D_\alpha} = |\overrightarrow{C_1}\overrightarrow{X_\alpha} - \overrightarrow{X}|\overrightarrow{D_\beta} = |\overrightarrow{C_2}\overrightarrow{X_\beta} - \overrightarrow{X}|\overrightarrow{D_\delta} = |\overrightarrow{C_3}\overrightarrow{X_\delta} - \overrightarrow{X}| \tag{3.14}$$

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1}(\overrightarrow{D_\alpha})\overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2}(\overrightarrow{D_\beta})\overrightarrow{X_3} = |\overrightarrow{X_\delta} - \overrightarrow{A_3}(\overrightarrow{D_\delta})| \tag{3.15}$$

$$\overrightarrow{X}(iter+1) = (\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3})/3 \tag{3.16}$$

At the beginning of the search, no previous information of the ideal parameters or their position in the search region is available. A better understanding of where the prey is located is expected to come from the first three best options ($\alpha\ \beta\ \delta$) e.g.

$\alpha\ \beta$ and δ make educated guesses about the victim's location and keep track of its distance during the iteration process. Following this, all other agents have to adjust their positions accordingly [18].

4 Conclusions

Our literature analysis led us to the conclusion that PID controllers are the most often used due to their easy structure and implementation. Automated tuning features have made PID control much easier to utilize. Numerous methods for improving PID control have been researched. A complete analysis of all the approaches that were concurrently tried under varied settings was carried out. In addition, because of its ability to enable automated tuning, PID has drawn increased interest from industrial uses. PID controller tuning would be a significant research topic.

References

[1] A. G. Hussien, M. Amin, M. Wang, G. Liang, A. Alsanad, A. Gumaei and H. Chen, *Crow search algorithm: theory, recent advances, and applications*, IEEE Access **8** (2020), 173548–173565.

[2] A. Askarzadeh, *A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm*, Comput. Struct. **169** (2016), 1–12.

[3] H.O. Bansal, R. Sharma and P.R. Shreeraman, *PID controller tuning techniques: a review*, J. Control Eng. Technol. **2** (2012), 168–176.

[4] B.W. Bequette, *Process control: modeling, design, and simulation*, Prentice Hall Professional, 2003.

[5] Z. Bingul and O. Karahan, *Comparison of PID and FOPID controllers tuned by PSO and ABC algorithms for unstable and integrating systems with time delay*, Optim. Control Appl. Meth. **39** (2018), no. 4, 1431–1450.

[6] R.P. Borase, D.K. Maghade, S.Y. Sondkar and S.N. Pawar, *A review of PID control, tuning methods and applications*, Int. J. Dyn. Control **9** (2020), 818–827.

[7] M.E. El-Telbany, *Tuning PID controller for DC motor: an artificial bees optimization approach*, Int. J. Comput. Appl. **77** (2013), no. 15, 18–21.

[8] T. George and V. Ganesan, *Optimal tuning of PID controller in time delay system: a review on various optimization techniques*, Chem. Prod. Process Model. **17** (2022), no. 1, 1–28.

[9] J. Han, P. Wang and X. Yang, *Tuning of PID controller based on fruit fly optimization algorithm*, IEEE Int. Conf. Mechatronics Autom. ICMA, 2012, pp. 409–413.

[10] A. Kandel, R. Pacheco, A. Martins and S. Khator, *The foundations of rule-based computations in fuzzy models*, W. Pedrycz, (eds), Fuzzy Model. Int. Ser. Intell. Technol. Springer, Boston, MA **7** (1996), 231–263.

[11] J.S. Kim, J.H. Kim, J. Park, S. Park, W. Choe and H. Heo, *Auto tuning PID controller based on improved genetic algorithm for reverse osmosis plant*, World Acad. Sci. Eng. Technol. **47** (2008), no. 2, 532–537.

[12] S.M.G. Kumar, R. Jain, N. Anantharaman, V. Dharmalingam and K.M.M.S. Begum, *Genetic algorithm based PID controller tuning for a model bioreactor*, Indian Chem. Eng. **50** (2008), no. 3, 214–226.

[13] Microstar Laboratories, Inc. *Ziegler-Nichols tuning rules for PID*, https://www.mstarlabs.com/control/znrule.html, 2023.

[14] S.K. Mosavi, E. Jalalian and F.S. Gharahchopog, *A comprehensive survey of grey wolf optimizer algorithm and its application*, Int. J. Adv. Robot. Expert Syst. **1** (2023), no. 6, 23–45.

[15] A. Myrtellari, P. Marango and M. Gjonaj, *Analysis and performance of linear quadratic regulator and PSO algorithm in optimal control of DC motor*, Int. J. Latest Res. Eng. Technol. **2** (2016), no. 4.

[16] C.A. Neto and M. Embiruçu, *Tuning of PID controllers: an optimization-based method*, IFAC Proc. **33** (2000), no. 4, 367–372.

[17] S.S. Rao, *Engineering optimization: theory and practice*, John Wiley & Sons, 2019.

[18] M.A. Sen and M. Kalyoncu, *Grey wolf optimizer based tuning of a hybrid LQR-PID controller for foot trajectory control of a quadruped robot*, Gazi Univ. J. Sci. **32** (2019), no. 2, 674–684.

[19] P. Srinivas, K. Lakshmi and V. Kumar, *A comparison of PID controller tuning methods for three tank level process*, Int. J. Adv. Res. Electr. Electron. Instrum. Eng. **3** (2014), no. 1, 6810–6820.

[20] V. Srivastava and S. Srivastava, *A comprehensive review of optimization algorithms for nonlinear systems*, In: R. Srivastava and A.K.S. Pundir (eds.), New Front. Commun. Intell. Syst. (2021), 535–546.

[21] N. Tandan and K.K. Swarnkar, *PID controller optimization by soft computing techniques-a review*, Int. J. Hybrid Inf. Technol. **8** (2015), no. 7, 357–362.

[22] M.J. Willis, *Proportional-integral-derivative control*, Dept. Chem. Process Eng. Univ. Newcastle, 1999.

[23] Y. Zhang, L. Zhang and Z. Dong, *An MEA-tuning method for design of the PID controller*, Math. Probl. Eng. **2019** (2019).