T.C.

ALTINBAŞ UNIVERSITY

Electrical and Computer Engineering

ENHANCE AND IMPROVE DC MOTOR SYSTEM BY USING OPTIMIZATION ALGORITHMS

Hayder Madeeh Hussein Al-Zamili

Master Thesis

Supervisor

Asst. Prof. Dr. Sefer Kurnaz

Istanbul 2019

ENHANCE AND IMPROVE DC MOTOR SYSTEM BY USING OPTIMIZATION ALGORITHMS

by

Hayder Madeeh Hussein Al-Zamili

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Master of Science

ALTINBAŞ UNIVERSITY

2019

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Sefer Kurnaz

Supervisor

Examining Committee Members

Asst. Prof. Dr. Sefer Kurnaz	School of Engineering and Natural Sciences, Altinbas University	_____
Assoc. Prof. Dr. Yasa Ekşioğlu	School of Engineering and Natural Sciences, Altinbas University	_____
Asst. Prof. Dr. Zeynep Altan	Faculty of Engineering and Architecture, Beykent University	_____

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Çağatay Aydin

Head of Department

Approval Date of Graduate School of Science and Engineering: __/__/__

Prof. Dr. Oğuz Bayat

Director

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Hayder Madeeh Hussein Al-zamili

DEDICATION

In the beginning, I would like to thank God for this grace and I would like to dedicate this work to the spirit of my father's grave, who supported me with his spirit despite everything, to my teacher and my mother's mother, who was everything to me, to my brothers who supported me and to my friends who did not leave me alone.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to all the instructors that have taught me more than just science, especially my supervisor Asst. Prof. Dr. Sefer Kurnaz, for all the time, support and guidance provided to me along the journey to accomplish this work. Thank you all for all the knowledge and advice that made me overcome all the difficulties that I have faces.

ABSTRACT

ENHANCE AND IMPROVE DC MOTOR SYSTEM BY USING OPTIMIZATION ALGORITHMS

Alzamili, Hayder Madeeh Hussein,

M.S. Electrical and Computer Engineering, Altınbaş University,

Supervisor: Asst. Prof. Dr. Sefer Kurnaz

Date: Oct 2019

Page: 58

DC motor is widely used in the industry. Its stability and speed of maintenance are important to maintain engine performance, which is affected by pregnancy fluctuations and environmental factors. Therefore, it is important to maintain the required speed and time of system stability. We have developed a different structure and characteristics of the controllers. One of them is the proportional integral derivative (PID) controller, which is mainly used for industrial purposes. Because of its durability and chassis, the PID is mainly used to control the speed and state of a DC motor. Also, PID has become more common due to the ease of integration in terms of both hardware and system software. The system we created is a DC motor. In this study, proportional, integral and derivative proportional (PID) regulators are used to control the speed of a DC motor, and the PID parameters are determined as a result of open-loop interaction in the operating system. Later, the PID parameters are estimated using Optimization algorithms, such as particle order optimization (PSO) and the Bat algorithm (BA). As a result, the results of the three algorithms are compared based on three criteria. The main goal of management theory is the analysis and design of the console. The system was created using MATLAB SIMULINK. For its design, we used the basic tools and the mathematical model, as well as the PID by the scale factor (Kp = 17.936 and Ki = 43.454, Kd = -0.776). The highest yield (9.267%) and a linear rate (0.0051) were given. These results are not ideal for DC

motor operation. For best results, PID optimization algorithms were used. PSO and BA algorithms are used. A comparison was made between the properties of PSO and BA to improve the PID control used to control the speed of a DC motor. In the PSO algorithm (1.780%), the rise time (1.38 seconds), the error SteadyState (0.0001), the maximum skip algorithm in the bat algorithm (3.7707%), the rise time (1.36 seconds) The error SteadyState (0.002) PSO algorithm better than Bat algorithm, in improving the function of a DC motor.

Keywords: Proportional derivative controller, Particle Swarm Optimization algorithm, DC Motor System, Bat algorithm, MATLAB.

TABLE OF CONTENTS

LIST OF TABLES

Pages

LIST OF FIGURES

Pages

LIST OF ABBREVIATIONS

DC : Direct Current

PID : Proportional Integral Derivative

PSO : Particle Swarm Optimization

BA : Bat Algorithm

GSO :Glowworm Swarm Optimization Algorithm.

CSO :Cat Swarm Optimization Algorithm

LIST OF SYMBOLS

N	:	The number of partcles in the collection
D	:	The dimension
T	:	The pointer of iterations(generations)
$v_{i,m}^{(t)}$	:	Velocity of particle I at iteration t, $v_d^{min} \leq v_{i,d}^{(t)} \leq v_d^{max}$
w	:	Inertia weight factor
$c1, c2$	:	The hastening constant, Random number between 0 and 1
$x_{i,d}^{(t)}$	:	The actual status of the particle i in a reiteration
$Pbest$	:	Best previous position of the i-th particle
$gbest$	:	Best particle among all the particles in the population
f	:	Frequency
v	:	Speed

1. INTRODUCTION

DC motors are one of the most substantial devices on a lot of control systems likes homes, cars, trains and control systems. DC Motor has inner advantages like simple control, great electromagnetic torque, the ability to set the speed with the large range[1]. The mathematical model is so important for the design of the system. For a DC motor, some types to accurately appear for the attitude of the device. However, parameters are important also because the mathematical form cannot give the correct attitude without the right parameters at the form [2].

DC motor that using for critical speed and status in the industry. The stability of the system and the output results in that dominant to control the speed of the DC. stability of speed which influenced load change and environmental factors. then, It must be at a constant speed as well as must be maintained[3].

Using the Proportional Integral Derivative (PID) controller with optimization algorithm bat algorithm (BA) and also, we will use a particle swarm optimization algorithm (PSO) algorithm to enhance and improve the dc motor. These algorithms will give best results (Scale Factors) to be used in PID controller instead of using PID controller (Conventional Previous Scale Factors) or Default Parameters of PID controller and the system of dc motor will be implemented in matlab simulink platform.

1.1 PROBLEM DEFINITIONS

DC motor is greatly used for sensitive speed and manufacturing location. System stability and performance are important to control the speed of a DC motor. Stable speed influenced by load fluctuations and factors. It should that the required speed value is constantly maintained as a value. In my study, it is aimed at achieving the value of speed achieved as the desired value, and maintaining it constantly using the proportional, integral and derived control unit (PID) synthesis parameters. PID parameters are calculation using metaheuristic algorithms, such as the optimization of the order of particles (PSO) and the algorithm Bat (BA).

1.2 CONTRIBUTION OF THESIS

There are many ways to enhance and improve the dc motor, we will use a PID controller with optimization algorithms to do this, in this study i will use PSO algorithm and bat algorithm and we will look at which one will give better results, also to improve the angular speed and reduce the error percentage by comparing between the default scale factors of PID controller when connected with dc motor system in matlab (without algorithms) and using the parameters (scale factors) that comes from bat algorithm and PSO algorithm by test these values of scale factors (comes from PSO algorithm and bat algorithm,), on PID controller and this will improve the efficiency and also the overshoot and steady state error(Ess) will be a test to see what the algorithm will import in using PID controller.

1.3 THESIS ORGANIZATION

The thesis organization fall into six chapters, in the first chapter we will give introduction about the topic, in chapter two we demonstrate the literature review that was introduced by previous researches, in chapter three give an introduction to optimization algorthim, (PID) controller and DC Motor System, our experiments are displayed and discussed in chapter three and four that include design, simulate and test our experiments, in chapter five we will give an explanation for our results, finally in chapter six will demonstrate the conclusion and suggestion for future works.

2. LITERATURE REVIEW

In this chapter, we present a literature review for the previous related works of DC MOTOR system as the following:

In (2019) [3] The DC motor is greatly used for high speed and location in the industry. System constancy and application are important for controlling the DC motor. which depends on load fluctuations and environmental factors. The objective of this study is to ensure that the value of speed, which is attained like the desired value and maintained steady, using (PID) controller to set parameters. First, Ziegler-Nichols (ZN) is one of the traditional methods used. The parameters of PID are determined by a system response that is open under an operating system. Later, the PID parameters are estimated using metaheuristic algorithms, like PSO and the Genetic algorithm (GA). As a result, the results of the three algorithms are compared based on five criteria. The PSO algorithm gives better results than the genetic algorithm for each standard.

In (2015) [4] Discuss the progress of the FPID controller to control the DC motor speed. The vague PID algorithms (PIDs) and the traditional PID (CPID) algorithms are implemented using PID and Fuzzy Logic simulation tools in the Lab View environment. Results show that the self-adjusting PID is better than the traditional PID.

In (2016) [5] they discuss how to set and modify the PID (Ziegler-Nichols, els..). Also, they analyze the DC motor echo to speed control by using the parameters result from the already aforesaid settings. Moreover, the advantage and disadvantages of all of these ways. GUL / MATLAB windows are used to do both ways to create a restful and better environment for an understanding of the PID control mode.

In (2017)[6] They discussed the short design phase of a fractional system controller (FO-PID) on an indirect approach method using a stable phase technique. A new dynamic tuning technique has been proposed to find control parameters. Also, demonstrate how the FO-PID controller is executed by using a point-to-point digital signal processor. it is designed and installed with each the parts of the 1.5W DC industrial motor. The strong process of the border difference is verified by experiment the controller using two separate DC motors of the same value but using various parameters. they Explain how to design and develop a FO-PID basis that enhances the DC motor.DSP has been used as the platform to implement the FO-PID digital control module. The Inertia Performance Improvement Technology was used in the system. The results explain improved DC

response accuracy by deploying the FO-PID controller. Traditional PID. The current consumption capacity is controlled by the control unit itself. This increases the potential of the FO-PID controller with a wide range of applications such as control, can enhance the industrial use of FO-PID controllers.

In (2017)[7] A methodology has been proposed to evaluate the quantitative durability of the PID controllers used in the DC motor. Durability analysis is performed using the first 23 FOPID designs, the correct order (IOPID) controller and the built-in Skogestad model (SIMC) controller. Experimental design The application uses FOPID and IOPID controllers to produce hardness analysis, static scale display and robust performance verification results compared to experimental factors. The comparison of the experimental design was multiplied by the FOPID, SIMC PID and IOPID controllers used to control the engine system. Factors used to measure survival were random noise in the feedback loop, external disturbances in system income, and uncertainty during pregnancy. Resistance examination was performed to examine the scale of formation of factors in the response system step and control procedures. This infection level was measured using RMSE, SD OUT, RMS CONTROL, and SD CONTROL output variables. The results of the statistical stability analysis showed that the presence of C (disturbance) with FOPID as the lowest frequency further affected the response step. This indicates that FOPID is more powerful against external interference. The controls are also affected by factors A (noise), B (load) and C (disturbance) for all controllers. FOPID provides the highest level of injury due to factors A (noise) and C (interference), but at least depends on factor B (load). This means that the FOPID control procedure is more powerful under load uncertainty conditions. Based on the reliability analysis, we can conclude that FOPID is an alternative to the management of undefined load and external interference systems affected by the low control voltage, although it is not a power control method. Finally, the test design methodology used this document to analyze the reliability of the control system and to measure and obtain numerical evidence of the actual performance of the controllers, taking into account the presence of various factors affecting the behavior of the control system.

In (2018) [8] DC motors are commonly used in the industry for various advantages as high efficiency, low cost and flexibility. Rotation speed of DC motor is controlled using conventional large-scale PI and PID controllers. However, taking into account the experimentally determined parameters Kp, Ki, Kd and the limits of PID agreements to achieve the ideal control effect for the upper level systems, a proportional-integral-

differential PID (FOPID) scheme was proposed based on optimization methods. The goal of this paper is to study the tuning of the FOPID control unit using intelligent soft computing technologies, such as differential evolution (DE) and improved particle swarm design (PSO) for the PID breakdown controller. The FOPID parameters are determined by reducing the absolute time error (ITAE) between the output of the reference model and the factory. The performance of the DE and PSO was compared with many simulations. Simulation results show that the DE-based FOPID tuning method provides improved performance for tracking tuning points, reducing errors and reducing measurement noise.

In (2017) [9] The controller implements a particle speed booster (PSO) for controlling the DC motor speed. By using MATLAB / SIMULINK Also, traditional relational units (P), proportional-integrative (PI), have been developed and modeled. The simulation results show that the PI controller has a large emission and sensitivity to gain the console. Optimization by using a swarm of particles confirmed the minimization of oscillations, as well as improved stable state error and uptime and increased response speed. Simulation results confirmed that the system is less sensitive to achieve. In this article, the P and PI controllers were designed and the parameters of the speed control module were improved using the Particle Swallow Optimization (PSO) method. The proposed PSO has a good contrast in accuracy and speed compared to P and PI based speed controllers, according to the estimated results. In addition, the designing of the PID control using PSO is due to the fact that the rotation rate, delay time and stability time of the step response curve are reduced compared to P and PI. PSO performance can be improved in the design and optimization process by increasing the number of duplicates.

In (2017) [10] Demonstrates a gentic algorithm for controlling the DC motor speed and is equipped with a PID regulator and a technique for improving the particle swarm. The use of particle swarm optimization to set the parameters of the PID controller, the DC motor and control based on the FID gene algorithm gives better results. Here, the DC motor speed control model is the second-generation system. The gentic algorithm is a easy calculation method used to improve for the better possible result. The gentic algorithm works on some major genetic selection factors, crossover, mutation. This article describes in detail the efficiency of DC motor parameters for the best result, using both improvements. The suggest approach to improves functions, having ease of application and good computational efficiency. PID control parameters for better output of high-quality solutions faster.

In (2017) [11] The DC motor speed control mode was discussed without knowing the specific engine parameters. An estimated mathematical for controlling system is obtained by major the system while measuring the system output by loading a specific input signal. The PID control algorithm is adopted, and parameters P, I and D are obtained by automatic tuning. HIL tests are conducted on the MATLAB and Arduino platforms. The system selector is used to determine the mathematical model of control system. The simulink form of the controller is set with Notes for the measuring system, and the parameters are set automatically from the PID console. Finally, tests are carried out on the Arduino Mega 2560 plate. For comparison, the method used in this article demonstrated that it is very convenient and efficient to develop the control system. The control system has the best effective performance and accuracy. During the experiments, we also found that with increased accuracy the speed would deteriorate. Given that the sensor signal collected is not smooth enough. A low pass filter can be considered for treatment in a prospective study.

3. OPTIMIZATION ALGORITHMS

3.1 MOTIVATION

The DC motor is one of the most important engines that are included in all industrial products or products consumed. Therefore, the control of the speed of the current that must be controlled and increased with the reduction of error. Therefore, I will use a Proportional, Integral and Derivative (PID) controller because it will control the speed for dc motor and in same way I will use the Optimization Algorithms with the Proportional, Integral and Derivative (PID) controller to reduce error rate and increase the speed at the angle in DC motor, I will use tow type from Optimization Algorithms first I will use particle swarm optimization algorithm (PSO) and second I will use bat algorithm (BA)because they will give values for (Scale Factors) better than (Scale Factors) normal that the PID give to operate the DC motor at its best.

3.2 PROPORTIONAL INTEGRAL DERIVATIVE (PID) CONTROLLER

Proportional, integral and derivative controller gives the simplest while the effective solution for any management problems. Since the invention of the PID controller in 1910. PID control has expanded significantly. However, 90% of the controllers are yet applied based on the PID algorithms, especially at lower levels, because other controllers do not correspond to the simplicity, functionality, ease of use provided by the PID[12]. PID is the feedback method for a control loop (console) That are used in many control systems. PID controller finds the error value like the difference among the required setpoint and the standard run variable. The console tries for reducing the error by setting the process by using a handler changeable. The PID algorithm includes three separate fixed parameters, occasionally called three-term control: relative, integral and the derivative, indicating P, I, and D. By these values can be simply explained in terms of time: firstly P depends on the current error, also Past errors, and the symbol D - Diagnosis of future errors, depend on current average change. The important step in implementing a PID console is setting a parameter. The PID calculates three parameters: relative gain (KP), integration (KI) also derivatives (KD).

The relative rate calculates in the actual error, determines the combined amount as the result of the amount of the last errors, determines the derived response value based on the

error rate. The performance of a closed-loop at every system can be accomplished by setting the parameters of the PID [13]. Can be represented in the following equation (3,1):

$$G(s) = kp + \frac{ki}{s} + kd^{s} \qquad (3,1)$$

The gain control of this module is PID control, the controller can supply a control procedure designed for given process requirements.

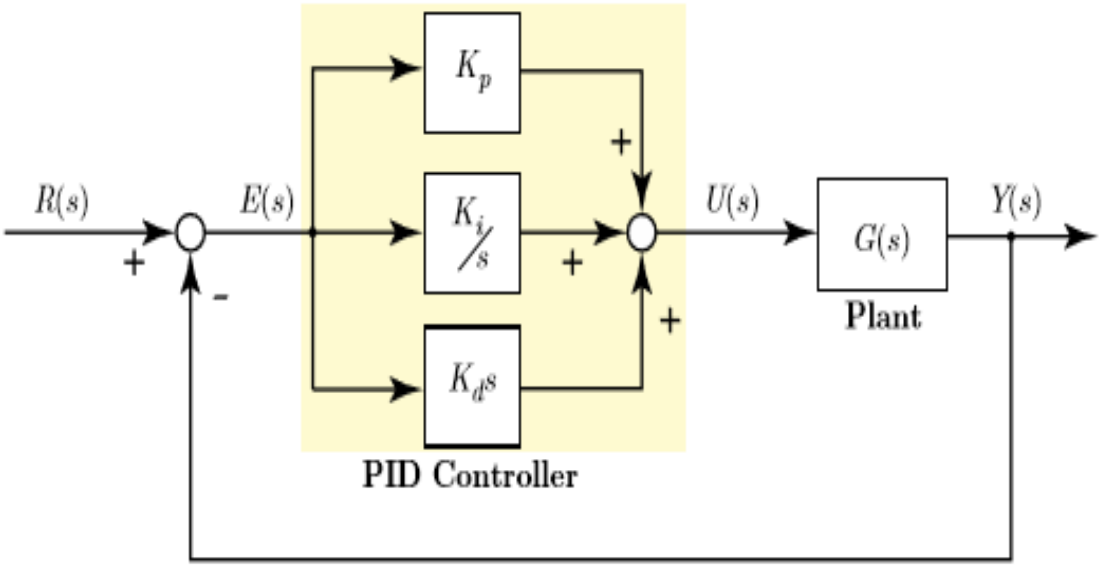


Figure 3.1: PID Controlled System[14].

3.2.1 Methods for Tuning a PID Loop

There are many ways to tweak the PID controller. The most active methods usually include the development of the process model form, the choice of P- I- D are based on the parameters of the dynamic model specific. Manual adjustment methods may be comparatively inefficient, especially if the rings have adjustment time in a minute order.

The choice of method usually depends on whether the “Stand-alone” cycle can be used for tuning and when the system is installed. If system can be used "offline", it often includes the best way to configure the system to change the point of input and attempt to output depending on time and use this restriction to specify control parameters.

Table 3.1: Choose a Setting Method.

Method	Advantages	Disadvantages
Manual Tuning	No math required , Online	Requires experienced personnel
Ziegler-Nichols	Proven Method, Online	Process upset, some trial-and-error, very aggressive tuning
Cohen-Coon	Good process models	Some math; offline; only good for first-order processes
Software Tools	Consistent tuning; online or offline - can employ computer-automated control system design (CAutoD) techniques;	Some cost or training involved

3.2.2 Ziegler-Nichols (Z-N) Method

In 1942, two ways for setting the parameters of PID controller was developed by Ziegler and Nichols8, the first method is Z-N open-loop and the second method is Z-N closed-loop, and show plain mathematical step to setting PID controllers. Often the operation response curve method is called Ziegler-Nichols, which is not closed in the ring setting method. The curve method operation response is a way of The operation parameters like retard time, an operation gain and time stable to the controller parameters[13]. ZieglerNichols is a one of classic method which defines as the first time retard order system. that is consists of two parameters first of lead time (L) and second time constant (T)[15]. Formulation of the method is given in equation (3.2) [3].

$$G_p(S) = \frac{Ke^{-sl}}{TS+1} \tag{3.2}$$

open loop transfer function Gp(s) Being controlled for the system is to explain how specific the way of an open loop response.

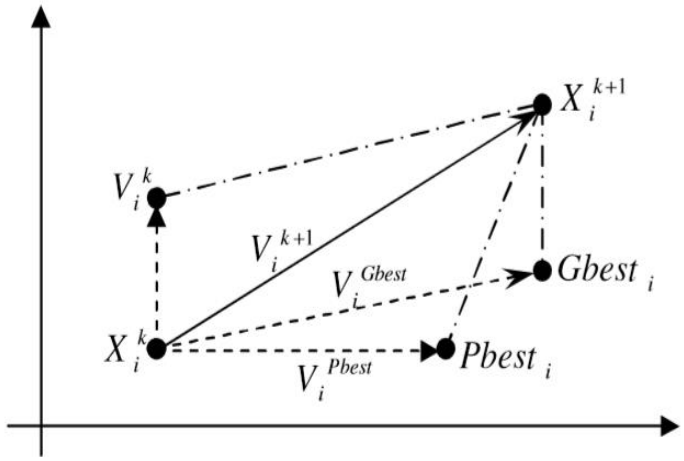
3.3 PARTICLE SWARM OPTIMIZATION ALGORITHM

The particle swarm optimization algorithm (PSO) is an advanced order-based method proposed by Eberhardt and Kennedy (1995) [16]. The (PSO) algorithm mimics the social behavior of an animal, inclusive insects, birds, herds, and fishes. And these swarms correspond to a joint feeding method, and each member of the swarm continues to change the search pattern in accordance with its learning experience and other Individuals. The major goal of the PSO algorithm project is nicely concerning two types of research: the first is an evolutionary algorithm, As an evolutionary algorithm, the PSO as well uses a number of modes that search for a large area of the optimal target function area [16]. Repeat the steps that start with a possible solution to get the best solutions. This means that possible solutions are molecules and move these particles in the search area based on the formula above the particle position. The motion of each particle is influenced by its local value, and its goal is to reach the most well-known positions in the search area, simultaneously updating the best positions found by other particles. The algorithm began with the initial points and the velocity of the vector. Try each iteration to find the best value, estimating the position and velocity vectors. Each particle has its own options. If the task consists of five different variables, you must choose a particle size as five. The best value for any particle is called the best local value and is written into the Pbest matrix. The best value for any particle is updated after each repetition if the best new value is found to control both the current particles and the previous locations. In addition, the position is recorded as the best (Gbest) after controlling the best matrix values at each frequency [3].

Millonas, who studied the behavior of social animals using old life theory, proposed basic principles to create practical life systems filled with collaborative behaviors using computers[16]:

1) closeness: it should be able to make simple time and space calculations.
2) Quality: it should be able to sense and respond to environmental quality changes.
3) Various response: it should not suspend access to resources on a small scale.
4) Stability: it should not change functioning with every change in the environment.
5) Adaptability: When this change is worth it, it should change the swarm behavior.

In PSO system, particles change their position, bypassing the multi-side looking space, until arithmetic restrictions are overtake. The concept of modifying the search point using (PSO) Fig. **3.2**.



x^k (the current position), x^{k+1} (the modified position), v^k (the current velocity), v^{k+1} (the modified velocity), v^{pbest} (the velocity based on $Pbest$), v^{Gbest} (the velocity based on $Gbest$).

Figure 3.2: Concept Point Search By PSO [17].

The PSO algorithm differs from another evolutionary algorithm, like genetic algorithms. Although the inhabitance is used to search for a field of study, thither are no factors inspired via the human DNA course of action applied to this population. in the PSO, inhabitance dynamics mimic the manner of the "swarm of birds", where public information is exchanged, and people can use previous discoveries and experiences of all other satellites in search of food. Every "companion", called a particle, in a society called a swarm, must "fly" through the looking area to discovery promising areas of the landcsape. For example, if they are minimized, these regions have lower functionality values than those previously visited. In this case, each particle is considered as a point in the field of measurement, that organizes its "journey" following its atmospheric experience, a particle is defined as a point transferring in space. all snippet in the actual time step tracks the position, speed in the search field. The presumption is the main concept of PSO [9]. And not using evolutionary factors, like alteration and intersection in (PSO) algorithm, for solving algorithms to improve the variable d, a cluster of particles is placed in the looking space with dimensions d at indiscriminately chosen speeds and locations to see the best values to date (Pbest).The speed of each particle is regulated depending on its flight experiment and the flight experiment of other particles.

For illustration, the particle is appeared for like xi = (xi, 1, xi, 2, ..., xi, d) in the different spaced. The better foregoing mode of the i-th particle is registered and perform as follows:

$$Pbesti = (Pbesti, 1, Pbesti, 2, \ldots, Pbest\ i, d)$$

Particles i represent the velocity vi = (vi, 1, vi, 2, ..., vi, d). The speed location of all particle that can be measured using the existing speed and distance from Pbesti, d to gbest, like in the following formulas (3.3) (3.4):

$$v_{i,m}^{(t+1)} = w \,.\, v_{i,m}^{(t)} + c_1 * rand() * \left(Pbest_{i,m} - x_{i,m}^{(t)}\right) + c_2 * rand() * \left(gbest_m - x_{i,m}^{(t)}\right) \quad (3.3)$$

$$x_{i,m}^{(t+1)} = x_{i,m}^{(t+1)} + v_{i,m}^{(t+1)} \quad ; \quad i = 1,2,\dots,n \quad ; \quad m = 1,2,\dots,d \quad (3.4)$$

wherever:

N: (The number of partcles in the collection).

D: (The dimension).

T: The pointer of iterations(generations).

$v_{i,m}^{(t)}$= (Velocity of particle I at iteration t, $v_d^{min} \leq v_{i,d}^{(t)} \leq v_d^{max}$).

w: (Inertia weight factor).

$c1, c2$: (The hastening constant, Random number between 0 and 1).

$x_{i,d}^{(t)}$: (The actual status of the particle i in a reiteration).

$Pbest$i: (Best previous position of the i-th particle).

$gbest$: (Best particle among all the particles in the population).

3.3.1 Social Behaviour of PSO

Several explanations have been proposed for the launch of PSO. Kennedy builds up the social and psychological perspective by conducting experiments to explore the functions of the different components in the rate-renewal equation [18]. Task Neural network training was used to correctly classify the XOR problem for comparing the performance of various models. Kennedy benefited from the lbest model, not the gbest model described above.

Consider the speed of updating the formula, repeat for convenience (3.5).

$$Vi,j\ (t\ +\ 1)\ =\ vi,j\ (t)\ +\ c1r1,j\ (t)[yi,j\ (t)\ -\ xi,j\ (t)]\ +\ c2r2,j\ (t)[\hat{}yj\ (t)\ -\ xi,j\ (t)] \tag{3.5}$$

The term $c1r1,j\ (t)[yi,j\ (t)\ -\ xi,j\ (t)]$ It refers to perception, because it takes into account only experiments with solid particles. If a PSO is created using only a cognitive term, the rate update equation becomes (3.6).

$$vi,j\ (t\ +\ 1)=\ vi,j\ (t)+\ c1r1,j\ (t)[yi,j\ (t)-\ xi,j\ (t)] \tag{3.6}$$

Kennedy found that the implementation of the "only for knowledge" model was lower than that of the main squadron, and he was unable to train the network in the maximum allowable frequency of some component settings. One of these reasons for abnormal PSO behavior is the lack of interaction between various particles.

The third expression in the velocity update equation, $c2r2,j\ (t)[\hat{}yj\ (t)\ -\ xi,j\ (t)]$, Represents social interaction among particles. A "social only" text of PSO can be created using the following equation for the update rate (3.7).

$$vi,j\ (t\ +\ 1)=\ vi,j\ (t)+\ c2r2,j\ (t)[\hat{}yi,j\ (t)-\ xi,j\ (t)] \tag{3.7}$$

The performance of this model exceeded the original PSO performance on a specific issue discussed by Kennedy.

In short, the term "PSO speed" content of a cognitive component and a public component. Much is not known about the proportional importance of these terms, although preliminary results show that the social component can be more important in some problems based on this, that PSO algorithm may be abstract like follows: This algorithm is a type of study using the spherical principle., where everyone is called a particle name as a possible resolve to an improved issue in a multidimensional looking space. This is just like speed. for all generation, the particle information is joint to adjust the speed of

any measurement, that used to calculate the new status of the particle. Molecules are constantly changing their positions in the multidimensional looking space to reach the perfect stability or state or exceed arithmetic border. An unrivaled connection between the different dimensions of the issue area is provided by using impartial functions. Several experimental data that shown this algorithm is an efficacious optimization implement [16]. **Fig 3.3** PSO flowchart. The following is a comparatively complete representation of the PSO algorithm. in this system of continuous coordinates of space can be mathematically described PSO.

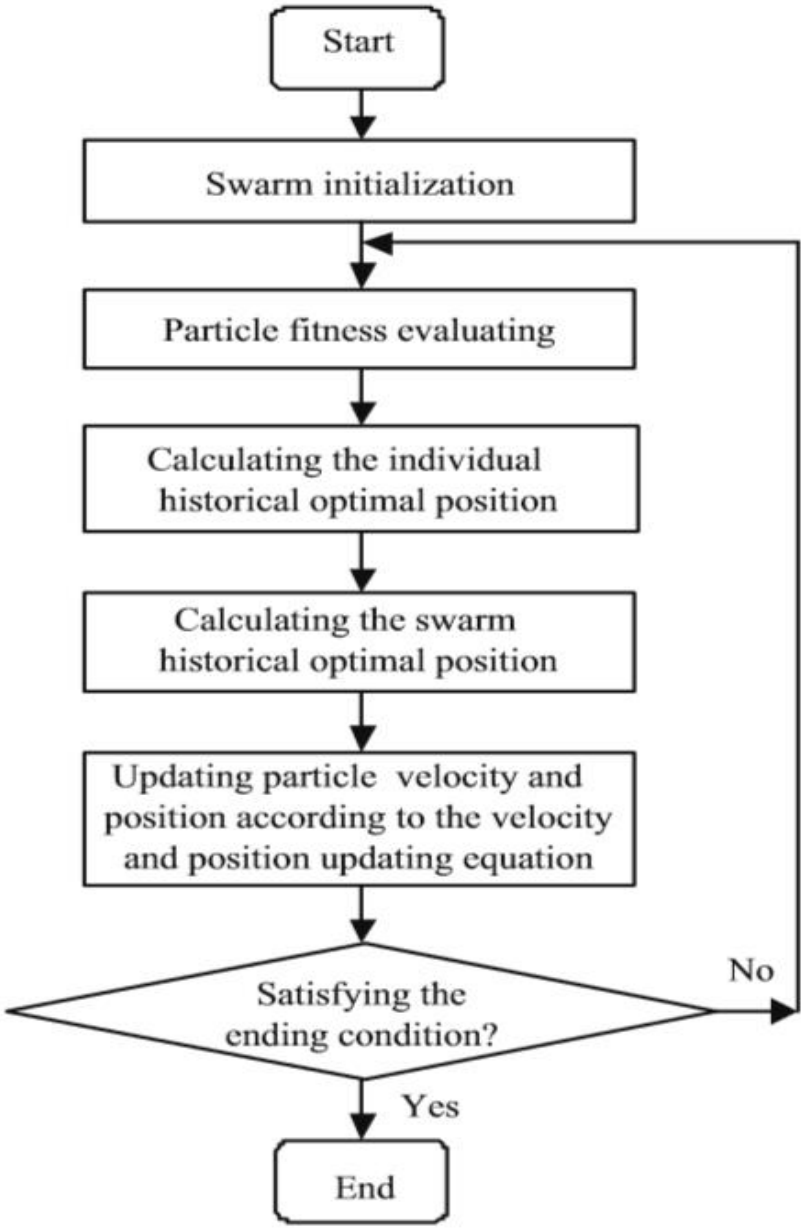


Figure 3.3: Flowchart of The PSO Algorithm [16].

3.3.2 Taxonomic Designation

The PSO is clearly related to some evolutionary algorithms. On the one hand, PSO supports a group of people who represent potential solutions, a function shared by all existing agencies. If the best personal relationships (Yi) are considered as part of the population, It becomes clear that there is a weak form of options [19]. In the ES $(\mu + \lambda)$ algorithm, Children compete with their parents and replace them if they are more appropriate.

$$yi(t + 1) = yi(t) if\ f\big(xi(t + 1)\big) \geq f\big(yi(t)\big),\ xi(t + 1) if\ f\big(xi(t + 1)\big) < f\big(yi(t)\big) \quad (3.8)$$

Equation (3.8) is similar to this mechanism, with the difference that each of the best characters (the father) can be replaced by their current (decreasing) position if the current position is more appropriate than the previous best personal position. In summary, we can say that there is a weak form of choice in the PSO.

The update rate equation is similar to the drive in GA real value. As a rule, the crossover produces two neural networks, which are combined in writing between the interested parents. The PSO refresh rate equation, without the term vi, j (t), can be interpreted as a form of a mathematical section, including religion, with a single line. Alternatively, the velocity equation, without the term vi, j (t), can be considered as a catalyst for transformation, with the force of the mutation determined by the distance that the particle has from the origin. This still leaves the term vi, j (t) lost, which can be interpreted as a form of mutation that depends on a person's position in the previous iteration. The best way to model vi, j (t) is to think of each iteration not as a process of replacing the previous population with a new group (death and birth), but as an adaptation. Therefore, the values of xi are not replaced, adapted using the speed vi. This makes the difference between other EAs and PSOs more evident: PSO stores information on position and speed (position changes), unlike traditional EAs that only follow situations. There seems to be some overlap between PSO and most other active agencies, but PSO has some features that are not currently available in other regions and operating agencies, in particular, the fact that PSO models particle speed and position models.

3.3.3 Origins and Terminology

The motion of the particles has been described as a "flight" over the n-dimensional space [20]. This term is partly related to experiments conducted to model a decomposing bird that produces to the development of the main PSO algorithm. [21]. A study of an article prepared by Reynolds [22]. (cited in the original PSO article by Kennedy and Eberhardt) reveals some interesting ideas. Reynolds was primarily interested in modeling bird flights for visual computer simulations, noting that the pack appears to be under central control. Obviously, the natural herd (or school) does not depend on the number of people involved in the formation from the observation of shoals of fish up to 17 miles long. Since birds (or fish) must have a limited "computing power", we can expect a significant extension of the size that a normal herd can reach if a person tracks all members of the herd. Since there is no upper limit in nature, it should be concluded that birds (or fish) deal only with a limited number of neighbors, which means local control rather than global control. Several reasons have been observed for mild behavior in nature.

Some of the developmental benefits include protection from predators, improved genetic survival and the use of more efficient research to produce food. This latter property is invaluable when food is distributed unevenly over a large area.

Reynolds began designing his flocks using three simple rules: avoiding the collision, choosing speed and concentrating the pack. Keep in mind that the pursuit of the pack will force the birds to fly close to their neighbors (be careful not to disturb the speed encounter), but at the same time keep a safe distance, controlled by the collision avoidance rule. Reynolds decided to use the engine of the calculated herd, taking into consideration only the nearest neighbors of the bird, instead of using the central center in the whole team, which he called " model of central authority ". This roughly corresponds to the best PSO model (shown below). It is interesting to note the observation of Reynolds [22]:

"Before implementing the current use of herd concentration behavior, the shepherds used a model of central authority. This leads to unusual effects, such as the wide distribution of all members of the herd at the same time towards the center point of the herd. This very accurately describes what happens in the PSO gbest model".

The flowing novels of Reynolds (a word derived from the oil of birds, meaning an organism resembling an ordinary bird), It was a common example of some principles of artificial life. Flow dynamics was a convincing example of emergent behavior:

Complex global behavior deriving from the interaction of simple rules. This is one of the features that make PSO a successful optimization algorithm: a simple application that leads to complex (and effective) search behaviors. Although the movement of the molecules appears to be optically regular, they do not fully correspond to certain definitions of regular behavior. Matari'c refers to the following concepts [23]:

- Safe passage: the capacity of a group of customers to transfer around, keep away from collisions with hurdles and between them.
- Dispersion: the capacity of a group of representatives to take sides to create and keep a minimum distance between representatives.
- Build: the capacity of a group of representatives to meet to create and keep the maximum distance between representatives.
- Rocket: possibility of finding a specific area or location.

Based on these definitions, Matharick argues that the current behavior consists of a controlled wave, which walks safely, disperses and aggregates. The PSO causes only controlled waves and assemblies and lacks safe displacement and dispersion. A safe tour is not so important. PSO, because it applies only to objects that can physically collide. Dispersion means that the particles will disappear when they approach each other, which is currently not the case with the PSO model: the PSO stimulates the molecules in the mass. The terms "swarm" and "swarm" are more appropriate. Millions of people have used this term to describe artificial life models[24]. Millions suggested that the reconnaissance squadron characterized the following characteristics:

- Proximity: make simple calculations of time and space.
- Quality: respond to environmental quality element.
- Different answers: do not fall into a limited set of solutions.
- Stability: the ability to maintain behavior ornament when the environment changes.
- Adaptability: the ability to change behaviors when considered profitable.

Eberhard et al. [20] It has been said that the molecules in PSO have these properties.

Finally, the term "serious" requires justification. Members of the population do not have enough mass and size, so they are called "points" they will be more precise. However, the concepts of speed and acceleration are more compatible with the term particle (referring to a small fraction of the matter) than the term. Other research areas, in particular,

computer graphics, use the term "particle systems" to describe the models used to provide effects such as smoke or fire.[25].

3.3.4 The Advantages of PSO

- Insensitive to extending the range of design variables.
- Simple implementation.
- Easy parallel parallel processing.
- Free derivative.
- Very few algorithm parameters.
- The global search algorithm is very effective [26].

3.3.5 The Disadvantages of PSO

• Slow convergence in the refined research phase (twice the local research capacity).

• The main disadvantage of PSO, as in other methods of learning optimization, is that it lacks some. What is a solid mathematical basis for analysis, which must be overcome in the future development of relevant theories. They can also have. Some real-time and other ED restrictions[27].

3.3.6 PSO Applications

- Neural network training.
- Improved power distribution network.
- Structural improvement.
- Biochemical process.
- System identification in biomechanics.
- Image processing.
- Analysis of human shaking to identify Parkinson's disease.
- Cancer classification.
- Examples of hominid biomechanics.
- Prediction of survival and survival.

3.4 THE BAT ALGORITHM (BA)

It proposed by (Xin-She Yang)[28]. The work of BA is very alike to the normal action of bats. by using the repeat to find all the goal. Similarly, the BA has taken the status of the microbe echo to discovery the best global solution. The BA haves three basics: The first is to assess the optimal distance for goal using the environmental situation. Secondly, the population moves to the search area at a constant speed and frequency. However, the wavelength and lightness of the sound may vary depending on the space among the goal and the status of the entire bat. The third basic, followed by the BA, decreases linearly in the actions of the bat growth factor[29]. The ability of this eco for small bats is excellent because the bats can discovery their victim and distinguish various types of goals flat in complete darkness [28]. The standard search algorithm is based on a bat echolocation process. By the notice the action and characteristics of small bats, Yang suggest obtaining a standard bachelor's degree depending on the three major characteristics of the echoing status for small bats. Ideal rules used in BA [28]:

A. each bat uses angular positioning to setting the space and the eleventh position of the bat is coded as a resolve to the improvement problem being studied.
B. The flight of the bat occurs randomly at point v in the case of xi with variable frequency (from the minimum fine to the maximum fmax) or with different wavelengths UD and A for looking the victim. and they can automatically set the wavelengths (or frequencies) of their released pulses and the pulse rate r based on the proximity to the target.
C. The sound value is different from a high positive value of A0 to a lower fixed amount. Some of the echo positioning functions are mainly identified in the optimization problem, that can be related to the main function, through which it may formulate a strong and bright algorithm. It radiates out loud and echoes nearby objects. The proper motivation they use depends on the chasing strategy. Studies indicate that microbes give sound waves at a frequency between 20 and 150 kHz. and also they have perfect landscape and listening skills. To calculate the position and size of the victim, they depend on the repetition of the echo that reaches their ears. The Bibliotheca Alexandrina has shown the characteristics of the bats captured during the search for their prey. The bat algorithm is formulated by initiall improving the echo positioning behavior in the bat, which includes the microbial behavior and acoustic properties of the echo position [30].

The following ideal rules usually follow:

(A) The bats use angular detection to determine area and find the variance among goal barriers and back baffles.

(B) Bats fly randomly at point vi in eleventh place with a constant frequency at wavelength k and $A0$ in search of prey. And they can set the wavelength of the released pulses and regulate the radiation of the pulses rate automatically r 2 [0, 1] depending on the closeness to the prey.

(C) Since the sound speed in the air is typical for v = 340 m / s, the wavelength k of an The sound of the ultrasonic pulse with a fixed frequency is determined by the formula (3.9).

$$f = \frac{v}{m} \tag{3.9}$$

(D) The volume can be changed in several methods, can be supposed, but the tone different from A0 (positive value) to a fixed steady minimum Amin.

3.4.1 Bat Motion

All racquet is quickly related with v_i^t and x_i^t, in t-redundancy, in the search space or in the resolution of measurements [31]. Between all bats there is a better solution about? From her, The above three rules can be expressed as update equations for x_i^t and v_i^t (3.10)(3.11)(3.12):

$$f_{i=\,f_{min}+(f_{min-f_{max}})} * \beta \tag{3.10}$$

$$v_i^t=(x_i^t - x_*)* f_i \tag{3.11}$$

$$x_i^t=x_i^{t-1} + v_i^t \tag{3.12}$$

Where $\beta \in [0,1]$ The random carrier is taken from one distribution. Here is x_* It is the best universal solution (solution), which depends on comparing all solutions between all n bits. Since the $\lambda_i f_i$ of the product is an increased speed, it can be used as f_i (or λ_i) to set the speed variation, however another factor λ_i (or f_i) is chosen, relying on the kind of problem. The choice of f_{min} and f_{max} (f_{min} = 0 and f_{max} = 100) rely on the size of the problem area of attention. At first, a frequency is randomly allocated to each bat taken from [f_{min}, f_{max}]. For the native search sector, when a solution is chosen from the best current solutions, a local solution is created for all racket using a path random.

$$x_{new} = x_{old} + \varepsilon A^t \tag{3.13}$$

Eq (3.13) Where $\varepsilon \in [-1,1]$ Random number, and $A^t << A_i^t >$ the sound of each bat is averaged over this time step.

3.4.2 The Structure of Bat Algorithm

The false algorithm code is presented in the basic bat algorithm 1. The major parts of BA can be summarized as follows [32]:

- In the first step is to configure (lines 1-3). At this stage, we format the algorithm parameters, build and assess the initial population, then select the best xbest solution in the group.

Input: Bat population $\mathbf{x_i} = (x_{i1}, \ldots, x_{iD})^T$ for $i = 1 \ldots Np$, MAX_FE.
Output: The best solution $\mathbf{x}_{best}$ and its corresponding value $f_{min} = \min(f(\mathbf{x}))$.

```
1:  init_bat();
2:  eval = evaluate_the_new_population;
3:  f_min = find_the_best_solution(x_best); {initialization}
4:  while termination_condition_not_meet do
5:    for i = 1 to Np do
6:      y = generate_new_solution(x_i);
7:      if rand(0,1) > r_i then
8:        y = improve_the_best_solution(x_best)
9:      end if{ local search step }
10:     f_new = evaluate_the_new_solution(y);
11:     eval = eval + 1;
12:     if f_new ≤ f_i and N(0,1) < A_i then
13:       x_i = y; f_i = f_new;
14:     end if{ save the best solution conditionally }
15:     f_min=find_the_best_solution(x_best);
16:   end for
17: end while
```

Figure 3.4: Original Bat algorithm

- The second step is to create a new solution (line 6).

Here the default bat moves to the search area according to update bat algorithm rules.

- The local search step is the third step (lines 7-9). The best optimized solution is offered using random paths.
- The new solution is presented in this fourth step (line 10).
- At this point, the new solution is saved (lines 12-14), conditional archiving is performed for a better solution.

- In this last step, the best solution is sought (line 15). The current solution is the best update.

In **figure 3.4** showing the flowchart of bat algorithm.

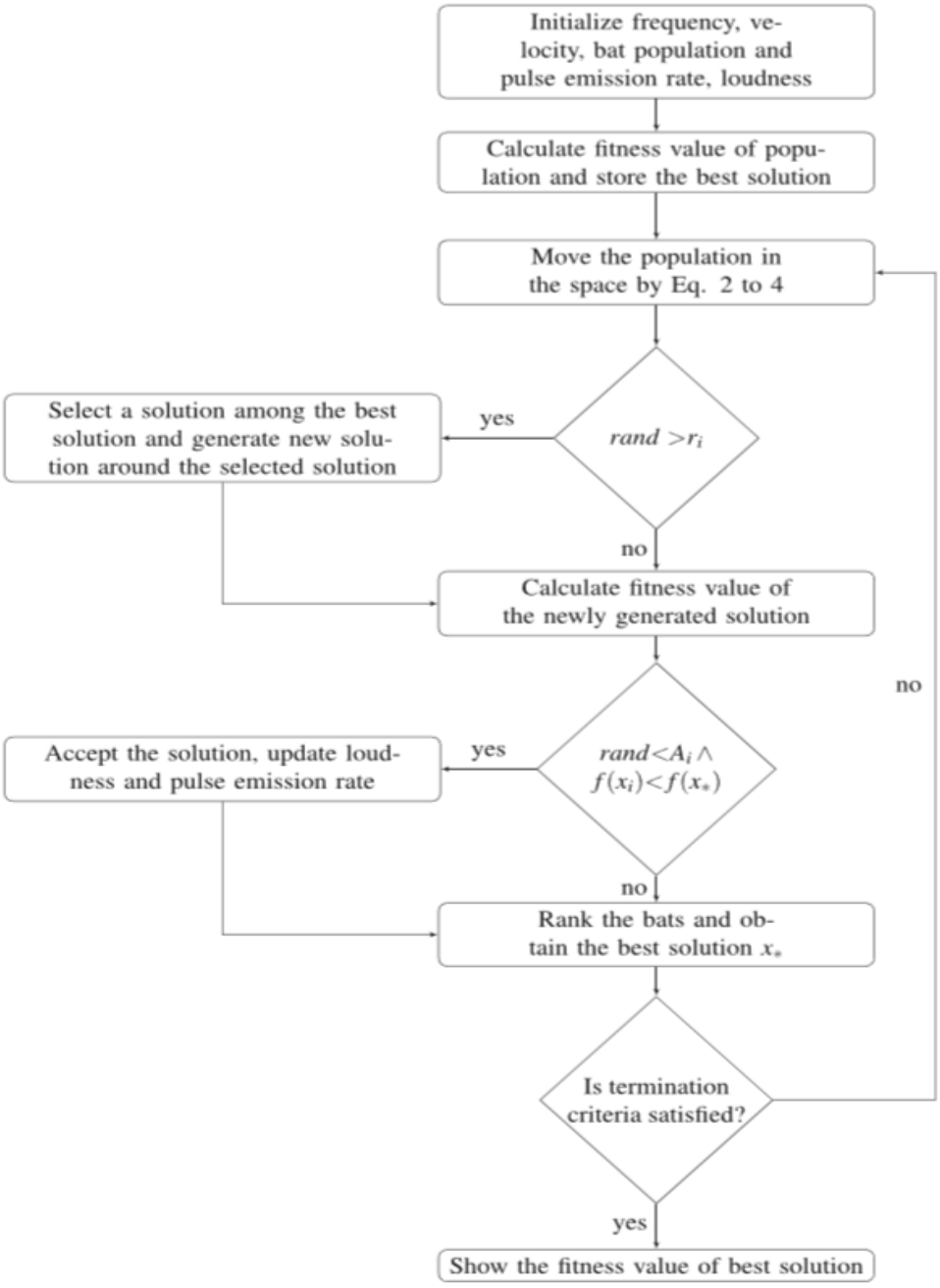


Figure 3.5: Flowchart of Bat Algorithm[33]**.**

3.4.3 Applications of Bat Algorithm

It has been used in almost every category of advance and ratings. Image processing, job selection planning, data mining, etc..[30]. Some application of bat algorithm[30][32][34]:

- Continuous Optimization.
- Combinatorial Optimization and programing.
- Inverse Problems and Parameter evaluation.
- Classification, Data Mining.
- Fuzzy Logic and Other Applications.
- Fuel arrangement optimization of reactor core.
- Feature selection.
- Application in multiple UCAVs.
- Multilevel image thresholding.
- Economic dispatch.
- Image Processing.
- Design of skeletal structures.
- Design of a conventional power system stabilizer.

3.5 DC MOTOR

3.5.1 The Electrical Part of Dc Motor

The idea of a DC motor that comes from reinforcing bars has resistance and mutual inductance, a related degree of electrical driving force (emf at the rear) that was created in the reinforcement file[35], and connected to a reinforcement bar to carry a mechanical load system an electric motor so that the input voltage to drive a DC voltage is the output of a DC motor is an angle (θ) or speed (ω) θ , $\omega m = \frac{d\theta}{dt} = \dot{\theta}$ this is due to the angle according to Because of the additional time that the motor deals with this section until Kirchhoff's law of voltage (KWL) is adopted to circulate around the ring from left to right, so that the input voltage and voltage of stability and stability Table D solenoid and reverse voltage back in the motor and, therefore, KVL will be calculated, because this formula $\sum Vi = 0$ collecting voltage in one loop = 0 Thus, the equation will be calculated, because this formula $Vi - VR - VL - eb = 0$ armature current within each element okru Stnost string and EMF deals with the torque of the engine Const ant speed.

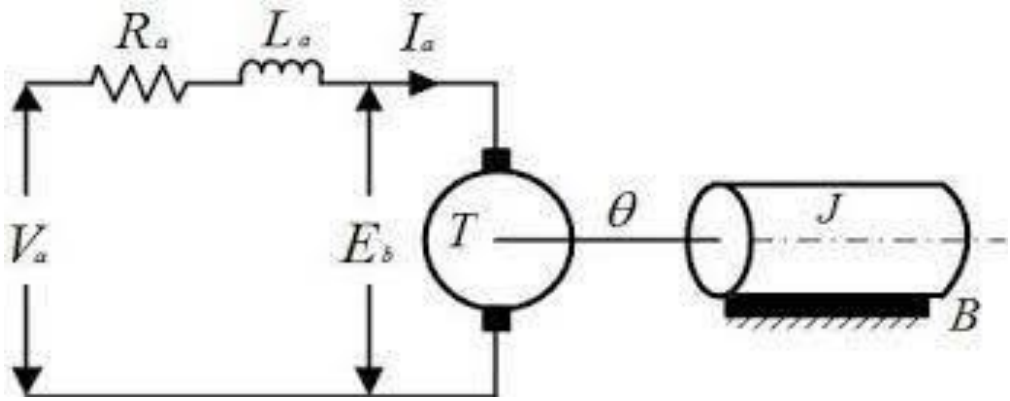


Figure 3.6: DC Motor Circuit Design [35].

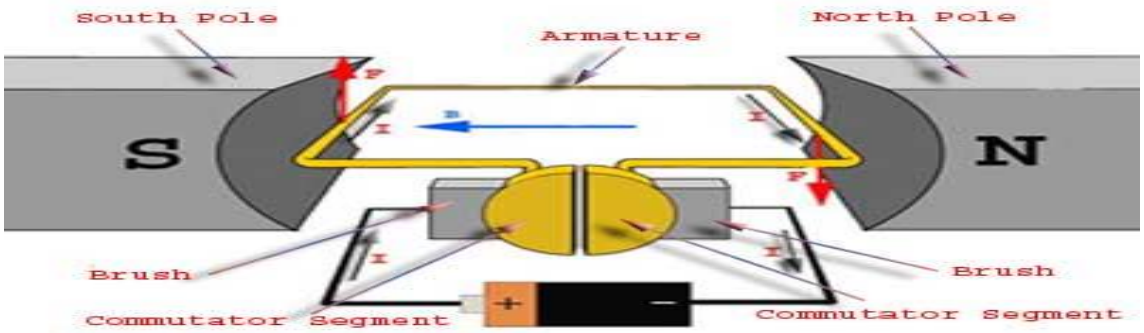


Figure 3.7: DC Motor Electrical Mechanism of Work [36].

V. Vemf, which deals with the engine rotational speed ($\dot{\theta}$) and the engine rotational speed, is called K. So $V.emf = K\omega m$ and ω its $\dot{\theta}$ so the formula $V.emf = K\dot{\theta} = K\frac{d\theta}{dt}$and ωm is the conversion rate. The coil voltage is the winding factor obtained from the coil multiplier current, which coincides with the motor current, because the time series circuit and $VL = L\frac{dIa}{dt}$and $VR = Ra\,Ia$ according to Ohm's law, and this law will be compensation in the Kirchhoff voltage code, and it will create this formula: $Vi - RaIa - L\frac{dIa}{dt} - k\frac{d\theta}{dt} = 0$ The electrode must be isolated from the mechanical part and each part in KVL, when theta has a part of the equations inside its part as a mechanical part, because its processing theta is therefore related to Rotation, so the new formula will change due to a change in the side of the equation, and each section will be moved to the part that you are processing and to the formula **(3.14).**

$$Vi - k\frac{d\theta}{dt} = Ra\,Ia + la\frac{dLa}{dt} \tag{3.14}$$

Thus, the left side of the equation is shown as the mechanical part, which deals with the back EMF, and the right side of the equation is represented as the electrical part, and the equation must be transformed into the Laplace transform, which must be performed and implemented in MATLAB SIMULINK, Laplace transforms of equation **(3.14)** [37].

Table 3.2: Transformation of the electric element in the Laplace transform.

Electrical element	**Laplace Transform**
R	R
$\boldsymbol{Z_L}$	LS
$\boldsymbol{Z_C}$	$\frac{1}{CS}$

The resistance in the table above, which uses resistance and Z_L has resistance for the file, which converted its resistance to LS and Z_C from power, which turned into $\frac{1}{CS}$. This is a transformation of laws into a Laplace transformation, which must be implemented on the Matlab platform and converted to Laplace transform you should to an understanding of this conclusion, as well as the basic principles [38] as follows:

$\frac{d}{dt} = s$ Thus, any conclusion for the period that must be converted to S (Laplace formula)

$\int dt = \frac{1}{s}$Any integral part of time convertible to $\frac{1}{s}$

These formulas will be replaced by equation (1), creating a new formula as follows:

$$Vi(S) - KS\theta(S) = Ra\ Ia(s) + LaS\ Ia(S) \quad (3.15)$$

Each element in the equation indicates that the derivative will be converted to (S), and the armature current is taken as a common factor for the file rotation. He must accept it, and it will create a new equation as follows:

$$Vi(S) - KS\theta(S) = Ia(s)[Ra + LaS] \quad (3.16)$$

This equation has an electric current Ia (s) for use in a DC motor system by dividing both sides by [Ra + Las]. [47] This formula will lead to the following formula:

$$Ia(s) = \frac{Vi(S) - KS\ \theta(S)}{Ra + LaS} \quad (3.17)$$

The current engine was rated. In a DC motor, the electrical torque associated with the current must refer to the power source and to the calculation of the torque, from which we know the speed of the DC motor, because the torque is the same as the mechanical torque, and therefore I convert the formula[39].

3.5.2 The Mechanical Part of DC Motor

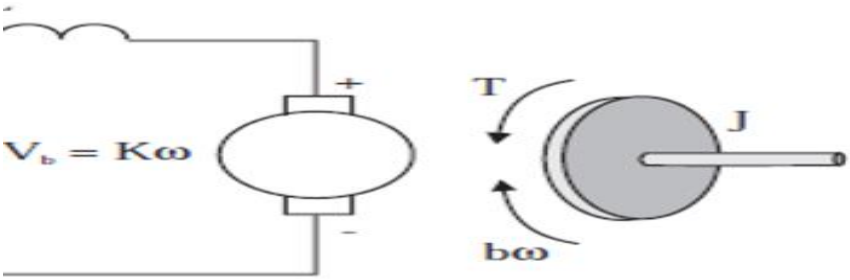


Figure 3.8: DC Motor Mechanical Torque [40].

In the mechanical part, any part can rotate the moment of inertia (J) and has a part around the engine, called the friction coefficient (B).

First, the current torque in a double multiplier with a double multiplier creates a new formula (**3.18**):

$$Te = Kt\ Ia \tag{3.18}$$

Current fittings for electric torque knobs and constant torque Kt

The basic definition, depend on the derivation of a dynamic model of a DC motor, is mechanical equal torque[40].

Secondly, the mechanical moment Tm doubles the moment of inertia during double formation of theta over time, and also adds a coefficient of friction multiplied by its displacement, shown below in the formula (**3.19**):

$$Tm = J\ \frac{d2\ \theta}{dt2} + B\ \frac{d\theta}{dt} \tag{3.19}$$

Thus, if we take efficiency = 100%, now all electrical torque turns into mechanical torque, this proves that Tm = Te, and this completes the new equation in the form of a new formula (**3.20**):

$$J\ \frac{d2\theta}{dt2} + B\ \frac{d\theta}{dt} = K.Ia \tag{3.20}$$

J : moment of polarity inertia.

B : damping constant.

K : constant mechanical moment.

TL : load torque

The parameters J, B and K vary from one motor to another depending on the motor size, current, etc. The above formula for J, B, K will be changed to prohibit the use of the circuit in MATLAB SIMULINK[41].

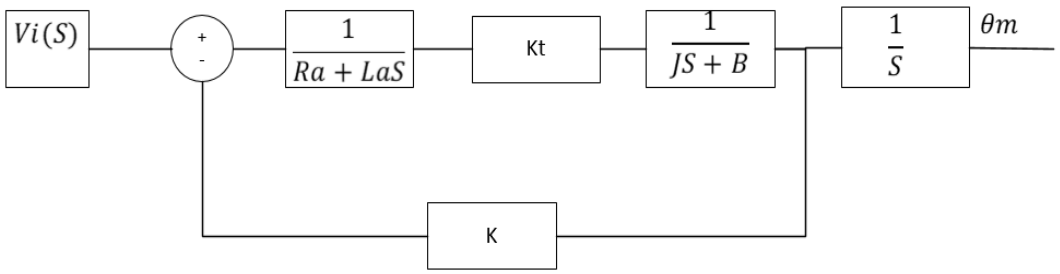


Figure 3.9: Theoretical Block Diagram Design (Laplace Transform) for DC Motor System.

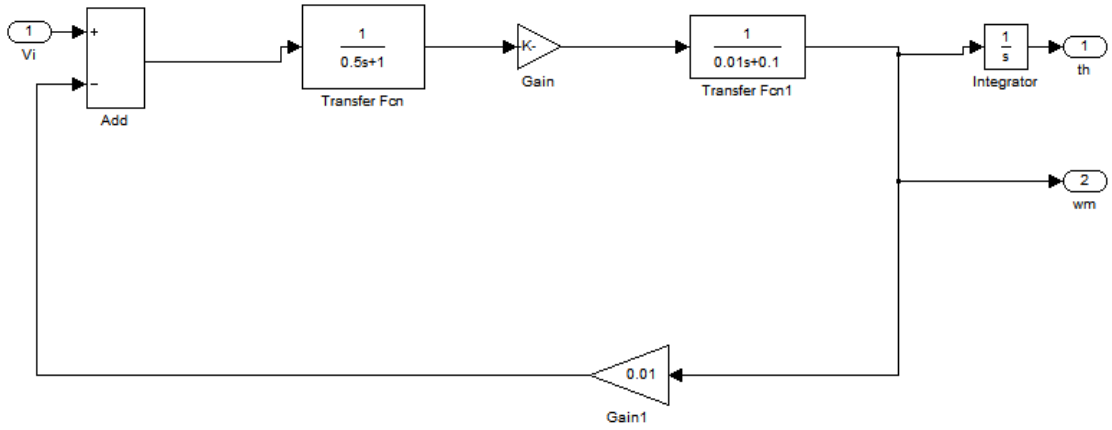


Figure 3.10: DC Motor design in MATLAB SIMULINK.

We will first generate a current, so we need the force VolVi (S) to enter the combination, and the comments will ultimately contain KS $\theta(S)$.To get this equation, we Vi(S)-KS $\theta(S)$ by $\frac{1}{Ra+LaS}$ will be multiplied by Kt, because the current multiplied by k will change to Torque and accept the Laplace transform of equation (4), so we get this formula (**3.21**):

$$Tm(S) = JS\ \omega m(S) + B\omega m(S) \tag{3.21}$$

And the mechanical torque equation is the second or first derivative of speed, therefore, each $\frac{d\theta}{dt} = \omega m$ we can write either in $\frac{d\theta}{dt}$ or ωm and $\omega m(S)$ [42].

$$Tm(S) = \ \omega m(S)\ [JS + B] \tag{3.22}$$

From the torque, we get the speed, so the torque will double by $\frac{1}{JS+B}$, so in MATLAB SIMULINK the torque will double as compared to the previous mass above $\frac{1}{JS+B}$, we will get speed and speed when making an integral part From this we get the angle.

$$\omega m(S) = Tm(S).\frac{1}{JS + B} \tag{3.24}$$

$$\omega m = \frac{d\theta}{dt} \tag{3.25}$$

$$\omega m(S) = S\theta(S) \tag{3.26}$$

$$\theta(S) = \omega m(S) * \frac{1}{S} \tag{3.27}$$

The parameter values that are used to model the DC motor system are listed in the **table 3.3** below:

Table 3.3: DC Motor system Parameters on Matlab.

Parameters	**Value**
1- Mutual Inductance (L)	0.5 H
2- Motor Inertia (J)	0.01 KG.M^2
3- Resistance (R)	1
4- Motor Torque Constant (K)	0.01
5- Damping Coefficient (B)	0.1

These parameters must first enter this value in the MATLAB file in order to apply these values on the Matlab desktop to see what Matlab wants to do.

Thus, MATLAB SIMULINK can take the values of a DC motor system from MATLAB Desktop to display as a symbol, as well as in MATLAB SIMULINK, when you want to communicate with it with symbols, because it also deals with these symbols, so we need the following To implement DC Motor in MATLAB SIMULINK:

Step one: compare the results shown to us in the step.

2) Snake: In addition to adding DC motor parameters and performing an operation.

3) subtraction: react to feedback using the gain obtained from the parameters of the DC motor and the addition to generate speed.

4) When we completed the design of the DC motor system in Simulink MATLAB, the DC motor system became a subsystem, and the external project described the input voltage and micrometer. The DC motor subsystem is shown below:

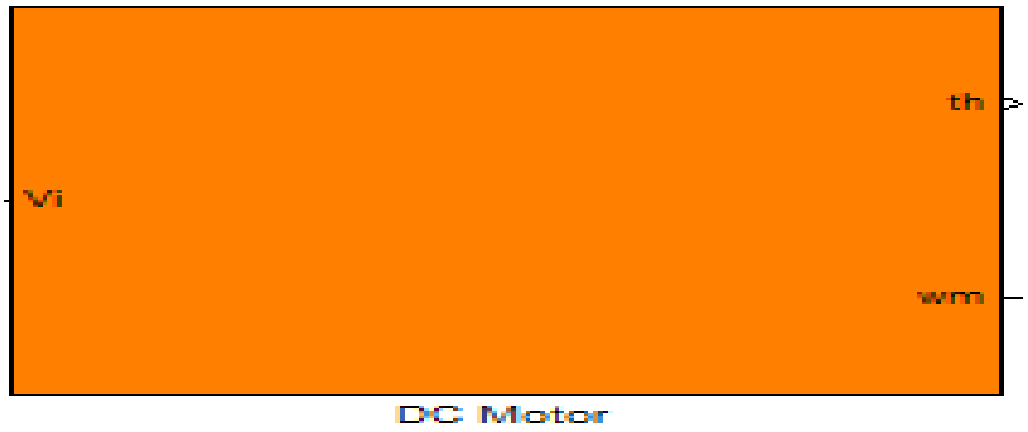


Figure 3.11: The Outside Design of DC Motor System in MATLAB SIMULINK

4. SIMULATION MODEL

4.1 PID WITH DC MOTOR

The console PID application includes a selection of KP, KI and KD. These parameters must be set in such a way that the following properties are respected: the response speed, the alignment time and the corresponding speed override speed, each of which ensures the stability of the system. The (PID) controller is built in the Matlab Which is connected with the DC motor as shown in **Figure 4.1** and is based on KP, KI and KD values. It is the primary form of connection. DC motor previously built in the MATLAB as shown in **Figure 4.2**.

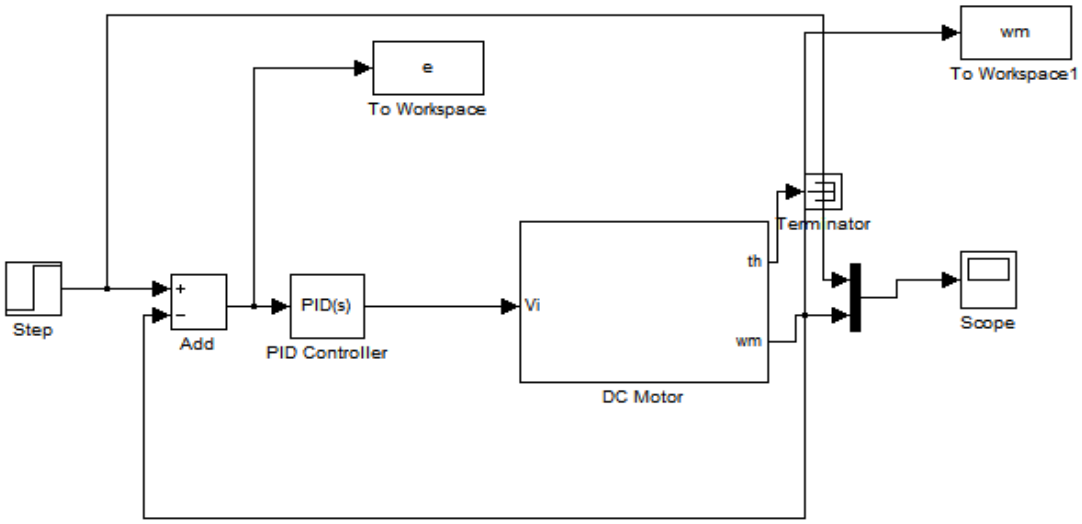


Figure 4.1: PID Controller with DC Motor in MATLAB.

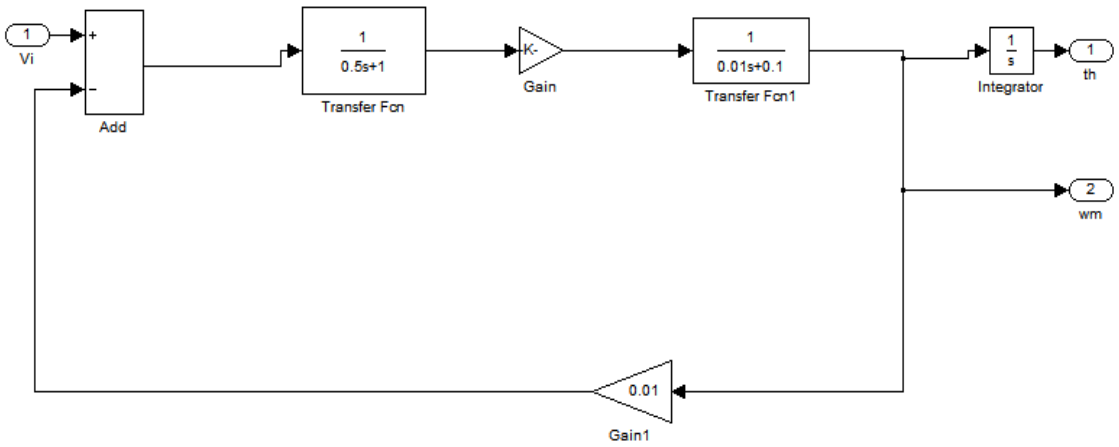


Figure 4.2: DC Motor Built in MATLAB

The main method for this purpose is based on trial and error methods that take time. There are different processes that are relatively different, integrated and differentiated. The responsibility of control technology is to adjust profits to reduce errors and dynamic responses. The function of the (PID) controller is defined like follows (4.1):

$$G(s) = kp + \frac{ki}{s} + kd^{s} \qquad (4.1)$$

(PID) control is a straight control methodology with no difficult control structure. And this type of controller works with the error signal, that is, the difference among the coveted output and the current output, and Creates a working signal that controls the installation. In the design of the PID controller, the number of KI is determined to achieve the destination error in a stable state. In the design of the PID console, KP, KI and KD are selected that are associated with a closed suspension system in less time and require a more period of test and error. As shown in **Figure 4.3**, the KP, KI and KD values used in the controller are ideal controller values.

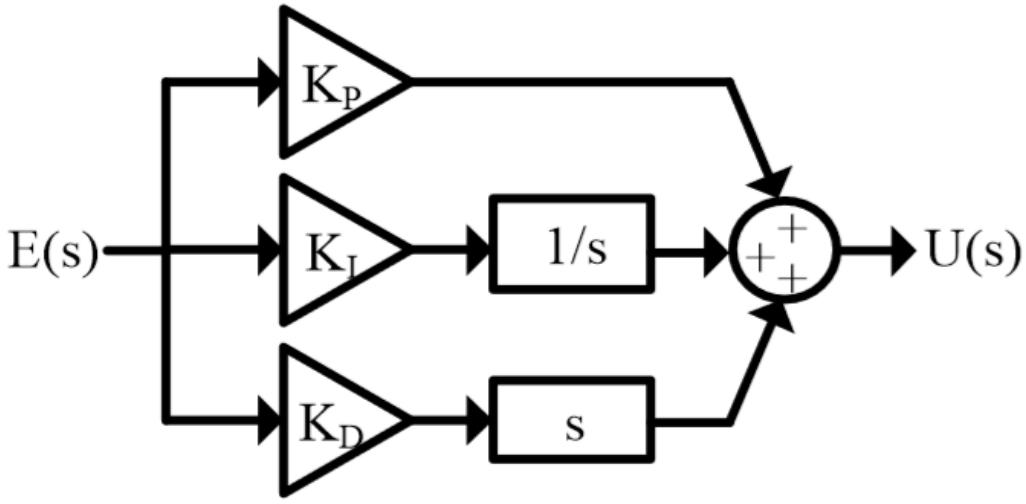


Figure 4.3: Block Diagram of a PID Controller

The values KP = (17.9369594256607) and KI = (43.4547543224494), as well as the value KD = (-0.776867313011485). PIDs contain three basic terms. The operating signal corresponds to the error signal and integrated operation, in which the operating signal corresponds to the received traffic and time of the integrated error signal , wherein the operating signal corresponds to the time derivative of the error signal.

4.2 PID CONTROLLER WITH PSO ALGORITHMS

The PSO algorithms are used by the PID controller to discover the best parameters of the DC motor speed control system. The (PID) system appears with the PSO algorithms.

Figure 4.4.

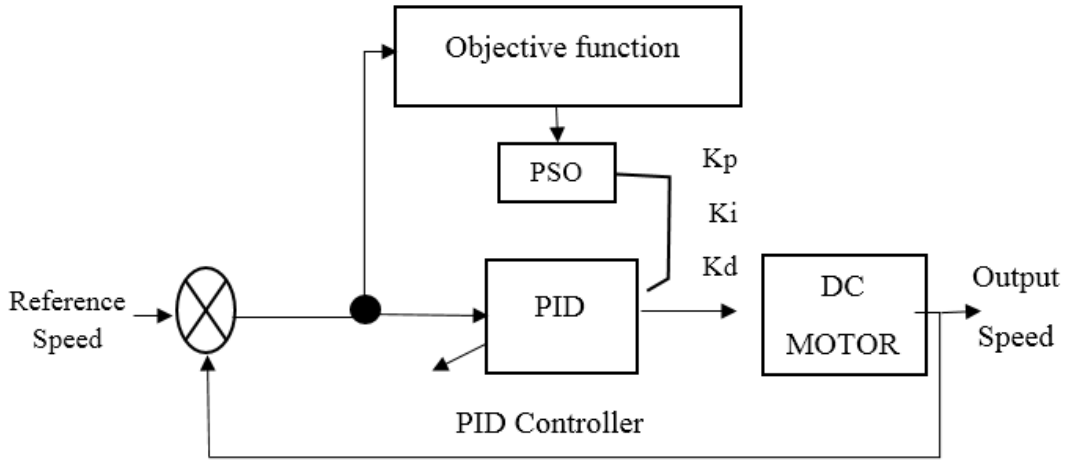


Figure 4.4: Schema A PID Control Block with PSO Algorithm.

The performance characteristics of the control system are low and become unstable if incorrect values are used for the control parameters in the console. For this reason, the parameters of the console must be adjusted to obtain good control characteristics with correct adjustment constants. Unlike conventional methods in which unsuitable particles are excluded and cost-effective particles are often produced, the coupling groups allow us to use the same position in the optimum solution. The Eith molecule consists of the best particles taking different positions on the surface of a hypothetical sphere in the first particle position, the radius of the Euclidean distance between this and the best particles. Each time the molecule obtains a new region, the best particle is updated by comparing the corresponding costs to these positions with the best-predetermined value of the particle. It is also assumed that, at a given moment, the best particle is "scattering" around the remainder of the mass and leads to the formation of a "gravity cone" with axes connecting the best particles of the mass and the rest of the population. The angle is drawn by the vector optimally binds the Eith particle, while the retainer that binds the Eith particle to it is present and next places is in new degrees. In the proposed PSO method, each particle contains three P, I and D members. This means that the search field is in three dimensions and that the particles must "fly" in three-dimension dimensions. KP, KI

and KD values obtained using the optimization equation are ideal values for the controller.

Kp= 9.20701585098555.

Ki= 8.27067408518599.

Kd= -0.009145

Figure 4.5 shows a block diagram of a PSO-PID control system.

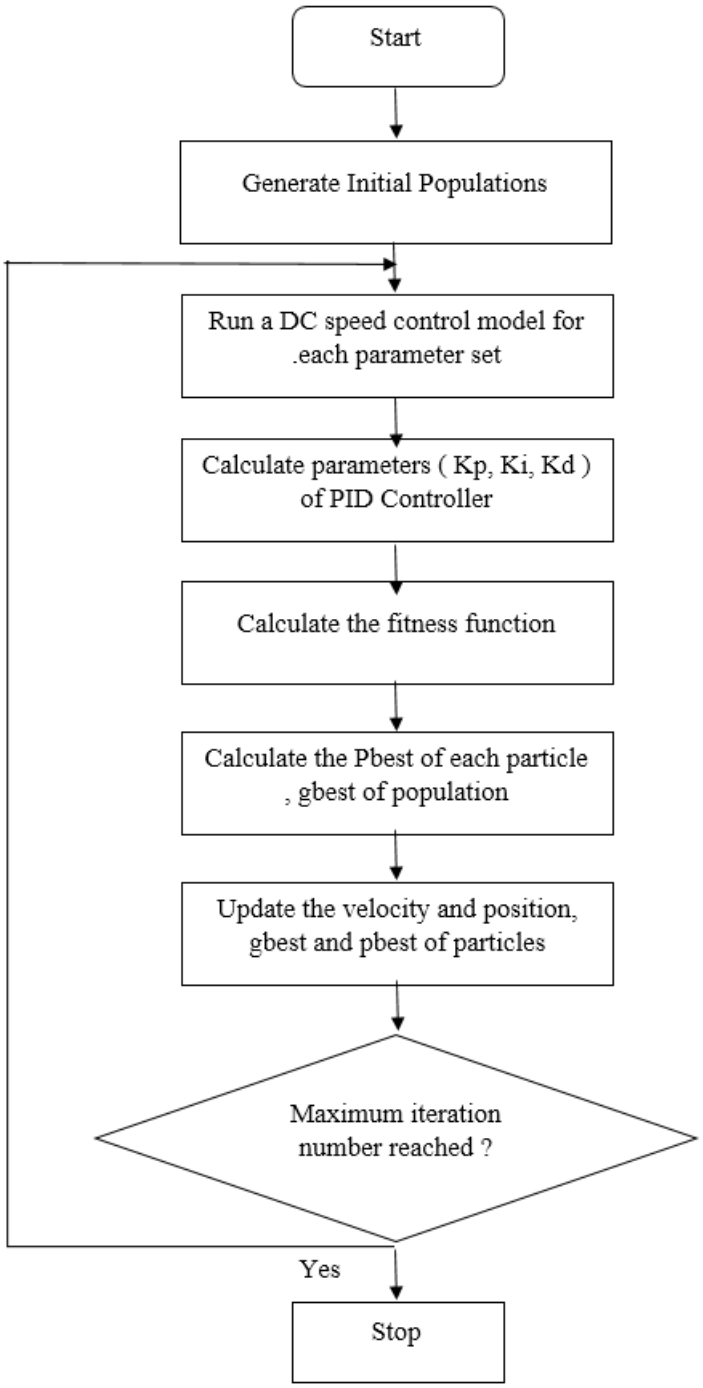


Figure 4.5: The Flowchart of the PSO-PID Control System.

4.3 PID CONTROLLER WITH BAT ALGORTHIM

The PID controller uses bat algorithms to find the optimal parameters for the DC motor speed control system. The PID structure with bat algorithms is shown in Figure 4.6.

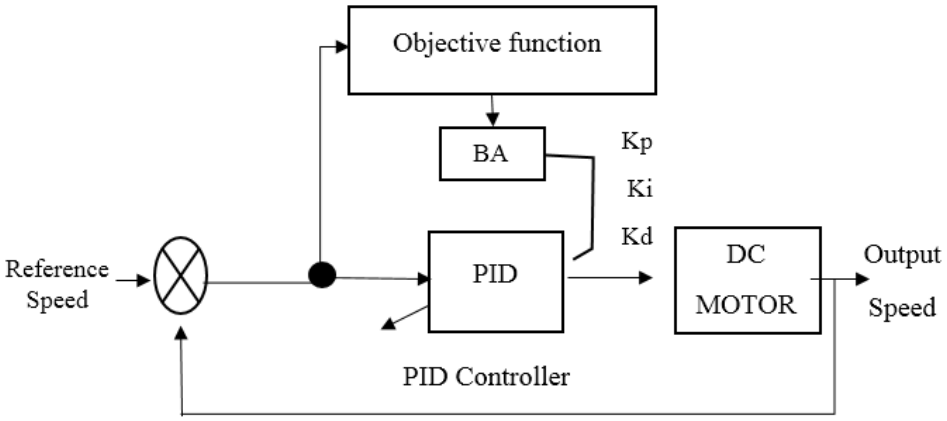


Figure 4.6: The Block Diagram PID Controller with Bat Algorithms.

The optimal settings (KP, KI, KD) can be found using the Beta algorithm using the hybrid tuning method, where the initial parameters are determined using the Ziegler-Nichols algorithm. In this simulation, as shown in Figure 4.7.

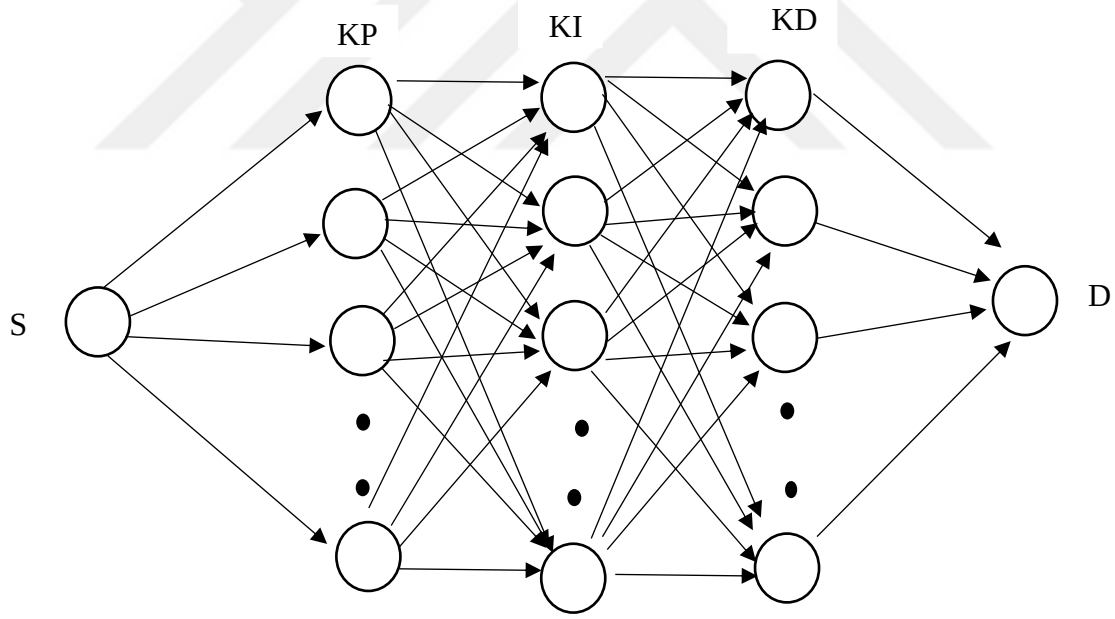


Figure 4.7: Bat Tours.

There are n nodes for KP, KI and KD, the start node and the end node. By default, a bat starts at the start node (S) and ends with a round at the end node (D). Each round in Virtual Bat represents a cost function (performance indicator) for a set of parameters (KP, KI, KD). The initial population of bats consists of bats scattered randomly across n nodes. Configure the capacitance (Aq), heart rate (Rq), frequency (Fq), and speed (Vq) of the

Q^{th}bits. BAT algorithm will determine the optimal values of the controller to work on the control of the DC motor and give the best results while reducing the error rate and increase the angular speed of the DC motor. The values given by the bat algorithm are values (KP, KI, KD). Where the values of Kp= 9.39794667682832 , Ki= 8.21698845205234 and Kd= -0.69345. These values are the values for the binary controller using the bat algorithm.

5. ANALYSIS OF SIMULATION RESULTS AND DISCUSSION

The use of the proportional integral derivative (PID) controller with optimization algorithms such as (PSO) algorithm and bat algorithm (BA) are an attractive solution for the DC motor, which improves the characteristics of the DC motor and increases its efficiency. Through the MATLAB platform, these models are designed and implemented.

5.1 PID CONTROLLER WITH DC MOTOR

The results were obtained in accordance with the MATLAB base system used and applied in the DC motor system, and the results are verified and verified for the proportional integral derivative (PID) controller. To test this control unit, which has been improved for the DC motor system, the numbers below describe the overflow, the time, and the steady state error following:

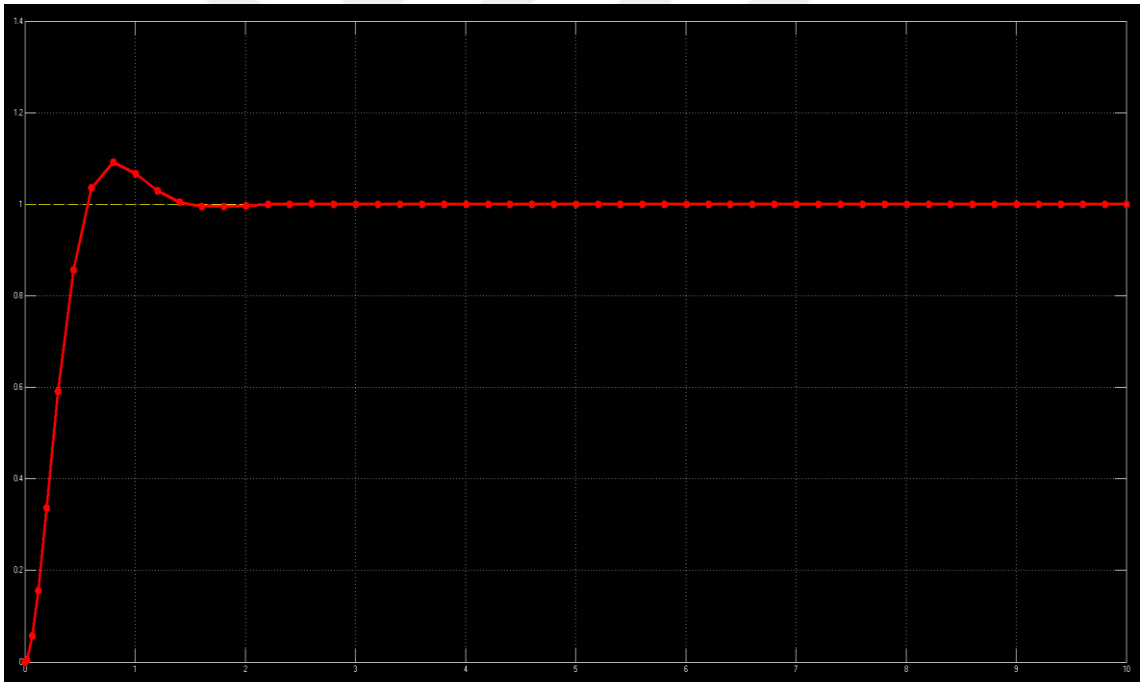


Figure 5.1: The General Curve for PID Controller with DC Motor System with Unit Step by Scope in MATLAB.

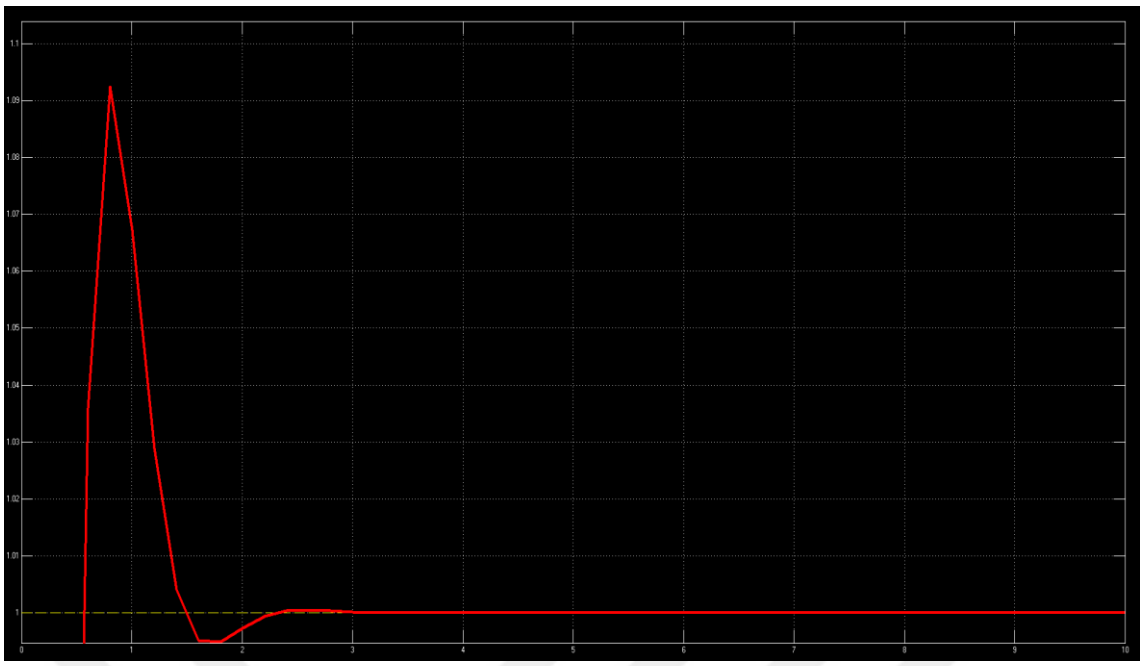


Figure 5.2: The Overshoot Curve for PID Controller with DC Motor System by Scope in MATLAB.

figure 5.2 describes a proportional_integral_derivative (PID) Controller that was stopped at 1.0924 to calculate the mathematical data for this override and represents the difference between the maximum override and the steady state (at any time when the curve is set)

Mathematical calculation (5.1) will take the following law:

$$MP = \frac{yp-yss}{yss} * 100\% \qquad (5.1)$$

figure 5.3 shows the value of yss, which is considered to be a constant value, since there are no changes, so this value is included in the maximum of the overridden rule, as indicated above.

$$MP = \frac{1.0924-0.9997}{0.9997} * 100\%$$

$$MP = 9.267\%$$

This value indicates the maximum deviation of the (PID) controller that is applied to the DC system of the motor.

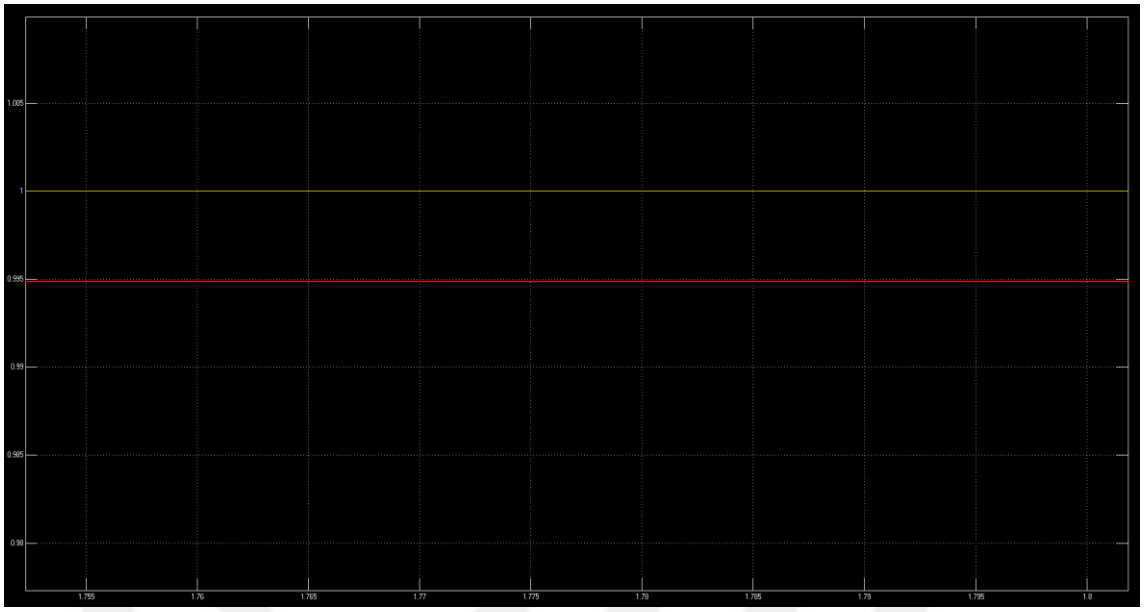


Figure 5.3: The Steady State Error Curve for PID Controller with DC Motor System by Scope in MATLAB.

The Math calculation (5.2) of steady state error for proportional_integral_derivative (PID) controller.

$$steady\ state\ error\ (ess) = 1 - yss = 1 - 0.9949 = 0.0051 \quad (5.2)$$

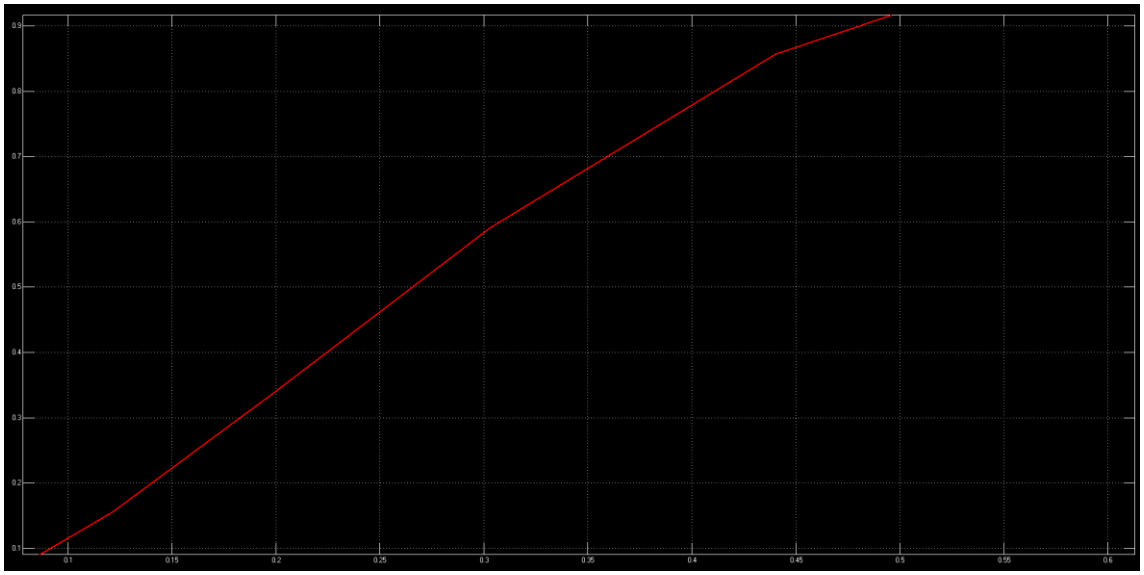


Figure 5.4: The Rising Time Curve for PID Controller with DC Motor System by Scope in MATLAB.

figure 5.4 shows the time during which the control curve of proportional_integral _derivative (PID) rises, representing 90% of the upper value at the beginning of the curve and stabilizing at the level of 0.4804, as shown in the figure above.

5.2 PID CONTROLLER USING PSO ALGORITHM WITH DC MOTOR

Results were obtained according to the MATLAB base system used and applied in the DC motor system. The PID was used with the particle swarm optimization (PSO) algorithm to improve and enhance the DC motor and obtain results that could be better for working with the DC motor. the numbers below describe the overflow, the time, and the steady state error following:

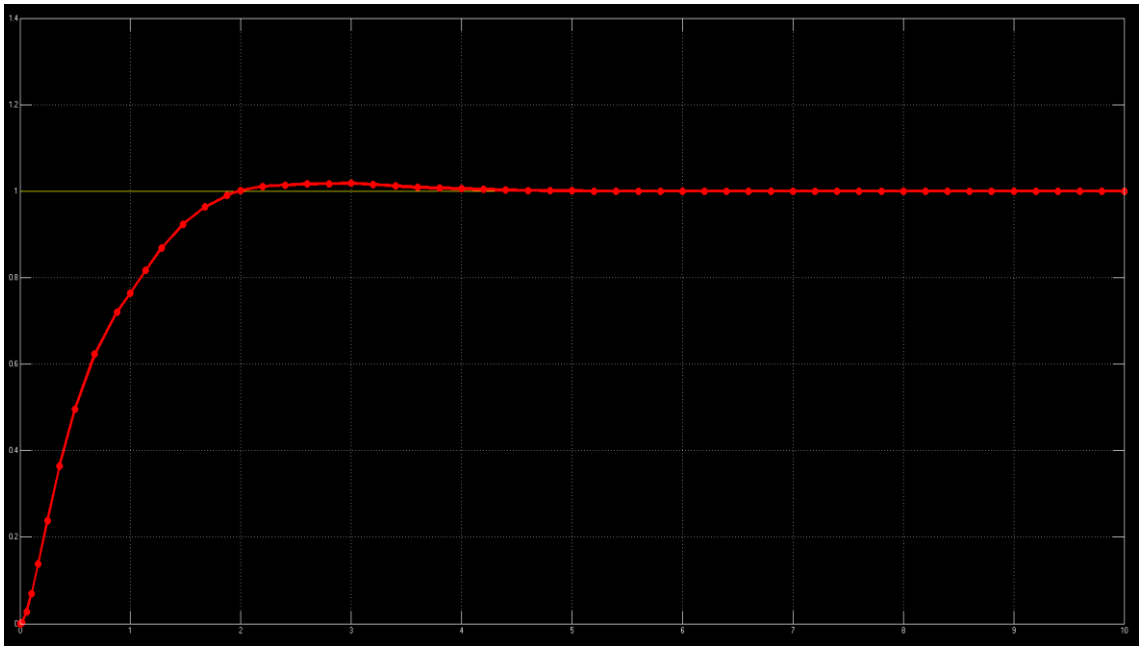


Figure 5.5: The General Curve for PID Controller Using PSO Algorithm with Dc Motor System with Unit Step by Scope in MATLAB.

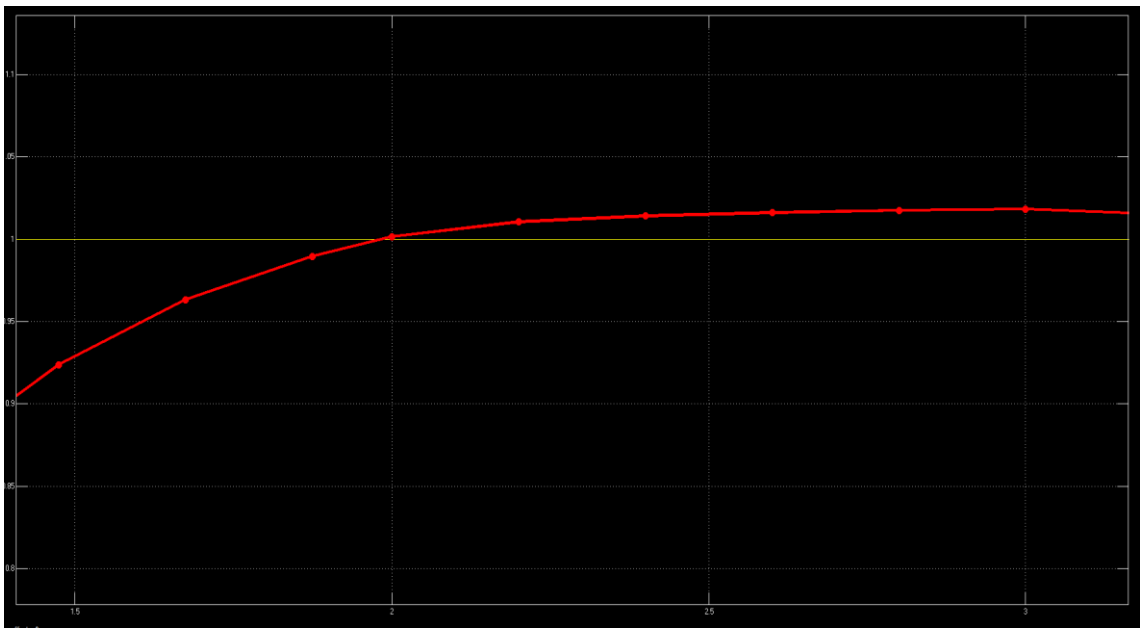


Figure 5.6: The Overshoot Curve for PID Controller Using PSO Algorithm with Dc Motor System with Unit Step by Scope in MATLAB.

Figure 5.6 describes a proportional_integral_derivative (PID) Controller with (PSO) algorithm that was stopped at 1.0183to calculate the mathematical data for this override and represents the difference between the maximum override and the steady state (at any time when the curve is set). By using eq (5.1) we will calculate maximum deviation of (PID) controller with (PSO) algorithm

$$MP = \frac{1.0183-0.9999}{0.9999} * 100\%$$

$$MP = 1.780$$

This value indicates the maximum deviation of the PID controller with PSO algorithm that is applied to the DC system of the motor.

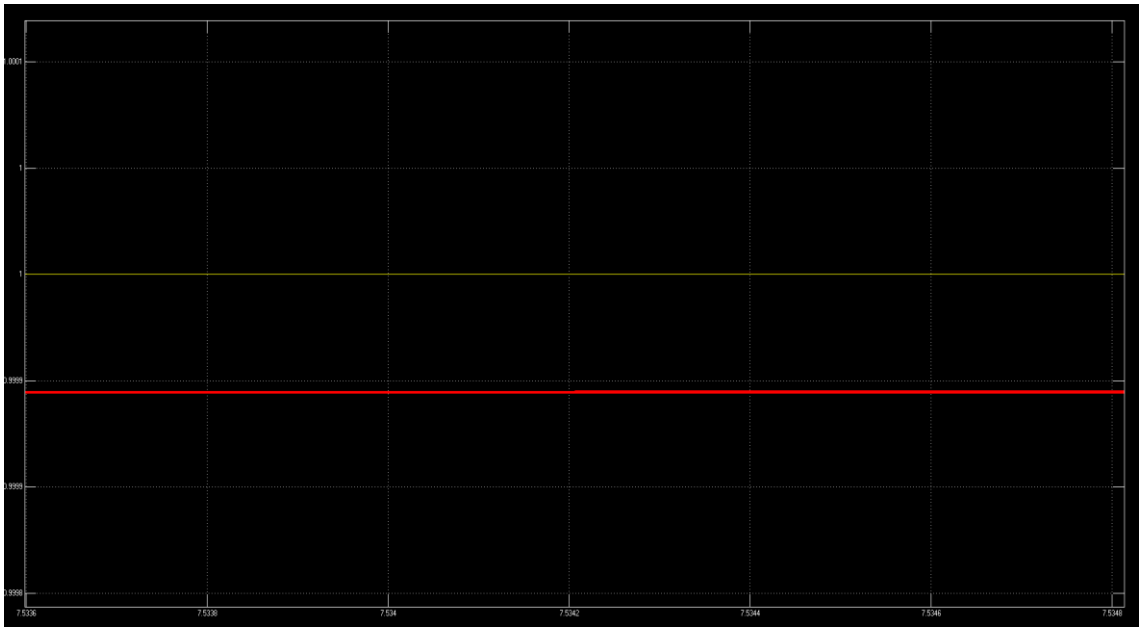


Figure 5.7: The Steady State Error Curve for PID Controller with PSO Algorithm for DC Motor System by Scope in MATLAB.

by using eq(5.2) we will calculate the steady state error for proportional_integral _derivative (PID) controller with (PSO).

$$steady\ state\ error\ (ess) = 1 - yss = 1 - 0.9999 = 0.0001$$

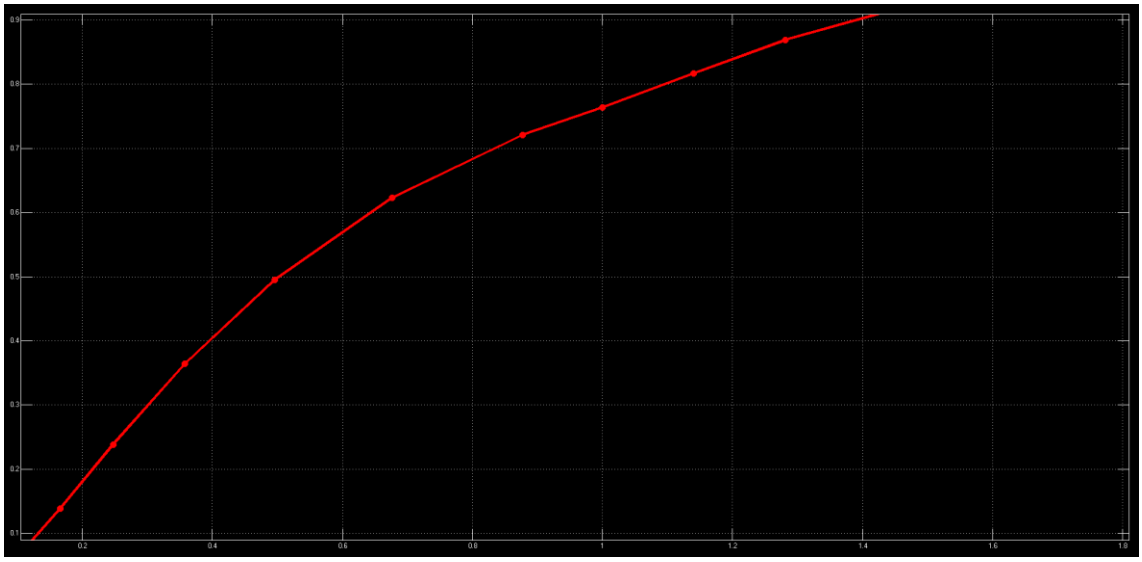


Figure 5.8: The Rising Time Curve for PID Controller with PSO Algorithm for DC Motor System by Scope in MATLAB.

Figure 5.8 shows the time during which the control curve of proportional integral derivative (PID) with (PSO) rises, representing 90% of the upper value at the beginning of the curve and stabilizing at the level of 1.38(sec), as shown in the figure above.

5.3 PID CONTROLLER USING BAT ALGORITHM WITH DC MOTOR

Results were obtained according to the MATLAB base system used and applied in the DC motor system. The proportional integral derivative (PID) was used with the bat algorithm to improve and enhance the DC motor and obtain results that could be better for working with the DC motor. the numbers below describe the overflow, the time, and the steady state error following:

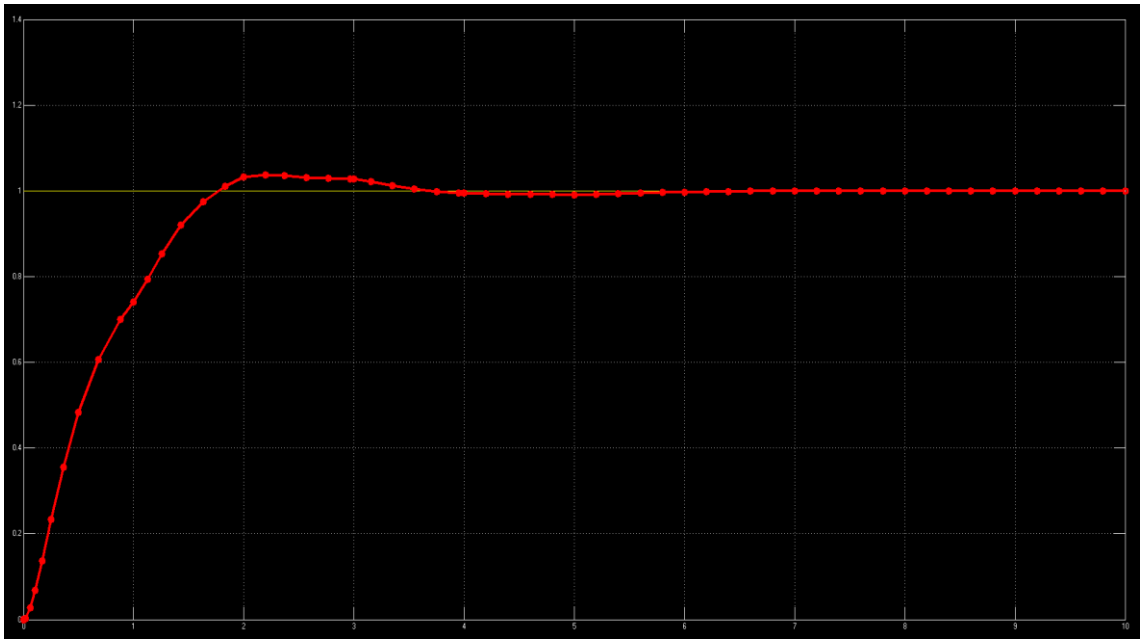


Figure 5.9: The General Curve for PID Controller Using Bat Algorithm with Dc Motor System with Unit Step by Scope in MATLAB.

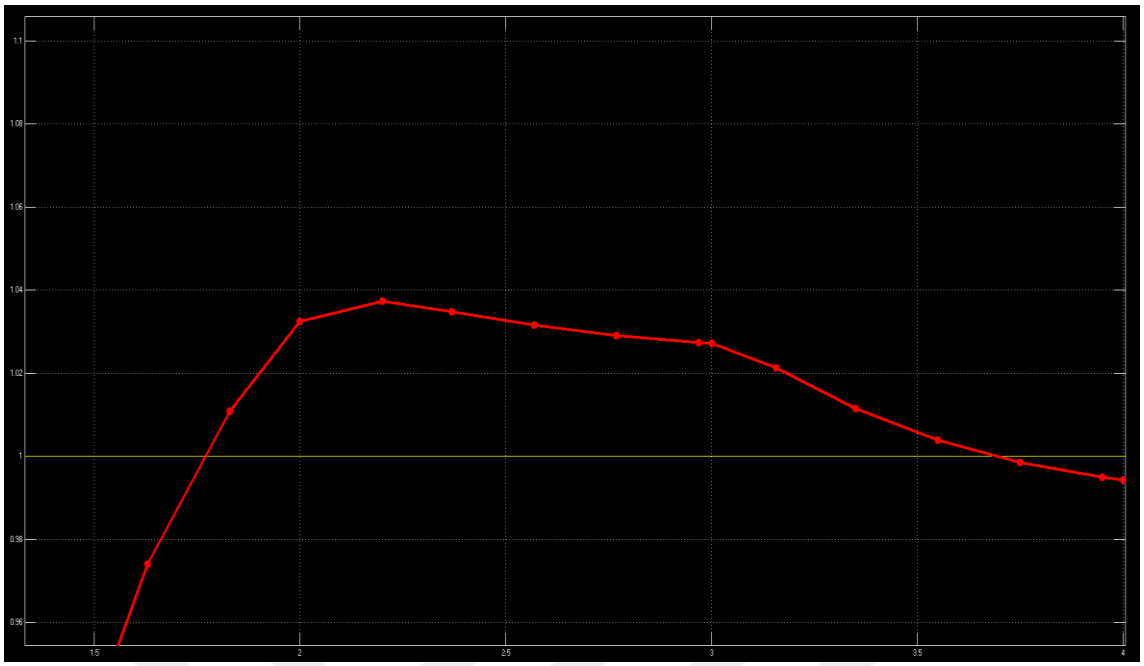


Figure 5.10: The Overshoot Curve for PID Controller Using Bat Algorithm with Dc Motor System with Unit Step by Scope In MATLAB.

Figure 5.10 describes a PID Controller with bat algorithm that was stopped at 1.0375 to calculate the mathematical data for this override and represents the difference between the maximum override and the steady state (at any time when the curve is set). By using eq (5.1) we will calculate maximum deviation of (PID) controller with bat algorithm

$$MP = \frac{1.0375-0.9998}{0.9998} * 100\%$$

$$MP = 3.7707$$

This value indicates the maximum deviation of the(PID) controller with bat algorithm that is applied to the DC system of the motor.

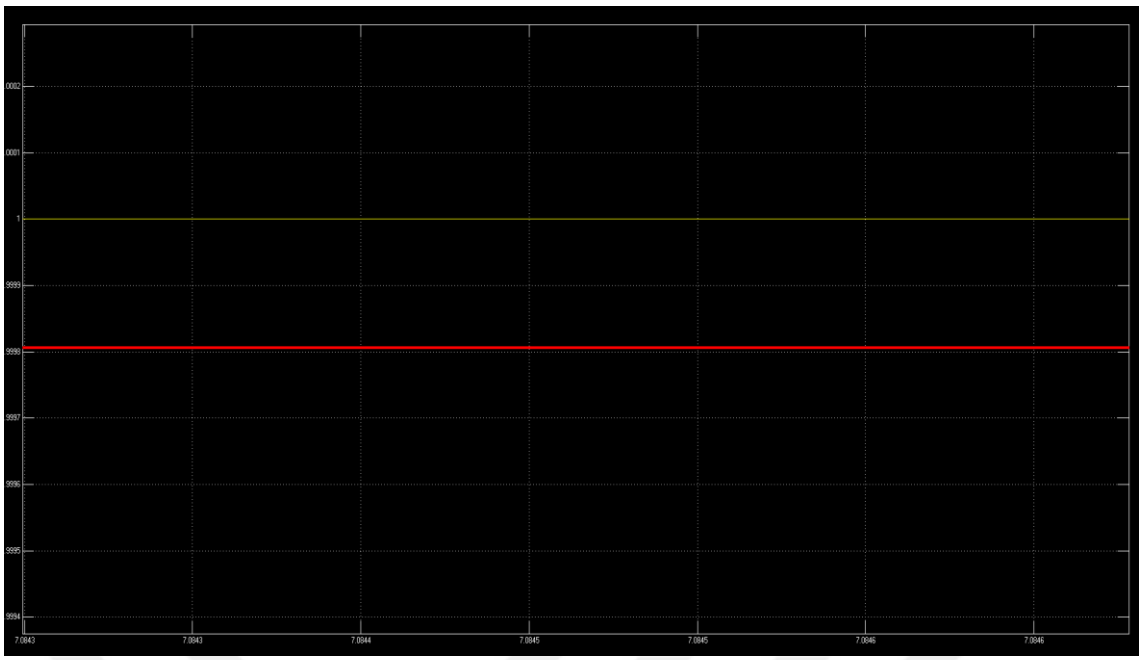


Figure 5.11: The Steady State Error Curve for PID Controller with Bat Algorithm for DC Motor System by Scope in MATLAB.

by using eq(5.2) we will calculate the steady state error for proportional integral derivative (PID) controller with (BA).

$$steady\ state\ error\ (ess) = 1 - yss = 1 - 0.9998 = 0.0002$$

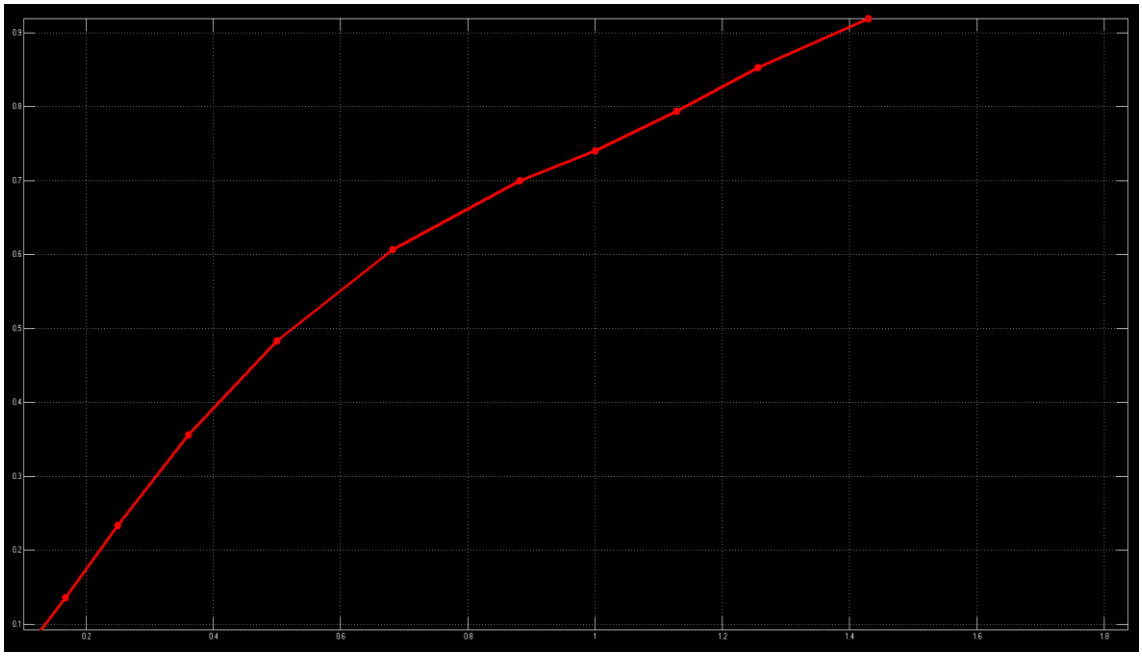


Figure 5.12: The Rising Time Curve for PID Controller with Bat Algorithm for DC Motor System by Scope in MATLAB.

Figure 5.12 shows the time during which the control curve of proportional integral derivative (PID) with (BA) rises, representing 90% of the upper value at the beginning of the curve and stabilizing at the level of 1.36(sec), as shown in the figure above.

5.4 COMPARISON BETWEEN PROPORTIONAL INTEGRAL DERIVATIVE (PID) WITHOUT ALGORITHMS AND (PID) WITH ALGORITHMS

The figures below describes the final figures, and it's the comparison between proportional integral derivative (PID) controller and when the proportional integral derivative (PID) controller used optimization algorithms.

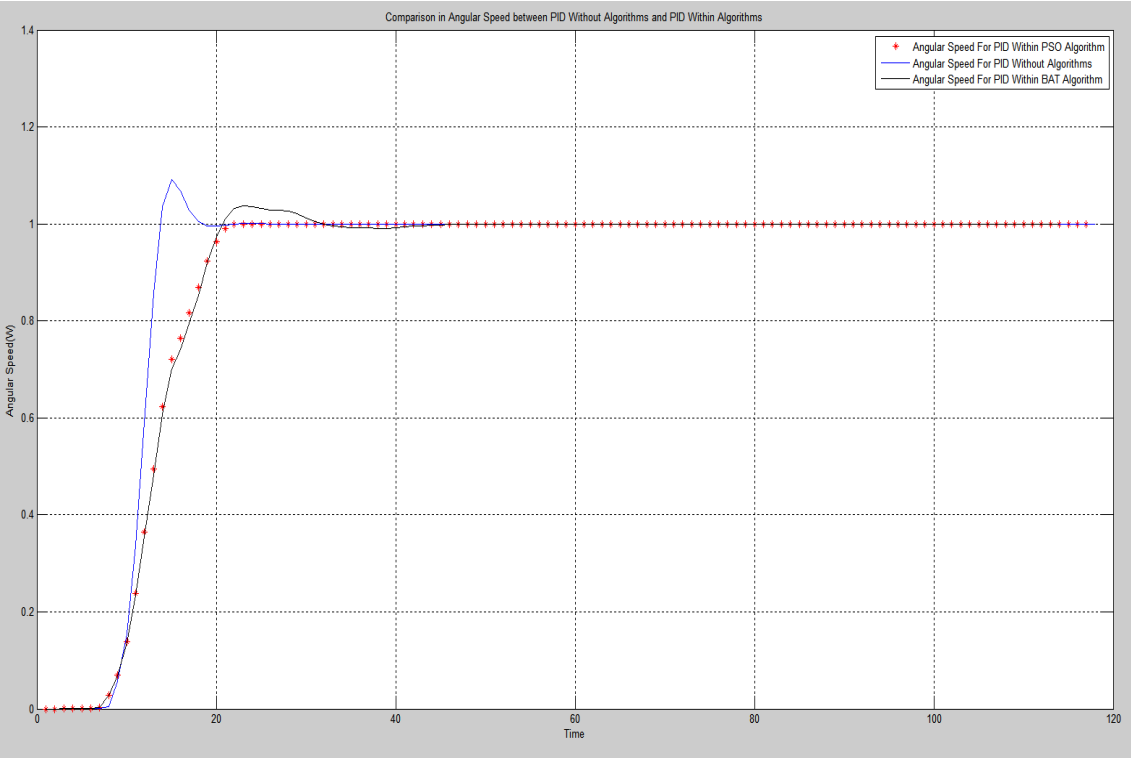


Figure 5.13: Comparison in Angular Speed Between PID Without Algorithms and PID Within Algorithms

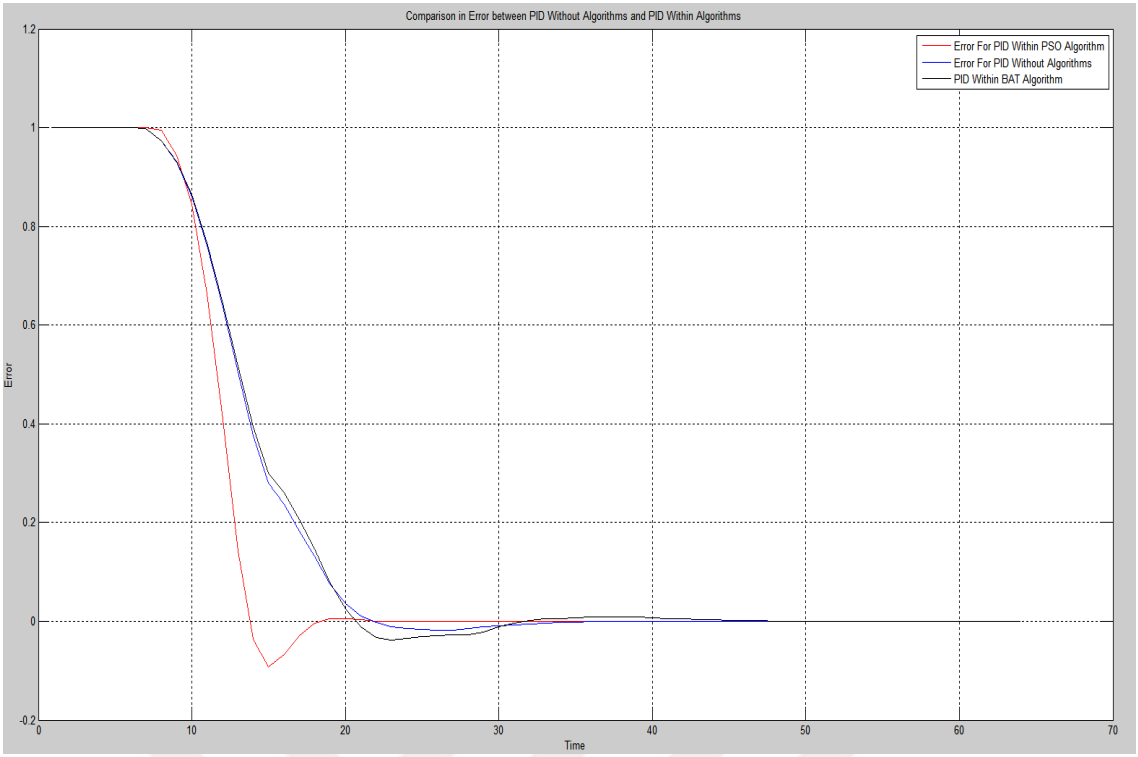


Figure 5.14: Comparison in Error between PID Without Algorithms and PID Within Algorithms.

Table 5.1: The final Results between PID and PID-PSO and PID-BAT.

Control Method	**PID**	**PID-PSO**	**PID-BAT**
Overshoot %	9.267%	1.780%	3.7707%
SteadyState Error	0.0051	0.0001	0.0002
Rising Time	0.48 (sec)	1.38 (sec)	1.38 (sec)

6. CONCLUSION

The system we created is a DC motor. The system was created using MATLAB SIMULINK. For their design, we used basic tools and a mathematical model, and also converted flaps, which we used to convert all equations into blocks, because MATLAB SIMULINK does not accept equations that are not related to the transformation and the Laplace model. Mathematics is similar to the model used in ordinary papers, and we used the PID in accordance with the scale factor (Kp = 17.936 and Ki = 43.454, Kd = -0.776). These values were tested and calculated based on volume factors based on previous studies. The highest yield was given (9.267%) and a linear rate (0.0051). These results are not ideal for a DC motor to work better. For best results, PID optimization algorithms were used. PSO and BA algorithms were used. A comparison was made between PSO and BA characteristics to improve the PID control used to control the speed of a DC motor. It got out to set up a console with the parameters of the PID controller signal, which is based on some of the criteria applied to the input signal of the block of the DC motor of the reference tone of the signal. It was found that skip, rise time and errors stabilized using MATLAB while simultaneously analyzing the output signals. Manual tuning is performed using MATLAB and SIMULINK. At the beginning of the study used manual tuning of DC motors. Then PSO and BA improve the PID controller used in the DC motor. Compared with exceeding the maximum, the height in time and the error of the fixed state. The maximum Overshoot in the PSO algorithm was (1.780%), the climb time (1.38 sec) and the SteadyState Error (0.0001), the maximum Overshoot in the bat algorithm was (3.7707%), the rise time (1.36 sec) SteadyState Error (0.002) These results show that the PSO algorithm with a microcontroller (PID) is better than the bat algorithm in improving and improving the DC motor.

In future work, I would recommend anyone who wants to complete thesis is to use another optimization techniques methods (algorithms) like glowworm swarm optimization algorithm (GSO) or cat swarm optimization algorithm(CSO) I think it will give best results.

REFERENCES

[1] T. N. Trong and A. Info, “The Control Structure for DC Motor based on the Flatness Control,” *Int. J. Power Electron. Drive Syst.*, vol. 8, no. 4, pp. 1814–1821, 2017.

[2] A. A. A. Emhemed and R. Bin, “Modelling and Simulation for Industrial DC Motor Using Intelligent Control Modelling and Simulation for Industrial DC Motor Using Intelligent Control,” *Procedia Eng.* ·, no. June 2014, pp. 420 – 425, 2012.

[3] F. Yener, S. Soysal, H. Yazgan, F. Yener, and S. Soysal, “Comparison Performances of PSO and GA to Tuning PID Controller for the DC Motor,” *Sak. Univ. J. Sci.*, vol. 23, no. 39539, pp. 162–174, 2019.

[4] J. Ohri, “FUZZY Based PID Controller for Speed Control of D . C . Motor Using LabVIEW 2 DC Motor Mathematical Model,” *Wseas Trans. Syst. Control*, vol. 10, pp. 154–159, 2015.

[5] A. Abdulameer, M. Sulaiman, M. S. M. Aras, and D. Saleem, “Tuning Methods of PID Controller for DC Motor Speed Control Tuning Methods of PID Controller for DC Motor Speed Control,” *Indones. J. Electr. Eng. Comput. Sci.*, vol. 3, no. August, pp. 343–349, 2016.

[6] S. W. Khubalkar, A. S. Junghare, M. V Aware, A. S. Chopade, and S. Das, “Demonstrative fractional order – PID controller based DC motor drive on digital platform Demonstrative fractional order – PID controller based DC motor drive on digital platform,” *ISA Trans.*, no. February 2018, 2017.

[7] J. Viola, L. Angel, and J. M. Sebastian, “Design and Robust Performance Evaluation of a Fractional Order PID Controller Applied to a DC Motor,” *IEEE/CAA J. Autom. Sin.*, vol. 4, no. 2, pp. 304–314, 2017.

[8] A. Idir, M. Kidouche, Y. Bensafia, K. KhettabSid, and sid A. Tadjer4, “Speed Control of DC Motor Using PID and FOPID Controllers Based on Differential Evolution and PSO,” *Int. J. Intell. Eng. Syst.*, vol. 11, no. 4, pp. 241–249, 2018.

[9] N. Q. Mohammed, “DC Motor Drive with P , PI , and Particle Swarm Optimization Speed Controllers,” *Int. J. Comput. Appl.*, vol. 166, no. 12, pp. 42–45, 2017.

[10] N. D. Pandey, M. T. Scholor, and T. Science, “COMPARISON BETWEEN SPEED CONTROL DC MOTOR USING GENETIC ALGORITHM AND PSO-PID,” *Int. J. Electr. Eng. Technol.*, vol. 8, no. 1, pp. 17–25, 2017.

[11] W. Tang, Z. Liu, and Q. Wang, “DC Motor Speed Control Based on System Identification and PID Auto Tuning,” *Proc. 36th Chinese Control Conf.*, no. 61403422, pp. 6420–6423, 2017.

[12] Y. Li, H. A. Kiam, and G. Chong, “PID Control System Analysis , Design , and Technology,” *IEEE Trans. Control Syst. Technol.*, vol. 13, no. May, pp. 559–574, 2013.

[13] G. Kasilingam and J. Pasupuleti, “Coordination of PSS and PID Controller for Power System Stability Enhancement – Overview,” *Indian J. Sci. Technol.*, vol. 8, no. January, pp. 142–151, 2015.

[14] R. Paz, “The Design of the PID Controller The Design of the PID Controller,” *Klipsch Sch. Electr. Comput. Eng.*, no. January 2001, 2014.

[15] E. Engineering and E. Engineering, “PID Controller Tuning using Ziegler-Nichols Method for Speed Control of DC Motor,” *Int. J. Sci. Eng. Technol. Res.*, vol. 03, no. 13, pp. 2924–2929, 2014.

[16] D. Wang, D. Tan, and L. Liu, “Particle swarm optimization algorithm : an overview Particle swarm optimization algorithm : an overview,” *Soft Comput.*, vol. 22, no. 2, pp. 387–408, 2018.

[17] B. Allaoua, B. Gasbaoui, and B. Mebarki, “Setting Up PID DC Motor Speed Control Alteration Parameters Using Particle Swarm Optimization Strategy,” no. 14, pp. 19–32, 2009.

[18] J. Kennedy, “The particle swarm: social adaptation of knowledge,” in *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC’97)*, 1997, pp. 303–308.

[19] P. J. Angeline, N. Selection, and C. Street, “Using Selection to Improve Particle Swarm Optimization,” pp. 84–89.

[20] R. Eberhart, P. Simpson, and R. Dobbins, *Computational intelligence PC tools*. Academic Press Professional, Inc., 1996.

[21] J. Kennedy, “Eberhart, R.: Particle Swarm Optimization,” in *Proceedings of IEEE international conference on neural networks*, 1995, vol. 4, no. 2, pp. 1942–1948.

[22] K. Parsopoulos and M. N. Vrahatis, “Modi cation of the Particle Swarm Optimizer for locating all the global minima,” no. January 2013, pp. 3–7, 2001.

[23] M. J. Matarić, “Designing and understanding adaptive group behavior,” *Adapt. Behav.*, vol. 4, no. 1, pp. 51–80, 1995.

[24] M. M. Millonas, “Swarms, phase transitions, and collective intelligence,” *arXiv Prepr. adap-org/9306002*, 1993.

[25] W. T. Reeves, “Particle SystemsmA Technique for Modeling a Class of Fuzzy Objects,” vol. 2, no. 2, pp. 91–108, 2009.

[26] A. Patel and K. Parikh, “Speed Control of DC Motor Using PSO Tuned PI Controller,” vol. 9, no. 2, pp. 4–8, 2014.

[27] S. J. Bassi, M. K. Mishra, and E. E. Omizegba, “Automatic tuning of proportional-integral-derivative (PID) controller using particle swarm optimization (PSO) algorithm,” *Int. J. Artif. Intell. Appl.*, vol. 2, no. 4, p. 25, 2011.

[28] A. N. M. B. Algorithm and X. Yang, “arXiv : 1004 . 4170v1 [math . OC] 23 Apr 2010,” pp. 1–10.

[29] W. H. Bangyal, “An Overview of Mutation Strategies in Bat Algorithm,” vol. 9, no. 8, pp. 523–534, 2018.

[30] X. Yang, “Bat Algorithm : Literature Review and Applications,” pp. 1–10, 2013.

[31] X. Yang, “Bat Algorithm : A Novel Approach for Global Engineering Optimization,” 2012.

[32] I. F. Jr, I. Fister, and X. Yang, “Bat algorithm : Recent advances.”

[33] S. Yılmaz and E. U. Küc, “A new modification approach on bat algorithm for solving optimization problems,” vol. 28, pp. 259–275, 2015.

[34] S. Induja and V. P. Eswaramurthy, “Bat Algorithm : An Overview and its Applications,” vol. 5, no. 1, pp. 448–451, 2016.

[35] K.-W. Yu and J.-H. Hsu, “Fuzzy gain scheduling PID control design based on particle swarm optimization method,” in *Second International Conference on Innovative Computing, Informatio and Control (ICICIC 2007)*, 2007, p. 337.

[36] A. Kumar and S. Suhag, “Multiverse optimized fuzzy-PID controller with a derivative filter for load frequency control of multisource hydrothermal power system,” *Turkish J. Electr. Eng. Comput. Sci.*, vol. 25, no. 5, pp. 4187–4199, 2017.

[37] A. Nabi, “Design of fuzzy logic PD controller for a position control system,” *Int. J. Eng. Manag. Res.*, vol. 3, no. 2, pp. 31–34, 2013.

[38] H. Kala, D. Deepakraj, P. Gopalakrishnan, P. Vengadesan, and M. K. Iyyar, “Performance evaluation of fuzzy logic and PID controller for liquid level process,” *Perform. Eval.*, vol. 2, no. 3, 2014.

[39] N. B. Mohamadwasel and O. Bayat, “Improve DC Motor System using Fuzzy Logic Control by Particle Swarm Optimization in Use Scale Factors,” vol. 8, no. 3, pp. 152–160, 2019.

[40] Y. Al-Dunainawi and M. F. Abbod, “Pso-pd fuzzy control of distillation column,” in *2015 SAI Intelligent Systems Conference (IntelliSys)*, 2015, pp. 554–558.

[41] I. Pan and S. Das, “Fractional order fuzzy control of hybrid power system with renewable generation using chaotic PSO,” *ISA Trans.*, vol. 62, pp. 19–29, 2016.

[42] B. Akbıyık, Đ. Eksin, M. Güzelkaya, and E. Yeşil, “Evaluation of the performance of various fuzzy PID controller structures on benchmark systems,” in *ELECO ‘2005, 4rd International Conf. on Electrical and Electronics Engineering*, 2005.

APPENDIX A

CODES

A.1 DC MOTOR PARAMETERS IN MATLAB CODE

```
clc;
clear all;
close all;
%Parameters of DC Motor
L=1;%Mutual Inductunce (H)
J=0.02; % Motor Inertia
R=1.5;%Resistance of Armeture
Kt=0.01;%Motor Torque constant
Kb=0.02;
B=0.2;% Damping coefficient
```

A.2 PSO ALGORITHMS

```
clear
clc
%Parameters of DC Motor
L=0.5;%Mutual Inductunce (H)
J=0.01; % Motor Inertia
R=1;%Resistance of Armeture
Kt=0.01;%Motor Torque constant
Kb=0.01;
B=0.1;% Damping coefficient
n = 50;             % Size of the swarm " no of birds "
bird_setp  =50;     % Maximum number of "birds steps"
dim = 3;            % Dimension of the problem

c2 =1.2;            % PSO parameter C1
c1 = 0.12;          % PSO parameter C2
w =0.9;             % pso momentum or inertia
fitness=0*ones(n,bird_setp);

                                        %------------------------------%
                                        %    initialize the parameter %
                                        %------------------------------%

R1 = rand(dim, n);
R2 = rand(dim, n);
current_fitness =0*ones(n,1);

                                    %-------------------------------------
------------%
                                    % Initializing swarm and velocities
and position %
                                    %-------------------------------------
------------%

 current_position = 17*(rand(dim, n)-0.5);
% current_position = 50*(rand(dim, n));
```

```
velocity = .3*randn(dim, n) ;
local_best_position  = current_position ;


                                %-----------------------------------
-------%
                                %     Evaluate initial population
%
                                %-----------------------------------
-------%

for i = 1:n
    current_fitness(i) = NOR_Tunning(current_position(:,i));
end


local_best_fitness  = current_fitness ;
[global_best_fitness,g] = min(local_best_fitness) ;

for i=1:n
    globl_best_position(:,i) = local_best_position(:,g) ;
end
                                                %--------------------%
                                                %  VELOCITY UPDATE   %
                                                %--------------------%

velocity = w *velocity + c1*(R1.*(local_best_position-
current_position)) + c2*(R2.*(globl_best_position-current_position));

                                                %-------------------%
                                                %   SWARMUPDATE     %
                                                %-------------------%


current_position = current_position + velocity ;

                                                %----------------------
--%
                                                %  evaluate anew swarm
%
                                                %----------------------
--%


%% Main Loop
iter = 0 ;        % Iterations'counter
while  ( iter < bird_setp )
iter = iter + 1;

for i = 1:n,
current_fitness(i) = NOR_Tunning(current_position(:,i)) ;
end


for i = 1 : n
        if current_fitness(i) < local_best_fitness(i)
           local_best_fitness(i)  = current_fitness(i);
           local_best_position(:,i) = current_position(:,i)   ;
        end
 end
```

```
 [current_global_best_fitness,g] = min(local_best_fitness);


if current_global_best_fitness < global_best_fitness
   global_best_fitness = current_global_best_fitness;

    for i=1:n
        globl_best_position(:,i) = local_best_position(:,g);
    end

end


 velocity = w *velocity + c1*(R1.*(local_best_position-
current_position)) + c2*(R2.*(globl_best_position-current_position));
 current_position = current_position + velocity;

 sprintf('The value of interation iter %3.0f ', iter );


end % end of while loop its mean the end of all step that the birds
move it


%               xx=fitness(:,50);
%               [Y,I] = min(xx);
%               current_position(:,I)
kk=globl_best_position;
K1=kk(1)
K2=kk(2)
K3=kk(3)
```

A.3 BAT ALGORITHMS

```
function [best,fmin,N_iter]=bat_algorithm(para)
% Display help
 help bat_algorithm.m
% Default parameters
clc;
clear all;
close all;
%Parameters of DC Motor
L=1;%Mutual Inductunce (H)
J=0.02; % Motor Inertia
R=1.5;%Resistance of Armeture
Kt=0.01;%Motor Torque constant
Kb=0.02;
B=0.2;% Damping coefficient
%%%%%%%%%%%%PID Parameters%%%%%%%%%%%%%%
% Kp=1;
% Kd=1;
% Ki=1;
if nargin<1,  para=[20 1000 0.5 0.5];  end
n=para(1);      % Population size, typically 10 to 40
N_gen=para(2);  % Number of generations
A=para(3);      % Loudness  (constant or decreasing)
r=para(4);      % Pulse rate (constant or decreasing)
% This frequency range determines the scalings
% You should change these values if necessary
Qmin=0;         % Frequency minimum
Qmax=1000;          % Frequency maximum
% Iteration parameters
N_iter=200;       % Total number of function evaluations
% Dimension of the search variables
d=200;            % Number of dimensions
% Lower limit/bounds/ a vector
Lb=9*ones(1,d);
% Upper limit/bounds/ a vector
Ub=10*ones(1,d);
% Initializing arrays
Q=zeros(n,1);   % Frequency
v=zeros(n,d);   % Velocities
% Initialize the population/solutions
for i=1:n,
  Sol(i,:)=Lb+(Ub-Lb).*rand(1,d);
  Fitness(i)=Fun(Sol(i,:));
end
% Find the initial best solution
[fmin,I]=min(Fitness);
best=Sol(I,:);
% =============================================================   %
% Note: As this is a demo, here we did not implement the        %
% reduction of loudness and increase of emission rates.         %
% Interested readers can do some parametric studies             %
% and also implementation various changes of A and r etc        %
% =============================================================   %
% Start the iterations -- Bat Algorithm (essential part)        %
for t=1:N_gen,
% Loop over all bats/solutions
        for i=1:n,
          Q(i)=Qmin+(Qmin-Qmax)*rand;
          v(i,:)=v(i,:)+(Sol(i,:)-best)*Q(i);
          S(i,:)=Sol(i,:)+v(i,:);
          % Apply simple bounds/limits
```

```
            Sol(i,:)=simplebounds(Sol(i,:),Lb,Ub);
            % Pulse rate
            if rand>r
            % The factor 0.001 limits the step sizes of random walks
                S(i,:)=best+0.001*randn(1,d);
            end
      % Evaluate new solutions
             Fnew=Fun(S(i,:));
      % Update if the solution improves, or not too loud
             if (Fnew<=Fitness(i)) && (rand<A) ,
                  Sol(i,:)=S(i,:);
                  Fitness(i)=Fnew;
             end
            % Update the current best solution
            if Fnew<=fmin,
                  best=S(i,:);
                  fmin=Fnew;
            end
          end
          N_iter=N_iter+n;
end
% Output/display
disp(['Number of evaluations: ',num2str(N_iter)]);
disp(['Best =',num2str(best),' fmin=',num2str(fmin)]);
% Application of simple limits/bounds
function s=simplebounds(s,Lb,Ub)
  % Apply the lower bound vector
  ns_tmp=s;
  I=ns_tmp<Lb;
  ns_tmp(I)=Lb(I);

  % Apply the upper bound vector
  J=ns_tmp>Ub;
  ns_tmp(J)=Ub(J);
  % Update this new move
  s=ns_tmp;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Objective function: your own objective function can be written here
% Note: When you use your own function, please remember to
%       change limits/bounds Lb and Ub (see lines 52 to 55)
%       and the number of dimension d (see line 51).
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function z=Fun(u)
% Sphere function with fmin=0 at (0,0,...,0)
z=sum(u.^2);
%%%%% ============ end ===================================
```